Agile Software Engineering: Interactive Workshop

IEEE Software Technology Conference
Long Beach, CA
March 31, 2014

Steven Baynes
Agile Engineering
Northrop Grumman
Steven.Baynes@ngc.com
Agenda

- Iteration 1 – Introduction and Agile Overview
- Iteration 2 – Team Structure
- Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
- Iteration 4 – Writing Product Backlog Items
- Iteration 5 – Estimating PBIs
- Iteration 6 – Sprint Planning
- Iteration 7 – Sprint Execution
- Iteration 8 – Sprint Review, Retrospective, and Delivery
- Iteration 9 – Summary
Agenda

• **Iteration 1 – Introduction and Agile Overview**
  – Getting to Know You, Training Outcomes and Expectations
  – Defining an Agile Environment and Review of Agile Methods
• **Iteration 2 – Team Structure**
• **Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning**
• **Iteration 4 – Writing Product Backlog Items**
• **Iteration 5 – Estimating PBIs**
• **Iteration 6 – Sprint Planning**
• **Iteration 7 – Sprint Execution**
• **Iteration 8 – Sprint Review, Retrospective, and Delivery**
• **Iteration 9 – Summary**
Getting To Know You!

Mr. Steve Baynes

- Product Group manager for Northrop Grumman’s commercial-off-the-shelf software products -- e.POWER® and InFlowSuite™
- Certified Scrum Professional, member of the Scrum Alliance and Agile Alliance organizations, and speaks often on Agile development
- About 9 years ago, Steve and his team got on board with the Agile Manifesto and Agile Principles
- Leveraging Agile principles and practices, Steve works with business development, project implementation teams and customers to continually improve e.POWER/InFlowSuite from both a feature and quality perspective
Today’s Outcomes

• Develop an understanding of the major agile engineering practices
• Identify the major components of the Scrum framework
• Gain insight into Agile testing principles
• Participate in a team retrospective
• Have fun working together!
Expectations

What are your expectations?

What questions do you have?

Time: 5 minutes
What is your level of experience?

Scale of 1 to 5
Agenda

• Iteration 1 – Introduction and Agile Overview
  – Getting to Know You, Training Outcomes and Expectations
  – Defining an Agile Environment and Review of Agile Methods

• Iteration 2 – Team Structure

• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning

• Iteration 4 – Writing Product Backlog Items

• Iteration 5 – Estimating PBIs

• Iteration 6 – Sprint Planning

• Iteration 7 – Sprint Execution

• Iteration 8 – Sprint Review, Retrospective, and Delivery

• Iteration 9 – Summary
Why Agile?
Global Project Failures

- Challenged and unsuccessful projects hover at 67%; big projects fail more often.
- Traditional projects specify too many requirements; More than 64% of features are never or rarely used.

![Features Pie Chart]

Agile Engineering

- **An empirical process**: control is through frequent inspection and adaptation
- Transparent, Inspect, Adapt
- Focuses on the value stream
- Delivers value to the customer through frequent and short iterations
- Incremental release of capabilities; each iteration (sprint) is potentially shippable
- Do a little bit of everything every iteration (sprint)
  - Plan, Test, Design, Build
Waterfall v Agile

Waterfall
- Plan
- Design
- Code
- Test
- Release

Agile
- Release
- Release
- Release
- Release
- Release

Visibility
- Time

Ability to Change
- Time

Business Value
- Time

Risk
- Time

Waterfall

Agile
Is Agile Better?

“The agile process is the universal remedy for software development project failure. Software applications developed through the agile process have three times the success rate of the traditional waterfall method and a much lower percentage of time and cost overruns. The secret is the trial and error and delivery of the iterative process.”

Source: The CHAOS Manifesto, Copyright 2011
Recognizing the Need for Change

• Rate of New Technology Adoption
  – 38 years for radio to attract 50 million listeners
  – 13 years for television to attract 50 million viewers.
  – 4 years for the Internet to attract 50 million surfers! (2000. Gabay. Successful Cybermarketing in a Week)
  – iPad sold 30 million within the first 80 days; 45 million in the first quarter

• NDAA 2010 signed by President Obama in October 2009
  – Agile DoD IT Guidance

• Terri Takai, DoD CIO (2011)
  – "Me saying I want to do agile and yet the large integrators not having agile developers is a challenge," she said. "And I'm sure that for the large integrators, it's a question of when we're going to move to agile and when are we going to move away from the laborious acquisition practices we've had in the past. It's not just a culture change, it's not just training for us, it's making sure we can convince our partners that we are truly going there and getting them to meet up." (5/27/2011. Federal News Radio)

• Stephen Welby, Deputy Assistant Secretary of Defense for Systems Engineering, AFEI Agile for Government Summit, November 21, 2013
The U.S. Government Accountability Office report identified 14 challenges and 10 effective practices.
Projects that Benefit Most from Agile

• When there are vague, uncertain requirements, or

• Where budget and/or schedule is fixed but the functionality can be reprioritized, or

• Where there are risks and unknowns such as unfamiliar technologies or need to place legacy systems, or

• Need for more transparency, or

• Desire for team ownership and accountability
### Agile Manifesto and Principles

**Manifesto**

- **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.

**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.
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.
7. Working software is the primary measure of progress.
8. Agile processes promote sustainable development.
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.
Agile is about how we believe people are best motivated to do work and about demonstrating high value on a regular basis particularly in environments that face high requirement volatility and unpredictability.
Agile Trends and Adoption

State of Agile
Agile Methods & Practices

AGILE METHODOLOGY USED
Once again Scrum and Scrum variants (73%) remain the most popular agile methodologies being used.

DSDM/Atelen: 1%
Agile Modeling: 1%
Agile Unified Process (AgileUP): 1%
XP: 2%
Other: 2%
Feature-Driven Development: 3%
Lean: 5%
Kanban: 7%
ScrumBan: 10%
Custom Hybrid: 11%
Scrum/XP Hybrid: 55%
What is “Scrum”?

“This new emphasis on speed and flexibility calls for a different approach for managing new product development. The traditional sequential or “relay race” approach to product development….may conflict with the goals of maximum speed and flexibility. Instead, a holistic or “rugby” approach –where a team tries to go the distance as a unit, passing the ball back and forth – may better serve today’s competitive requirements….”

The Scrum Framework

Capabilities and User Stories
Prioritized by Product Owners
Creates the release plan

The Daily Tasks managed by the Cross Functional Team

Design, Code, Integrate, Test

Commitment
Identification of Impediments
Communication

Release Plan

Product Backlog

Sprint Backlog

System Architecture
System Design Requirements

Inspect and Adapt
Visibility and Transparency

24 Hours
Daily Scrum Meeting

1-4 Week Sprint

Sprint Review and Retrospective

Potentially Shippable Product Increment
Scrum and Extreme Programming Core Practices

Scrum
- 1 - 4 week Sprints
- Self-directed and self-organizing teams (6 +/- 3)
- Cross functional teams
- Daily Scrum meetings
- Product Backlog
- Sprint planning
- Don’t change the Sprint goal
- Scrum Master
- Time boxing
- Removing impediments
- No specific engineering practices defined
- Product Burndown
- Sprint Review
- Sprint Retrospective

Extreme Programming
- Planning game
- Small, frequent releases
- System metaphors
- Simple design
- Testing
- Frequent refactoring
- Pair programming
- Team code ownership
- Continuous integration
- Sustainable pace
- Whole team together
- Coding standards
Agile Keywords & Phrases

- Each Project or Product consists of one or more Releases
  - Each Release consists of one or more Sprints
    - Each Sprint consists of one or more Product Backlog Items (PBIs)
      - Each PBI is implemented by Tasks

**Deliver working functionality every Sprint**
Agenda

• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure
  – Creating the Agile Project Team
• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
• Iteration 4 – Writing Product Backlog Items
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
• Iteration 8 – Sprint Review, Retrospective, and Delivery
• Iteration 9 – Summary
# Roles and Responsibilities

<table>
<thead>
<tr>
<th>Role</th>
<th>Specific Role</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Chief Engineer/Architect</strong></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Ensures the integrity of the architecture</td>
</tr>
<tr>
<td></td>
<td>Communicates the systems design</td>
</tr>
<tr>
<td></td>
<td>Has no predefined team</td>
</tr>
<tr>
<td></td>
<td>Adds to the Product Backlog</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Product Owner</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Defines the items in the Product Backlog</td>
</tr>
<tr>
<td></td>
<td>Manages project features and release to optimize return on investment (ROI)</td>
</tr>
<tr>
<td></td>
<td>Prioritizes features according to user and stakeholder needs</td>
</tr>
<tr>
<td></td>
<td>Inspects increment and makes adaptations to Product Backlog</td>
</tr>
<tr>
<td></td>
<td>Can change features and priority every sprint</td>
</tr>
<tr>
<td></td>
<td>Communicates project progress and status</td>
</tr>
<tr>
<td></td>
<td>Accepts or rejects work from the Development Team</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Development Team</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Cross-functional, six plus/minus three members</td>
<td></td>
</tr>
<tr>
<td>Selects the sprint goal and specifies work results</td>
<td></td>
</tr>
<tr>
<td>Commits to what it feels it can accomplish</td>
<td></td>
</tr>
<tr>
<td>Has authority to do everything within existing standards and guidelines to reach the sprint goal</td>
<td></td>
</tr>
<tr>
<td>Manages itself and its work</td>
<td></td>
</tr>
<tr>
<td>Collaborates with Product Owner to optimize value</td>
<td></td>
</tr>
<tr>
<td>Demonstrates Sprint results to the Product Owner</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Scrum Master</strong></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Ensures that the team is fully functional, productive and improves quality</td>
<td></td>
</tr>
<tr>
<td>Enables close cooperation across all roles and functions and removes barriers</td>
<td></td>
</tr>
<tr>
<td>Shields the team from external interferences</td>
<td></td>
</tr>
<tr>
<td>Ensures that the process is followed</td>
<td></td>
</tr>
<tr>
<td>Teaches Product Owner and Team how to fulfill their roles</td>
<td></td>
</tr>
<tr>
<td>Does not make decisions for the team</td>
<td></td>
</tr>
</tbody>
</table>
Isolated progress with too many hand-offs and barriers
Agile Project Team Structure (An Example)

Supports Cross Functional Teams

- Push accountability and ownership to the team level
- Everyone trained
Cross Functional Teams (Development Teams)
Scrum Master Responsibilities

- Ensures Scrum is understood and enacted
- Facilitates Scrum events as needed or requested
- Help everyone adhere to Scrum’s theory, practices, and rules
- Servant leader for the Scrum Team
- Cause change that improves quality or productivity
- Embody agility to the organization
Agenda

• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure
• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
  – Defining the Business Strategy & Product Roadmap
  – Project Startup
  – Release Planning: What we are going to get done over the next several months?
• Iteration 4 – Writing Product Backlog Items
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
• Iteration 8 – Sprint Review, Retrospective, and Delivery
• Iteration 9 – Summary
High Level Agile Stages

- Vision
- Customer Needs
- Product Roadmap
- Release Planning
- Sprint Planning
- Sprint Execution
- Sprint Review and Retrospective
- Delivery
- Product Backlog

High Level Agile Stages

30
5 Levels of Planning

1. **Vision**
   - Capabilities identified
   - Some initial analysis

2. **Product Roadmap**
   - Inflows from Vision and Customer Needs
   - Outflows to Product Backlog

3. **Product Backlog**
   - Goals & PBIs
   - Inflows from Product Roadmap
   - Outflows to Release 1 (Planning)

4. **Release 1 (Planning)**
   - 1 or more Sprints
   - Outflows to Sprint 1, Sprint 2, Sprint 3, etc.
   - PBIs: Task, Task, Task

5. **Sprint 1 (Planning)**
   - ~ 1 - 4 weeks (fixed)
   - Inflows from Product Backlog
   - Outflows to Daily Stand-Up (5)
Product Roadmap Workshop

Product Roadmap provides the vision usually 6-12 months

INPUT

- Business/Mission Objectives
- Technology
- Vision

Defining the Product Roadmap

OUTPUT

- High-Level product capabilities mapped to several releases
- Priorities
- People & Resources
- Product Backlog updated

Product Roadmap Workshop
Today’s Scenario: RestEZ

Online hotel reservation system for RestEZ

• Based on customer needs, your team has defined a logical architecture for the online hotel reservation system.

• The system is a traditional 3 tier architecture:
  – a database layer (to persist reservations),
  – a business logic layer (to manage reservations),
  – and a browser-based user interface (to receive customer input).
Product: Hotel Website for RestEZ

**Product Roadmap**

**Release 1**
- March 31
  - Room reservations and payment
  - User profiles for future visits
  - Hotel amenities

**Release 2**
- June 30
  - Conference offerings
  - Online chat support
  - Local information

**Release 3**
- Sept. 30
  - Special discounts for room reservations
  - Improve usability
  - Google maps

**Release 4**
- Dec. 31
  - Air and hotel package deal
  - Meeting and Business plans and reservations
Release 0: Project Start up

• Start building the team.
  – Begin with at least one or two senior developers, the Scrum Master and Product Owner and one or more stakeholder representatives.
  – Training

• Create an initial architecture for the system.
  – You need to have at least a general idea of how you're going to build the system.
  – Identify an architectural strategy. Work through the design details later during future Sprints in model sessions.
  – Every Sprint must deliver at least some piece of business functionality

• Setting up the environment.
  – You need workstations, development tools, and work areas. Start with just enough to get the team going and continue to build on this in future releases.

• Determine first release date and Sprint length

• Create a transition backlog
• Reflecting on the section *Creating the Team* address the following.

• **Discussion**: Whose responsibility?
  – The Product Owner is micromanaging the team making self-managing impossible.
  – The team is struggling to understand the priorities of the work.
  – A team member is constantly late for the daily standup.
  – The team is not able to deliver on their commitments.

Time: 5 minutes
Agenda

• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure

• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
  – Defining the Business Strategy & Product Roadmap
  – Project Startup
  – Release Planning: What we are going to get done over the next several months?

• Iteration 4 – Writing Product Backlog Items
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
• Iteration 8 – Sprint Review, Retrospective, and Delivery
• Iteration 9 – Summary
High Level Agile Stages

- Vision Customer Needs
- Product Roadmap
- Release Planning
- Sprint Planning
- Sprint Execution
- Sprint Review and Retrospective
- Delivery
- Product Backlog

High Level Agile Stages
Agile Iterations

The length of a Release may vary.

The length of a Sprint remains constant.
The Product Backlog

- A list of all desired work on the project
- Expressed such that each item has value to the users or customers of the product
- Prioritized by the product owner

Vacationer searches for room availability...
Vacationer changes existing reservation...
Vacationer cancels existing reservation...
Vacationer reserves room with a credit card...

Ordered by the Product Owner
How do we create the Product Backlog?

- Product Owner works with the Development Team to begin drafting Product Backlog Items
  - User Stories, features (threads), use cases
  - May also include “acceptance criteria,” notes, comments, other explanatory information

- Workshops for generating the product backlog (PBL)
  - To clarify requirements and generate or review use cases, scenarios, user stories, etc.
  - Use Cases, Scenarios, Workflows

- PBL may contain hierarchy
  - Breaking down big items into smaller ones
  - Epics -> User Stories

- The PBL is ordered so the most important items are worked first
Use Case Flow: Make Room Reservations

<table>
<thead>
<tr>
<th>PBI</th>
<th>Effort Estimates</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vacationer searches for room availability...</td>
<td></td>
</tr>
</tbody>
</table>
  - Search by date  
  - Search by room type | 5 |
| Vacationer saves search request... |  
  - Save a request with a name  
  - Recall a request by name | 8 |
| Vacationer reserved room with a credit card... |  
  - Reserve with Visa  
  - Reserve with MC  
  - Do NOT reserve with AmEx | 13 |

The Release Plan
What is our velocity?  
How many points can we take on for this release?  
What can we commit to?

Ordered according to Business Value

Owned by the Product Owner with the opportunity to reprioritize each Sprint
Velocity (Based on history)

- Velocity is the amount of work a development team completes in a Sprint (points completed).
- Velocity is a range; Look for the high, the low, and the mean.

**Team A Velocity**
- High: 45 story points
- Low: 30 story points
- Mean: 37 story points

**Project Velocity**
- High: 155 story points
- Low: 120 story points
- Mean: 137 story points
Release Planning Meeting

- Meeting is time-boxed
- Usually ½ - 1 day
- Occurs with the entire project team
- Final review before Release Development
Commit to the Release Plan

- Capabilities/Goals identified
- High level requirements (CRs) and initial PBIs mapped and reviewed by stakeholders
- PBIs (functional and non-functional requirements) captured and estimated
- Known or forecast velocity by development team and project team
  - Ex.: Project Teams achieves 120 to 150 points per Sprint; for a release with 6 Sprints this is about 800 points. The scope is 720-900 points of work.
- Dependencies identified
- Total number of PBIs planned
- Planned hours / staff (WBS element)
Agenda

• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure
• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
• Iteration 4 – Writing Product Backlog Items
  – Three critical aspects: Card, Conversation, and Confirmation
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
• Iteration 8 – Sprint Review, Retrospective, and Delivery
• Iteration 9 – Summary
Product Backlog Items (PBIs)

**Typically include**

- Description
- Acceptance criteria (not test procedures)
- Estimated Value
- Estimate Cost (effort)

**Format, syntax**

- Whatever format works for your project!
- Transparent and appropriate for use
- Common techniques:
  - User Stories
  - Use Cases
  - Acceptance Tests (TDD)
PBIs

- **Functional**
  - Ex: based off a scenario of a use case
  - A user can be another system

- **Non-functional**
  - Become compliant with things such as reliability, availability, maintainability, security, ...

- **“Spike”**
  - Short time (a few hours) of research and design to understand “how to do it”

- **Defects**
  - Non-critical deficiencies that can wait for planning into a Sprint
  - Fix “critical” defects NOW!

- **Needs a Definition of Done**
  - Design, Write tests, code, unit tests, documentation, etc.
  - Includes maintaining compliance with non-functional requirements

- **No credit for partial work – either done or not done**

- **Includes Acceptance Criteria**

### Functional PBI

<table>
<thead>
<tr>
<th><strong>Vacationer searches for available rooms</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Search by location</td>
</tr>
<tr>
<td>Search by date</td>
</tr>
<tr>
<td>Search on type</td>
</tr>
</tbody>
</table>

Include critical information as needed:
Design Notes, Assumptions, Constraints
Creating PBIs

High-level Analysis and Design (on-going)

Slices cut across all sub-systems

Presentation / User Interface

Business Logic / Services

Database / Integration

Information from each Sprint feeds into the next

Sprint 1  Sprint 2  Sprint 3  Sprint 4  Sprint 5  Sprint 6  Sprint 7  Sprint 8

Release 1

Release 2
A User Story is Comprised of…

- **Card** – A written description of the story, used for planning and as a reminder

- **Conversation** – Represents a discussion about the story to flesh out the details of the story

- **Confirmation** – Acceptance Tests that convey and document details that can be used to determine when a story is complete
  - [http://www.mountaingoatsoftware.com/articles/advantages-of-user-stories-for-requirements](http://www.mountaingoatsoftware.com/articles/advantages-of-user-stories-for-requirements)
“Shall” requirements might say:
- The product shall have a gas engine
- The product shall have four wheels
  - The product shall have a rubber tire mounted to each wheel
- The product shall have a steering wheel
- The product shall have a steel body

What is it?

Source Mike Cohn:
www.mountaingoatsoftware.com
User Stories Convey Meaning

As a lawn service provider I want to mow lawns quickly and easily so that I can mow more lawns per day.

As a lawn service provider I want to sit comfortably while mowing lawns so that I can mow more lawns per day.

Reference: Mike Cohn, mountaingoatsoftware.com
Writing User Stories

- Often written by the Product Owner or as a team
- Brainstorm to generate ideas
- Some stories start out as epic stories; break them down
- Stories should be drafted and estimated prior to the release planning meeting

**INVEST** in good User Stories.....

Reference: Bill Wake
The hotel reservation system shall enable the user to complete a reservation transaction.

As a premium member, I want to search for available discounted rooms so I can save money while reserving the room of my choice.

As vacationer, I want to search for available rooms so I can see what is available and compare rooms.

As vacationer, I want to save my selections so I can come back later and make changes or complete my reservation.

STEP #1
Write User Stories
The hotel reservation system shall enable the user to complete a reservation transaction.

**User Story**

- As a vacation planner I want to reserve a room with a credit card so I can confirm my reservation.

**Acceptance Criteria and Verification**

- Reserve with Master Card
- Reserve with Visa
- Do not reserve with American Express—issue error message

**STEP #2**

Write Acceptance Criteria
As a vacationer and user of the hotel website, I want the system to be available 99.99% of the time so I can search the website for information any time…

As vacationer, I want web pages to download in <4 seconds so I don’t have to wait!!

As the hotel website owner, I want 1,000 concurrent users to be able to access the site at the same time with no impact to performance so users are not turned away due to speed.

**Stories for non-functional requirements**

**Describes system behavior or characteristics**

**STEP #3**
Identify constraints

Reference: Mike Cohn, mountaingoatsoftware.com
Scenario: Based on customer needs, your team has defined a logical architecture for an online hotel reservation system. The system is a traditional 3 tier architecture:

- a database layer (to persist reservations),
- a business logic layer (to manage reservations),
- and a browser-based user interface (to receive customer input).

Your product owner started to create the product backlog and has provided the following Epic:

**As a “vacationer” I want to search the hotel’s facilities and related amenities.**

- As a group we will write 1 story for the epic story above including acceptance criteria.

Time: 5 minutes
Agenda

- Iteration 1 – Introduction and Agile Overview
- Iteration 2 – Team Structure
- Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
- Iteration 4 – Writing Product Backlog Items
- Iteration 5 – Estimating PBIs
  - Estimating the effort to develop each user story
- Iteration 6 – Sprint Planning
- Iteration 7 – Sprint Execution
- Iteration 8 – Sprint Review, Retrospective, and Delivery
- Iteration 9 – Summary
Estimating PBIs

• Traditional estimates focus on absolutes
• Agile Estimates focus on relative size estimates

Why are relative size estimates preferred?

• People are better at comparing versus measuring in absolutes
• Estimates are faster
• Basic math still applies
• Easier to reach consensus

House made of straw
Small effort (3)

House made of sticks
Medium effort (5)

House made of bricks
Large effort (8)

Reference: Rally Software
Recommendations for Estimating PBIs

• Keep your estimates high-level, intuitive guesses
  – You’re only estimating the effort of one PBI relative to other PBIs in your Backlog
  – Don't over-analyze the details

• Estimating using a number sequence (Fibonacci numbers)
  – 1, 2, 3, 5, 8, 13, 21, …
  – The precision of your estimate should match the accuracy (cone of uncertainty)

• Estimate as a team

• You can also track your reliability
  – i.e. the number of points delivered as a percentage of the number of points committed at the start of the Sprint.

• Track your Velocity over time
  – At the end of each Sprint, look back at your Velocity

• Don't try to reconcile points with hours (ever!)

• Commit as a team
Estimating Technique: Planning Poker

- Estimating the PBI for a release. A release is one or more Sprints.
- Going into the estimation phase, PBIs for the release have been identified and each has verification objectives; PBIs have been discussed with the team.
- Steps:
  - Each estimator is given a deck of cards, each card has a valid number such as (1, 2, 3, 5, 8, 13, 21, ?)
  - The teams read the PBIs
  - An “average” PBI is selected (not the hardest, not the easiest)
  - The story is read to the team and discussed briefly
  - Each estimator selects a card to reveal his estimate (keep the card hidden)
  - Cards are turned over at the same time so everyone can see them
  - Differences in estimates are discussed; especially outliers
  - Re-estimate until estimates converge to within 1 increment or maximum 3 times (use the highest #)

Reference: www.mountaingoatsoftware.com
Estimating Technique: T-Shirt Sizes

1. Find the smallest PBI in your Product Backlog and place it on the left end of a wide space on a table
   - Assign size “XS” to trivial PBIs. If it’s easily conceivable there might be smaller PBIs some day (defects?), assign “S” to this story.

2. Find the biggest PBI the team believes it can complete in one Sprint and place it on the right end of the table space
   - Using the “right” number of team members and starting on the item at the start of the Sprint
   - Assign “XL” or “XXL” to this story
   (Note: the team will plan other new PBIs later.)

3. Place all other PBIs into 5 or 6 “buckets”: XS, S, M, L, XL (, XXL)
   - If the team identifies a PBI as too big to complete in one Sprint, place it to the right of the biggest PBI; these are really “epics” and need to be broken down into PBIs that can be completed in one Sprint. Size them as “XXX” (too big)

4. Number the buckets to get points: 1, 2, 3, 5, 8 (, 13)
   - “Too big” PBIs can be given place-holder points, if desired (to be replaced by the points of smaller PBIs broken down later)
What could these story points possibly mean?

Example

1: Trivial. This will be an easy point for my burn-down chart. (Low uncertainty, low complexity, and low effort)

2: A little bit of thought is required, but I’ve done so many, I won’t even break a sweat. Or, it sounds trivial and I’m going to hedge my bets a bit.

3: I’ve done this a lot, I know what needs to be done, maybe a few extra steps, but that’s it. Doubtful that I’ll need to Google anything.

5: OK, now we’re getting into the heart of the problem or, I don’t do this very often. I’ll need to think through this a bit. (At least one of the variables (complexity, effort, uncertainty has increased)

8: This is going to take some time and planning. Or, I’ve seen other people do this and I prototyped something like it a few years ago.

13: Possibly the maximum amount I can do and still get it done in one sprint. If this item is truly reduced to its simplest form, then it’s a complex piece of work. We need to consider our velocity.

21: This is very complex or contains a high level of uncertainty and/or effort.

?: Large. Needs more information and possibly broken down into several stories
## Backlog

<table>
<thead>
<tr>
<th>Backlog Item</th>
<th>Relative Estimate</th>
</tr>
</thead>
<tbody>
<tr>
<td>Putting away a load of laundry</td>
<td></td>
</tr>
<tr>
<td>Building a shed</td>
<td></td>
</tr>
<tr>
<td>Vacuuming the floors</td>
<td></td>
</tr>
<tr>
<td>Packing lunch</td>
<td></td>
</tr>
</tbody>
</table>

**Time:** 10 minutes
Agenda

- Iteration 1 – Introduction and Agile Overview
- Iteration 2 – Team Structure
- Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
- Iteration 4 – Writing Product Backlog Items
- Iteration 5 – Estimating PBIs
- Iteration 6 – Sprint Planning
  - What we are going to get done over the next few weeks
- Iteration 7 – Sprint Execution
- Iteration 8 – Sprint Review, Retrospective, and Delivery
- Iteration 9 – Summary
High Level Agile Stages

- Vision
- Customer Needs
- Product Roadmap
- Release Planning
- Sprint Planning
- Sprint Execution
- Sprint Review and Retrospective
- Delivery
- Product Backlog

High Level Agile Stages

1. **Vision**
2. **Customer Needs**
3. **Product Roadmap**
4. **Release Planning**
5. **Sprint Planning**
6. **Sprint Execution**
7. **Sprint Review and Retrospective**
8. **Delivery**

- **Product Backlog**

Diagram shows the flow from Vision and Customer Needs to Product Roadmap, leading to Release Planning, Sprint Planning, and eventually to Sprint Execution and Delivery. The Product Backlog is central to the process, feeding into the planning stages and driving the sprint execution and delivery.
Meeting is time-boxed.

Guideline = 2 hrs/week of Sprint without Product Backlog refinement.

Or, 1 hr/week of Sprint with PBL refinement.
Sprint Goal

- A Sprint’s Goal is fixed and may not change for that Sprint
- Sprint scope is a forecast and refines as a Sprint progresses
- Backlog outside the current Sprint is always changing, evolving, and being reordered

68
As a vacationer, I want to search room availability…

- Search for rooms available by date
- Search by type of room

As a vacationer, I want to change my reservation…

- Test objectives…

As a vacationer, I want to cancel my reservation…

- Test objectives…

---

**Sprint Backlog (PBIs with Tasks)**

<table>
<thead>
<tr>
<th>Task</th>
<th>Hours</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design</td>
<td>8</td>
</tr>
<tr>
<td>Design Review</td>
<td>3</td>
</tr>
<tr>
<td>Write Tests</td>
<td>8</td>
</tr>
<tr>
<td>Code</td>
<td>20</td>
</tr>
<tr>
<td>Code peer review</td>
<td>4</td>
</tr>
<tr>
<td>Automate Test</td>
<td>12</td>
</tr>
</tbody>
</table>
Scrum Board in Action!

Sprint Goal: ________________

<table>
<thead>
<tr>
<th>PBI</th>
<th>To Do</th>
<th>In Work</th>
<th>Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>As a vacationer, I want to search room availability…</td>
<td>Peer review …</td>
<td>Create tests …</td>
<td>Design the …</td>
</tr>
<tr>
<td>21</td>
<td>2</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td></td>
<td>Integrate the …</td>
<td>Static analysis …</td>
<td>Review the …</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Verify the …</td>
<td>Document the …</td>
<td></td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>As a vacationer, I want to change my reservation…</td>
<td>Integrate the …</td>
<td>Test the …</td>
<td>Design the …</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Static analysis …</td>
<td>Code the …</td>
<td>Review the …</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td>Verify the …</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Document the …</td>
<td>Peer review …</td>
<td></td>
</tr>
<tr>
<td></td>
<td>12</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Create tests</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Code the …</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>16</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
Team Capacity

- Capacity is the team members’ available hours to work in a Sprint
- Revisited each Sprint
- Compare planned task hours to capacity hours

Example for a two-week Sprint

<table>
<thead>
<tr>
<th>Team Member</th>
<th>Clock Hours per day</th>
<th>Total clock hours this Sprint</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bill</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>Scott</td>
<td>5</td>
<td>50</td>
</tr>
<tr>
<td>Chris</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>Andy</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>Cindy</td>
<td>7</td>
<td>70</td>
</tr>
<tr>
<td>Mike</td>
<td>8</td>
<td>80</td>
</tr>
<tr>
<td>TEAM TOTAL</td>
<td>40</td>
<td>400</td>
</tr>
</tbody>
</table>
Development Team owns and defines what *Done* means
  – Product Owner “accepts” Done work … so there better be agreement!

Examples of *Done*
  – Designed, coded, unit tested, integrated
  – Refactored
  – Documented
  – Acceptance tested, deployed

*Done* defines the current technical capability of the team.
  – Over time *Done* should include everything needed before release
  – Not Done backlog items may not be demonstrated at the Sprint Review (transparency—it’s not ready)

Reference: Ken Schwaber, ADM
• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure
• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
• Iteration 4 – Writing Product Backlog Items
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
  – Managing the Sprint Backlog, The Daily Scrum, and Agile Testing
  – Executing the Tasks to get the user stories “DONE”
• Iteration 8 – Sprint Review, Retrospective, and Delivery
• Iteration 9 – Summary
High Level Agile Stages
Refining the Product Backlog

• Top ordered PBIs are well understood and easily selected in Sprint Planning. They are “ready.”
• Product Backlog is continuously refined to increase understanding, granularity and transparency.
• Refinement usually consumes no more than 10% of the capacity of the Development Team.
Managing the Sprint Backlog

- Any team member can add, delete or change the Sprint backlog (tasks)
- Individuals sign up for work of their own choosing (self-organizing)
- Estimated work remaining is updated daily (it may go up!)
- Work for the Sprint emerges
- If the “how” is unclear, define a Sprint backlog task placeholder with a large amount of time. Add a small task (1-2 hours) to “spike” (design) the solution and break down the placeholder with the spike results.
- The team collectively is responsible for delivering on their commitments
Sprint Execution:
Plan, Design, Build, Test, Document, …

Product Backlog

• Unit Testing/Component Testing
• Continuous Integration
• Test Automation
• Peer/Code Reviews
• Documentation
• Requirements mapping

**The Daily Scrum**

**Daily Scrum**

1. What did I do yesterday that helped the Development Team meet the Sprint Goal?

2. What will I do today to help the Development Team meet the Sprint Goal?

3. Do I see any impediment that prevents me or the Development Team from meeting the Sprint Goal?

**OBJECTIVE:**

- Update the Sprint Plan to achieve the Sprint Goal

**PARAMETERS:**

- Daily
- Development Team mandatory
- Time boxed to 15-minutes
- Stand up (optional)
- Not for problem solving
- Scrum Master optional
  - To facilitate only!
- Product Owner optional
  - To clarify PBIs & priorities only
Scaling Agile: Scrum of Scrums

• Usually two or three times per week…as needed

• What has your team done since we last met?

• What will your team do before we meet next?

• What’s in your team’s way?

• What are you about to put in another team’s way?
You are on a Scrum Team for an International Project:

- You and two other members live in Los Angeles, CA
- One is in McLean, VA
- Two are in London, UK
- One is in Melbourne, Australia

How might you conduct the Daily Scrum?
<table>
<thead>
<tr>
<th>Tasks</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
<th>Fri</th>
</tr>
</thead>
<tbody>
<tr>
<td>Code the user interface</td>
<td>8</td>
<td>4</td>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Code the middle tier</td>
<td>16</td>
<td>12</td>
<td>10</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>Test the middle tier</td>
<td>8</td>
<td>16</td>
<td>16</td>
<td>11</td>
<td>8</td>
</tr>
<tr>
<td>Write online help</td>
<td>12</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

![Line graph showing hours worked each day](Mountain Goat Software)
A Team’s Sprint Burndown

<table>
<thead>
<tr>
<th>Iteration</th>
<th>Tasks</th>
<th>Owner</th>
<th>Status</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
<th>Fri</th>
<th>Mon</th>
<th>Tues</th>
<th>Wed</th>
<th>Thur</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design Review</td>
<td>Scott</td>
<td></td>
<td>Completed</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Install baseline</td>
<td>Bill</td>
<td></td>
<td>Completed</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>ICD updates</td>
<td>Scott</td>
<td></td>
<td>Completed</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>3</td>
<td>3</td>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Acquire test data</td>
<td>Bill</td>
<td></td>
<td>Completed</td>
<td>8</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Code</td>
<td>Scott</td>
<td></td>
<td>Completed</td>
<td>24</td>
<td>20</td>
<td>16</td>
<td>14</td>
<td>10</td>
<td>10</td>
<td>3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Develop tests</td>
<td>Scott</td>
<td></td>
<td>Completed</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>4</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>Run Tests</td>
<td>Scott</td>
<td></td>
<td>Completed</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>8</td>
<td>5</td>
<td>3</td>
<td>4</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

PBI: As a vacationer, I want to search room availability…

The Team
Manages its work and progress
Meets daily to discuss progress and commit to plan

Managed by the Team
Agile Testing

Test Early

Test Often

Automate
“The error [is] typically 100 times more expensive to correct in the maintenance phase than in the requirements phase.” *Software Engineering Economics, Barry Boehm*

*Agile attempts to change the cost of change equation.*

*Extreme Programming Explained: Embrace Change, Kent Beck*

- With Agile, it’s okay to make changes
- Change is unavoidable
- Agile embraces change
Theory and Practice

- Agile testing is about the people (Development Team, Users) and communication
  - Lengthy test documents will be incomplete, out-of-date, ambiguous
  - Agile testing is like agile coding: incremental & iterative

- Test results should be big, public, easy-to-read charts

- Testers part of the Cross Functional Team
  - The reasons? Speed, accountability, transparency, transfer of knowledge
  - In many cases, the developer is the tester

- Testing is not a phase

- Automated testing is a crucial

Test Early, Often, Automate
Agile Testing Quadrants

Automated & Manual

Supporting the Team

Automated

Technology-Facing

Q1

Q2

Q3

Q4

Business-Facing

Automated Tools

Critiquing the Product

Manual

Feature Tests
Story Tests
Functional Tests
Prototype Tests
Simulations

Exploratory Testing
Scenarios
Usability Testing
Acceptance Testing
Alpha/Beta

Performance Testing
Load Testing
Security Testing
“ility” Testing

Unit Tests
Component Tests

www.agiletester.ca
Agenda

• Iteration 1 – Introduction and Agile Overview
• Iteration 2 – Team Structure
• Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
• Iteration 4 – Writing Product Backlog Items
• Iteration 5 – Estimating PBIs
• Iteration 6 – Sprint Planning
• Iteration 7 – Sprint Execution
• Iteration 8 – Sprint Review, Retrospective, and Delivery
  – Reviewing the completed work during the sprint (inspect and adapt)
  – What went well and what could be improved (sprint and releases)
  – Every sprint is potentially releasable
• Iteration 9 – Summary
High Level Agile Stages

- Vision
  - Customer Needs
- Product Roadmap
- Release Planning
- Sprint Planning
- Sprint Execution
- Sprint Review and Retrospective
- Delivery
- Product Backlog

High Level Agile Stages
The Sprint Review

- Demonstrates new DONE functionality
- Transparency and information sharing
- Informal
- Time-boxed
- All stakeholders invited
- Product Owner accepts the DONE work
- Revisit the backlog
- Update Metrics
Sprint Retrospective

- Take a look at what is and is not working well
- Time-boxed
- Done after every Sprint
- Facilitated by the Scrum Master
- Focus is on process improvement
- Scrum Team participates
  - ScrumMaster
  - Product owner
  - Team
  - Consider customers and others

Ways to focus the discussion

Goal we want to accomplish

1. Worked well
2. Could be improved
3. Actions and Priorities
Agile Transparency

Plan, Execute, Track, Improve

Teams know their velocity
Implements future planning
Planning the Next Sprint

• The backlog of user stories may have changed
  – Different priorities
  – New user stories

• Customer/Client manager is present to discuss the next set of user stories

• Team reviews estimates in the release or estimates new user stories

• Team chooses user stories that can be done in this next Sprint
  – We have new data about the team velocity

• Team commits to the user stories for the next Sprint

• Team does task breakdown for the chosen user stories
End of the Release

- Similar to end of Sprint – every Sprint is potentially releasable
- If practices are not “agile” enough, hardening Sprint may be required
- PBIs have been demonstrated and accepted
- UAT, IRR, etc.
- Version description document
- Although not ideal, many agile teams have a parallel testing effort during Sprints where defects are found and fed back into the process. In addition, the working software becomes a working “system”
- QA testing or similar
Agenda

- Iteration 1 – Introduction and Agile Overview
- Iteration 2 – Team Structure
- Iteration 3 – Vision and Product Roadmap, Project Startup & Release Planning
- Iteration 4 – Writing Product Backlog Items
- Iteration 5 – Estimating PBIs
- Iteration 6 – Sprint Planning
- Iteration 7 – Sprint Execution
- Iteration 8 – Sprint Review, Retrospective, and Delivery
- Iteration 9 – Summary
• Reflect on today’s session.
  – What worked well
  – Suggestions for improvement

• What are your next steps?
  – What can you do to make a difference and lead change in your team, program, or organization?

Time: 5 minutes
• Agile is about creating an adaptive organization that is able to respond to the changing needs of customers and technologies
• Agile is not just about software development
• Agile practices affect the entire organization
• There are several Agile methods under the umbrella of “Agile Practices”
• Agile development emphasizes the need for ongoing iterative development with completed, demonstrable functionality at the end of every Sprint
• Agile methods emphasize the need for team and customer collaboration
Many of the ideas in this presentation originated from:
- Ken Schwaber and Mike Cohn
- Other contributions/research are noted throughout the presentation
- My experiences with the many programs and projects across Northrop Grumman

References and Recommended Agile Reading List
- Adaptive Enterprise by Steven Haeckel
- Agile and Iterative Development: A Manager’s Guide by Craig Larman
- Agile Estimating and Planning by Mike Cohn
- Succeeding with Agile by Mike Cohn
- Agile Project Management with Scrum by Ken Schwaber
- Agile Testing by Lisa Crispin and Janet Gregory
- Scrum and The Enterprise by Ken Schwaber
- Software in 30 Days by Ken Schwaber and Jeff Sutherland
- Agile Retrospectives: Making Good Teams Great by Esther Derby
- www.mountaingoatsoftware.com by Mike Cohn
# Agile Terminology

<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Sprint</strong></td>
<td>Fixed time-box in which development occurs also referred to as an “iteration”</td>
</tr>
<tr>
<td><strong>Product Backlog</strong></td>
<td>List of prioritized Requirements or Product Backlog Items (PBIs)</td>
</tr>
<tr>
<td><strong>Product Burn Down Chart</strong></td>
<td>Progress for the release; Focuses on the remaining user story points for the given timeframe</td>
</tr>
<tr>
<td><strong>Product Owner</strong></td>
<td>Owns the product backlog, assigns priority to user stories based on customer input</td>
</tr>
<tr>
<td><strong>Release</strong></td>
<td>Usually a 2 – 9 month timeframe; formal committed delivery of product</td>
</tr>
<tr>
<td><strong>Scrum Master</strong></td>
<td>Helps the Agile team through the process and removes impediments</td>
</tr>
<tr>
<td><strong>The Team</strong></td>
<td>Cross functional team</td>
</tr>
<tr>
<td><strong>User Story</strong></td>
<td>A format for PBIs. Similar to a requirement. “As a <strong>user</strong> I want <strong>what</strong> so that <strong>purpose</strong>”</td>
</tr>
</tbody>
</table>