Large-Scale Adoption of Agile Development Lessons Learned

Eugene Levin

Citi's Markets and Banking Technology

283 King George Road, E-4 B-168

Warren, NJ 07059

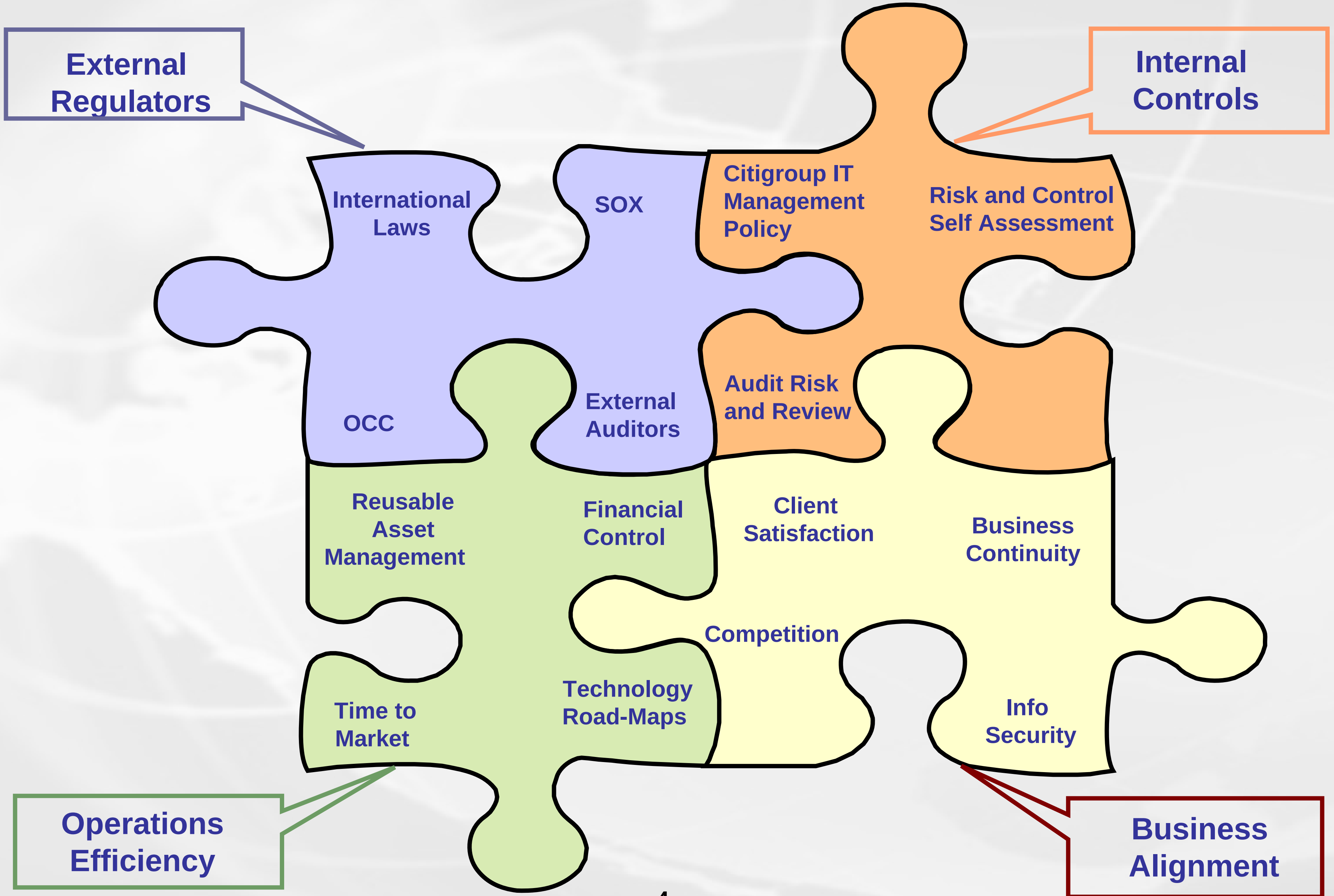
Eugene.Levin@citi.com

Agenda

- Challenges of Developing Software in a Regulated Environment
- Defining Agile Development
- Overview of Citi's Disciplined Agility Framework
- Citi's Agile Rollout Approach
- Pilot Projects Selection Criteria
- Lessons Learned

Challenges of Developing Software in a Regulated Environment

Investment Bank Technology – Process and Governance Drivers



A Baseline Level of Control Has Been Implemented Years Ago

Citi's Information Technology Management Policy

Each organization must establish and maintain a *documented software engineering process* to be used in conjunction with its architecture and project management process.

The *independent confirmation* of the adherence of the software product and software process to applicable standards, procedures, and product requirements.

A Systems Development Life Cycle (SDLC) has been institutionalized throughout CIB-Tech

- A standard, fixed set of phases for software projects has been created.
- Several "route maps", allowing various degrees of choice in process execution and deliverable creation and approval are available.
- Within each route map, certain deliverables are required to be produced; failure in doing so results in a non-compliance issue being raised to various parties.
- The Software Quality Manager role has been created and staffed within each individual group, performing reviews and audits to ensure compliance.
- An Architect role has been created and staffed within each local group, ensuring that software designs are kept aligned with the architectural frameworks established within an area.
- Certain Project Management basics, such as cost planning and base lining, have been instilled as a natural by-product of the governance surrounding the SDLC.

Global Business Drivers Challenge SDLC Methodology

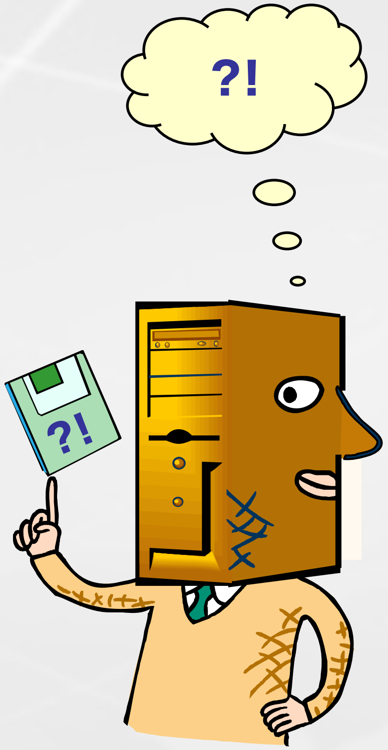
- Although our baseline had enabled a degree of control, it had to be enhanced to enable *faster* software delivery:
  - Several business areas have cited cycle time from request to delivery as being insufficient to keep pace with the needs of the business.
  - Controls were established quickly using a least common denominator approach; this has yielded multiple impedance points between local and corporate practices.
  - Training the several thousand required employees has been a tremendous challenge, resulting in misunderstanding, confusion and frustration with the process.
  - Application of certain measurements in an unfamiliar process has engineered undesirable behaviors into individuals and teams; “gaming” the system is common.
  - “ Route Maps” had extremely strong controls with constrained adaptability and mostly waterfall approach

Defining Agile Development

Foundation of Agile Development

- Development is exploration and learning
- Learning is accelerated by frequent feedback
- Use and review of a working system provides the most effective and credible feedback

"When we build software…the product is not the software; it is the knowledge contained in the software."
"… for the most part, engineers do not know how to build the systems they are trying to build; it is their job to find out how to build such systems."*



* Phillip Armour, *The Laws of Software Process*, ISBN 0849314895, 2004

Agile Manifesto

We are uncovering better ways of developing software by doing it and helping others do it. Through this work we have come to value:

Individuals and interactions over **processes and tools**

Working software over **comprehensive documentation**

Customer collaboration over **contract negotiation**

Responding to change over **following a plan**

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

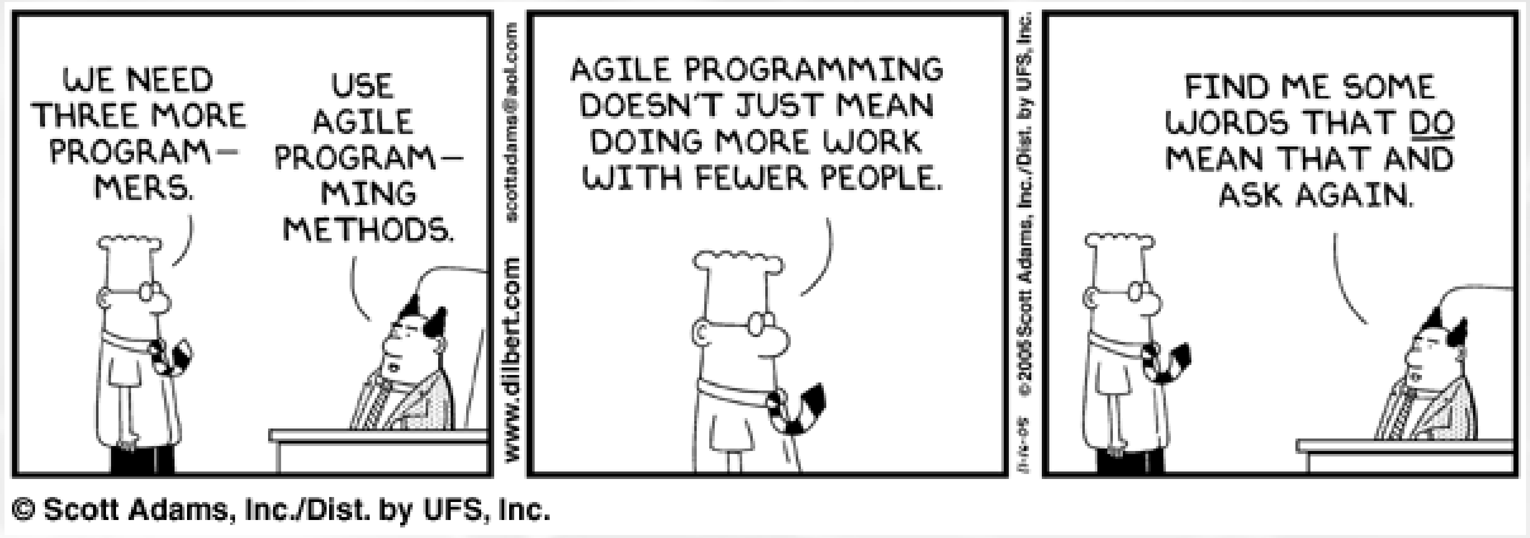
* Source: http://www.agilemanifesto.org/

Agility is the Ability to Create and Adapt to Change

- Focus on *customer value* via constant business-driven prioritization of features and regular customer feedback;
- Manage *uncertainty and risk* through evolutionary (iterative and incremental) product development;
- Utilize *intense collaboration* and feedback via maximum face-to-face communication;
- Unleash team members' creativity and productivity through *light touch leadership* and *self-organization*;
- Deliver high quality through plan-do-study-act *continuous improvement* cycles; and
- Facilitate *learning and adaptation* to change via practices like team *retrospectives* (mini *lessons learned* done while the project is in-flight).

* Source: http://lithespeed.blogspot.com/2007/09/1-page-introduction-to-agile-methods.html

What is Agile Development?


WE NEED THREE MORE PROGRAM-MERS.
USE AGILE PROGRAM-MING METHODS.
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.
www.dilbert.com
scottadams@aol.com
© 2005 Scott Adams, Inc./Dist. by UFS, Inc.
© Scott Adams, Inc./Dist. by UFS, Inc.

"Agile is an iterative and incremental (evolutionary) approach to software development which is performed in a highly collaborative manner with "just enough" ceremony that produces high quality software which meets the changing needs of its stakeholders. " – Scott W. Ambler

*** See Appendix for a list of Agile Manifesto Principles**

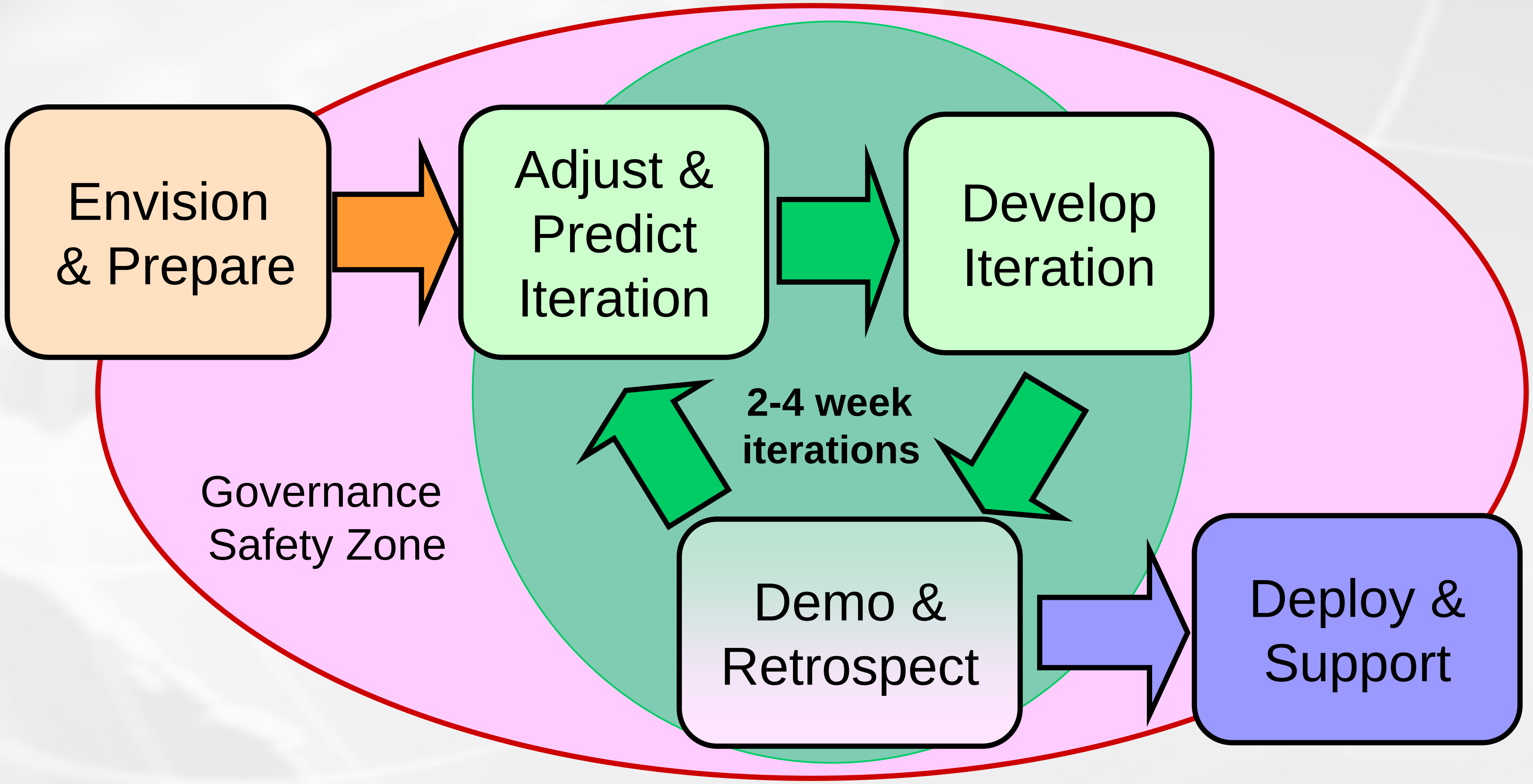
Core Shared Themes of Agile Development

- Evolve applications in multiple short 2-4 weeks iterations
- Client (or surrogate) reviews each release and directs priorities for next iteration
- Track project progress by features completed
- Never slip a release date, reduce scope instead
- Leverage human strengths

Comparing various interpretations of agile development, these themes are common and essential

Overview of Citi’s Disciplined Agility Framework

Citi's Disciplined Agility Framework



Disciplined Agility Framework was developed in collaboration with
Systems and Software Consortium, Inc.
http://www.systemsandsoftware.org/

Disciplined Agility Project-Level Artifacts

Project Request	**[Mandatory]**
Project Control Questionnaire	**[Mandatory]**
Project Charter	**[Mandatory]**
Process Tailoring	[Mandatory]
Backlog	[Mandatory]
Stakeholder Commitments	[Conditional]
Release Plan	[Conditional]
Solution Concept	[Conditional]
Project Charter Authorization	**[Mandatory]**

Project Charter is a living document, created during Envision & Prepare, and updated as needed

Disciplined Agility Iteration-Level Artifacts

Iteration Feature List	[Conditional]*
Acceptance Tests (AT)	[Mandatory]
Feature/AT Status	[Mandatory]
Developer Tests	[Conditional]
Code and Executables	[Mandatory]
Project Status	[Mandatory]
Delivery Signoff	[Mandatory]**

* May be combined with the Backlog/Release Plan

** Required for Release Iterations

Key Points of Disciplined Agility Framework

- Disciplined Agility is NOT a methodology
- Framework defines the minimal set of controls that constitute the *Governance Safety Zone*
- Team has the flexibility to customize the process within governance safety zone
- Fixed iteration length is encouraged
- Teams perform retrospectives for continuous improvement
- Any agile practice can be applied within this framework. (more on the next slide...)

Disciplined Agility Accommodates Various Methods

- **SCRUM** – management framework
- **XP** – integrate software development teams
- **Crystal** – provides guidelines for tailoring a set of elements for a given effort.
- **SCRUM + XP** – increases XP’s scalability to larger projects
- **Crystal+SCRUM+XP** – provides logic for why/when to scale and add weight.

Will Disciplined Agility Projects Pass Audits?

What do Auditors Look For?

Audit is a "systematic, independent and documented process for obtaining audit evidence and evaluating it objectively to determine the extent to which audit criteria are fulfilled."

Audit criteria are a "set of policies, procedures, or other requirements against which collected audit evidence is compared."

Audit **evidence** consists of "**records**, statements of fact or other information, relevant to the audit and which are verified."

-ISO CD2/ISO 19011 and ISO 9000:2000

Compliance audits are fundamentally **documentation reviews.** Audit verifies that **documentation** complies with the standard's requirements and verifies implementation to the '*say what you do and do as you say*' criteria.

Will Disciplined Agility Projects Pass Audits?

Yes, recognizing the fundamental audit rule of thumb which is *"Say what you do and do what you say."*

Say What You Do

- Audit does not define standards, operating groups define standards.
- The articulation of the *Governance Safety Zone* in the Disciplined Agility Framework was developed with significant input from audit/risk/control.
- Every project tailors, fine-tunes and documents unique process during Project Charter workshop and Iteration retrospectives

Do What You Say

- Auditors check that operating groups comply with their own standards.
- The bridge between the standard and compliance with the standard is education of the Auditors and the development personnel on governance requirements of the Discipline Agility framework.

Citi’s Agile Rollout Approach

Objectives for the Early Adoption Phase

- Improved time-to-market. High priority features implemented first
- Each project customizes process appropriate to its needs while ensuring mandatory controls
- Pilots Develop a Self-Sustaining Agile Capability
- Organization form a culture of continuous improvement
- Agile development gains broad acceptance and participation across business clients, development groups, specialists, process support
- An internal pool of experienced agile coaches is formed
- Agile training materials grow and are adjusted based on real examples
- A body of Lessons Learned is captured and formed
- The infrastructure recognizes and supports agile process

Pilot Projects Selection Criteria

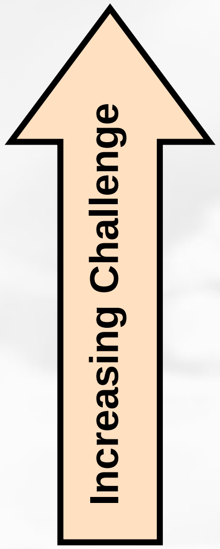
Pilot Projects Selection Criteria During the Early Adoption Phase

- People Factors
- Team Size and Experience
- Project Characteristics

People Factors for Early Adoption

- **Motivated**: see a need to change for improved performance
- **Proactive Client**: has time to actively steer the project
- **Safety**: a feasible project, enough trust to try something new
- **Attitude**: willingness to explore as a team, learn from results

Team Size and Experience Challenges



- Teams with little history of development testing, version control, or process discipline in general
- Prior hostile relationships between team and other project stakeholders
- Volatile/part-time team members
- Distributed teams
- Large teams

Of course, these will add risk to *any* project, but they have even greater impact on first-time agile teams

Agile Project Characteristics

Factor	Traditional	Agile Methodologies
Scope (requirements)	Well known Size well understood Will not change	**Uncertain (Are the requirements correct?)** **Unknown (What is the scope?)** **Subject to change**
Resources (money, infrastructure, people)	Approved and available Has been done before Budget is sufficient and funded People familiar with tasks and tools	**Not fully approved or available** **Need proof of concept** **Money is tight** **Uncertain budget** **New skills needed**
Time	Clearly defined Clear milestones	**Not well defined/open** **Unclear milestones** **Subject to change**
Risks	Well understood Minor impact	**New technologies** **Unknown risks** **Major impact**

Source: http://www.newarchitectmag.com/documents/s=7567/na1002e/index.html

Lessons Learned

Organizational Challenges of Broader Adoption

- Different notions of agile
- Difficulty in bringing the “extended team” together
- Business partnership
- Training and Support
- Existing culture, attitudes and beliefs
- Existing organization/team/work structures
- Tools not geared to support Agile process

Typical Problems and Challenges

- Giving feedback on every iteration [clients]

Not true!

- Slicing requirements into features [team]
- Testing commitment, and test automation [team]
- Getting the team's attention
  - Generic presentation [coach]
  - Daily distraction [team]
  - Development frenzy [PM]
  - Multi-tasking...

More to come...

- Identifying the scope of the viable team... [all stakeholders]
- Embracing the empowered team... [PMs and tech leads]
- Keeping the faith... [PM and team]

Multi-Tasking

- Who/What causes multi-tasking?
  - Who assigns developers to multiple projects?

- Problems in earlier projects cause multi-tasking
  - Delays keep developers from moving on to the new project
  - Serious bugs crop up in operation and interrupt developers in their new project to do emergency fixes in the old one

- Some multi-taxing arises from an attempt to keep people with specialty skills (e.g., DBAs) fully utilized

Multi-tasking causes far more problems than it solves!

– Opt for fewer people dedicated to a project

– Cohesive teams will assist each other across specialty boundaries

Who’s On the Team?

- **Distributed teams**
  - Remote SMTs (e.g., BAs in London and Tokyo)
  - “Cost Reduction”
    - Contractors in India
    - Junior engineers in Shanghai (2nd-class team members)
    - UAT team in Belfast

- **Dependencies on other teams**
  - Are they separate teams for technical or accounting reasons? Act as sub teams or separate teams? For example...
    - Coordinate with other applications up/downstream
    - Infrastructure component we need upgraded
    - Vendor product we need upgraded
    - SMEs that specify the algorithm (e.g., for risk model)
    - Architect (developer or reviewer?)
    - Building different application versions for different clients

Embracing the Adventure, Mastering the Fear

- PM’s and technical leads can have a particularly difficult transition to an agile culture—they need to
  - Encourage the team to take charge (challenge the team to take responsibility)
  - Refrain from intervention at the first sign of uncertainty or trouble (give the team time and nudges to shift into a new mode)
  - Let the team do “the wrong thing” for an iteration just to let them confirm what results they get (it’s only 1 iteration!)
  - Facilitate (not assert!) team estimates and stand up for them (push back on clients trying to wish the system into reality)

Keeping the Faith, Staying Agile

- **Over time do teams start to**
  - Backslide into old habits?
  - Lose their religion under stress and revert to cutting corners?
  - Stray into strange practices?
- **Frankly, we do not always know!**
  - Many projects never send a postcard after the kickoff
  - We struggle and sweat to train and support them and then they never write ☹

Questions & Answers

Appendix

12 Agile Manifesto Principles

- Our highest priority is to satisfy the customer through early and continuous delivery of valuable software.
- Welcome changing requirements, even late in development. Agile processes harness change for the customer's competitive advantage.
- Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale.
- Business people and developers must work together daily throughout the project.

12 Agile Manifesto Principles - Cont

- Build projects around motivated individuals. Give them the environment and support they need, and trust them to get the job done.
- The most efficient and effective method of conveying information to and within a development team is face-to-face conversation.
- Working software is the primary measure of progress.
- Agile processes promote sustainable development. The sponsors, developers, and users should be able to maintain a constant pace indefinitely.

Evolution of SQM Role

- **From Process Compliance to Process Facilitation**
  - Become trusted, credible partner, working closely, not at arm’s length
  - Assess teams and their readiness for Agile development
  - Coach team members on various pragmatic Agile practices
  - Assist in the development of the project customization guidelines
  - Facilitate project’s process customization and fine tuning
  - Help changing the culture
  - Foster Cooperative Relationships among those who are critical to success
  - Monitor process compliance
    - Check artifacts for timeliness and some content. No templates.
    - Compliance monitoring shifts from reactive to proactive – alerts rather than non-conformances

FAQ: Can Agile Work with Offshore Teams?

Yes, when prudent investments are made in multiple modes of communication, seamless environments and team empowerment extends to the offshore team as well.

- **May require more documentation, but not as a replacement for higher bandwidth forms of communication like telephone, net meetings, video presentations and video conferences.**
- **Multiple modes of communication include collaboration tools such as wikis, issue lists, and various tracking tools.**
- **Don’t discount the soft side -- mutual visits promote stronger communication and relationship building.**
- **Seamless environments should include source control systems and continuous integration, automatic build and test suites.**
- **Go beyond offshore coders and invest in offshore analysis and design.**
- **Make the offshore team a key part of the iteration planning meetings and demonstrations.**
- **Flexible give and take on time zone differences, like staggering early morning and late evening meeting times.**

Source: Adopted from Martin Fowler ***Using an Agile Software Process with Offshore Development***

http://martinfowler.com/articles/agileOffshore.html