BITS Pilani

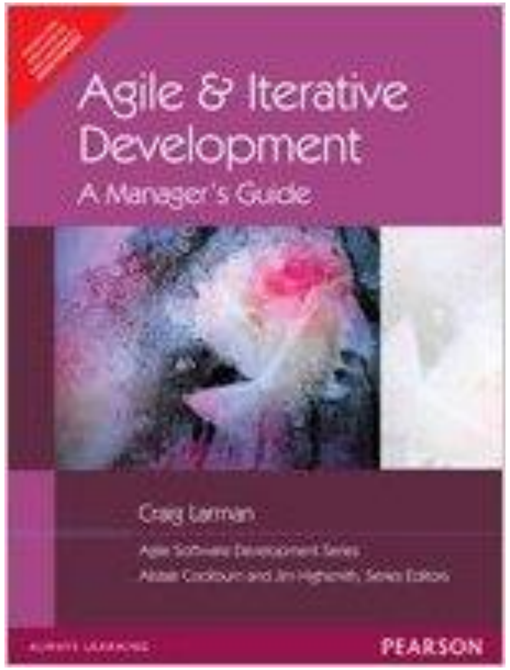
Pilani | Dubai | Goa | Hyderabad

Agile Pitfalls

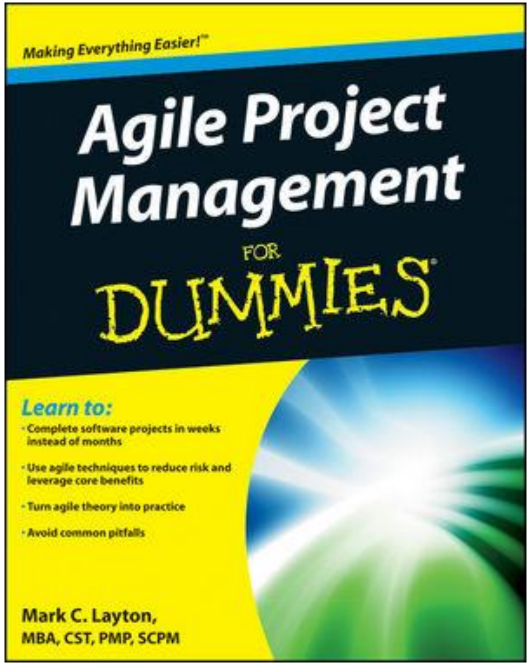
- Prof K G Krishna

Text/Reference Books

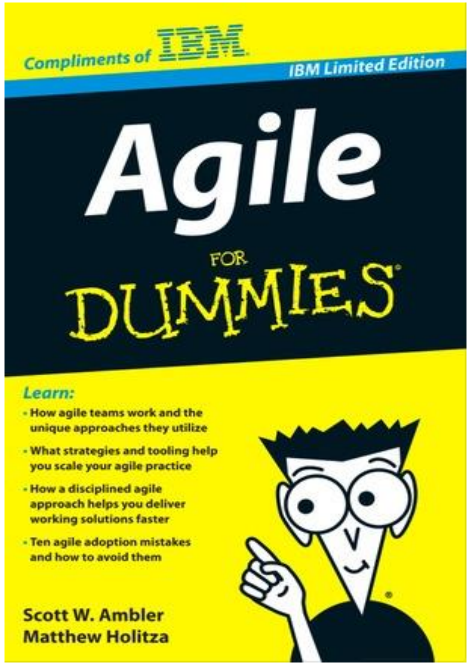
T1


Agile & Iterative Development
A Manager's Guide
Craig Larman
Agile Software Development Series
Alistair Cockburn and Jim Highsmith, Series Editors
PEARSON

T2


Making Everything Easier!
Agile Project Management FOR DUMMIES
Learn to:
Complete software projects in weeks instead of months
Use agile techniques to reduce risk and leverage core benefits
Turn agile theory into practice
Avoid common pitfalls
Mark C. Layton, MBA, CST, PMP, SCPM

Compliments of IBM

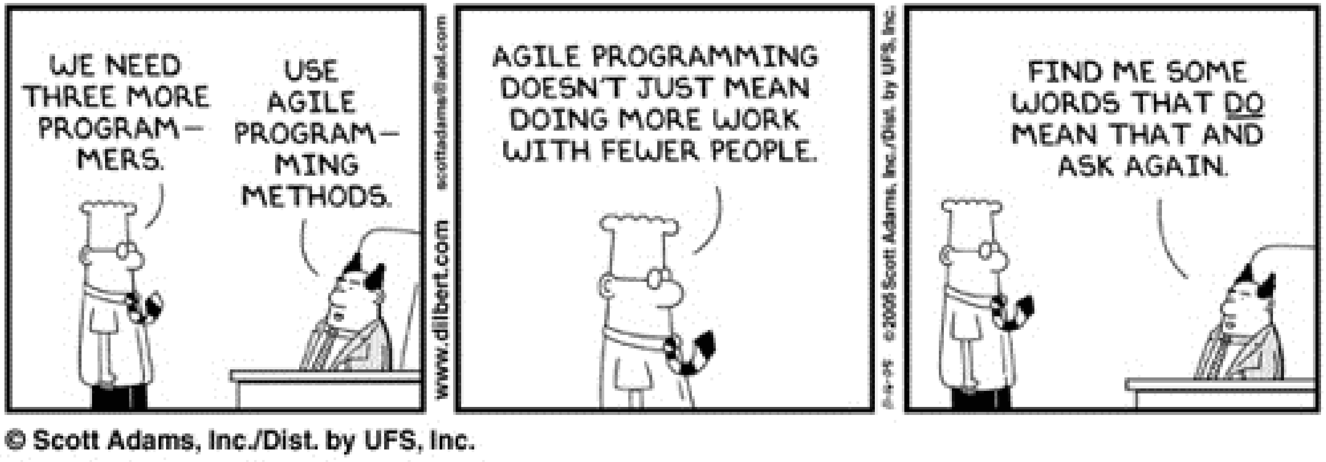

Compliments of IBM
IBM Limited Edition
Agile FOR DUMMIES
Learn:
How agile teams work and the unique approaches they utilize
What strategies and tooling help you scale your agile practice
How a disciplined agile approach helps you deliver working solutions faster
Ten agile adoption mistakes and how to avoid them
Scott W. Ambler
Matthew Holitza

➔ As this field is evolutionary, the student is advised to stay tuned to the current and emerging practices by referring to their own organization's documentation as well as Net sources

Topics

Agile Myths & Pitfalls

- Common Mistakes/Myths in Agile
- Predictive Planning vs. Adaptive Planning
- CMMI vs. Agile
- Distributed Agile


WE NEED THREE MORE PROGRAM-MERS.
USE AGILE PROGRAM-MING METHODS.
www.dilbert.com scottadams@aol.com
AGILE PROGRAMMING DOESN'T JUST MEAN DOING MORE WORK WITH FEWER PEOPLE.
© 2008 Scott Adams, Inc./Dist. by UFS, Inc.
FIND ME SOME WORDS THAT DO MEAN THAT AND ASK AGAIN.
© Scott Adams, Inc./Dist. by UFS, Inc.

Common mistakes/misunderstandings

- Agile Process is Adoption/Adaptation of ‘Waterfall-inspired’ *Predictable* Iterative and Incremental Process

  “…We have adopted iterative method for two months and finished use case analysis and plan and schedule of what we’ll be doing in each iteration. After review and approval of final requirements set and iteration schedule, we’ll start programming…”

- Scrum Masters/Managers Direct the Team
  - Scrum Master is a only a Facilitator of the Process, removes roadblocks
- No Daily-update of Sprint-backlog by Members
- New Work added to Iteration or Individual
  - Stability is required at least once an Iteration starts
- Owner is Not Involved or take Decisions--too Many ‘cooks’ involved
  - Scrum is Customer-driven and Product Owner represents Customer’s Requirements and hence defines/prioritizes Product-backlog
- Design/Diagramming is followed, Documentation is bad
  - Scrum is pragmatic in its choice of tools—simple and value-adding (“don’t use sword when nail is enough”)
- Too many Meetings, too often
  - Short (<15mins) Daily Meeting to sync-up on Issues (not problem-solving which happens in constant communication/collaboration among all team members)
- ıteration is Not complete and without Integration testing
  - Each Iteration is Fully Done—coded, unit-tested, integration-tested and documented; can be a release to customers (though after couple of iterations)

Common mistakes/misunderstandings *(contd.,)*

- Applying a subset of Practices that do not complement/compensate each other
  - e.g., Collective Code Ownership is not feasible with Testing, Continuous Integration and Coding Standards; minimal requirements documentation is not feasible without onsite customer; frequent refactoring doesn't work without testing, etc.
- Not Writing Unit Tests First
  - Writing the tests first influences how one conceives, clarifies, and simplifies the design; test-first has an interesting psychological quality of satisfaction: I write the test, and then I make it succeed; there is a feeling of accomplishment that sustains the practice
- No Customer-owned Tests (UAT)
- Minimal Refactoring
  - The XP avoidance of design thought before programming is meant to be compensated by a relatively large refactoring effort, such as 25% of total effort applied to refactoring
- Many Fine-grained Tasks (Story cards)
  - Most task cards should be in the one or two-day range of effort. Most in the "few hours" range creates unnecessary information management.
- Pairing with One Partner Too Long
  - XP pairing changes frequently, often in two days or less. Variation also helps spread the learning.
- Customer or manager is tracker
  - Programmers will feel awkward reporting slow progress.

Common mistakes/misunderstandings *(contd.,)*

- Not integrating the QA Team
  - Many organizations have a separate Quality Assurance team, used to having a completed system “thrown over the wall” to them.
- Post development design is wrong
  - XP isn't anti-documentation, but prefers programming if that's sufficient to succeed. XP can support the creation of design documentation to reflect the existing code base, once the program is complete
- In Pair programming, not willing to learn or explain
  - For successful pair programming, an attitude of openness to learning and of explaining yourself is required**.**
- Iterations are NOT Time-boxed
  - It is a misunderstanding to let the iteration length expand when it appears the goals can't be met within the original time frame--rather, the preferred strategy is usually to remove or simplify goals for the iteration and analyze why the estimates were off the mark
- Each iteration ends in a production release
  - The baselined software produced at the end of an iteration is an internal release rather than shippable code. It represents a subset of the final production release, which may only be ready after a dozen or more short iterations
- Predictive Planning
  - It is a misunderstanding to create, at the start of the project, a believable plan laying out exactly how many iterations there will be for a long project, their lengths, and what will occur in each. This is contrasted with the agile approach: adaptive planning.

Agile Myths in a Nutshell…

- Should customize the choice of practices without having first applied them all.
- “Doing XP” means to avoid the waterfall model and develop iteratively, or just to avoid documentation, or just to write some unit tests.
- Customers are not involved in the Planning Game, creating acceptance tests, or reviewing the iteration results.
- Create a plan laying out how many iterations there will be for the project, their lengths, and what will occur in each (*Predictive Planning*)

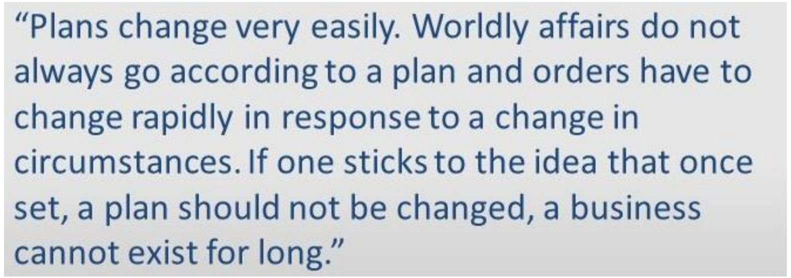
Agile Pitfalls/Challenges

- **Culture Change**: Top-down, big-bang culture change, adaptation, and institutionalization; traditional management methods clash with Agile concepts and practices
- **Fixed-price Contracting**? Not amenable to traditional fixed-price contracts for large agile projects which are complex
- **Resistance to Drop Established QA Practices**: Not willing to integrate or totally dissolve structures related to QA and Testing functions
- **Inadequate Management/Development Tools**
- **Fuzzy Interface** with Architecture, Quality, Security and Usability functions
- **Lack of Planning**: Non-standardization of Release and Iteration Planning
- **Lack of Trust/Teamwork**: Tendency to Micromanage, Organizational Politics, and Conflicts between Developers and Project Managers
- **Inadequate QA:** Inconsistent use of Agile testing, Usability, Security and other QA Practices; Inadequate testing to meet iteration time-box constraints
- **Lack of Expertise**: Inadequate Skills and Experience (SME and Coaches)
- **Multisite/Multiteams:** Working on complex R&D/Large Projects across Locations

Planning (Waterfall vs. Iterative vs. Agile) ➔➔

Predictive Planning vs. Adaptive Planning (Agile)

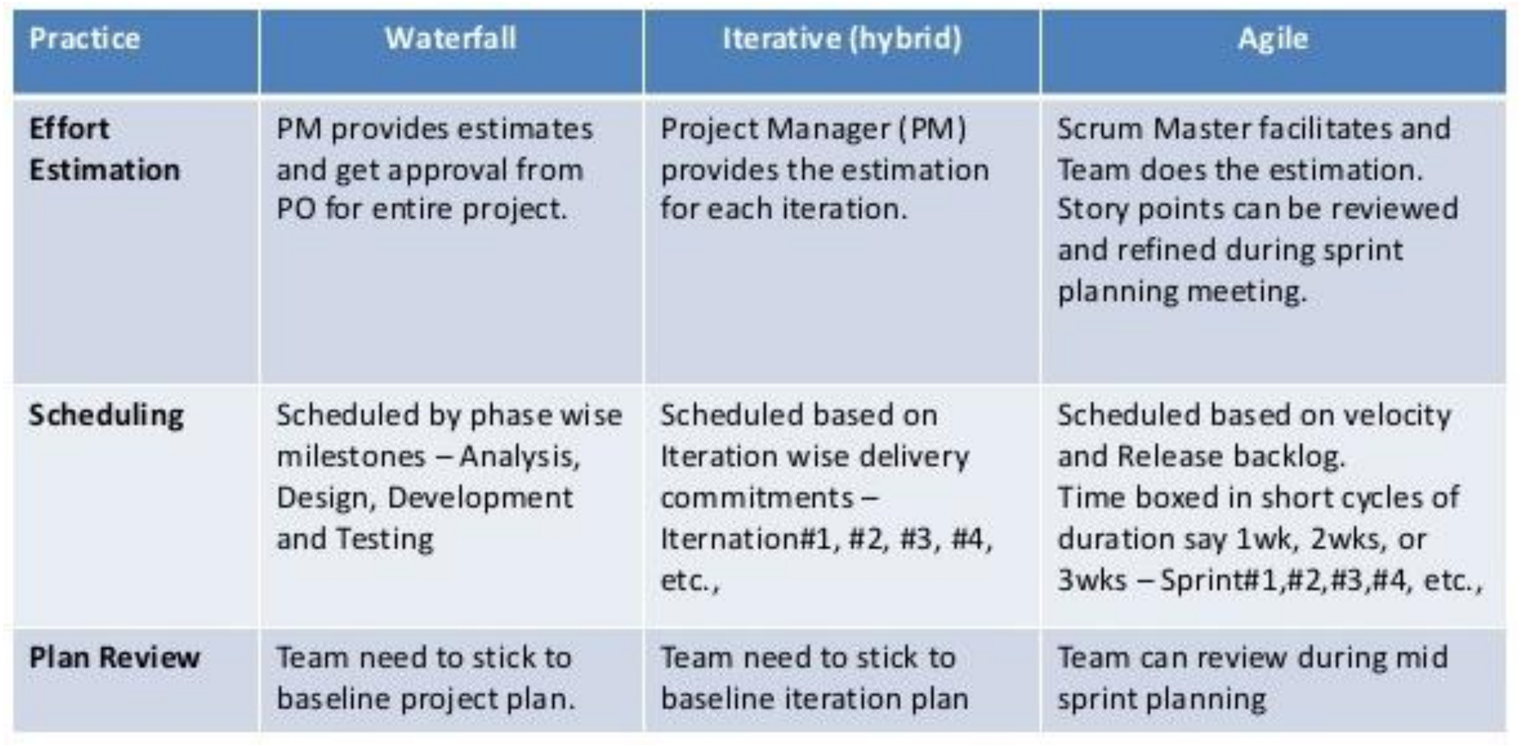
- Predictive Planning (flavor of ‘waterfall’)
  - Planning for few Large Milestones
  - Determining in advance Number of Iterations
  - Plan in Detail All The Iterations
- Adaptive Planning (flavor of ‘agile’)
  - Though Milestones can be fixed, but the Path towards Milestones is flexible
  - Milestones can themselves change later in the best interests of Project
  - Plan in Detail for the Just Next Iteration



“Plans change very easily. Worldly affairs do not always go according to a plan and orders have to change rapidly in response to a change in circumstances. If one sticks to the idea that once set, a plan should not be changed, a business cannot exist for long.”

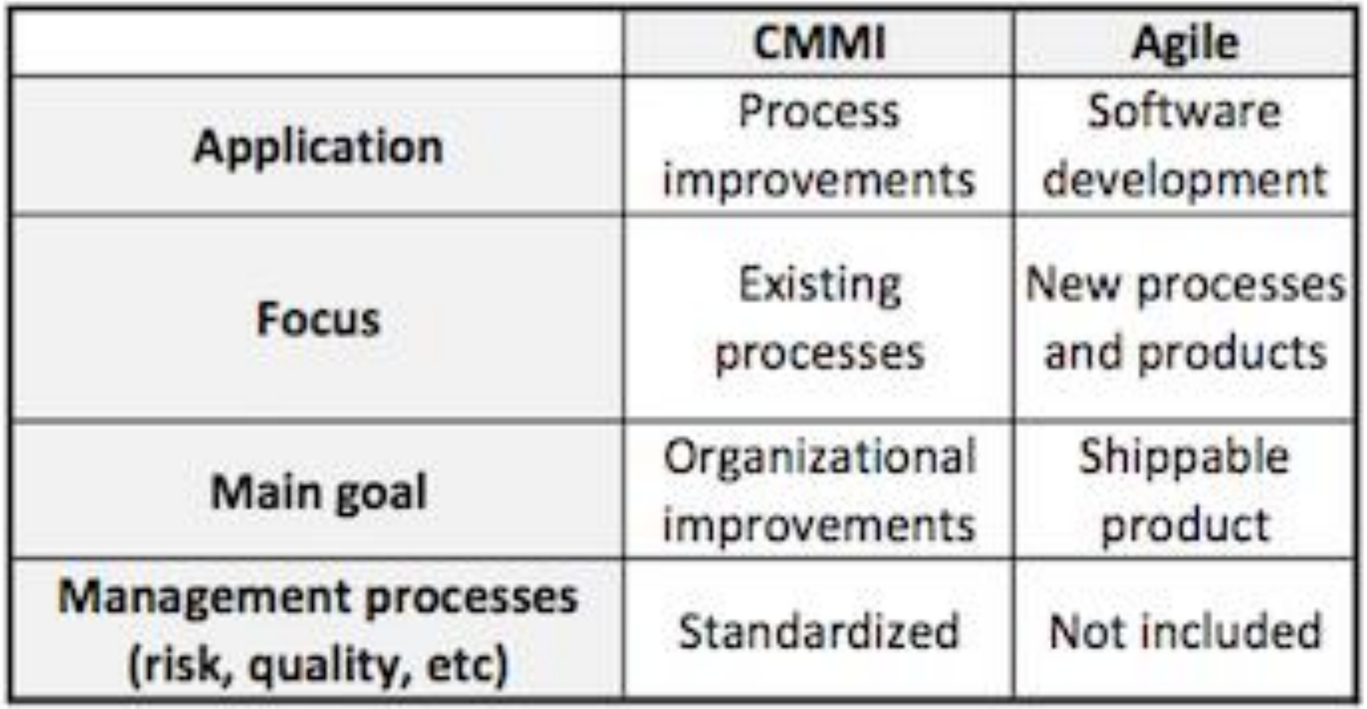
Taiichi Ono
Father of Lean (Toyota Production System)

Planning: Waterfall vs. Iterative (Predictive Planning) vs. Agile (Adaptive Planning)



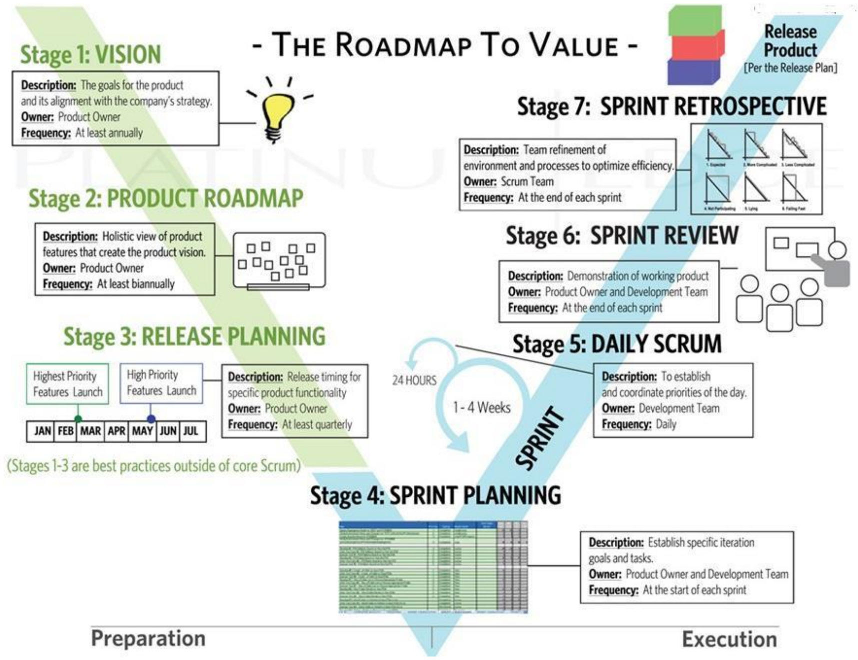
Practice	Waterfall	Iterative (hybrid)	Agile
Effort Estimation	PM provides estimates and get approval from PO for entire project.	Project Manager (PM) provides the estimation for each iteration.	Scrum Master facilitates and Team does the estimation. Story points can be reviewed and refined during sprint planning meeting.
Scheduling	Scheduled by phase wise milestones – Analysis, Design, Development and Testing	Scheduled based on Iteration wise delivery commitments – Iternation#1, #2, #3, #4, etc.,	Scheduled based on velocity and Release backlog. Time boxed in short cycles of duration say 1wk, 2wks, or 3wks – Sprint#1,#2,#3,#4, etc.,
Plan Review	Team need to stick to baseline project plan.	Team need to stick to baseline iteration plan	Team can review during mid sprint planning

The ‘Big’ Cultural Differences between CMMI and Agile

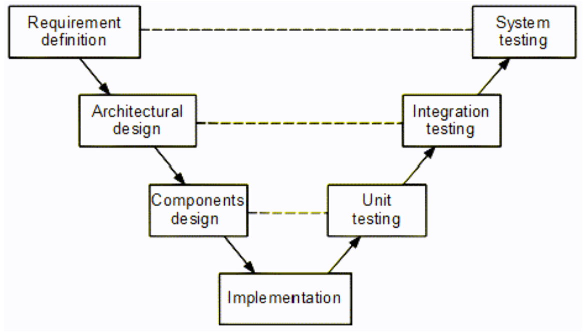


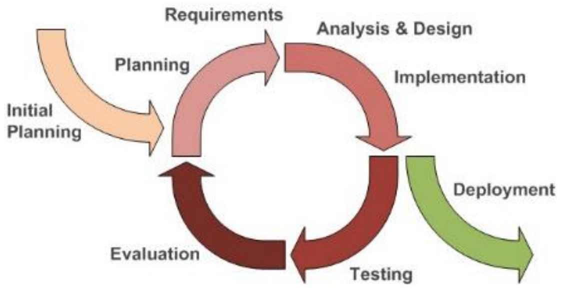
	CMMI	Agile
Application	Process improvements	Software development
Focus	Existing processes	New processes and products
Main goal	Organizational improvements	Shippable product
Management processes (risk, quality, etc)	Standardized	Not included

Agile is NOT an Iterative Adaptation of Waterfall (V-Process) Model!


- The Roadmap To Value -
Release Product
[Per the Release Plan]
Stage 1: VISION
Description: The goals for the product and its alignment with the company's strategy.
Owner: Product Owner
Frequency: At least annually
Stage 2: PRODUCT ROADMAP
Description: Holistic view of product features that create the product vision.
Owner: Product Owner
Frequency: At least biannually
Stage 3: RELEASE PLANNING
Highest Priority Features Launch
High Priority Features Launch
JAN FEB MAR APR MAY JUN JUL
Description: Release timing for specific product functionality
Owner: Product Owner
Frequency: At least quarterly
(Stages 1-3 are best practices outside of core Scrum)
Stage 4: SPRINT PLANNING
Description: Establish specific iteration goals and tasks.
Owner: Product Owner and Development Team
Frequency: At the start of each sprint
Stage 5: DAILY SCRUM
Description: To establish and coordinate priorities of the day.
Owner: Development Team
Frequency: Daily
24 HOURS
1 - 4 Weeks
SPRINT
Stage 6: SPRINT REVIEW
Description: Demonstration of working product
Owner: Product Owner and Development Team
Frequency: At the end of each sprint
Stage 7: SPRINT RETROSPECTIVE
Description: Team refinement of environment and processes to optimize efficiency.
Owner: Scrum Team
Frequency: At the end of each sprint
Preparation
Execution

$\neq$





Source: (T2)

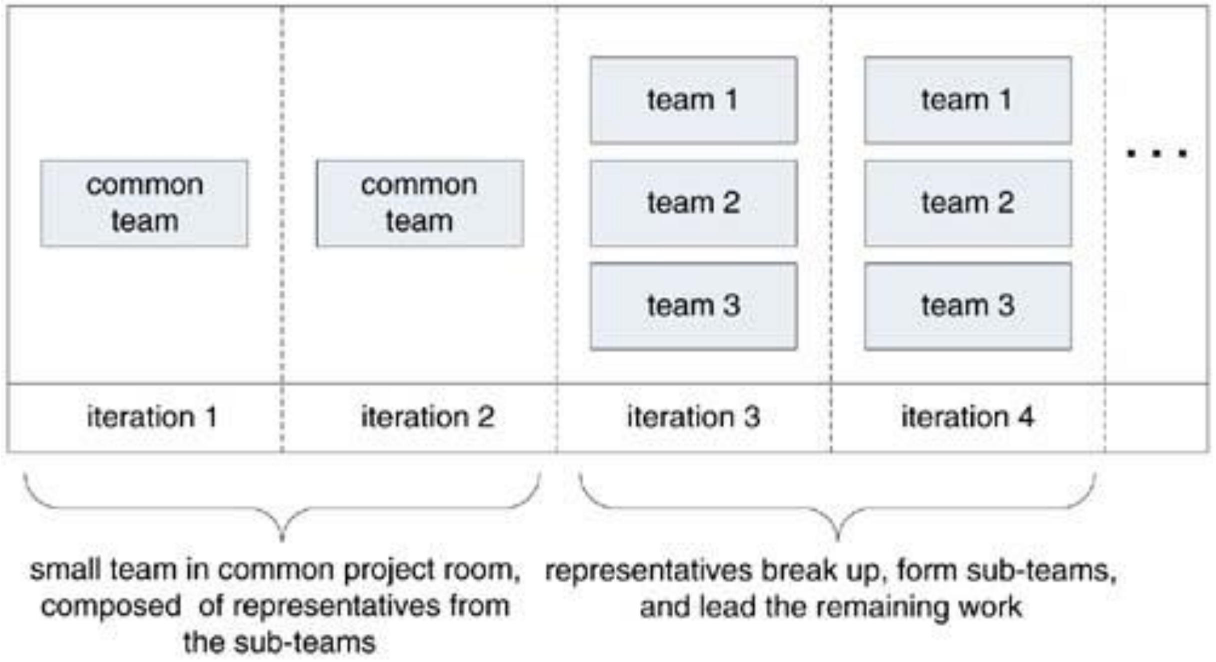
Distributed Agile (Multisite/Multiteam) ➔➔

Multisite / Multiteam Planning

- For projects that will be composed of multiple teams, spread across different locations, consider **doing the early iterations at one location**, in one common project room, with a small group ideally composed of one or two skilled representatives from each of the subteams.
- During these iterations, there is an **emphasis on requirements analysis** and development to discover and build the core architecture of the system—the foundation.
- Major components, and their collaborations and **interfaces are ideally clarified through early programming and testing** rather than just speculative design; the early project benefits from the close communication, collaboration, common vision, and technical strength of the initial group.
- Once the core is built, the representatives return to their respective locations, form larger teams, and the **remaining work is done in parallel with multiple subteams**.
- Each representative has developed a clearer picture of the vision and architecture, and can better convey and maintain that for the remainder of the project. Further, each acts as a liaison to the other teams. Also, after having spent some close time with the other subteam leaders, there is **improved communication between the subteams**.

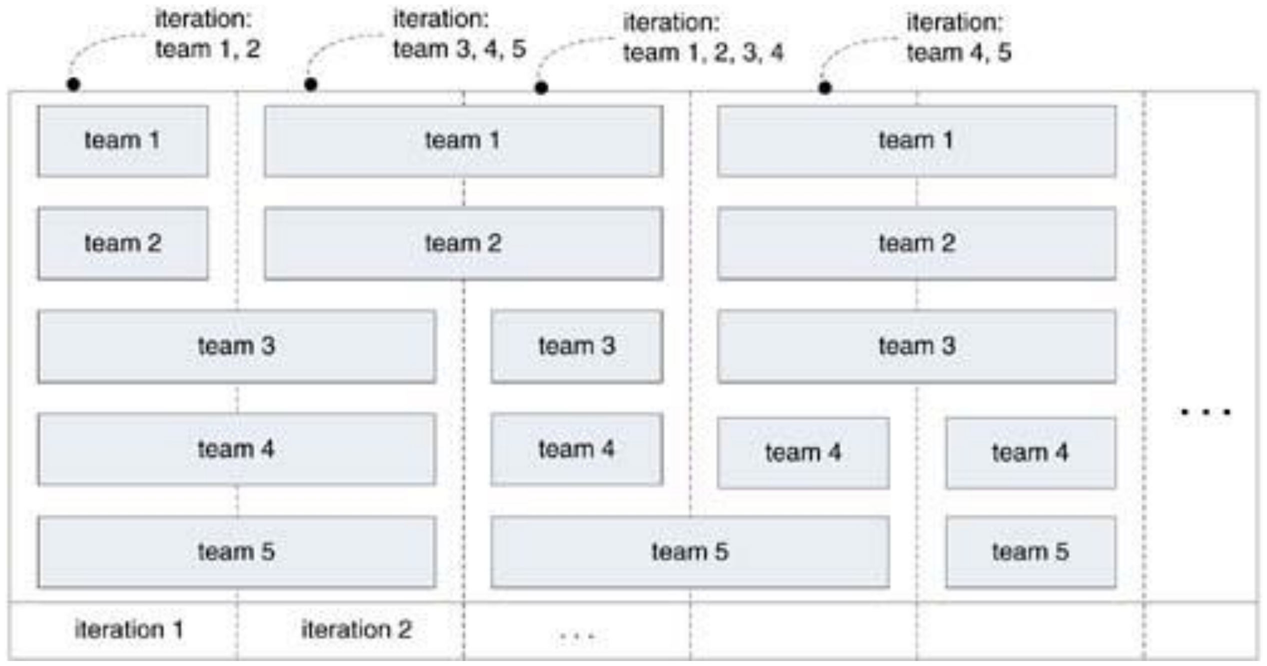
Multiteam Development

- Have all components, across all subteams or sites, integrated and tested together to conclude the release of a common iteration (to be relaxed in certain projects involving R&D, e.g. development of 5G protocol stack)


common team
common team
team 1
team 2
team 3
team 1
team 2
team 3
. . .
iteration 1
iteration 2
iteration 3
iteration 4
small team in common project room, composed of representatives from the sub-teams
representatives break up, form sub-teams, and lead the remaining work

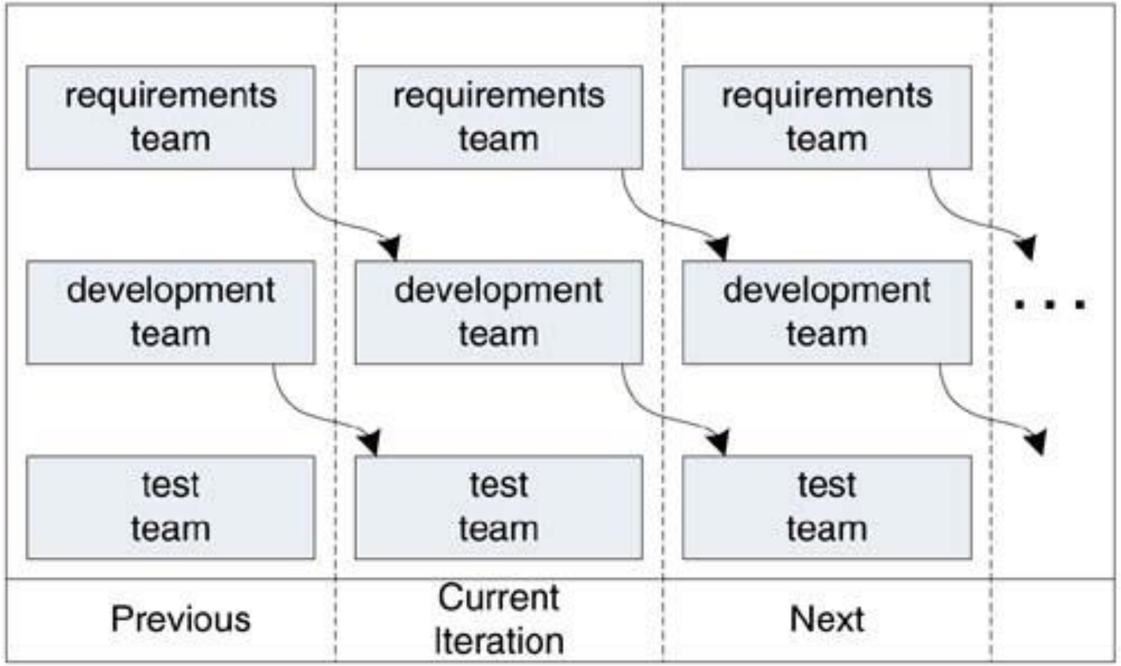
Subteam Iterations (for R&D Projects)

- Ideal for Multisite research-oriented project, the preferred guideline is that all teams work together towards a common goal and integrate on a common iteration end date



Pipelining Iterations (for large projects)

- More appropriate for larger projects, those with offsite requirements donors, or those with long, complex testing that must be performed by a separate expert team (e.g. complex testing which includes labour-intensive manual testing, security testing and memory-leak stress testing, etc. where the system needs to run for many hours or days to discover defects)



Summary: Agile Myths – Debunked / Major Challenges

- Agile is NOT an adoption of Spiral and other IID methods of 'Waterfalll era'
- There IS just enough Planning for meeting major milestones, but the path towards milestones can involve multiple (and unpredictable number of) iterations
- Agile does NOT avoid Design, but evolves Design with each Iteration
- Agile (Scrum, in particular) DOES support minimum (and agile) Process vs. (rigid and monolithic) process in Waterfall model
- Not EVERY Iteration should lead to Production Release
- Handling Fixed-price Contracts is a Challenge in Agile—but NOT impractical
- Culture change is the BIG Transformation effort required before going for Agile
- Multisite/Multiteam Projects too can be handled using Agile using appropriately planned and coordinated Iterations across locations

Thank You