


Agile For Data Professionals





IBM Software Group

Agile Strategies for Data Professionals

Scott W. Ambler
Chief Methodologist for Agile/Lean, IBM Rational

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



© 2010 IBM Corporation

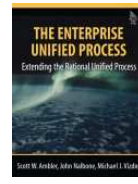
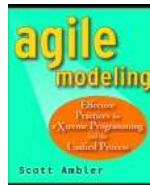
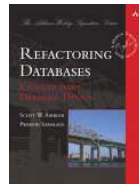
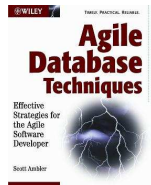
Workshop goals:

- Learn about the fundamentals of disciplined agile delivery
- Explore the challenges surrounding data quality within modern organizations
- Learn about the fundamentals of data quality management
- Overview agile database techniques for addressing data quality
- Discover how to address enterprise concerns in an agile/lean manner
- Learn how to scale agile strategies to meet the real-world situations which organizations find themselves in

Agile For Data Professionals

About this Workshop

- This one day workshop was delivered by Scott W. Ambler on April 15, 2010 for DAMA Phoenix
- Material from this workshop was adapted from several sources:
 - ▶ IBM Rational's 2-day Disciplined Agile Delivery (DAD) workshop (RP250)
 - ▶ Agile Database Techniques, www.agiledata.org
 - ▶ Refactoring Databases, www.agiledata.org
 - ▶ Agile Modeling, www.agilemodeling.com
 - ▶ Enterprise Unified Process, www.enterpriseunifiedprocess.com
 - ▶ Several Dr. Dobb's Journal Surveys, www.ambysoft.com/surveys/



Agile For Data Professionals

Workshop Overview

- Comparing strategies
- Overview of agile fundamentals
 - ▶ The Agile Manifesto
 - ▶ Agile methodologies
 - ▶ Agile criteria
 - ▶ Disciplined agile development
 - ▶ The Agile Scaling Model
- Challenges with data-oriented activities
- Agile database techniques
 - ▶ Agile data modeling
 - ▶ Database testing
 - ▶ Test-driven database development
 - ▶ Database refactoring
- Agile strategies for data warehouses/BI
- Agile enterprise data
 - ▶ Agile enterprise architecture strategies
 - ▶ Agile data administration strategies
 - ▶ Agile data management
- Scaling strategies

3



Agile For Data Professionals

Logistics

Morning 9-12

2 10-minute breaks

Lunch 12-1

Afternoon 1-4

2 10-minute breaks



Agile For Data Professionals

Questions?

Feel free to ask questions at any time



Also, please turn off your phones

5



Agile For Data Professionals

Warning!

- I'm spectacularly blunt at times
- Many new ideas will be presented
- Some may not fit well into your existing environment
- Some will challenge your existing notions about software development
- Some will confirm your unvoiced suspicions
- Don't make any "career-ending moves"
- Be skeptical but open minded



6



Agile For Data Professionals

Getting to Know Each Other

- Quick poll:

- ▶ Who is an “agilist”?
- ▶ Who is a data person?
- ▶ Who works in an organization currently running agile projects?
- ▶ Who works in an organization thinking about agile?
- ▶ Who works in an organization where you’ve had a “questionable” experience with agile?



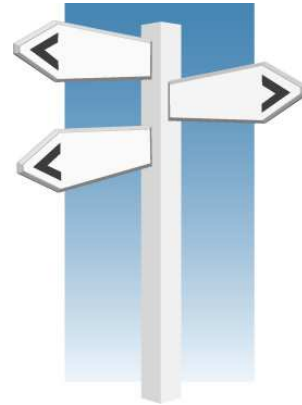
- Quick discussion:

- ▶ What personal objectives do you have for this workshop?

Agile For Data Professionals

Agenda

- ***Comparing strategies**
- Agile fundamentals
- Challenges with data activities
- Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- Scaling agile
- Parting thoughts



*=Current topic

8



Agile For Data Professionals

Organize Into Groups

- You must form groups of 4-6 people
- Requirements:
 - ▶ Have at least one person with a data background
 - ▶ Have at least one person with an agile background
 - ▶ Have at least one person with a development background
- If possible:
 - ▶ Work with people you have never met before
 - ▶ Work with people from different organizations
 - ▶ Have fun, but stay focused on the assignments

Agile For Data Professionals

Greeting Card Assignment: Assigning Roles

- **Four roles:**
 - ▶ Stakeholder (1)
 - ▶ Modeler (1)
 - ▶ Developer(1 or more)
 - ▶ Tester (1)
- **Stakeholder:**
 - ▶ Come to the front of the room for directions from the instructor
- **Everyone else:**
 - ▶ Get to know each other and read the instructions on the next slide

10



The stakeholder needs to come to the front of the room for instructions as quickly as possible.

Agile For Data Professionals

Greeting Card Assignment: Traditional

Follow these steps to create the greeting card:

1. Modeler interviews stakeholder (2 minutes), producing requirements document. Everyone else may listen but not speak.
2. Modeler hands over requirements document to development team (designer and developers). Modeler and Stakeholder now quiet.
3. Developers create card (2 minutes). Tester watches.
4. Developers provide card and original requirements document to the tester.
5. Tester inspects card and suggests improvements (30 seconds).
6. Developers “fix” card (30 seconds).
7. Developers provide card to the stakeholder.
8. Stakeholder rates the end product.

Model

Construct

Test

Fix

Rate

11



While you are “doing nothing”, observe what is working, what isn’t working, challenges that your team is having with this approach. Jot down pertinent observations.

When the card is rated, the stakeholder may share the requirements for the first card only.

Agile For Data Professionals

Greeting Card Assignment: Mini-Waterfalls

1. Organize your project into three “mini-waterfalls”:
 1. Modeler interviews stakeholder to understand requirements (30 seconds)
 2. Developers work on card (30 seconds)
 3. Tester inspects card (10 seconds)
 4. Developers update card (10 seconds)
 5. Developers present card to stakeholder (10 seconds) who indicates what they like and dislike
2. Stakeholder rates the end product

Model

Construct

Test

Fix

Model

Construct

Test

Fix

Model

Construct

Test

Fix

Rate



12

While you are “doing nothing”, observe what is working, what isn’t working, challenges that your team is having with this approach. Jot down pertinent observations.

When the card is rated, the stakeholder may share the requirements for this new card.

Agile For Data Professionals

Greeting Card Assignment: Agile

1. Organize the project into 3 one-minute iterations:
 1. Stakeholder is an active participant in the development effort. They can describe their requirements at any time, provide feedback at any time, ...
 2. Develop the card (45 seconds)
 3. Retrospective: How can you work together more effectively? (15 seconds)
2. Stakeholder rates the end product

Model
Construct
Test

Improve

Model
Construct
Test

Improve

Model
Construct
Test

Rate

13



While you are “doing nothing”, observe what is working, what isn’t working, challenges that your team is having with this approach. Jot down pertinent observations.

When the card is rated, the stakeholder may share the requirements for this card.

Agile For Data Professionals

Discussion

- **Instructions:**

- ▶ Get back together in your groups
- ▶ Discuss for 5 minutes
- ▶ Be prepared to present one or two important points with the rest of the class

- **Potential discussion points:**

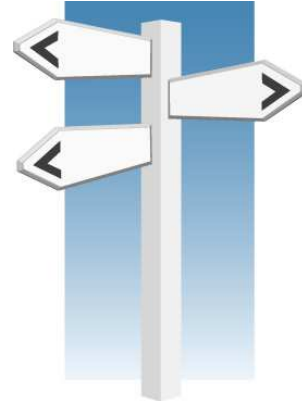
- ▶ What strategy worked best for you? Why?
- ▶ What are the risks with each approach?
- ▶ What are the trade-offs with each approach?
- ▶ Was there overhead between hand-offs? Why?
- ▶ What can you apply to “the real world”?



Agile For Data Professionals

Agenda

- Comparing strategies
- *Agile fundamentals
- Challenges with data activities
- Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- Scaling agile
- Parting thoughts



*=Current topic



Agile For Data Professionals

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

16



Focus on collaboration:

- Less paperwork and more conversation
- Stakeholders actively involved

Focus on quality:

- Have a full regression test suite for your systems
- Develop loosely-coupled, highly cohesive architectures
- Refactor to keep them this way

Focus on working solutions:

- Greater feedback makes agile projects easier to manage
- Less documentation is required
- Less bureaucracy

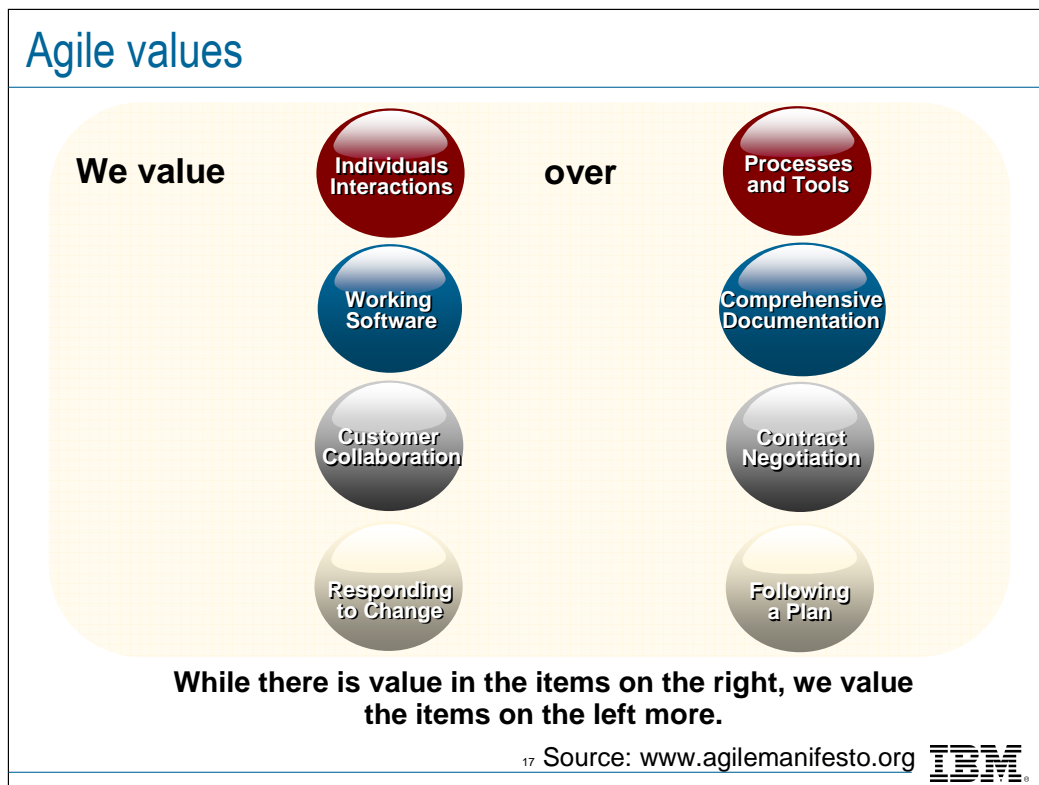
Agilists are generalizing specialists:

- Fewer hand offs between people
- Fewer people required
- Specialists find it difficult at first to fit into the team

Agile is based on practice, not theory:

- This is a significant change from traditional
- You need to see how agile works in practice to truly understand it

Agile For Data Professionals



Many people point to the values of the agile manifesto as a definition for agile development. The values are a great set of philosophies, but likely do not provide a good definition of agile development.

Value #1: Individuals and interactions over processes and tools. Teams of people build software systems, and to do that they need to work together effectively. Teams include, but are not limited to, programmers, testers, project managers, modelers, and customers.

Value #2: Working software over comprehensive documentation. Documentation has its place. Written properly, it is a valuable guide to help people understand how and why a system is built, and how to work with the system. However, never forget that the primary goal of software development is to create software, not documents; otherwise, it would be called documentation development wouldn't it?

Value #3: Customer collaboration over contract negotiation. Only your customer can tell you what they want. Having a contract with your customers is important. Having an understanding of everyone's rights and responsibilities may form the foundation of that contract, but a contract isn't a substitute for communication.

Value #4: Responding to change over following a plan. Change is a reality of software development, a reality that your software process must reflect. There is nothing wrong with having a project plan; it would be risky to not have one. However, a project plan must be malleable. There must be room to change it as your situation changes. Otherwise, your plan will quickly become irrelevant.

Agile For Data Professionals

Agile principles

1. Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
 2. Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
 3. Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
 4. Business people and developers must work together daily throughout the project.
 5. Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
 6. The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- (Continued...)**

Source: www.agilemanifesto.org/principles.html



Agile For Data Professionals

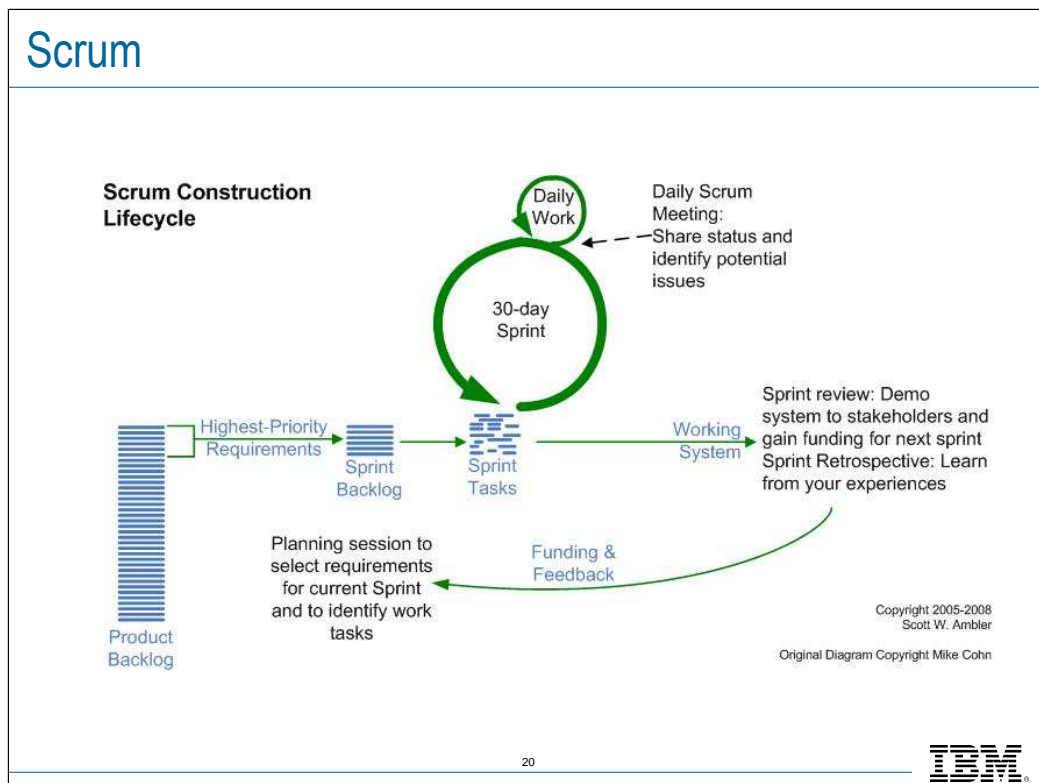
Agile principles (continued)

7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.
9. Continuous attention to technical excellence and good design enhances agility.
10. Simplicity – the art of maximizing the amount of work not done – is essential.
11. The best architectures, requirements, and designs emerge from self-organizing teams.
12. At regular intervals, the team reflects on how to become more effective, then tunes and adjusts its behavior accordingly.

Source: www.agilemanifesto.org/principles.html



Agile For Data Professionals



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

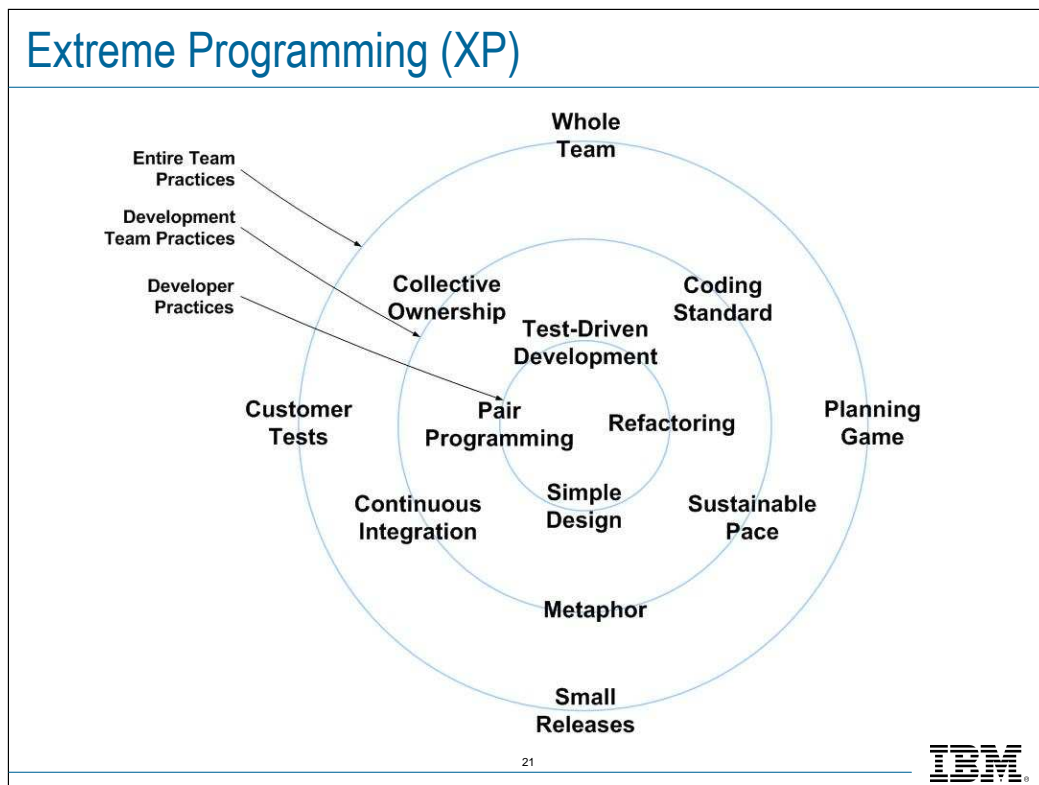
Scrum practices:

- **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

Agile For Data Professionals



Coding standard: Team members should follow the same development conventions.

Collective ownership: Everyone is responsible for all the project artifacts, including code; this, in turn, means that everybody is allowed to change any part of an artifact.

Continuous integration: Integrate and test changes at least once every few hours, preferably more often. This approach enables development teams to find defects early, thereby reducing the average cost of addressing the defects, it also helps them to deliver higher quality code and to move forward safely when adding or changing functionality.

Customer tests: Capture requirements in the form of executable tests (executable specifications).

Metaphor: The metaphor is a simple evocative description of how the system works.

Pair programming: Two programmers working together at one work station, will add as much functionality as two working separately except that it will be much higher in quality.

Planning game: The purpose of the Planning Game is to guide the product into delivery. Instead of predicting the exact dates of when deliverables will be needed and produced, which is difficult to do, it aims to "steer the project" into delivery.

Refactoring: A refactoring is a simple change to your code which improves the quality of the design without changing the semantics.

Simple design: XP teams build software to a simple design. They start simple and they keep it that way.

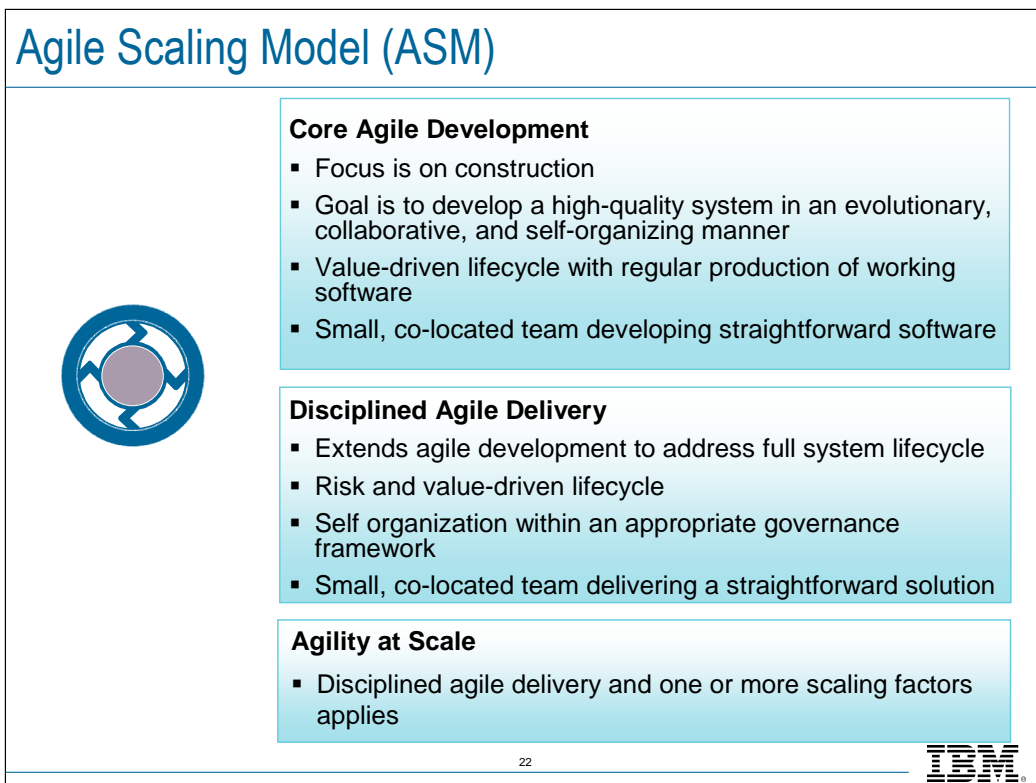
Small releases: The team releases running, tested software, delivering business value chosen by the customer, every iteration.

Sustainable pace: Work only as many hours as you can to be productive in a sustainable manner.

Test driven development (TDD): Write a single test, either at the requirements or design level, and then just enough code to fulfill that test. TDD is a JIT approach to detailed requirements specification and a confirmatory approach to testing.

Whole team: Whole team, together have the skills to successfully communicate the essentials of a problem and deliver an agreed-upon solution.

Agile For Data Professionals



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 Core 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.

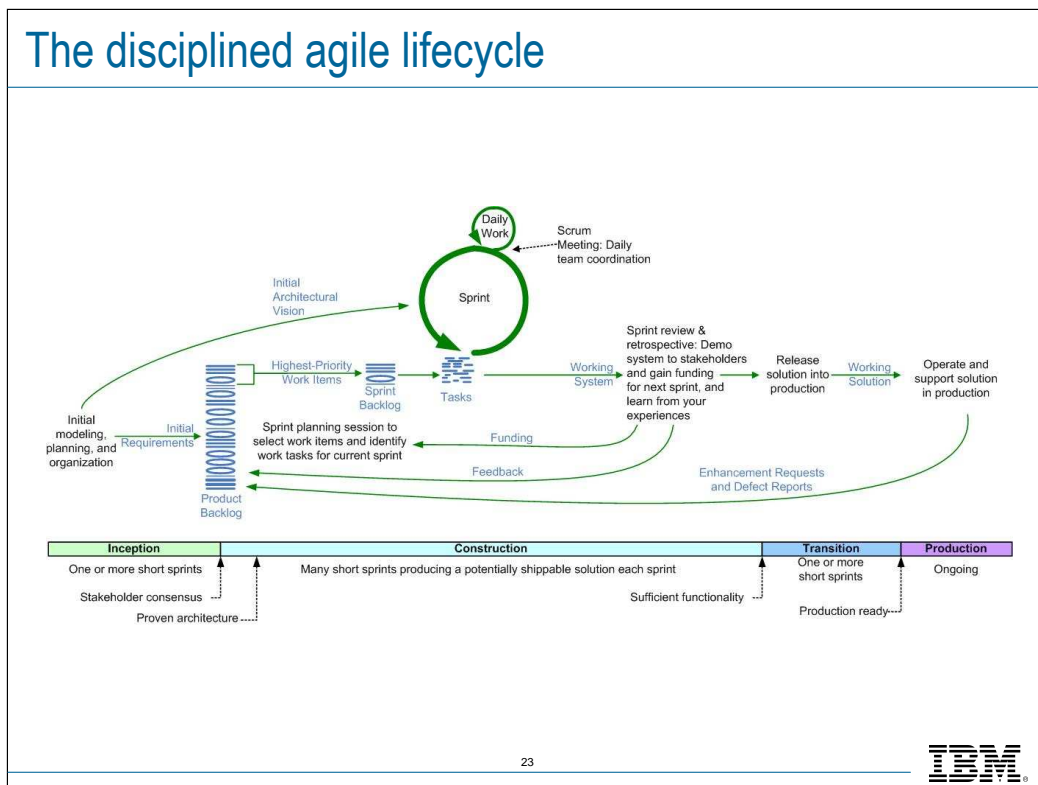
Disciplined agile delivery addresses the full delivery lifecycle from project initiation to production. It adds appropriate, lean governance to balance self organization. It adds a risk-driven viewpoint to the value-driven approach in order to increase the chance of project success. The focus is on small (<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.

Agile For Data Professionals

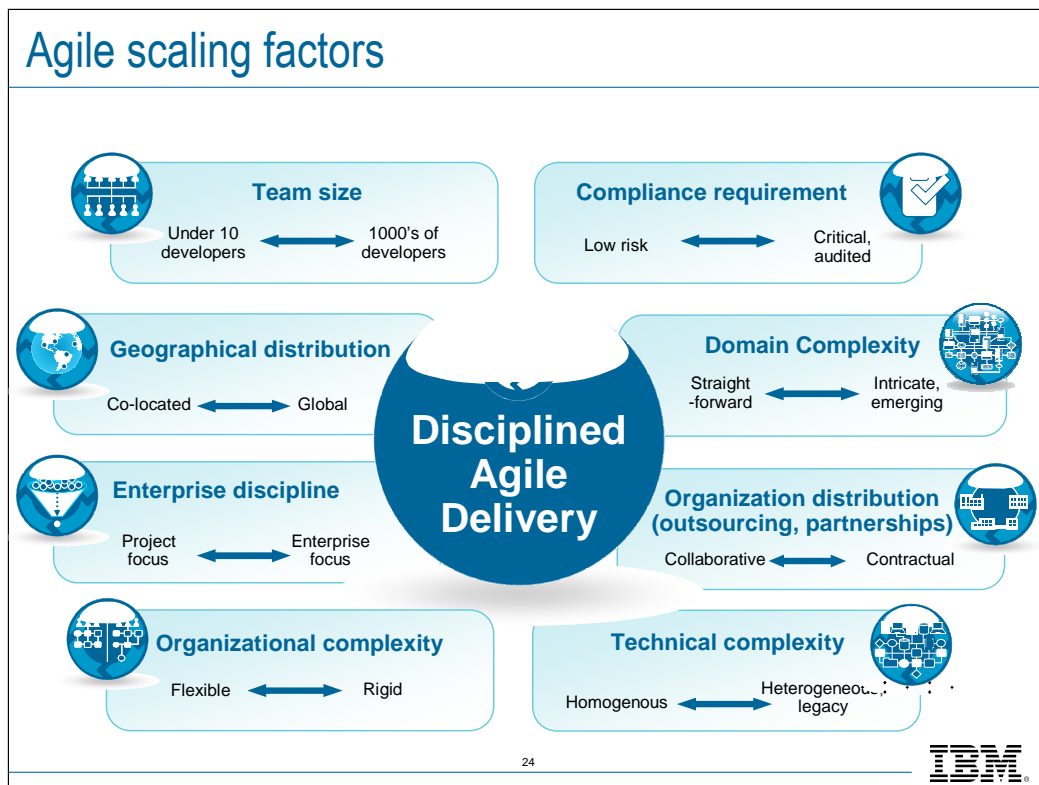


The disciplined agile delivery lifecycle expands upon the Scrum construction lifecycle in three important ways:

1. It has explicit project phases, recognizing that agile delivery is really iterative in the small and serial in the large.
2. 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.
3. 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.

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

Agile For Data Professionals



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 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.

Agile For Data Professionals

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 work within an appropriate governance framework.
5. Regularly reflect on, and measure, how they work together and then act to improve on their findings in a timely manner.



25



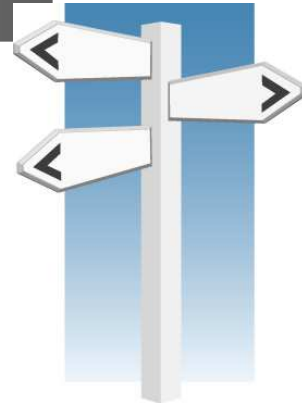
Criteria to determine if a project team is taking a disciplined approach to agile

See www.ibm.com/developerworks/blogs/page/ambler

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- ***Challenges with data activities**
- Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- Scaling agile
- Parting thoughts



*=Current topic



26

Agile For Data Professionals

Discussion

- Instructions:
 - ▶ Get back together in your groups
 - ▶ Discuss for 10 minutes
 - ▶ Be prepared to present one or two interesting points
- Potential discussion points:
 - ▶ What is the quality of your production data?
 - ▶ What does your organization do to prevent data quality problems? To fix them once found?
 - ▶ How well are these strategies working in practice?
 - ▶ What “data goals” need to be fulfilled on an application development project?
 - ▶ What activities need to occur to fulfill these goals?
 - ▶ How do data professionals prefer to approach data activities?
 - ▶ How do agile developers prefer to approach data activities?



Agile For Data Professionals

Harsh Observation

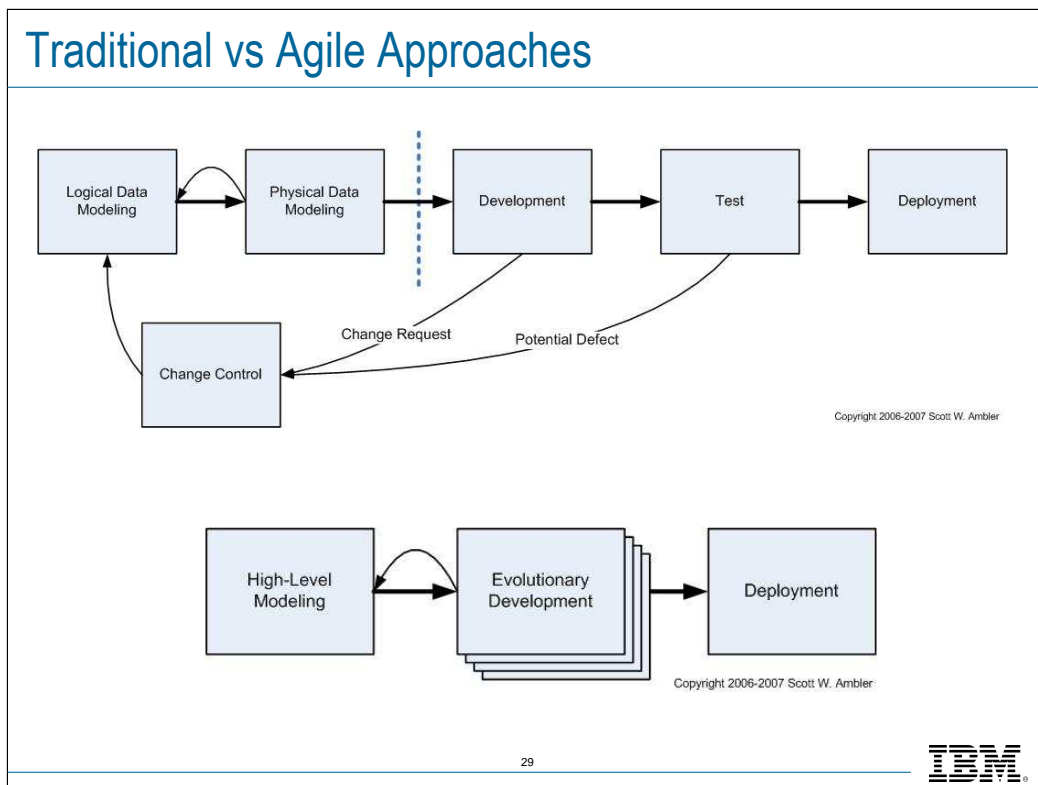
You get the data quality that you deserve

28



TDWI estimates that data quality problems are a \$600 billion (yes, \$600,000,000,000) a year problem for US organizations.

Agile For Data Professionals



There is a disconnect between agile strategies and traditional strategies.

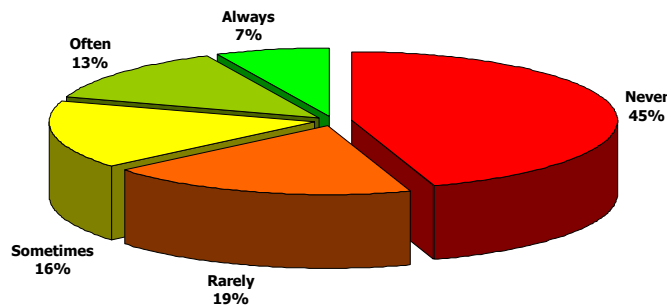
Diagrams used with permission.

See <http://www.ambysoft.com/essays/agileLifecycle.html>

Agile For Data Professionals

Questioning Big Requirements Up Front (BRUF)

Feature Usage Within Deployed Applications



- Detailed requirements specs increase project risk
- Traditional “change mgmt” approaches are little more than “change prevention” strategies
- Stakeholders want software which fulfills their needs, not software that fulfills a spec
- The real goal is to understand and then implement requirements, not document them

Source: Chaos Report v3, Standish Group

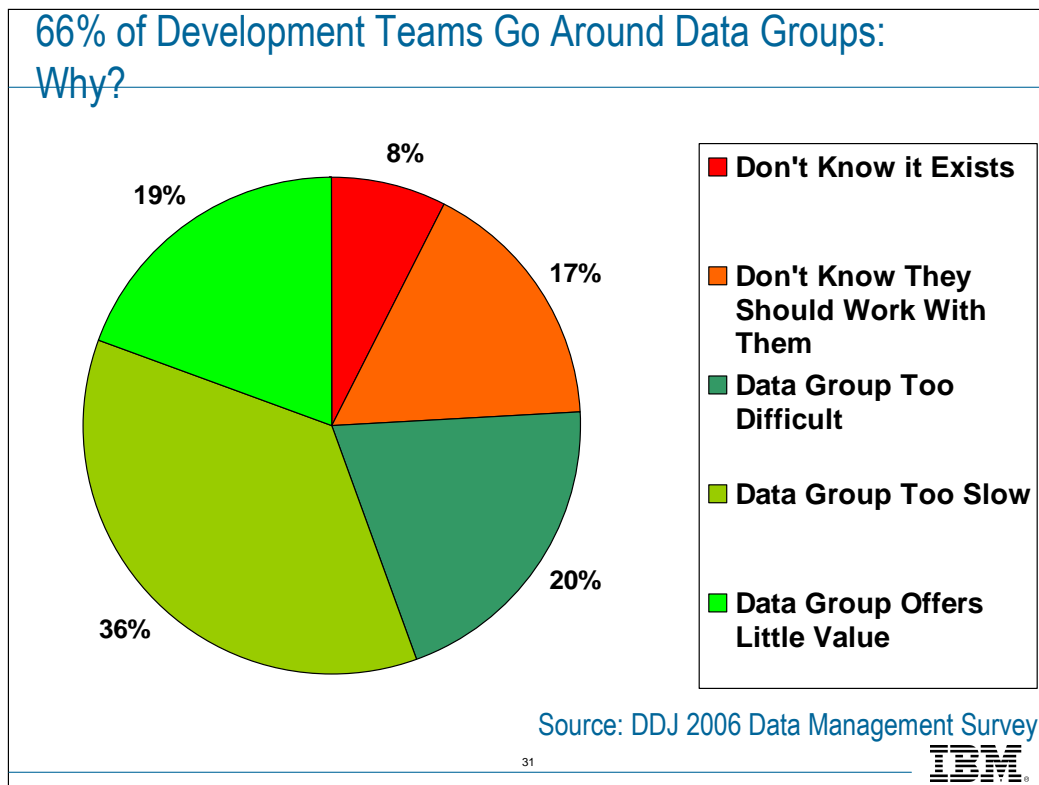
30



The Chaos Report shares some interesting statistics pertaining to the effectiveness this sort of approach. The Chaos Report, looks at thousands of projects, big and small, around the world in various business domains. The study reports that 15% of all projects fail to deliver at all, and that 51% are challenged (they’re severely late and/or over budget). However, the Standish Group has also looked at a subset of traditional teams which eventually delivered into production and asked the question, “Of the functionality which was delivered, how much of it was actually used?” An astounding 45% of the functionality was never used, and a further 19% is rarely used. In other words, on these so-called “successful” projects there is significant wastage.

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

Agile For Data Professionals

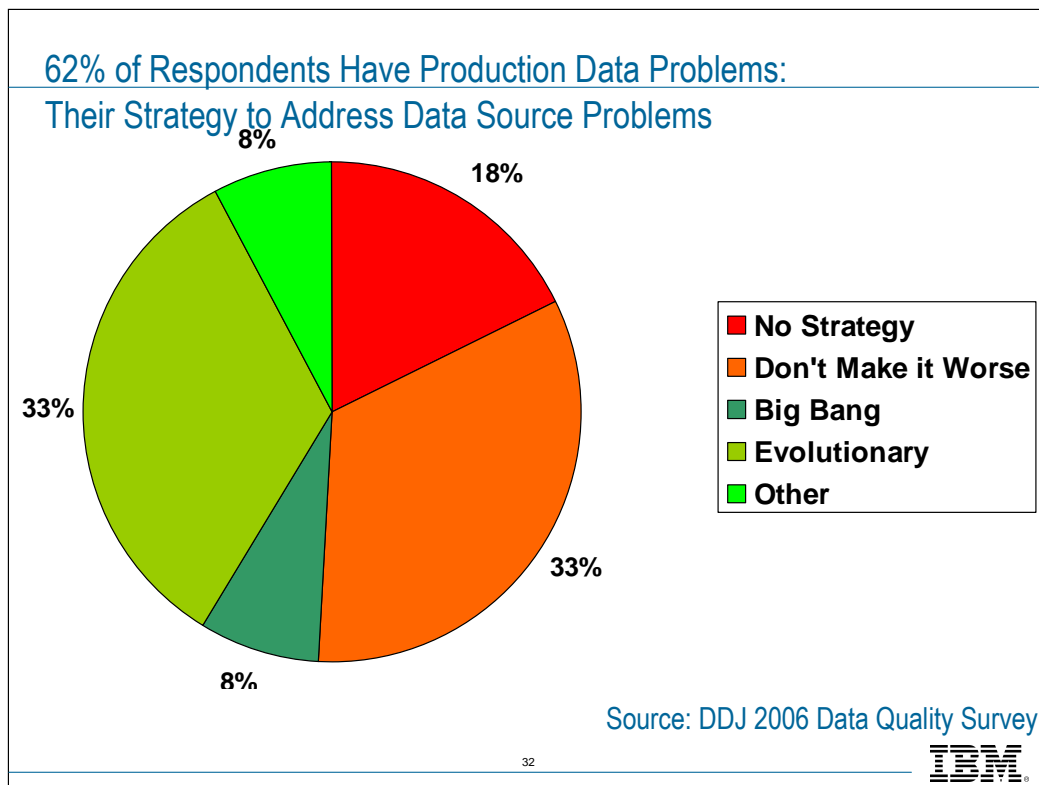


Source: <http://www.ambysoft.com/surveys/dataManagementAugust2006.html>

Of the 1,176 respondents 618 indicated that they were developers, 188 IT management, and 98 data professionals. 66% of respondents indicated that development teams sometimes go around their data management (DM) groups. Of those, 20% found that their DM group was too difficult to work with, 36% felt the DM group was too slow to respond, and 19% felt the DM group offered too little value. I believe that this is a clear indication that a cultural impedance mismatch (<http://www.agiledata.org/essays/culturalImpedanceMismatch.html>) exists between developers and data professionals. Symptoms of the cultural impedance mismatch include:

- Application developers that claim relational technology either shouldn't or can't be used to store objects
- Data professionals that claim that your object/component models must be driven by their data models
- Application developers that claim that because they're using a persistence framework they don't need to understand anything about the underlying data technology
- Data professionals that disparage agile software development approaches, yet when pressed know very little about agile
- Application developers complain about the "useless data bureaucracy" without understanding why the data activities have been adopted
- Data professionals complain about the data messes created by application developers (yet they rarely seem to want to train developers to do the job right)

Agile For Data Professionals

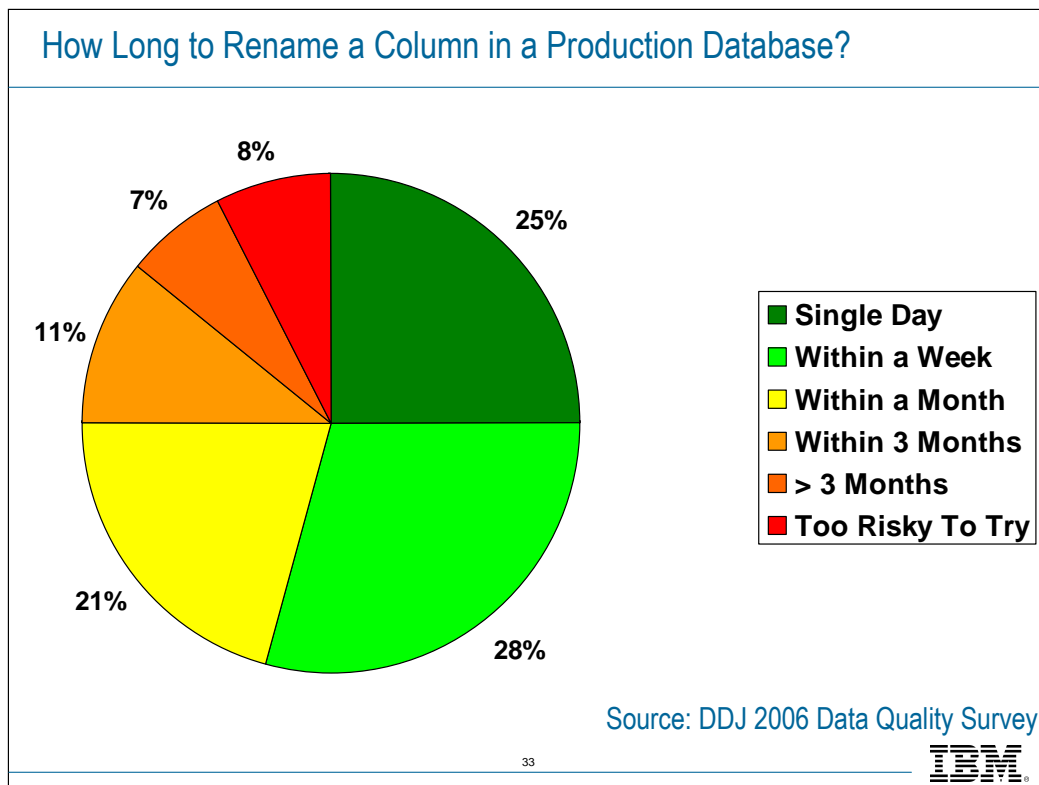


Source: <http://www.ambysoft.com/surveys/dataQualitySeptember2006.html>

Hoping that things won't get worse isn't much different than having no strategy at all.

Big bang = Rewrite all apps, rework all data sources, release all at once.

Agile For Data Professionals



Source: <http://www.ambysoft.com/surveys/dataQualitySeptember2006.html>

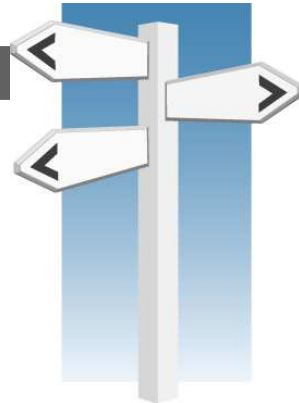
If you can't easily do something such as rename a column, how can you possibly support more complex changes which would likely provide greater value?

Some of the time required to make the changes in production include the time to go through your organization's release process. However, most organizations have the ability to deploy fixes very quickly.

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- Challenges with data activities
- *Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- Scaling agile
- Parting thoughts



*=Current topic



34

Agile For Data Professionals

Agile Data Modeling

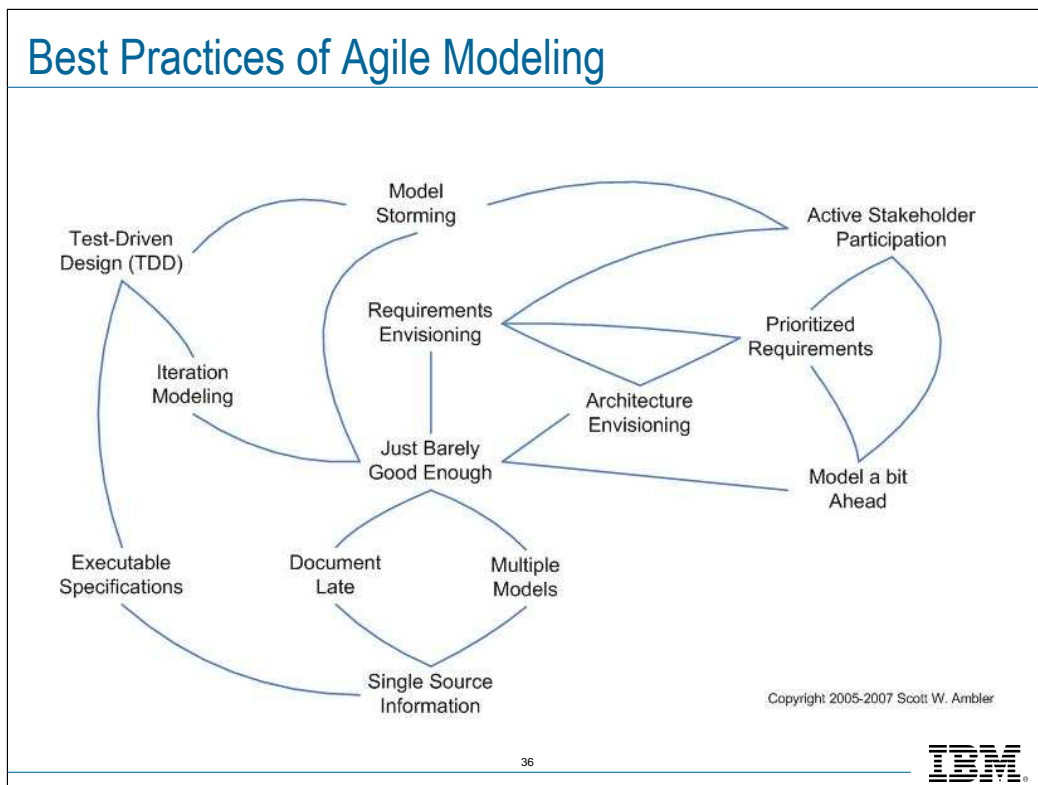
- Data modeling is the act of exploring data-oriented structures.
- Evolutionary data modeling is data modeling performed in an iterative and incremental manner.
- Agile data modeling is evolutionary data modeling done in a collaborative manner.
- Data modeling is only one of many aspects of modeling, see www.agilemodeling.com/artifacts
- See Agile Model Driven Development (AMDD) www.agilemodeling.com/essays/amdd.htm

35



See <http://www.agiledata.org/essays/agileDataModeling.html>

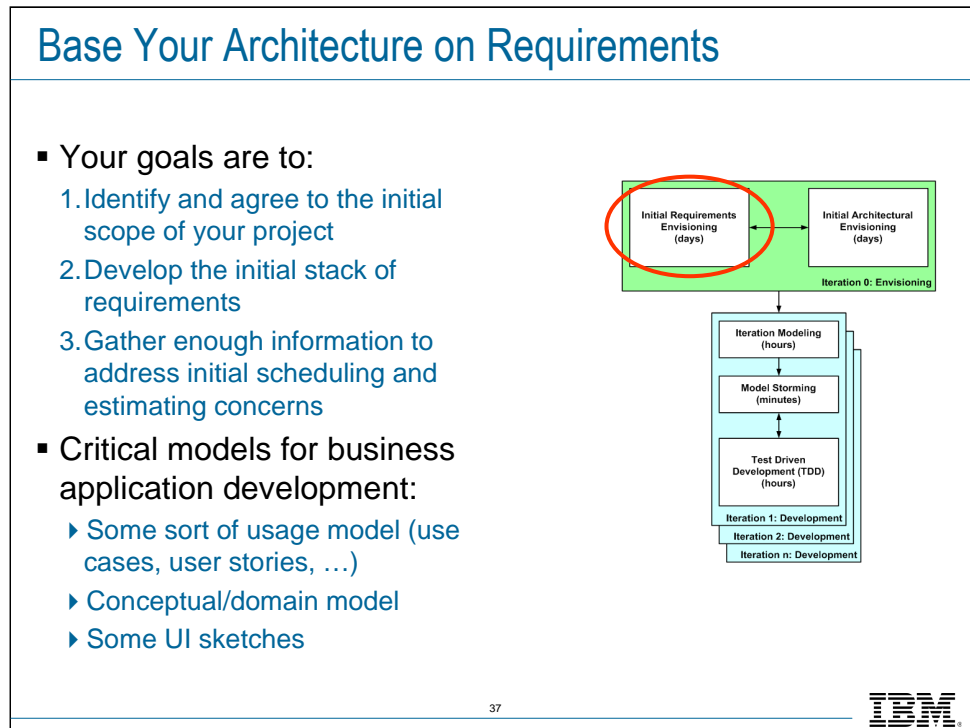
Agile For Data Professionals



www.agilemodeling.com/essays/bestPractices.htm

This is a “practices map”, showing several practices and relationships between them.

Agile For Data Professionals



The Agile Model Driven Development (AMDD) lifecycle is described at:
<http://www.agilemodeling.com/essays/amdd.htm>

Requirements envisioning is described at:
<http://www.agilemodeling.com/essays/initialRequirementsModeling.htm>

Diagrams used with permission

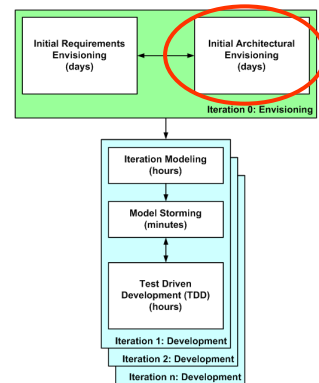
Agile principle # 2: Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.

Agile principle # 11: The best architectures, requirements, and designs emerge from self-organizing teams.

Agile For Data Professionals

Initial Architecture Envisioning

- Your goals are to:
 1. Identify and agree to a potential initial architecture of your system
 2. Provide sufficient technical vision for estimating and scheduling concerns
- Critical models for business application development:
 - ▶ Some form of deployment diagram
 - ▶ A free-form “technology stack” diagram
 - ▶ A UI flow diagram



38



Keep the modeling as light as possible, the details will come out on a just in time basis later in the project.

The fundamental goal is to come up with a viable technical strategy, not to produce a detailed technical specification.

<http://www.agilemodeling.com/essays/initialArchitectureModeling.htm>

Diagram used with permission.

Agile principle # 10: Simplicity – the art of maximizing the amount of work not done – is essential.

Agile principle # 11: The best architectures, requirements, and designs emerge from self-organizing teams.

Agile For Data Professionals

Database Refactoring

- A database refactoring is a simple change to a database schema that improves its design while retaining both its *behavioral and informational semantics*.
- A database schema includes structural aspects such as table and view definitions; functional aspects such as stored procedures and triggers; and informational aspects such as the data itself

Customer
CustomerID <<PK>>
Fname

Original Schema

Customer
CustomerID <<PK>>
Fname { drop date = November 14 2009 }
FirstName
SynchronizeFirstName { event = update insert, drop date = November 14 2009 }

Transition Period

Customer
CustomerID <<PK>>
FirstName

Resulting Schema

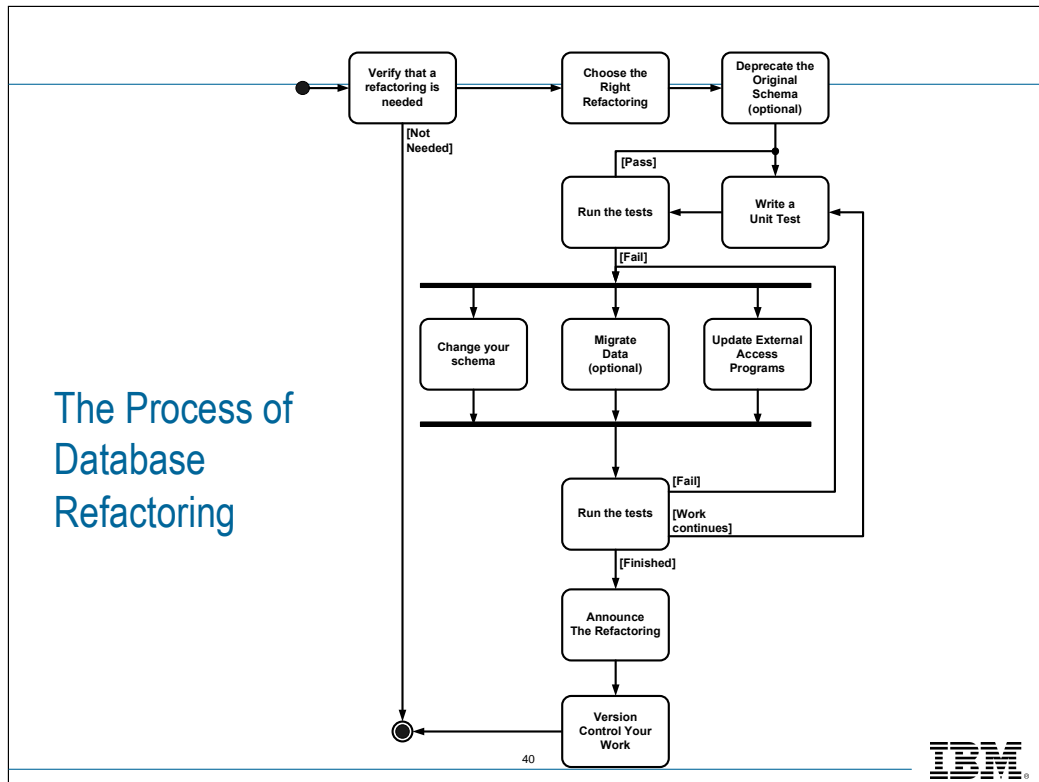
39



Source: www.agiledata.org/essays/databaseRefactoring.html

Agile principle #9: Continuous attention to technical excellence and good design enhances agility.

Agile For Data Professionals

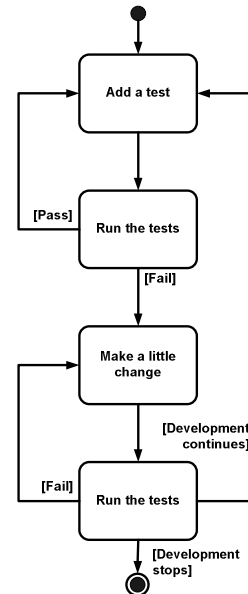


Source: www.agiledata.org/essays/databaseRefactoring.html

Agile For Data Professionals

Test-First Design

- Automated developer testing drives the detailed design of the system
- Tests form the majority of your detailed design specifications
- You'll still need some high-level models to provide overviews
- This is the first time in IT history that developers are motivated to keep the specifications up-to-date
- An example of single sourcing information
- TDD = TFD + Refactoring



41



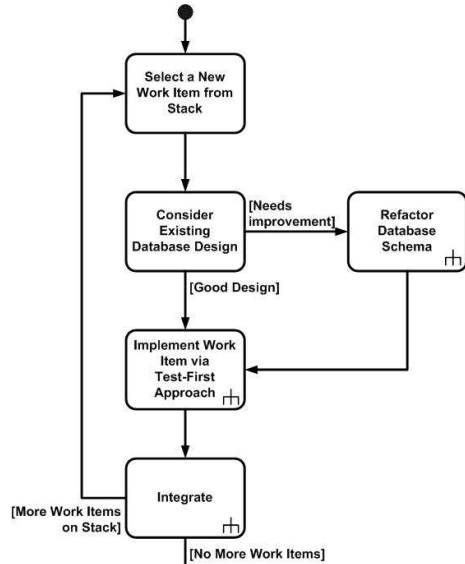
Source: www.agiledata.org/essays/tdd.html

Agile principle #9: Continuous attention to technical excellence and good design enhances agility.

Agile For Data Professionals

Test/Behavior Driven Database Development (TDDD/BDDD)

- An evolutionary approach to database design
- Driven by tests, not models
- A practical approach to ensuring database quality
- Completely different than traditional approaches to database design
- Apparently very threatening to traditional data professionals
- One part of BDD, not a stand-alone activity
- May/June 2007 issue of IEEE Software



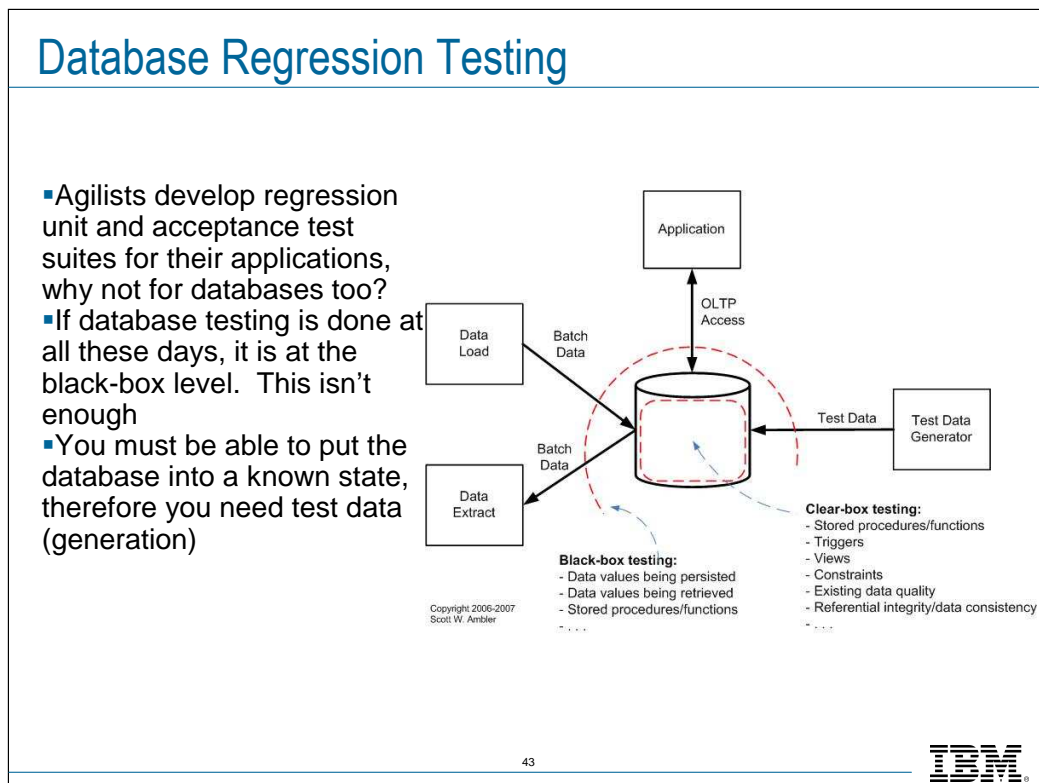
Copyright 2006-2007 Scott W. Ambler



Agile principle #9: Continuous attention to technical excellence and good design enhances agility.

Observation: There is nothing special about databases. They can and should be developed via a quality-focused, evolutionary approach.

Agile For Data Professionals



Source: www.agiledata.org/essays/databaseTesting.html

Database regression testing is just part of your overall regression testing efforts.

Here's a few interesting questions to ask someone who isn't convinced that you need to test the DB:

1. If you're implementing code in the DB in the form of stored procedures, triggers, ... shouldn't you test that code to the same level that you test your app code?
2. Think of all the data quality problems you've run into over the years. Wouldn't it have been nice if someone had originally tested and discovered those problems before you did?
3. Wouldn't it be nice to have a test suite to run so that you could determine how (and if) the DB actually works?

Agile principle #9: Continuous attention to technical excellence and good design enhances agility.

Agile For Data Professionals

Current State of Database Testing (Data Quality Survey)

- 95.7% of organizations considered data to be a corporate asset
 - ▶ Of those, only 40.3% had a regression test suite in place
 - Of those, Only 63.3% allowed developers to run this test suite when they needed to
 - ▶ Of the organizations that didn't have a test suite, only 31.6% had discussed putting one in place
- 63.7% of organizations implemented mission-critical functionality in the database
 - ▶ Of those, only 46% had a regression test suite in place
 - Of those, Only 66.3% allowed developers to run this test suite when they needed to
 - ▶ Of the organizations that didn't have a test suite, only 38.6% had discussed putting one in place



Source: DDJ 2006 Data Quality Survey



44

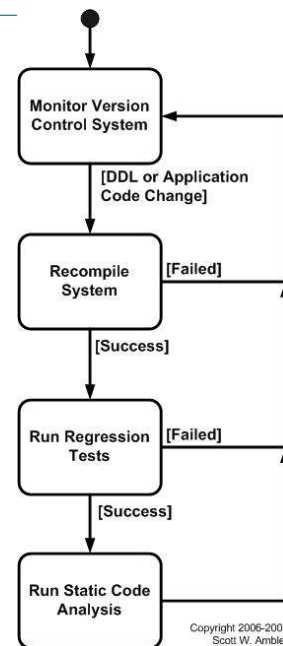
<http://www.ambyssoft.com/surveys/dataQualitySeptember2006.html>

See Whence Data Quality? at <http://www.drdoobbs.com/database/196900212>

Agile For Data Professionals

Continuous Database Integration

- Whenever new code is checked in you should rebuild your system
- Part of building the system is building the database (if it changed)
 - You may even want to rebuild the database from scratch
- You want to run your regression test suite to determine if changes have injected defects
- Static code analysis is becoming common for application code. We could do the same for database code.



Copyright 2006-2007
Scott W. Ambler



45

Database integration is really part of your overall continuous integration efforts, although might only be done within your nightly CI efforts due to performance issues

Agile For Data Professionals

Supporting Practices

- Naming conventions and data standards
- Configuration management of all project artifacts
- Encapsulate database access

46



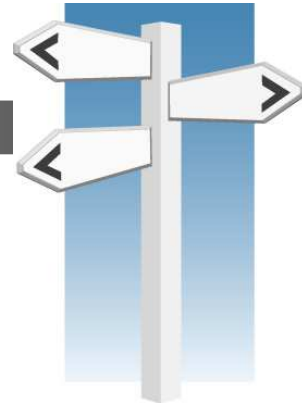
Conventions:

- See <http://www.ambysoft.com/surveys/stateOfITUnion200907.html>
- 59% of respondents indicated that their organization has enterprise-wide coding conventions
- 47% have enterprise-wide UI/usability conventions
- 51% have enterprise-wide data conventions.
- But, as you see in the survey, just because the conventions exist doesn't mean that people are actually following them (similar trends apply for UI and data conventions,

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- Challenges with data activities
- Agile database techniques
- ***Agile strategies for DW/BI**
- Agile enterprise data
- Scaling agile
- Parting thoughts



*=Current topic



Agile For Data Professionals

Discussion

- **Instructions:**
 - Get back together in your groups
 - Discuss for 10 minutes
 - Be prepared to present one or two interesting points
- **Potential discussion points:**
 - What activities need to occur on a DW/BI project?
 - What's the difference between DW and BI projects?
 - What are the primary risks?
 - Do the “agile database” techniques change the game?
 - What types of artifacts are important?
 - What would an agile DW/BI project timeline look like?



Agile For Data Professionals

Harsh Observation

Rhetoric aside, there should be nothing special about data warehouse/business intelligence projects compared with other types of software development projects

49



The DDJ 2007 Project Success survey found that DW/BI projects had the lowest success rates amongst all project types explored. See www.ambyssoft.com/surveys/

Agile For Data Professionals

Agile DW/BI Suggestions

- Adopt short iterations
 - ▶ Two to four weeks
 - ▶ Deliver new data sources or reports each iteration
- Take a usage-driven approach
 - ▶ Data-driven approaches are too narrow
 - ▶ How people use, or work with, the data to fulfill business goals is the true issue → focus on usage
 - ▶ Work closely with your stakeholders to explore their intent
- Strive to fix the data at the source
 - ▶ Fixing the data as part of your Extract-Transform-Load (ETL) efforts should be seen as a band-aid, not a solution
- Accept data inconsistencies
 - ▶ Perfection is the enemy of good enough

50

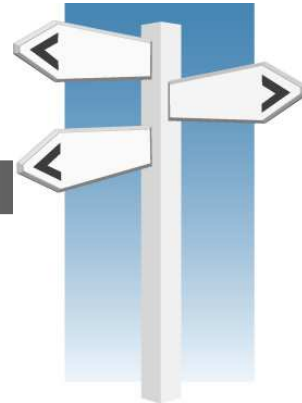


<http://www.agiledata.org/essays/dataWarehousingBestPractices.html>

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- Challenges with data activities
- Agile database techniques
- Agile strategies for DW/BI
- ***Agile enterprise data**
- Scaling agile
- Parting thoughts

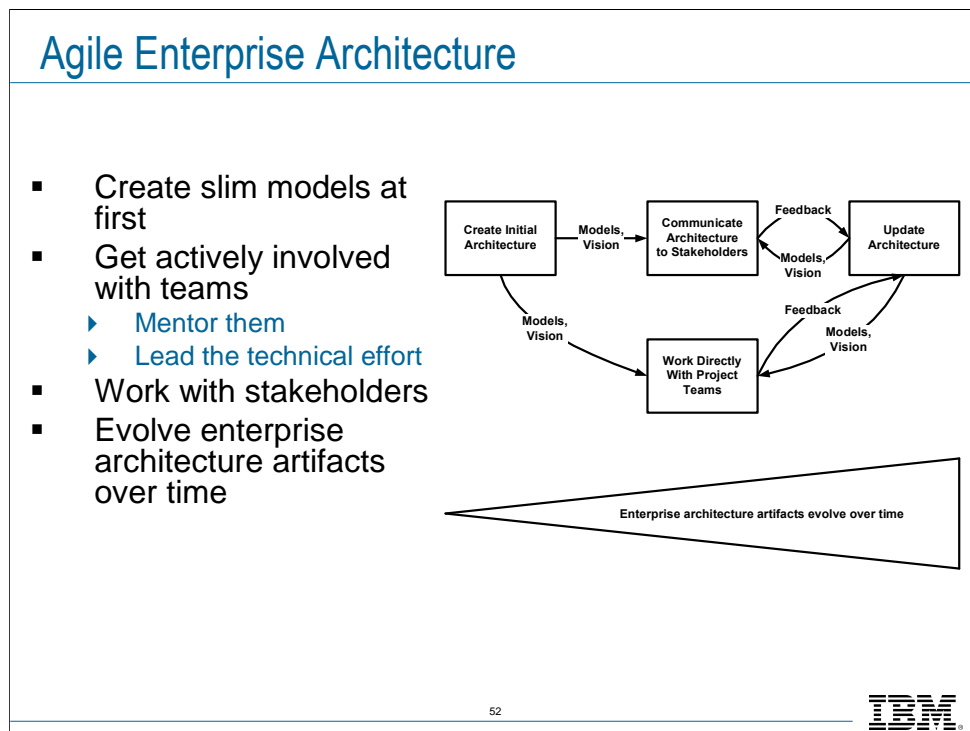


*=Current topic



51

Agile For Data Professionals



The Agile Data Method's third philosophy is "Enterprise groups exist to nurture enterprise assets and to support other groups, such as development teams, within your organization. These enterprise groups should act in an agile manner that reflects the expectations of their customers and the ways in which their customers work." Let's explore what that actually means.

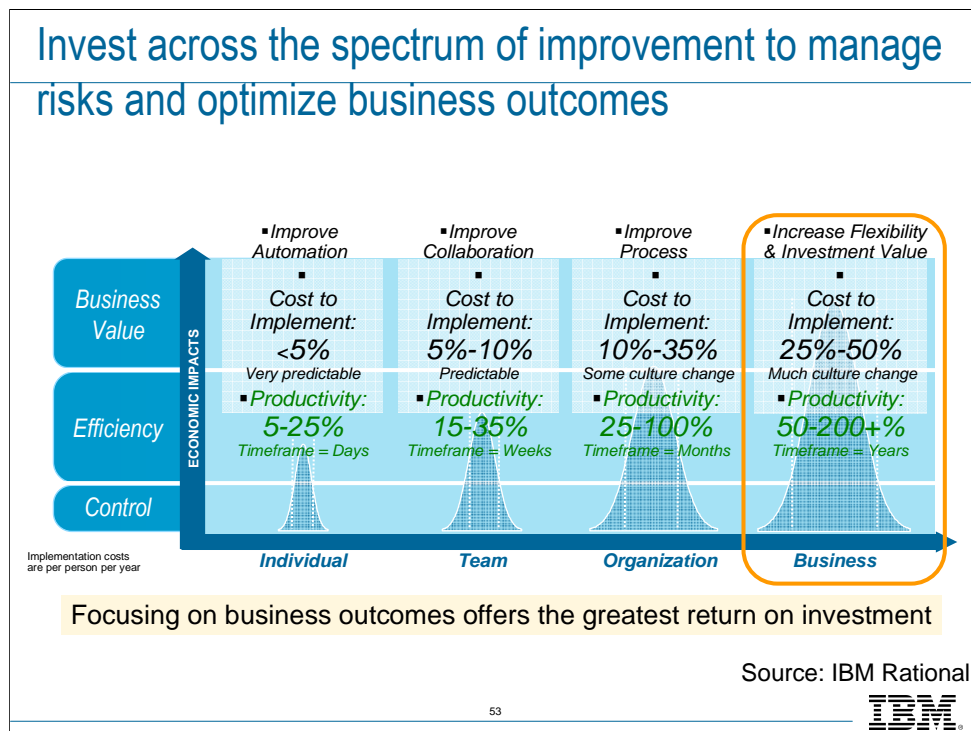
To be successful at enterprise architecture you need to rethink your overall approach and address five fundamental issues. These issues are connected in a synergistic manner; you must address all of them otherwise you will put your effort at risk. These issues are:

- Focus on people, not technology or techniques
- Keep it simple
- Work iteratively and incrementally
- Roll up your sleeves
- Look at the whole picture
- Make enterprise architecture attractive to your customers (business stakeholders and development teams)

Diagram used with permission

Source: <http://www.agiledata.org/essays/enterpriseArchitecture.html>

Agile For Data Professionals



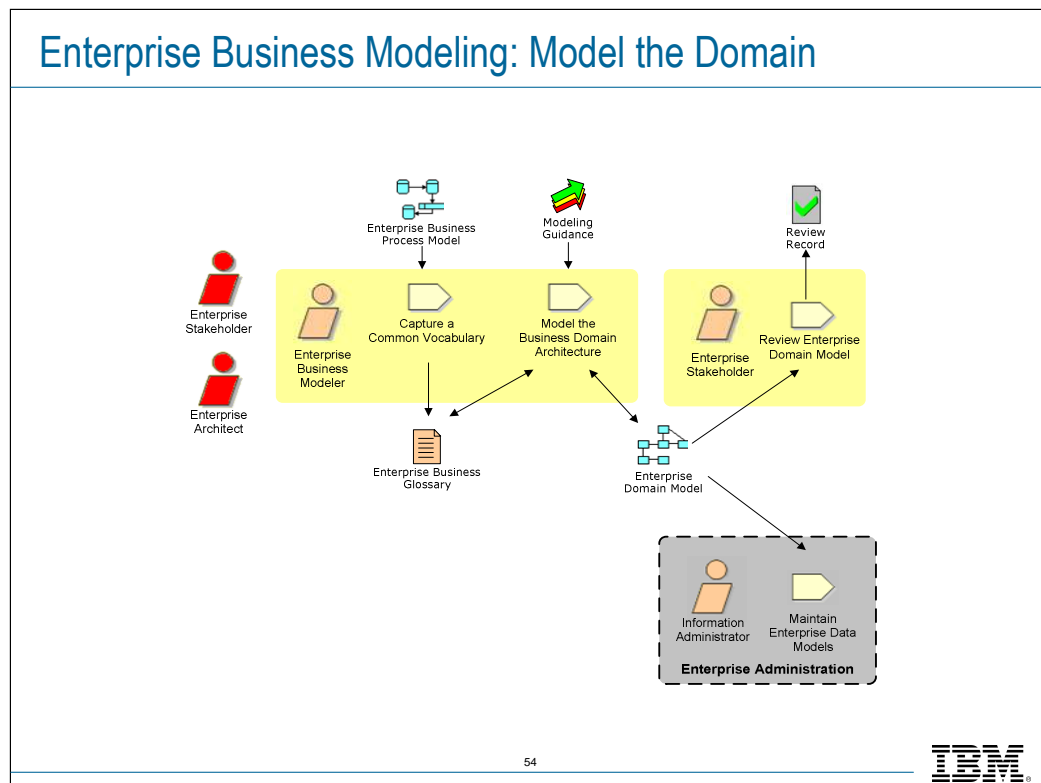
EA generally requires a change in your approach to the business.

These are long term changes, but as we saw previously the leading cause of EA program failure is not giving it enough time to succeed.

Implication is that you need to get some early and mid-term wins, not just long-term ones

So, EA might not be the only thing you should be working on (consider getting better at application architecture as well, for example)

Agile For Data Professionals



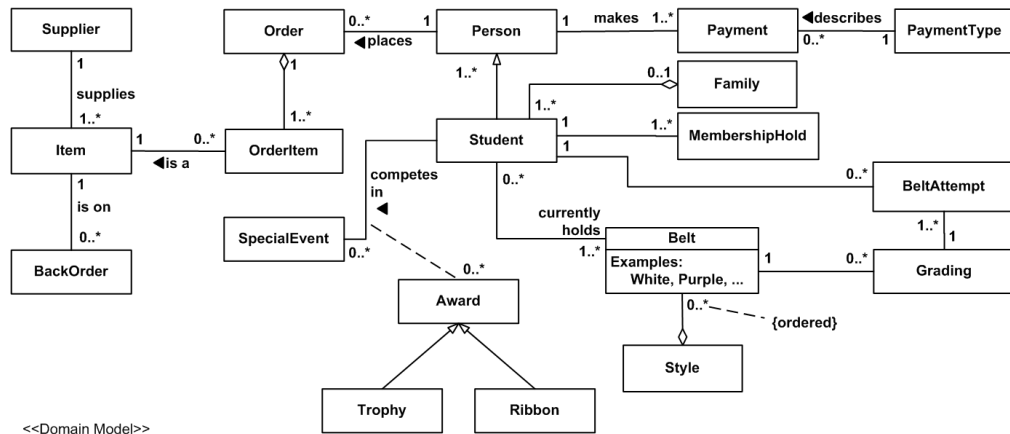
An important part of enterprise business modeling is the creation of a high-level domain/conceptual model that depicts the main business entities and their relationships that are of interest to your organization. This model, which does not need to be very detailed, is valuable input into the enterprise data modeling efforts of your information administration group as well as the requirements and business modeling efforts of project teams. In both cases the enterprise domain model provides a basis from which to begin more detailed modeling efforts: the project teams won't have to reinvent your domain model each time. In parallel you will also create an enterprise business glossary, which defines a common vocabulary within your organization. This enterprise business glossary is a valuable resource that should be shared with all of your project teams. It doesn't make sense for individual project teams to define common business terms such as "customer" over and over again because this can lead to wrong assumptions and incorrect data. Instead, teams should reference the commonly accepted enterprise business glossary and then focus on the terminology that is unique to their effort.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseBusinessModeling.html>

Agile For Data Professionals

But Don't Go Crazy – You Don't Need Details



<<Domain Model>>
Copyright 2004 Scott W. Ambler

55

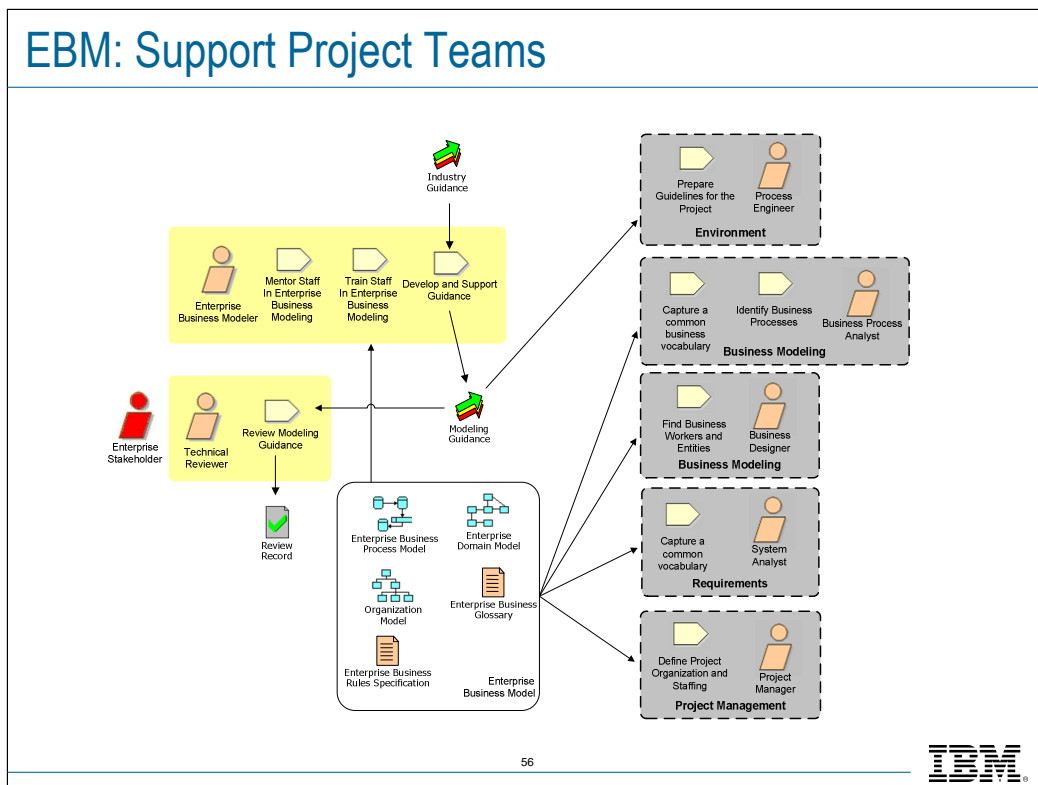


This model should be very slim, capturing the main business entities and the relationships between them. The only supporting documentation which I would create for this model would be a definition of the entities, information I'd be tempted to capture in a [glossary](#).

Diagram used with permission

Source: <http://www.agiledata.org/essays/agileDataModeling.html>

Agile For Data Professionals

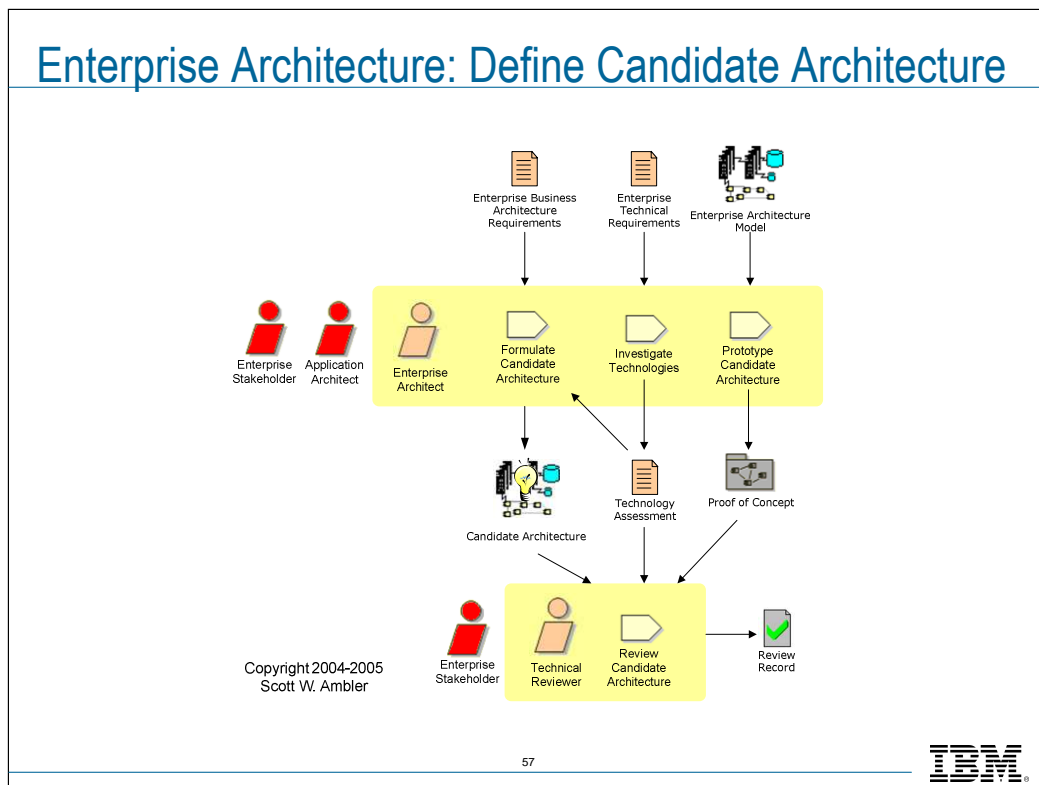


It isn't sufficient to simply create enterprise business models and give them to your project teams because they probably will simply ignore the models. I've worked on development projects in several organizations where I discovered that many times an enterprise business model is available but that the teams never think to take advantage of them, if they even know they exist. Successful enterprise business modelers communicate their work to development teams and are willing to support the teams in taking advantage of the enterprise business models. The strategies of the agile community, particularly around improved communication and collaboration, are absolutely critical to your success at enterprise business modeling.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseBusinessModeling.html>

Agile For Data Professionals



When you are first formulating your enterprise architecture model, it isn't reasonable to try to address all of these issues at once; instead you should tackle individual aspects of your enterprise architecture one at a time. This strategy is also appropriate for evolving your enterprise architecture: you want to do so gradually so that your organization can successfully adopt a new infrastructure in a low-risk manner. A good approach is to identify and then validate candidate architectures, which are the building blocks from which your enterprise architecture is built. In other words, be as agile as possible in your EA approach.

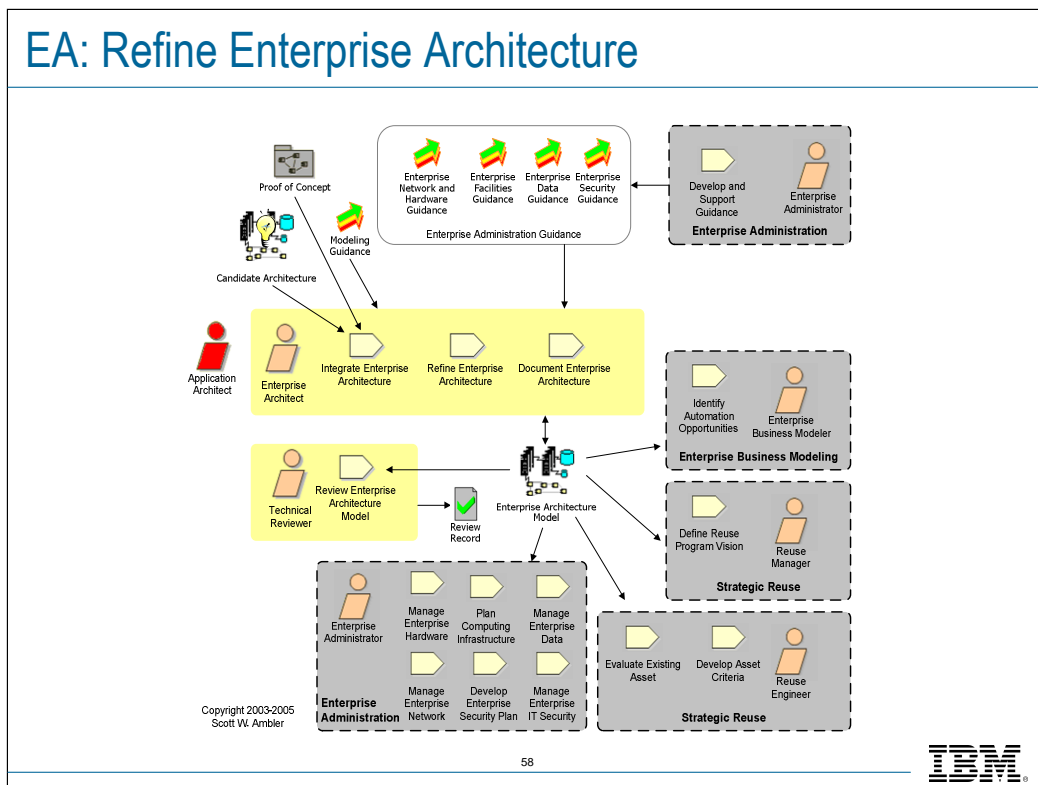
A candidate architecture is a proposed addition or change to the enterprise architecture that has not been fully detailed or implemented; it may be as nebulous as the statement "start exposing our applications to a wider audience of users, possibly via web services" or as concrete as the idea that you should adopt a persistence framework such as Hibernate. Candidate architectures can be described in various formats, from something as simple as a paragraph describing the idea to a simple position paper to a detailed white paper describing a new technology. Depending on the scope of the candidate architecture, you may need to investigate it further through online research, attendance at conferences or training sessions, online discussion groups, or by talking with your peers at other organizations that have experience in the technologies.

The next step is to show that the candidate architecture actually works and fits into your existing enterprise architecture.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseArchitecture.html>

Agile For Data Professionals



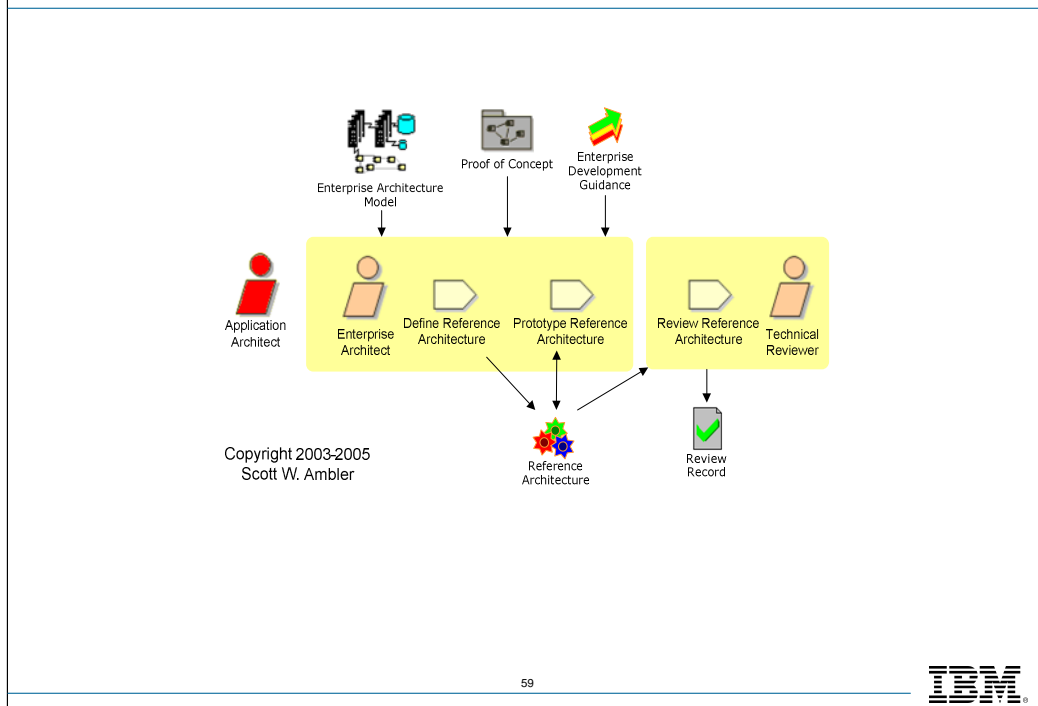
With a suitable candidate architecture in hand, the enterprise architects integrate it into your enterprise architecture model. Your enterprise architecture model should reflect accepted enterprise administration guidance, such as security and data standards and guidelines, which is maintained and supported by your enterprise administrators. It should also reflect the modeling and enterprise development guidance that the enterprise architects develop and support for your organization. Your enterprise architecture is probably very complex, so you will need several views to adequately model it.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseArchitecture.html>

Agile For Data Professionals

EA: Define Reference Architecture



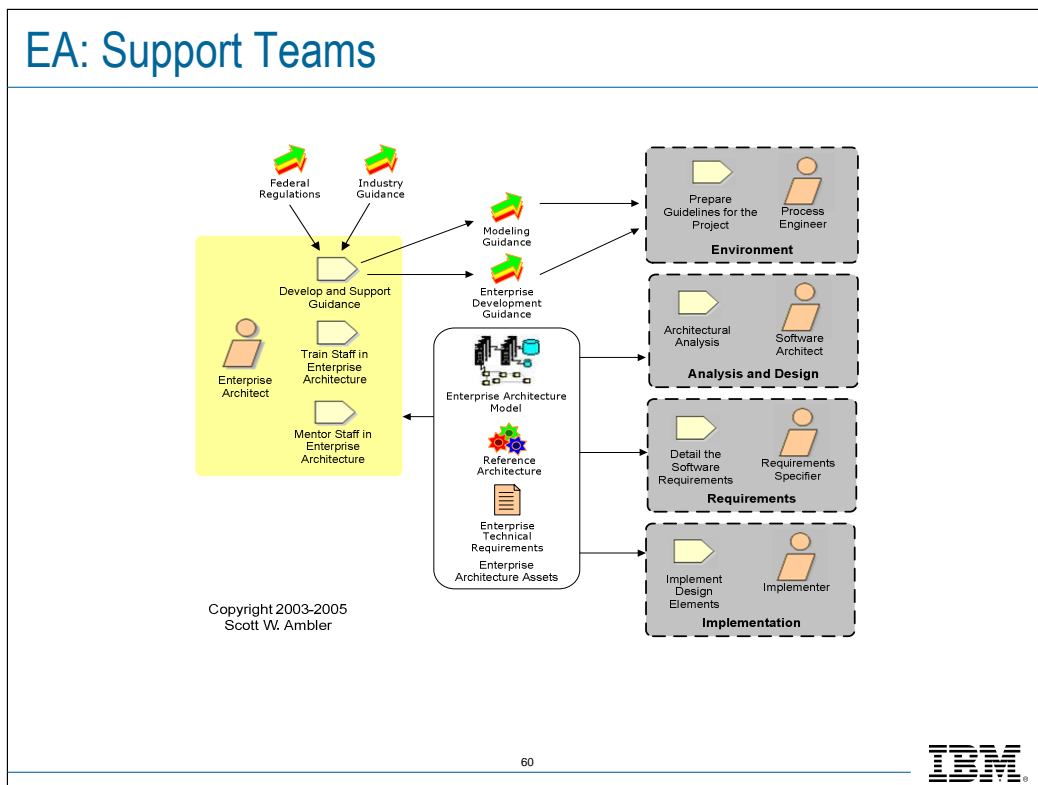
A reference architecture is a working example designed and proven for use in a particular domain, together with supporting artifacts to enable their use; it at least serves as an example and at best provides the basis for creating an application architecture. Reference architectures help application teams to craft application architecture that utilize your enterprise architecture. They can greatly speed application development effort and minimize support costs because applications are architected in standard, proven ways. Reference architectures are not theoretical models of how things might work. They are proven patterns of how things do work. Nothing demonstrates that better than working code that teams can inspect to see how to apply it.

A reference architecture may be geared towards a particular technology or domain. The reference architecture should be documented in a manner that is easy to use by application developers. One way to do so is to create a standard RUP Software Architecture Document (SAD). Remember to keep your documentation concise and well written; it is possible to be agile when creating documents.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseArchitecture.html>

Agile For Data Professionals



A vital part of the enterprise architecture discipline, as with the other enterprise management disciplines, is supporting project teams. It does no good to define a great enterprise architecture if application teams cannot understand it or do not adhere to it. Enterprise architects must work with project teams to help them succeed at adopting and following the enterprise architecture.

Enterprise architects work closely with application architects (the software architect role in RUP) to help them formulate architectures for specific applications. They do this by helping them to apply reference architectures and to craft application architectures with enterprise needs in mind. A good enterprise architect will spend the majority of his or her time mentoring others. Enterprise architects will also hold training sessions. This may take the form of formal, classroom-style lectures, brownbag lunches, or presentations. Training is often called for when a significant change or addition to the enterprise architecture is made that needs to be communicated to the entire IT organization. Enterprise architects can get the word out to the largest audience quickly by presenting some form of training followed up with mentoring to specific application teams or individuals.

Diagram used with permission

See <http://www.enterpriseunifiedprocess.com/essays/enterpriseArchitecture.html>

Agile For Data Professionals

For existing Enterprise Architecture (EA) programs, what has improved?

(Rating between -10 and +10)

1. System integration (3.6)
2. IT governance (3.3)
3. Team follows common technology infrastructure (3.3)
4. Business efficiency (3.2)
5. Data integrity (3.2)
6. Continuity of organizational knowledge (3.0)
7. Business governance (3.0)
8. Audit compliance (2.9)
9. Risk management (2.9)
10. Technical integrity (2.8)
11. Operating costs (2.5)
12. Enterprise decision making (2.5)
13. Reduction of waste (2.3)
14. Support for multi-vendor projects (1.8)
15. Outsourcing initiatives (1.3)
16. Reduction of technical complexity (0.8)

Source: Dr Dobb's January 2010 State of the IT Union Survey

61



Ratings are -10 to +10. Rating was calculated as a weighted average. Very improved = 10, improved = 5, worse = -5, and much worse = -10

The greater the number, the greater the average improvement.

The EA reality doesn't seem to live up to the EA rhetoric, although there are some clear benefits for orgs succeeding at EA.

Survey results are posted at

<http://www.ambysoft.com/surveys/stateOfITUnion201001.html>

Agile For Data Professionals

For existing Enterprise Architecture (EA) programs, what were the importance of success factors/strategies? (Rating between -10 and +10)

1. Active involvement of business leaders (5.8)
2. Active involvement of IT leaders (5.7)
3. Enterprise architects are active participants on project teams (5.5)
4. Enterprise architects are trusted advisors of the business (5.5)
5. Flexible enterprise architects (5.1)
6. Having a business case for EA efforts (4.5)
7. Continuous improvement/evolution of EA artifacts (4.5)
8. Architecture reviews (4.1)
9. Appropriate governance (4.1)
10. Cost reduction (3.5)
11. Master data management (MDM) (2.8)

Source: Dr Dobb's January 2010 State of the IT Union Survey

62



Ratings are -10 to +10. Rating was calculated as a weighted average. Very improved = 10, improved = 5, worse = -5, and much worse = -10

The greater the number, the more important the success factor.

The success factors seem focused mostly on “soft” people issues

Survey results are posted at

<http://www.ambysoft.com/surveys/stateOfITUnion201001.html>

Agile For Data Professionals

For cancelled Enterprise Architecture (EA) programs, why was it ended? (Rating between -10 and +10)

1. Insufficient time provided (3.3)
2. Project teams didn't take advantage of the EA (3.2)
3. Too difficult to measure benefits (2.5)
4. Enterprise architects perceived as "ivory tower" (2.5)
5. Development teams couldn't wait for enterprise architects (2.5)
6. No perceived benefit of EA program (2.0)
7. No executive endorsement (1.7)
8. Enterprise architects weren't sufficiently flexible (1.5)
9. Enterprise architects perceived as impediment to success (1.5)
10. Insufficient funding (1.5)
11. EA perceived as not viable (0.0)
12. Cancelled due to political issues (-0.6)
13. EA program successful but terminated (-1.9)

Source: Dr Dobb's January 2010 State of the IT Union Survey

63



Ratings are -10 to +10. Rating was calculated as a weighted average. Very improved = 10, improved = 5, worse = -5, and much worse = -10

The more positive the score, the greater the factor contributed to cancellation on average

EA programs take years to see benefits, as we saw on a previous slide, so it's little surprise that the leading cause of failure is not giving the EA programs enough time. However, most of the other failure factors are focused around the EA team not executing effectively with the development teams that or around business issues.

Survey results are posted at
<http://www.ambysoft.com/surveys/stateOfITUnion201001.html>

Agile For Data Professionals

Lean Governance

- Governance defines responsibilities, decision rights, and accountabilities
- Data governance, SOA governance, ... must be an overall part of your IT governance strategy, not separate efforts
- Traditional IT governance is based on command and control strategies which often prove ineffective in practice
- Lean IT governance is based on motivation, collaboration and enablement

64



See IBM Whitepaper "Lean Development Governance" at www.ibm.com

Lean governance practices:

- Pragmatic Governance Body – cost effective techniques, focus on enabling development teams, not controlling them
- Staged Program Delivery – roll out the program in chunks / increments. Subprojects must sign up to release date. If subproject misses, it skips to the next release, but this minimizes the impact to customers.
- Manage Project Pipeline by Business Value – invest in the projects that make the most sense, with less focus on "cool technology"
- Iterative Development – incremental approach to development. Build a system in time-boxes, not as a single-big bang effort
- Adapt the Process – one process size does not fit all, tailor the process to meet a team's exact needs. Processes must be evaluated and evolve over time. Shouldn't have a "because we've always done it that way" mentality
- Risk-based Milestones – huge feature of RUP, Mitigate business, technical, ... risk early in the lifecycle
- Continuous Improvement – Identify and act on lessons learned throughout the project, not just at the end
- Embedded Compliance – Build in compliance into your day-to-day process, do not have a separate compliance process which causes unnecessary overhead. Automation is critical
- Simple and Relevant Metrics – Automate metrics collection, minimize the number of metrics collected, know why you're collecting them
- Continuous Project Monitoring – Use automated metrics to monitor projects, identify potential issues and work with teams to resolve them early
- Integrated Lifecycle Environment – Automate as much of the drudge work as possible, tools and processes should fit together effectively throughout the lifecycle.
- Valued Corporate Assets – People should follow guidance, reuse assets, and conform to common infrastructure because these things add value, not because they're forced to do so. Make it easier to follow existing guidelines rather than creating their own.
- Flexible Architectures – SOA, components, objects, patterns, ... lead to greater levels of consistency, reuse, enhancability, and adaptability
- Self-Organizing Teams – IT professionals are intelligent people who can determine their own strategies for working, for planning their work within established parameters (such as iteration boundaries) and are provided the right coaching.
- Align HR Policies with IT Values – Hiring, retaining, and promoting technical staff requires different strategies than non-technical staff. Promote rewards for people to learn new skills. Limit bureaucracy, and put incentives/rewards in place for timely delivery or other key accomplishments.

Agile For Data Professionals

Agile Data Administration Strategies

- Agile data administrators:
 - ▶ Work in a collaborative manner
 - ▶ Work in an evolutionary manner.
 - ▶ Communicate their role
 - ▶ Mentor agile software developers in corporate standards and guidelines
 - ▶ Be willing to actively address issues
 - ▶ Work with the enterprise architects
 - ▶ Adopt a lean approach to data governance
 - ▶ Take an agile approach to Master Data Management (MDM)

65



Work in a collaborative manner. They work side by side with people on project teams as needed. They don't dictate or try to enforce standards or procedures nor do they try to control the project team. They realize that their primary goal is to enable teams to effectively work within the bounds of their organization, and that the most effective way to do this is through collaboration.

Work in an evolutionary manner. Because agile teams work in an evolutionary manner the enterprise administrators must also work this way. The implication is that the administrators won't have complete project artifacts to work with – e.g you can't insist that a development team puts a logical data model in place before you'll help them to identify potential data sources.

Communicate their role. Agile developers are self governing, they do not work on a “command and control” approach. Trying to “impose your will” through onerous processes or management edicts won't work very well, instead people will find ways to either ignore you or go around you. In fact, there is a technique called [blocking](#) which works incredibly well in practice.

Mentor agile software developers in corporate standards and guidelines. The goal is to support the standards and guidelines, not enforce them. A good rule of thumb is that if you need to act as the “standards police” then you have lost the battle.

Be willing to actively address issues. When pushback occurs an enterprise administrator works with the project team(s) explore and address the problem. Perhaps a standard isn't appropriate for a new technology. Perhaps existing logical data definitions need to be updated to reflect new usage. Perhaps the network needs to be upgraded to support a new application. Enterprise administrators must be willing to actively work with project team members to resolve issues such as this.

Work with the enterprise architects. Enterprise administrators work closely with enterprise architects to communicate the constraints imposed by the current environment to the architects. More importantly the enterprise administrators need to understand the future direction envisioned by the enterprise architects to ensure that their efforts support the long-term direction of your organization.

Adopt a lean approach to data governance.

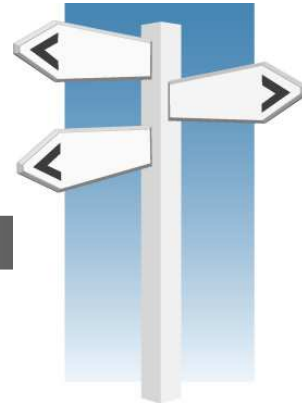
Take an agile approach to Master Data Management (MDM). The goals of MDM are to promote a shared foundation of common data definitions within your organization, to reduce data inconsistency within your organization, and to improve overall return on your IT investment. There is nothing inherently special about MDM -- it doesn't need to be complex nor bureaucratic. The only way for MDM to succeed is for it to enhance data-oriented efforts on development projects and not hamper their overall progress. Traditional approaches to MDM typically do the exact opposite, and as a result struggle to provide value.

Source: <http://www.agiledata.org/essays/enterpriseAdministration.html>

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- Challenges with data activities
- Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- ***Scaling agile**
- Parting thoughts

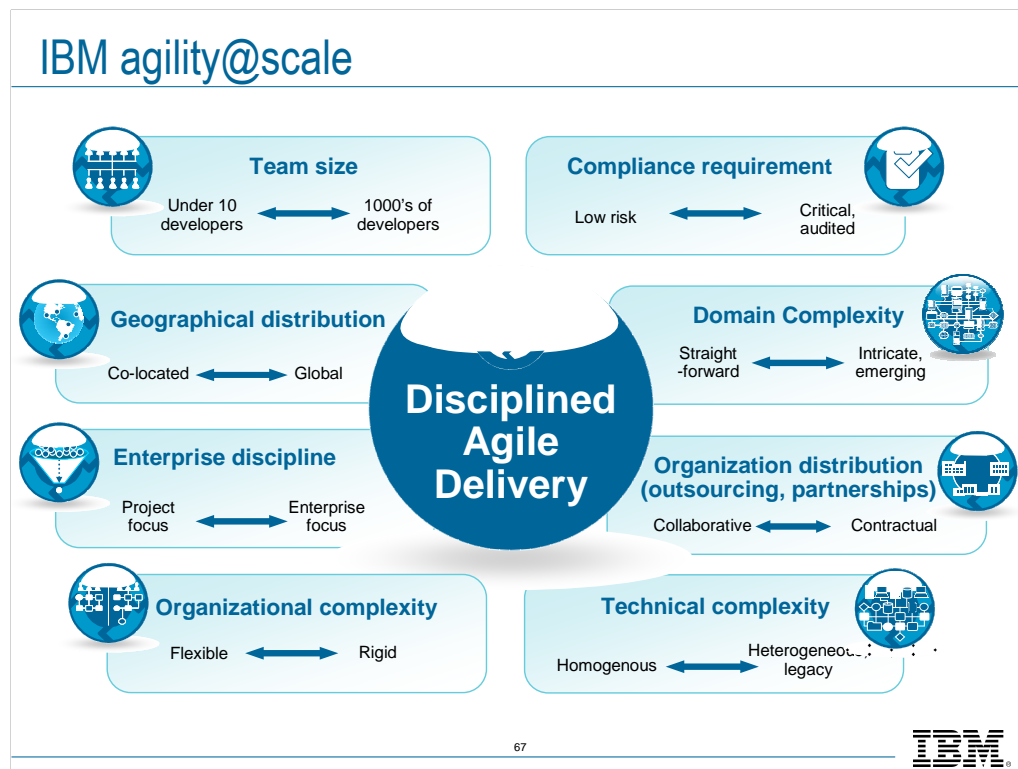


*=Current topic



66

Agile For Data Professionals

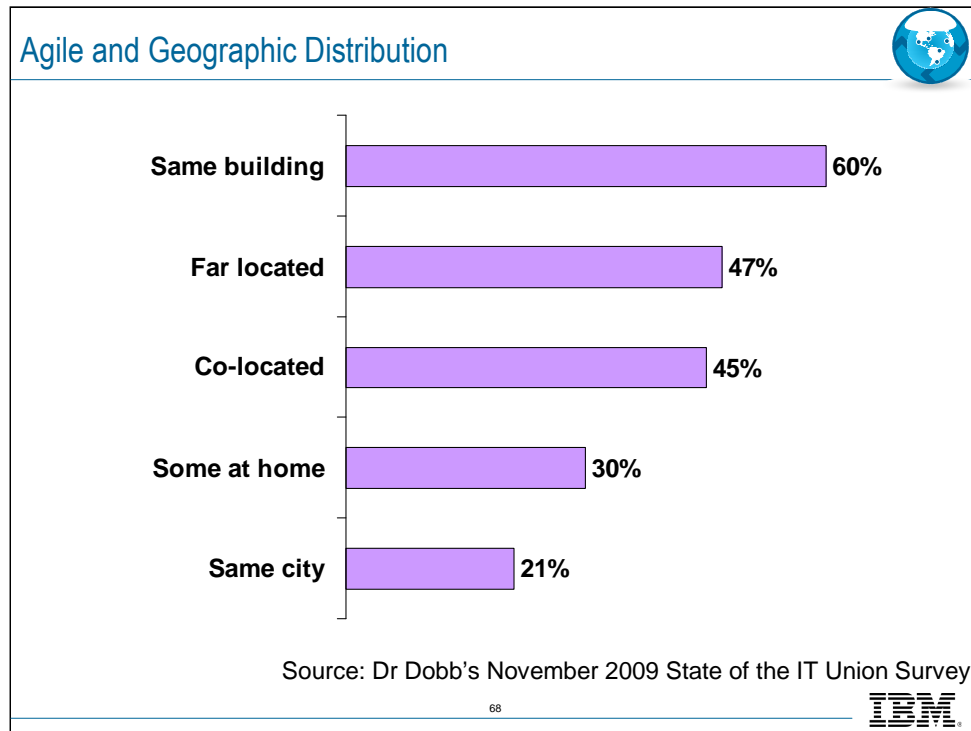


In the early days, agile development was applied to projects that were small in scope and relatively straightforward. Today, organizations want to apply agile development to a broader set of projects. Agile needs to adapt to increasing complexity. Agility@Scale is about explicitly addressing the complexities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

1. **Geographical distribution.** What happens when the team is distributed within a building or across continents?
2. **Team size.** Mainstream agile processes work well for small teams (10-15), but what if the team is fifty people? One hundred people? One thousand people?
3. **Compliance requirement.** What if regulatory issues – such as Sarbanes Oxley, ISO 9000, or FDA CFR 21 – are applicable?
4. **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.
5. **Organization distribution.** Sometimes a project team includes members from different divisions, different partner companies, or from external services firms.
6. **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.
7. **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.
8. **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.

Agile For Data Professionals



Same building => Some teams had people working in different cubicles or rooms in the same building

Far located => Some teams had people working at very distant locations

Co-located => Everyone in single room

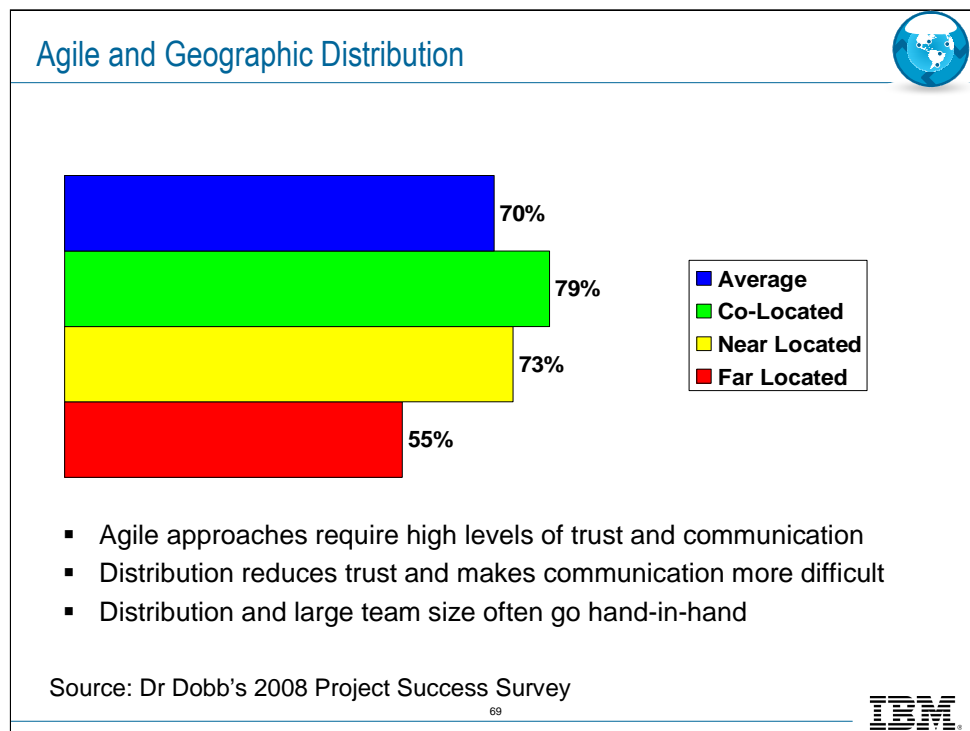
Some at home => Some teams have people working from home

Same city => Some teams have people working in different buildings that are within driving distance

Question asked if you'd been successful with agile at a given level of team distribution.

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

Agile For Data Professionals



The more distributed a team is, the greater the risk. The same holds true for other paradigms, and at no distribution level was a traditional strategy more successful than either an agile or iterative strategy.

Definitions used in the survey:

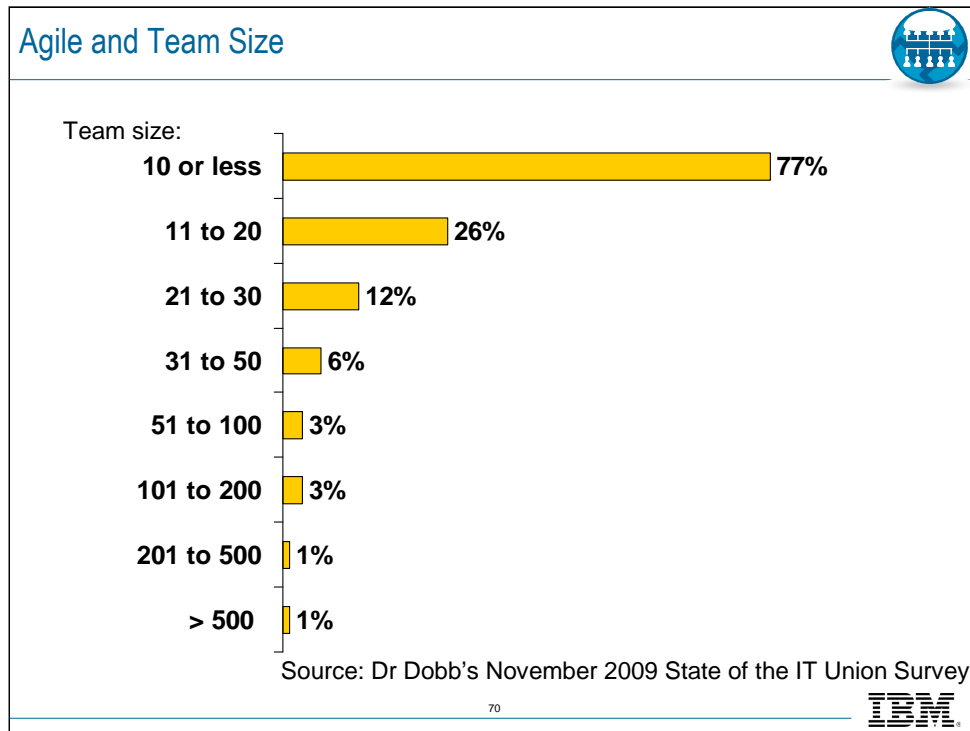
On an agile software development project the team follows an iterative process which is also lightweight, highly collaborative, self-organizing, and quality focused. An example of an agile process is OpenUP, Scrum, and XP.

Co-located teams: Everyone, including primary stakeholders, are in the same room.

Near-located teams: Some people may be in cubes, or on different floors, or different buildings, or working from home BUT everyone is in the same geographic area and could potentially get together each day for a physical group meeting if need be.

Far-located teams: Some people are at such a distance that they would need to take a plane to attend a physical meeting of the entire team.

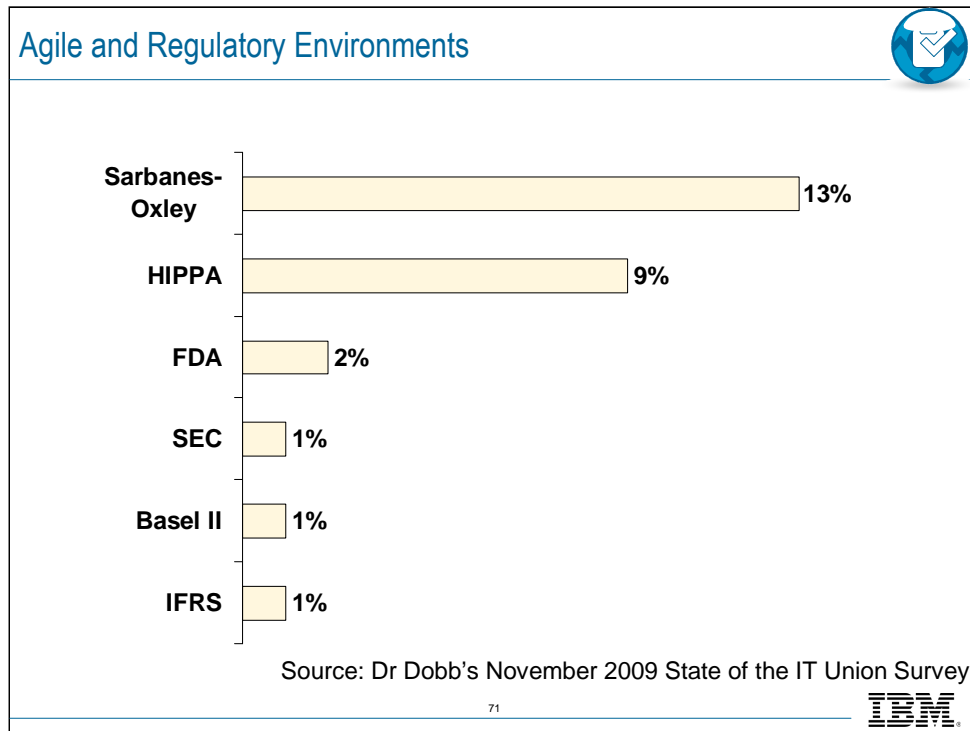
Agile For Data Professionals



Question asked if you had been successful with agile on a given team size.

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

Agile For Data Professionals

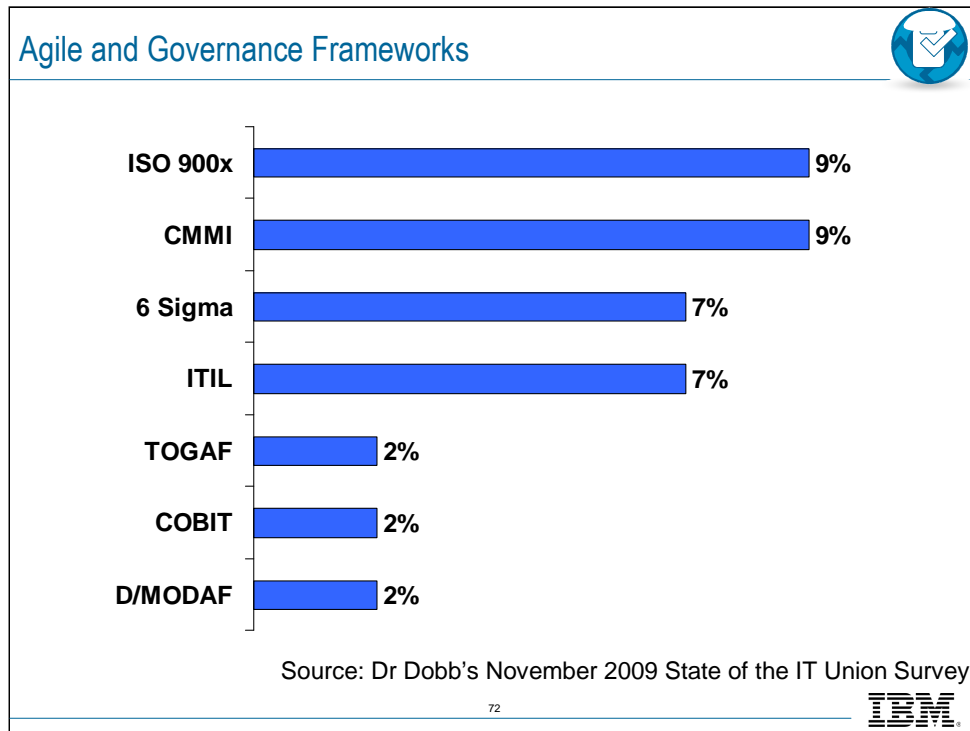


Question asked if you had been successful with agile with given regulations.

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

The 2009 Ambysoft Agile Project Initiation Survey (www.ambysoft.com/surveys/) found that 1/3 of agile teams work within a regulatory environment.

Agile For Data Professionals

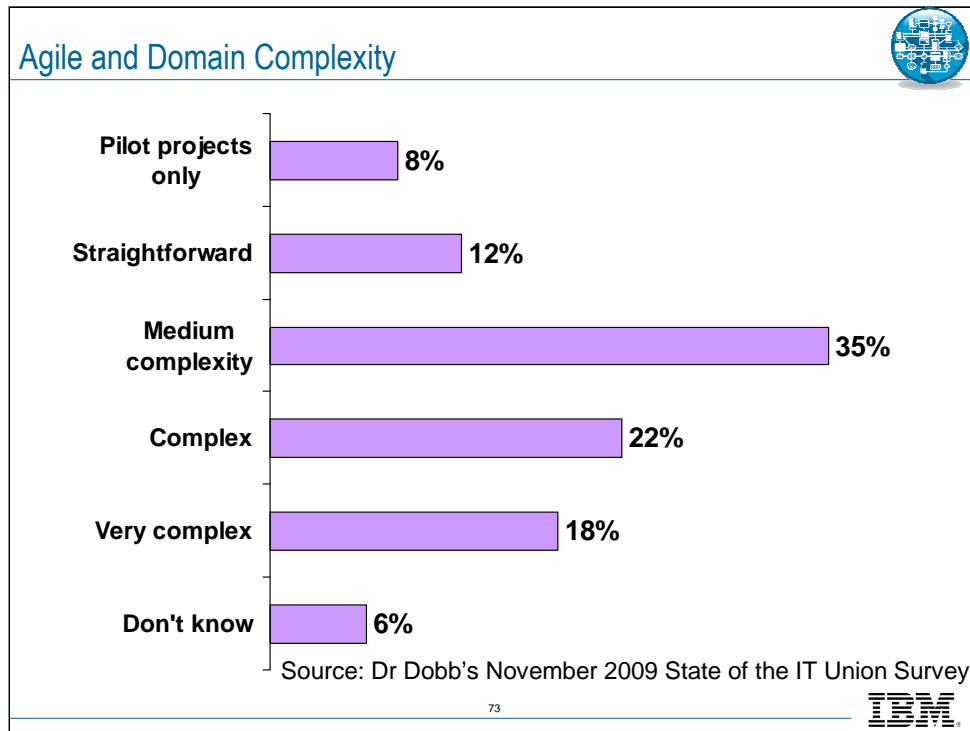


D/MODAF = DODAF or MODAF

Question asked if you had been successful with agile with given frameworks.

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

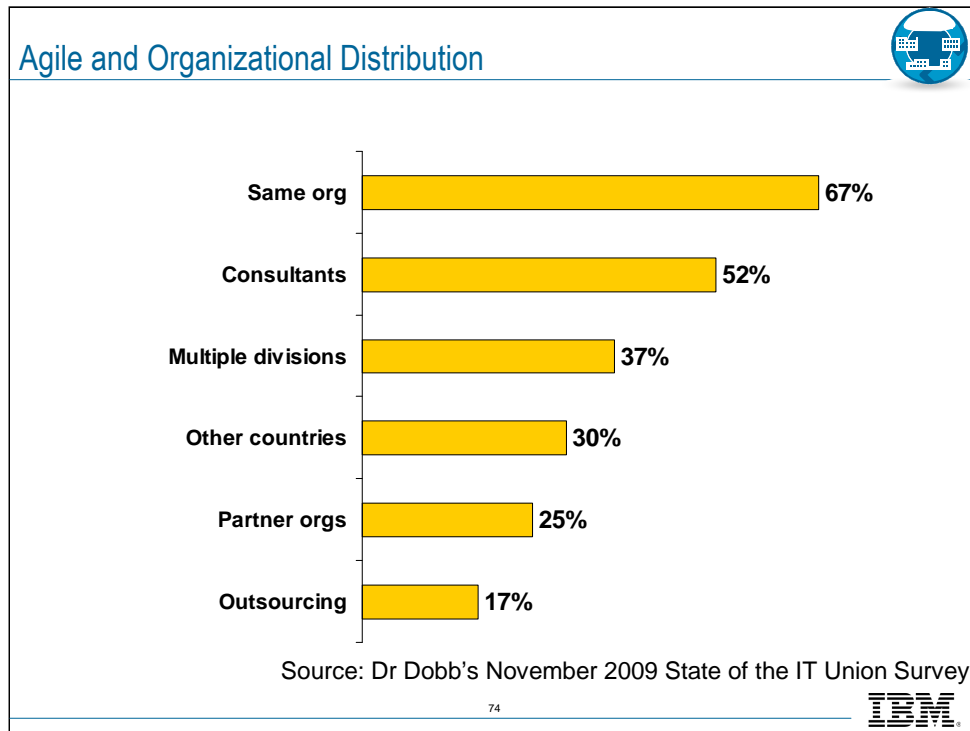
Agile For Data Professionals



Question asked about how you perceived the complexity of the “hardest domain complexity” you had been successful with agile.

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

Agile For Data Professionals



Question asked about whether you had been successful with agile with various levels of organizational distribution.

Same org → Everyone works for same org

Consultants → Project includes contractors/consultants

Multiple divisions → Some projects include people from more than one business division within our org

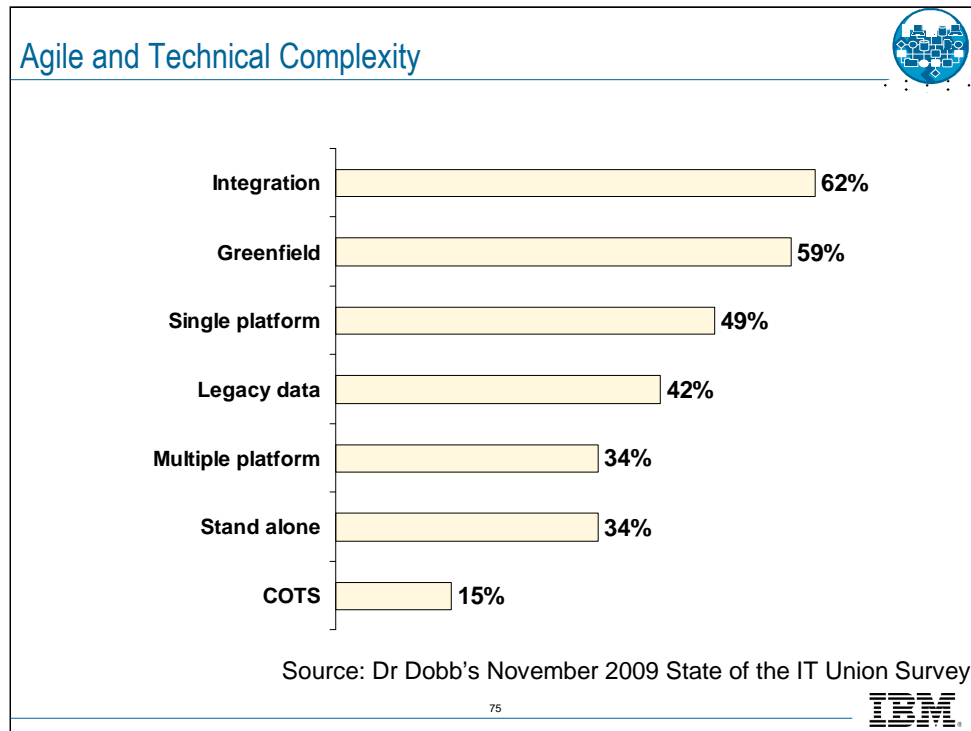
Other countries → Some projects include people from other countries who work for our org

Partner orgs → Some projects include people from partner organizations

Outsourcing → Some projects outsource some of the work to external organizations

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

Agile For Data Professionals



Question asked about how you perceived the complexity of the “hardest technical complexity” you had been successful with agile.

Integration → Integration with existing systems/services/components

Greenfield → New system

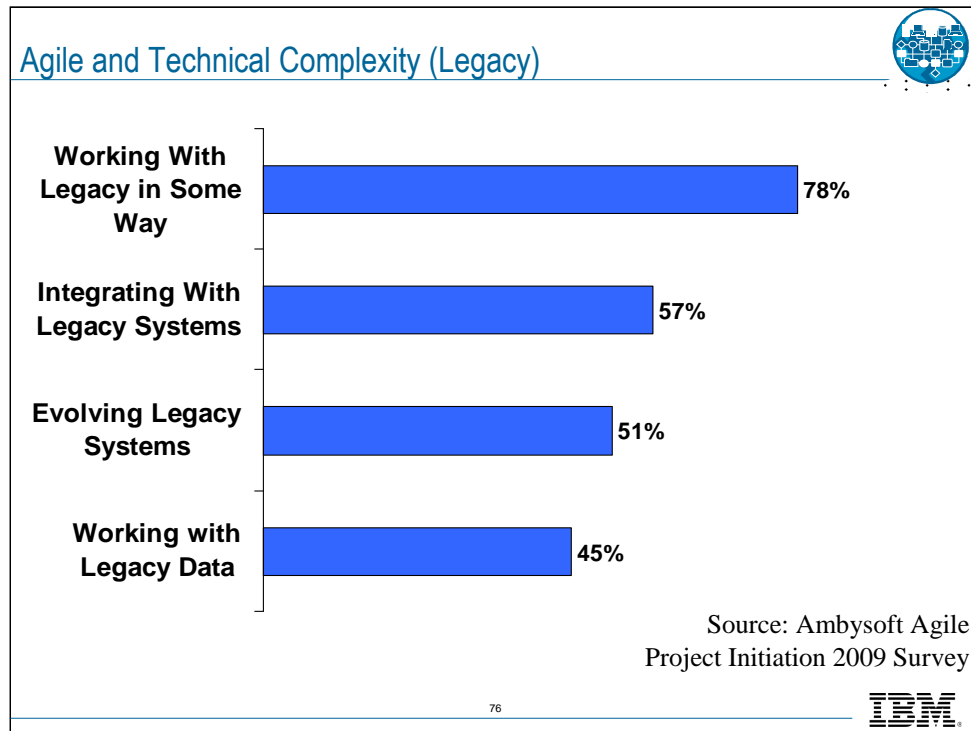
Legacy data → working with legacy data sources

Stand alone → building a silo system

COTS → Package implementation/Commercial off the shelf

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

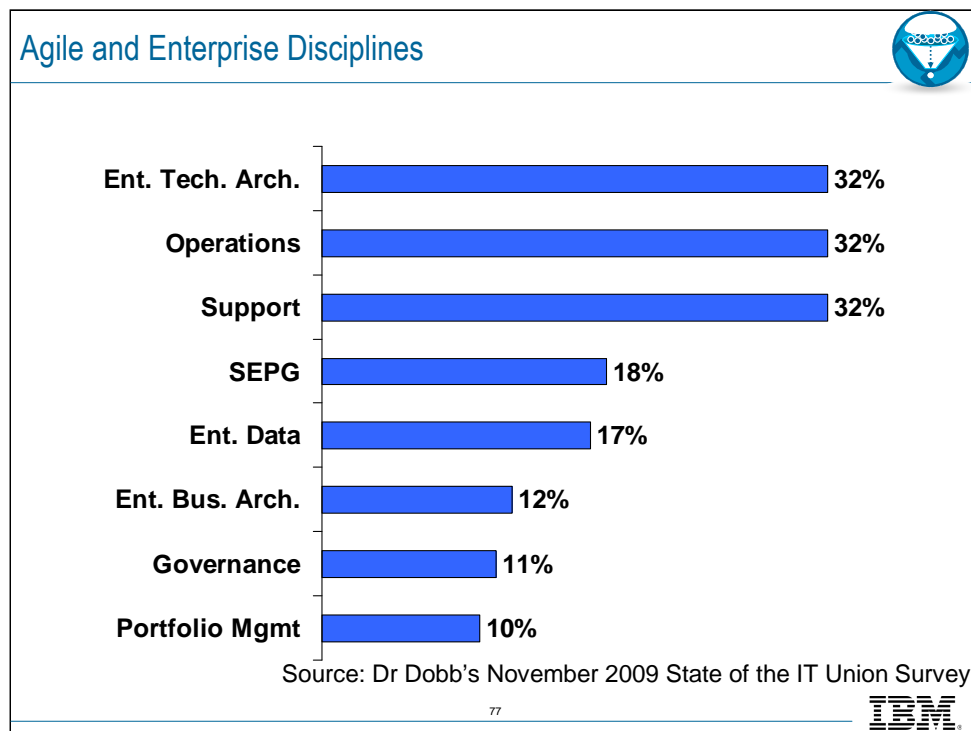
Agile For Data Professionals



Working with legacy in some way was calculated if the people were doing one or more of the three legacy options

<http://www.ambysoft.com/surveys/projectInitiation2009.html>

Agile For Data Professionals



Question asked about whether given enterprise groups had successfully engaged with agile teams. DID NOT ask whether they had the groups or not.

Ent Tech Arch → Enterprise technical architecture

SEPG → Process engineering/improvement team

Ent Data → Enterprise Data

Governance → Governance team/body

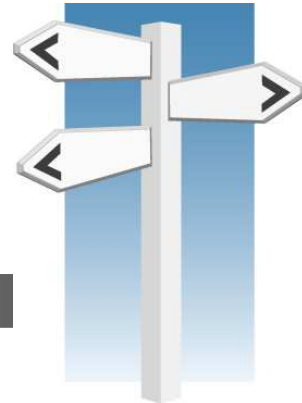
Ent bus arch → enterprise business architecture

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200911.html>

Agile For Data Professionals

Agenda

- Comparing strategies
- Agile fundamentals
- Challenges with data activities
- Agile database techniques
- Agile strategies for DW/BI
- Agile enterprise data
- Scaling agile
- ***Parting thoughts**



*=Current topic



78

Agile For Data Professionals

Training and Education

- Developers should have skills in:
 - ▶ Agile database techniques
 - ▶ Architecture fundamentals
- Data professionals should have skills in:
 - ▶ Agile development techniques
 - ▶ Architecture fundamentals
- Architects should have skills in:
 - ▶ Agile development techniques
 - ▶ Agile database techniques
- Promote the concept of generalizing specialists
 - ▶ www.agilemodeling.com/essays/generalizingSpecialists.htm

79



Being overly specialized in a single area (data, programming, security, ...) likely isn't good for your long term career prospects nor is it good for your overall productivity. See www.agilemodeling.com/essays/generalizingSpecialists.htm for a detailed discussion

Agile For Data Professionals

Non-Solo Development

- Examples:
 - ▶ (Promiscuous) Pair programming
 - ▶ Modeling with others
- Advantages:
 - ▶ Reduces need for formal reviews and inspections
 - ▶ Knowledge and skills disseminated amongst team
 - ▶ Motivates common standards and guidelines
 - ▶ Improves productivity
 - ▶ Breaks down political barriers
- Disadvantages:
 - ▶ Management doesn't understand it
 - ▶ Very small minority (> 5%) don't like it

80



Agile For Data Professionals

Strategies for Developers to Learn Data Skills

- Data professional-led skills transfer:
 - ▶ 32% offered either mentoring by data professionals and/or pairing with data professionals to developers
 - ▶ 22% work in organizations where developers are mentored by data professionals in data skills
 - ▶ 21% work in organization where developers pair with data professionals to gain data skills
- Training in data skills:
 - ▶ 18% work in organizations which offer some form of data training to developers
 - ▶ 13% work in organizations which offer data modeling training to developers
 - ▶ 10% work in organizations which offer data administration training to developers
- Less than ideal:
 - ▶ 61% work in organizations that expect developers to pick up data skills on their own
 - ▶ 45% work in organizations where developers pair with other developers to gain data skills
 - ▶ 10% work in organizations which believe that developers don't need data skills

Source: DDJ September 2009 State of the IT Union Survey

81



62 (10%) out of 613 respondents deliberately did not provide data skills to developers (49 because data activities left to data professionals and 26 because access encapsulated)

110 (18%) out of 613 offered some form of data training to developers

197 (32%) out of 613 offered mentoring by data professionals or pairing with data professionals

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200909.html>

Agile For Data Professionals

Strategies for Data Professionals to Learn Development Skills

- Developer-led skills transfer:
 - ▶ 36% work in organizations where data professionals either receive mentoring in development skills from developers and/or pair with developers
 - ▶ 26% work in organizations where data professionals pair with developers to gain development skills
 - ▶ 21% work in organizations where data professionals are mentored by developers in development skills
- Training in development skills:
 - ▶ 17% work in organizations which provide development training to data professionals
- Less than ideal:
 - ▶ 40% work in organizations where data professionals are expected to pick up development skills on their own
 - ▶ 23% work in organizations where data professionals pair with other data professionals to gain development skills
 - ▶ 20% work in organizations where which believe that data professionals do not need development skills

Source: DDJ September 2009 State of the IT Union Survey

82



124 (20%) out of 613 respondents had no training in development stuff (117 because left to development, 20 because of encapsulation)

219 (36%) out of 613 indicated that data professionals get mentoring by developers or pair with developers

Source: <http://www.ambysoft.com/surveys/stateOfITUnion200909.html>

Agile For Data Professionals

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



83



IBM is helping organizations around the world to scale agile strategies to meet their unique situations.

Agile For Data Professionals



IBM Software Group

Backup Slides

 IBM agility@scale™



© 2010 IBM Corporation

Agile For Data Professionals

Agile has gone mainstream

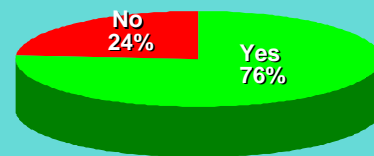
"Thirty-five percent ofrespondents have projects or pilots underway, and *only 12 percent do not see a fit* for agile processes in their organizations.

The fact that 88 percent of these organizations (one-third of which have over 10,000 employees) are using or evaluating agile processes proves that agile processes have truly hit the mainstream."

- Excerpt from "And the Agile Survey Says..."
Agile Journal, March 6, 2006

Third-party research suggests even wider adoption

Have you adopted any agile techniques?



Source: State of the IT Union Survey, Dr. Dobb's Journal, July 2009

85



While agile was once considered viable only for small, co-located teams, the improvements in product quality, team efficiency, and on-time delivery have caused larger teams to take a closer look at adopting agile principles in their environments. In fact, in a recent study conducted by the Agile Journal, it was determined that 88% of companies, many with over 10,000 employees, are using or evaluating agile practices on their projects. Agile is truly poised to become the dominant software development paradigm.

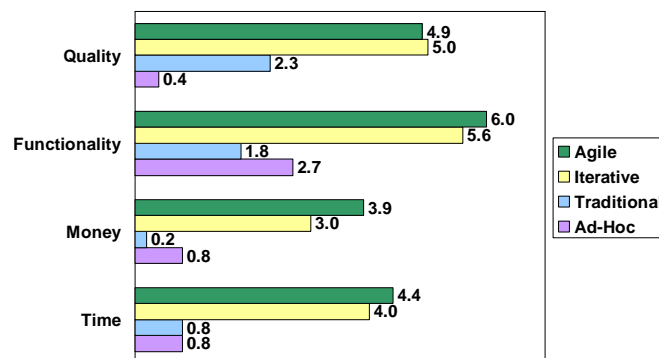
This trend is also echoed in other industry studies, including one conducted by IBM® Rational® agilist, Scott Ambler. Ambler's study suggests a 76% adoption rate of agile techniques.

Agile techniques, such as test-driven development (TDD), continuous integration, iterative development, agile modeling, daily stand-up meetings, and so on are quickly being adopted within large IT organizations.

Agile For Data Professionals

Why are organizations adopting agile strategies?

- Agile teams have an average success rate of 70% compared with 66% for traditional/waterfall teams
- 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
- Detailed results online at www.ambyssoft.com/surveys/



Source: Dr. Dobb's Journal (DDJ) 2008 Project Success Survey



Iterative and agile approaches are more effective delivering higher quality, greater ROI, better stakeholder satisfaction, and deliver in a timely manner compared with traditional and ad-hoc approaches. Traditional approaches were better than ad-hoc when it came to quality, but ad-hoc approaches were better at delivering functionality.

People really don't define success in terms of "on time, on budget, to specification" regardless of what the theory folks may claim. For example:

- 83% of respondents believe that meeting actual needs of stakeholders is more important than building the system to specification.
- 82% believe that delivering high quality is more important than delivering on time and on budget
- 70% believe that providing the best ROI is more important than delivering under budget
- 58% believe that delivering when the system is ready to be shipped is more important than delivering on schedule

Agile For Data Professionals

What is disciplined agile?

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.



87



Agile can mean different things to different people. Finding common ground can help to avoid misunderstandings. As a result of working with companies who are undertaking Agile Development, IBM Rational has made some key observations:

- Agile software development can be significantly different from one organization to another. Agile is not a “one size fits all” proposition
- Agilists do a lot more testing than traditional teams. In fact they often write a test before they write sufficient production code to fulfill that test; then they iterate
- Agilists work together very closely and, ideally, work closely with their stakeholders
- Changing requirements are seen as a competitive advantage, as long as you can act on them, not as something that you need to prevent
- Agilists deliver working software every iteration, providing concrete evidence of actual progress on the project. Daily builds, or even multiple times a day are highly desirable
- Shorter iterations are desirable, from as short as one-to-two weeks, 4 weeks as a common recommendation, although up to 8 weeks will occur at scale

Agile For Data Professionals

Tower of Babel: Agile terminology

Disciplined Agile	XP	Scrum	Other
Team lead	Coach	Scrum master	Project manager
Iteration	Iteration	Sprint	Timebox
Daily stand up meeting	Daily stand up	Scrum Meeting	Coordination meeting
Retrospective	Retrospective	Sprint retrospective	Reflection meeting
Product owner	Customer	Product owner	Stakeholder representative
Team Member	Extreme programmer	Team member	Developer
Iteration review	-	Sprint review	-

Important: Each agile method has its own terminology. It is highly unlikely that the industry will ever standardize the terminology.



Don't worry too much about the terminology, instead worry about the philosophies and strategies surrounding agile/lean.

Agile For Data Professionals

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 development
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



IBM

Points 1&2: DDJ 2008 Modeling and Documentation survey found that agile teams are just as likely, and sometimes more likely, to model and write documentation as compared to traditional teams (www.ambyssoft.com/surveys/). Just because agile teams are writing detailed specifications up front doesn't mean that they're not modeling and documenting. See www.agilemodeling.com for more information about modeling and documentation

Point 3: Producing working software on a regular basis takes significant discipline. Doing test-driven development (TDD) takes significant discipline. Traditionalists have mistaken bureaucracy for discipline, but filling out paperwork and reviewing it takes very little discipline in practice. See the blog posting at www.ibm.com/developerworks/blogs/page/ambler on this topic.

Point 4: Agile teams take a 2-level approach to planning where the high-level release plan is created early in the project, and updated throughout. Also, detailed iteration task lists are created by the team at the beginning of each iteration. Just because agile teams aren't created big detailed gantt charts doesn't mean that they're not planning.

Point 5: Traditional methodology proves very unpredictable in practice. According to the Standish group, the average traditional project comes in at 189% of budget. Regardless of paradigm, it is futile to try to predict up front to any sort of accuracy the cost, the schedule, and the scope of a development project. However, it is safe to predict that agile teams will produce higher quality, result in greater stakeholder satisfaction, provide better ROI, and get to market faster than traditional teams. See DDJ 2008 Project Success Survey at www.ambyssoft.com/surveys.

Point 6: IBM has delivered software into the marketplace with agile teams of over 200. They currently have agile programs of 5-600, do distributed agile development, perform agile development in regulatory environments, and so on.

Point 7: Many people have written about this over the years. The Level 1 agile community is doing a pretty good job of reinventing a lot of material and practices that have been in The IBM® Rational® Unified Process (RUP) methodology for many years.

Point 8: Various adoption surveys show that the majority of organizations have adopted agile techniques on one or more projects. IBM Software Group has very clearly indicated its support for agile

Point 9: There are various strategies for doing fixed price and scope projects using agile approaches.

Agile For Data Professionals

Agile and Lean are complementary

▪ Agile

- ▶ A philosophy that concentrates on delivering things that have value to a customer
- ▶ Avoid things that have no value to the customer
- ▶ Don't believe in the "big detailed plan"



▪ Lean

- ▶ Started as a management approach for streamlining production.
- ▶ Avoid **all** waste
- ▶ Get the customer involved at the earliest opportunity



90



In *Implementing Lean Software Development*, Mary and Tom Poppendieck show how the seven principles of Lean manufacturing can be applied to optimize the whole IT value stream. These principles offer practical, measurable ways to transform software delivery processes.

Eliminate waste. Lean thinking advocates regard any activity that does not directly add value to the finished product as waste. The three biggest sources of waste in software development are the addition of extra features, churn, and crossing organizational boundaries. Crossing organizational boundaries can increase costs by 25 percent or more by creating buffers that slow response time and interfere with communication. It is critical that development teams be allowed to organize and operate in a manner that reflects the work that they're trying to accomplish—rather than the functional roles of team members.

Build in quality. The Poppendiecks make a simple observation: if you routinely find problems with your verification process then your process must be defective. When you regularly find that your developers are doing things that you don't want them to do—or are not doing what they should be doing—then your approach to governance must be at fault. It's important not to make governance yet another set of activities layered on top of your software process. Instead, the strategy should be to embed governance into your processes, making it as easy as possible for developers to do the right thing.

Create knowledge. Planning is useful, but learning is essential. You want to promote strategies, such as iterative development, that help teams discover what stakeholders really want and act on that knowledge. It's also important to have a body of reusable standards and guidelines that people can easily modify to meet specific project needs. In addition, consistent and timely feedback is important, both within the team and at the program level through continuous monitoring of simple and relevant metrics.

(continued on next page)

Agile For Data Professionals

Principles of Lean software development

1. Eliminate Waste
2. Build Quality In
3. Defer Commitment
4. Deliver Fast
5. Focus on Learning
6. Respect People
7. Optimize the Whole



91



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.