Using Agile Measurements

Dottie Acton

Senior Fellow

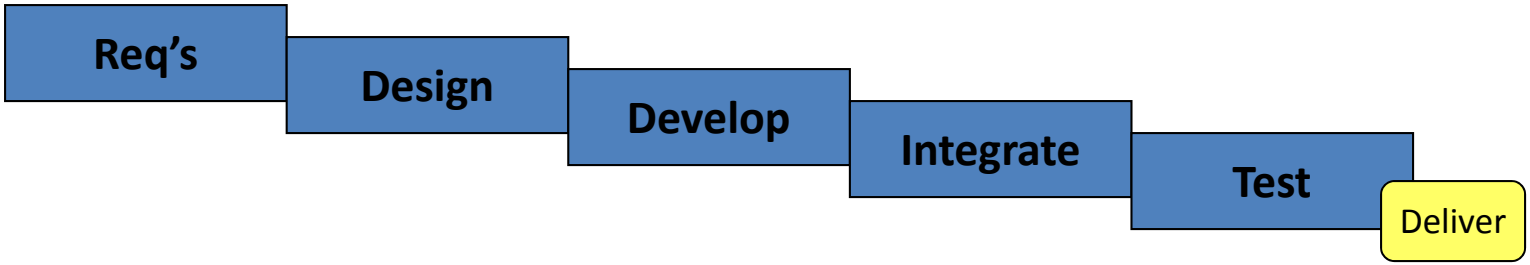
Lockheed Martin

LOCKHEED MARTIN

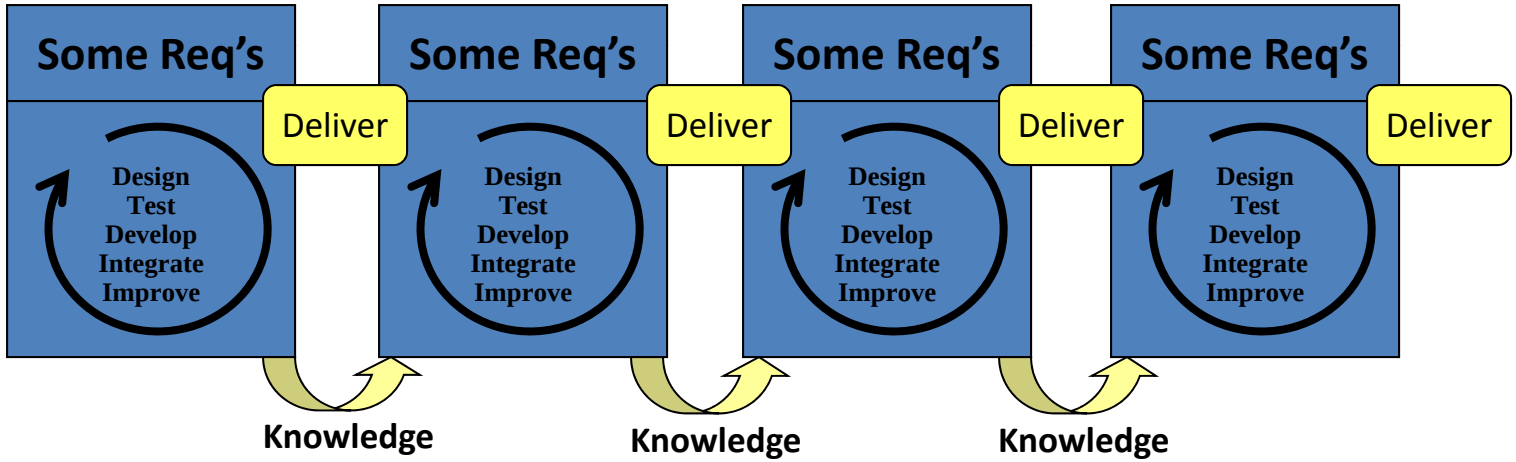
Topics

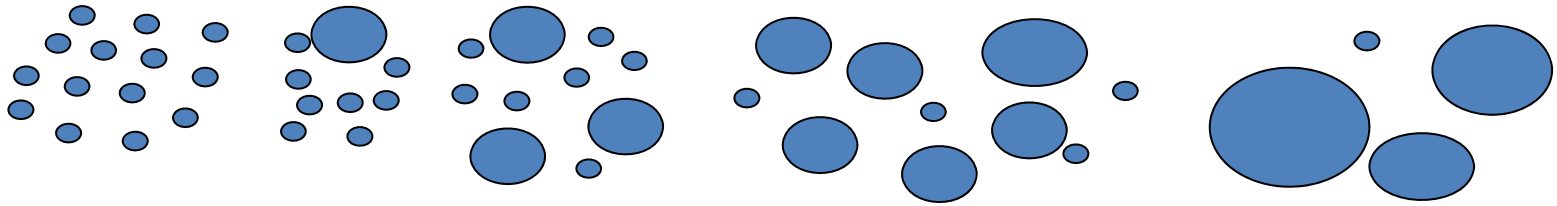
- A Quick Look at Agile
- Standard Agile Progress Measurements
- Measuring Quality
- Using Measurements at Retrospectives
- Reporting Agile Status
- Summary
- Questions

Traditional Development Approach



Agile Approach





Comparing Agile and Traditional Approaches

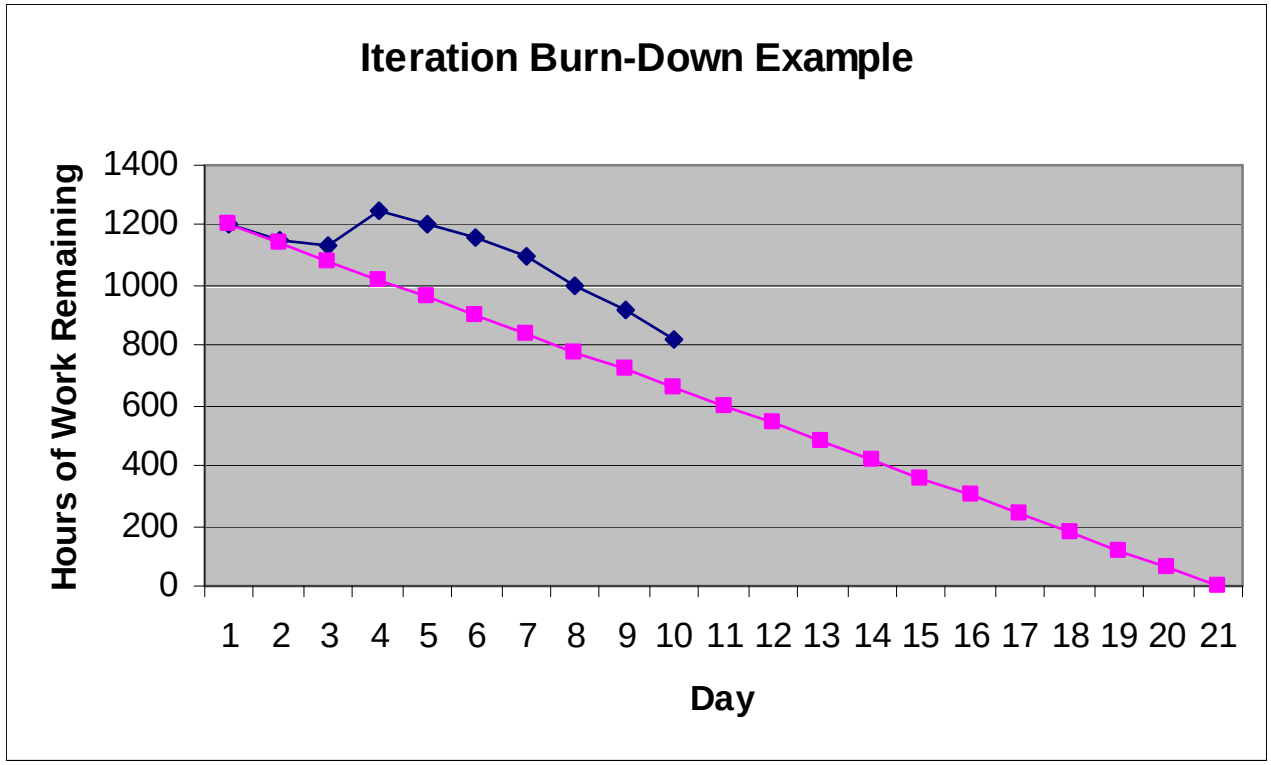
Agile programs characteristics make it challenging to fit into the traditional measurement profile

	Traditional	Agile
Development Cycle	Typical long, phase-based development cycle	Short, time-boxed Iterations with scheduled releases
Planning	Entire program planned in detail up-front; detailed requirements that define program scope are baselined early in the lifecycle	High level plans only up front; detailed plans for each iteration; product backlog, consisting of features, defines the initial scope; user stories add detail for each iteration
Estimation	SW estimates are SLOC based	SW estimates are based on story points – a unit-less relative size
Tracking	Fixed Scope – track to planned schedule	Fixed Schedule – track to planned scope
Change Management	Scope changes are discouraged	Scope changes are welcomed and current scope can be *negotiated-out* based on customer priorities

Measuring Iteration Progress

- At the start of an iteration, each team plans its work for the iteration
  - Selects highest priority stories from the product backlog
  - Defines the tasks needed to complete those stories
    - Size of task is measured in ideal hours
    - Goal is to define tasks in the 2 hour to 2 day range
  - Daily, each team tracks its progress in completing the tasks
  - Stories are not considered complete until all the associated tasks are complete

Agile Iteration Tracking

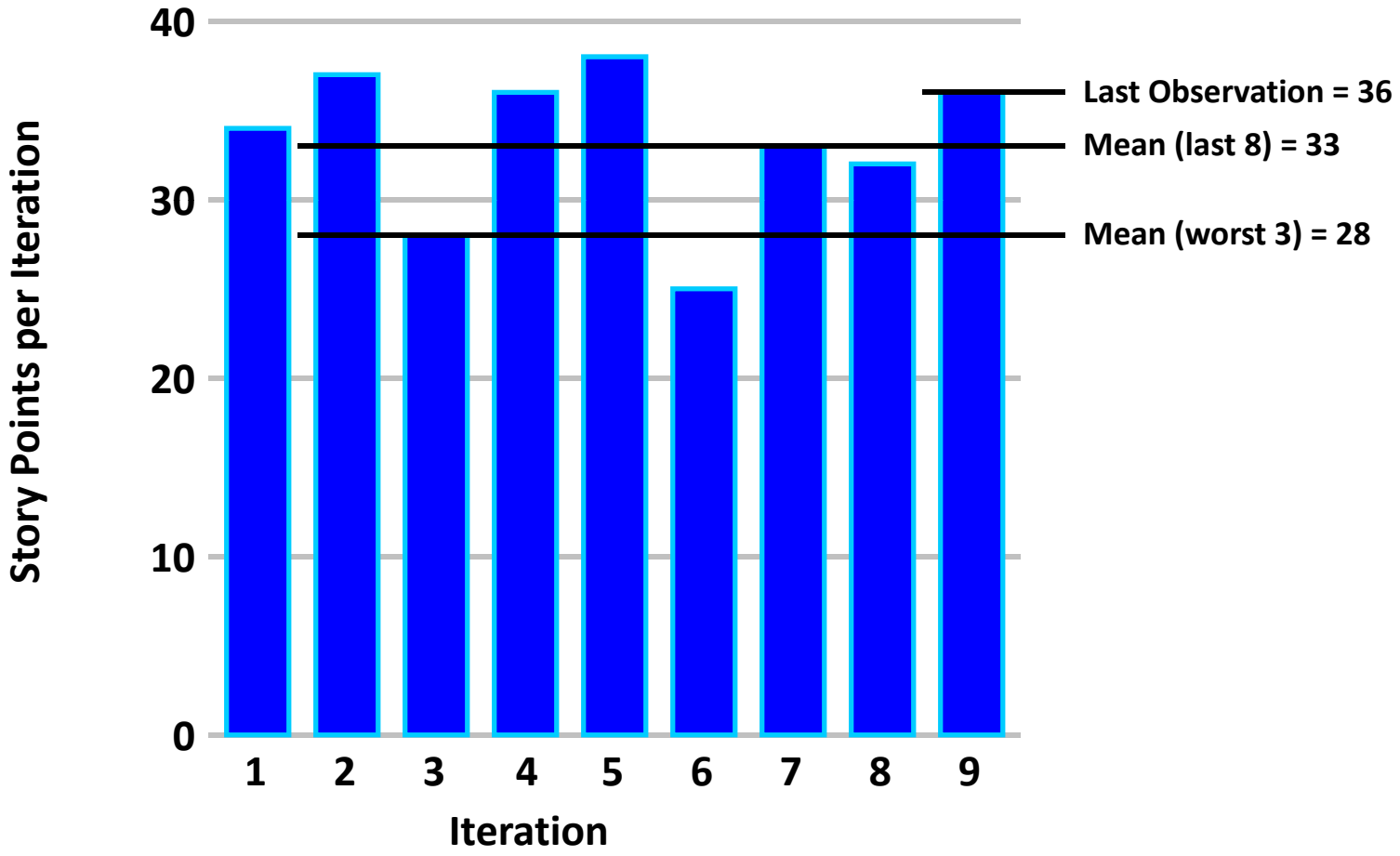


Tracking at this level is a team responsibility, not a management task.

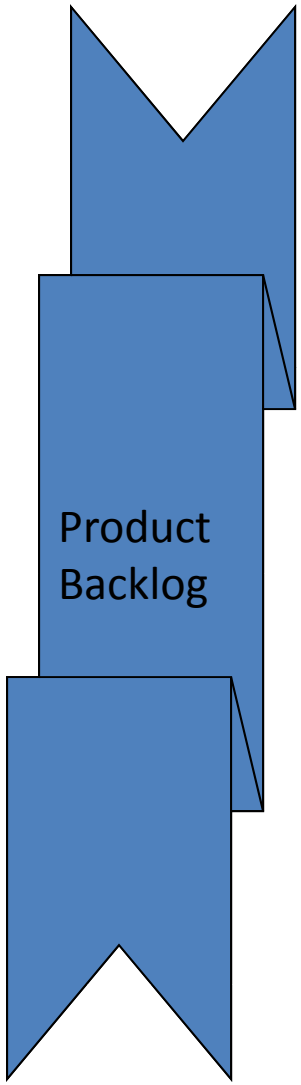
Measuring Release Progress

- Measure progress toward completing the work in the product backlog at the end of each iteration
  - Base measure is number of story points completed
    - Story point is a unit-less relative measure
    - Usually based on Fibonacci numbers or powers of 2
  - Use a burn-down chart similar to iteration burn-down chart, but measure story points at iteration boundaries
- Velocity calculations can be used to predict release completion ranges
  - Velocity = story points per iteration

Calculating Velocity



A Simple Predictive Model



Calculate a range of options for the amount of work that can be completed in the remaining iterations

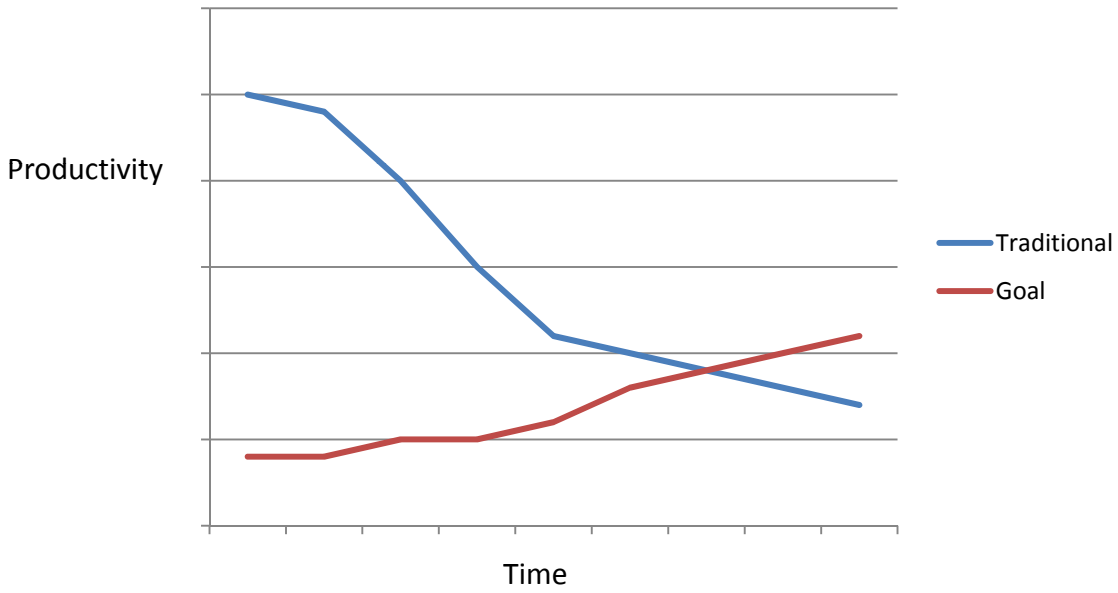
Amount complete at slowest velocity (average of the lowest 3)

Amount complete at long-term average velocity (average of the last 8)

Amount complete at current velocity

What's the Magic Behind Agile

Any time you touch the code, make it better – or at least no worse



Change the economics of change

For Agile to be effective, it is essential to focus on software quality.

Measuring Code Quality

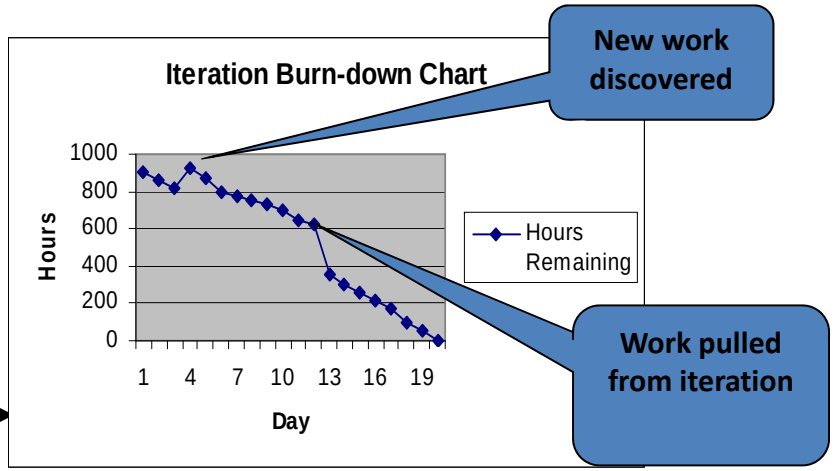
- Identify measures that will indicate how hard it will be to change the code
  - Cyclomatic Complexity
    - Complex modules are harder to change
  - Afferent and Efferent Coupling (Ca and Ce)
    - High incoming and outgoing dependencies impede change
  - Abstractness (A)
    - Modules with many abstractions are more open to change
- Why this matters more with Agile
  - After the first iteration, you will be changing existing code
  - Cannot maintain velocity with hard to change code
- How to automate
  - Add standards checkers to the build process
  - Reject modules that fail the checks

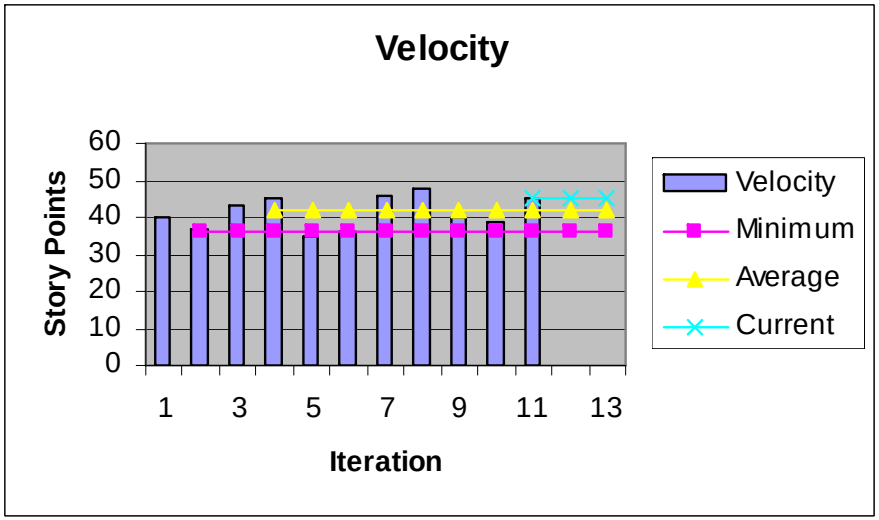
Measuring Test Coverage

- Coverage can be measured each iteration
  - Early indicator of problems if coverage goals are not being met
    - Potential schedule slip due to late discovery of defects
  - Trends in test coverage can be early indicator of process problems
    - Less testing each iteration is not a good trend
- If desired coverage is not being achieved, can add tasks to enhance test coverage
  - Prioritize additional testing against more features
- Measuring code coverage is essential with Agile
  - Cannot make changes effectively without a safety net of tests
- How to automate
  - Use test coverage tools when running automated tests associated with each build

Using Measurements at Retrospectives

Things that worked well	Things we should change
	Things we should stop doing





Reporting Agile Measurements

- Most managers don't speak *Agilese* and they don't see through Agile colored glasses

Agile Language
Geek

Enterprise Language
English

Must Translate from Geek → English ! !

Some Translations

- Iteration and product backlog burn-down charts can translate easily to plan vs. actual
- Data for EVM calculations can be derived from iteration task data
- Teams still need to track their risks and roll up risks to the program level as appropriate

- Techniques
  - Start by presenting the traditional measurements, followed by the agile equivalents
  - Over time, transition to presenting the agile measurements first, followed by the traditional measurements
  - Evolve to focusing only on the agile measurements, with traditional measurements as backup data

Summary

- Measurements are used on Agile programs for the same purpose they are used on other programs – to provide data for making good decisions
- Automation is essential
  - Fortunately Agile project management tools like Rally or VersionOne automate the generation of burn-down charts
  - It is easy to include quality and code coverage measurements as part of the nightly builds