
Setting Agile-Centric Release Criteria aka Driving Done-Ness

Bob Galen

President & Principal Consultant,

RGCG, LLC – Leading you down the path of agility...

www.rgalen.com bob@rgalen.com

Introduction

Bob Galen

- Somewhere 'north' of 20 years experience ☺
- Various lifecycles – Waterfall variants, RUP, Agile, Chaos, etc.
- Various domains – SaaS, Medical, Financial, Computer Systems, and Telecommunications
- Developer first, then Project Management / Leadership, then Testing
- 'Pieces' of Scrum in late 90's; before Agile was Agile
- Agility @ Lucent in 2000 – 2001 using Extreme Programming
- Formally using Scrum since 2000
- Most recently at ChannelAdvisor as Dir. Product Development and Agile Architect/Coach (2007-2008)
- Connect w/ me via LinkedIn if you wish...

Bias Disclaimer:
Agile is THE BEST Methodology for Software Development...
However, NOT a Silver Bullet!

Outline

Introduction

- 1. Agile Goals & Criteria**
- 2. Team Level: Done-Ness**
- 3. Sprint Level: Story & Acceptance**
- 4. Release Level: Freeze, Entry & Exit Criteria, and Release Criteria**
- 5. Release Train / Influence Points**
- 6. Q&A**

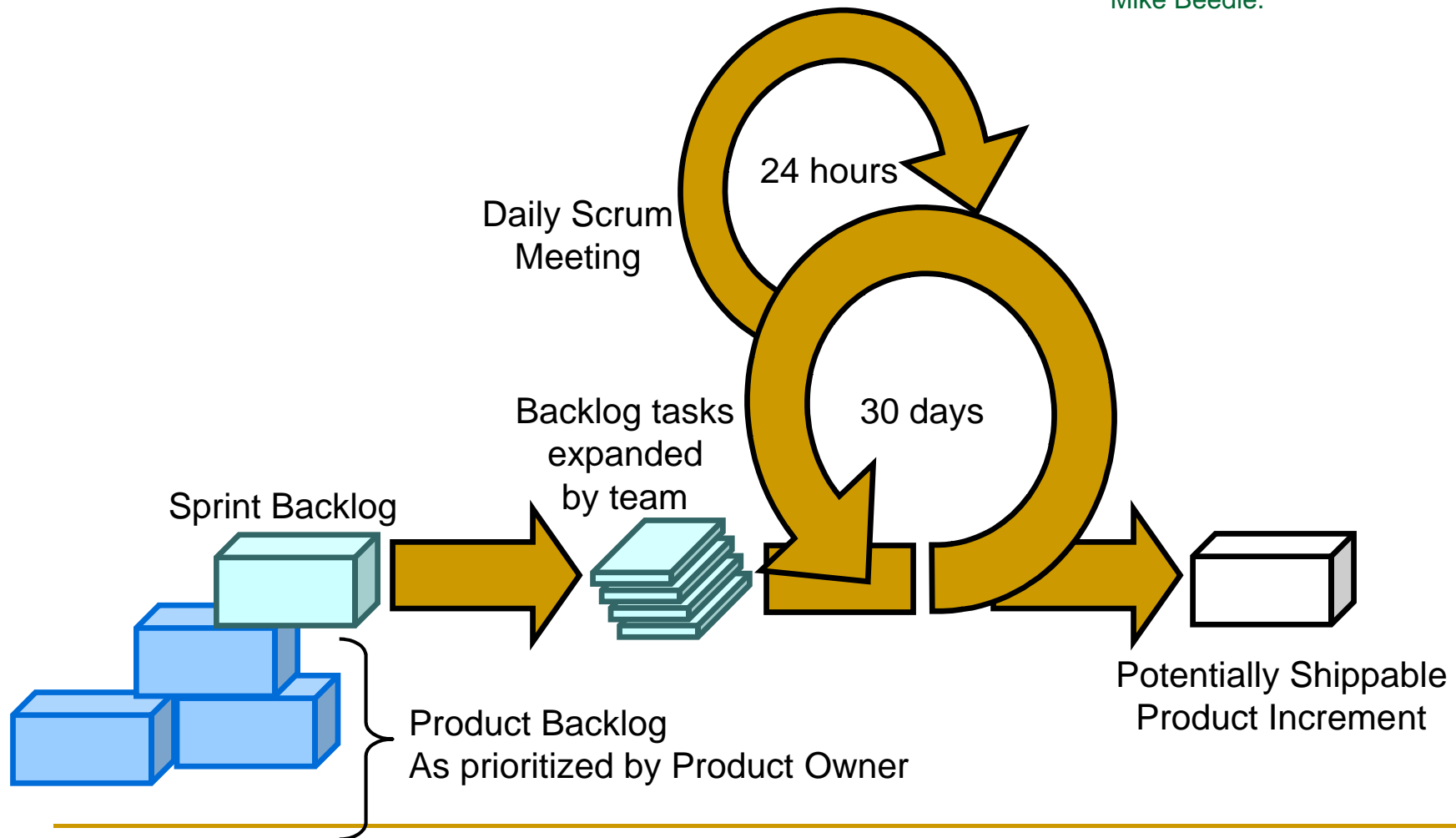
Scrum

We'll be using Scrum as the Agile reference framework for the class

- Agile Project Management Framework
- Roles:
 - Scrum Master, Product Owner & Team
- Sprint – iteration (15 – 30 days)
- Product Backlog – a prioritized list of: requirements, work to do, user stories
- Sprint Planning – mapping PB to tasks
- Daily stand-up (15 minutes, Pigs & Chickens)
- Sprint Review (demo) & Retrospective

Process Overview

Source: Adapted from *Agile Software Development with Scrum* by Ken Schwaber and Mike Beedle.



Establishing Goals & Criteria

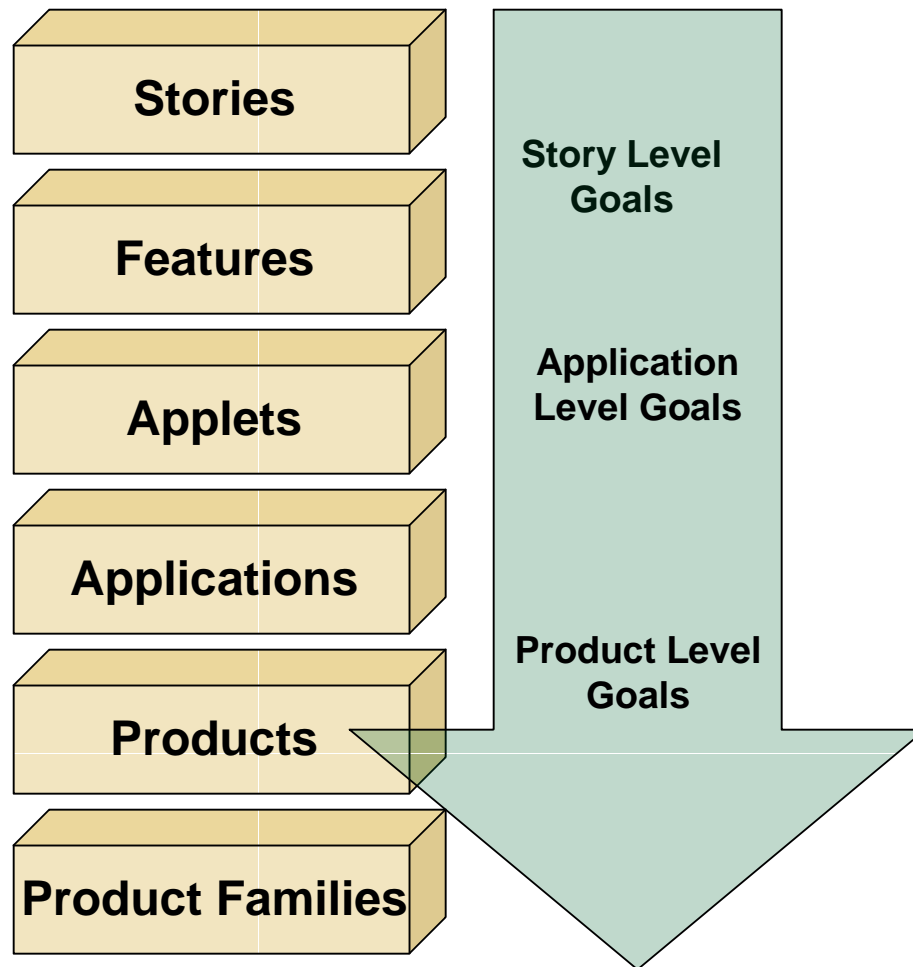
Why They Are Crucial?

- Agile teams are essentially self-directed, so plans don't drive behavior or success...
- People do and Goals drive the Team!
- The teams then swarm around the goals, using their creativity and teamwork to figure out:
 - What's most important
 - How to achieve it
 - Always looking for simple & creative—20% solutions

Lean Principles Applied

- Most important aspects first
- KISS principle; no Gold-plating
- Small deliverables; worked on serially
- Deliver End-to-End behavior
 - In thin *'Slices'*
 - UI to Backend DB
- Driven to done, into inventory as completed components, as soon as possible!
 - Avoiding 90% done syndrome
 - Post-Sprint integration collaboration
- Rarely re-visited; mindset is to ruthlessly minimize rework

Product Owners – Hierarchical Goals



- The PO organization must coordinate
 - A hierarchical set of goals that drive the execution of the teams
 - Include release-centric goals
 - Include quality centric goals
 - Include timing & workflow centric goals

In conjunction with technology and project leadership

While often interfacing to operations and customer facing organizations

Notions of Done-Ness Development

- Need to define “Done” from team members perspectives
- If you’re a developer, what does “I’m done with that story” mean?
 - ✓ Code complete
 - ✓ Code reviewed (paired)
 - ✓ Checked in – build successful
 - ✓ Unit tests developed – passed
 - ✓ Integration
 - ✓ QA collaboration
 - ✓ Run by the Product Owner
- Every type of task should have a definition of Done-Ness! How else could you estimate the work?

Notions of Done-Ness Testing

- If you're a tester, what does "I'm done with that story" mean?
 - ✓ Test cases designed w/a broad view to test cases (unit, functional, acceptance, regression)
 - ✓ Test cases pair-reviewed with development & test team members
 - ✓ Test cases - checked into repository
 - ✓ Ran test cases successfully; no issues
 - ✓ Ran Acceptance Tests with the Product Owner
 - ✓ Automated the Acceptance Test cases
 - ✓ Connected the automation to the Continuous Integration environment
 - ✓ Validated independent execution

Story Acceptance

- Each User Story should have acceptance criteria as part of the card
- They should focus on the verifiable behavior, core business logic, key requirements for the story
- Typically, they are crafted by the Product Owner
 - Leveraging skills of Business Analysts and Testers
- Story acceptance tests are normally automated and run as part of feature acceptance AND regression
 - FIT & FitNesse are among the Open Source tools that enable this

User Story

Examples

As a dog owner, I want to sign-up for a kennel reservation over Christmas so that I get a confirmed spot

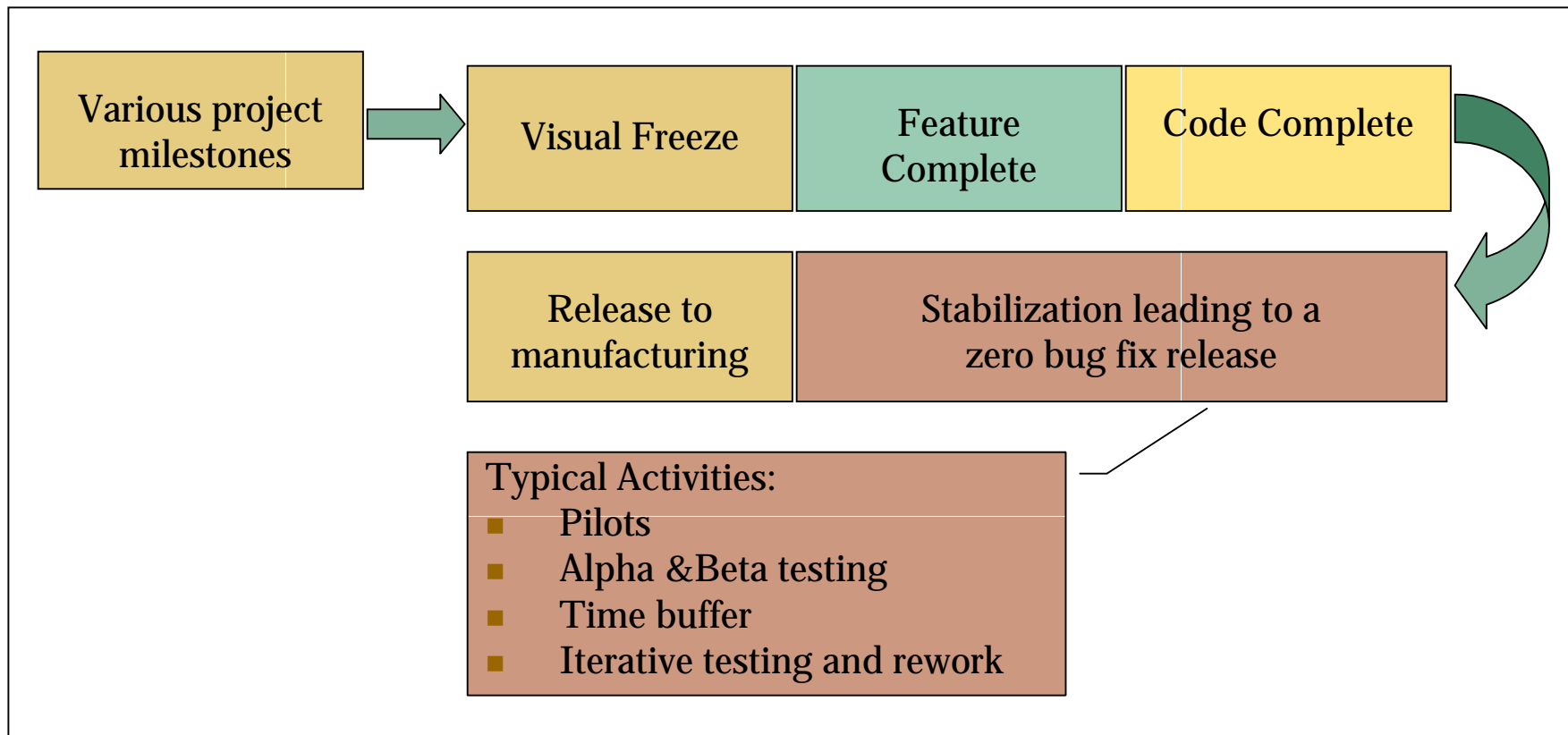
Verify individual as a registered pet owner
Verify that preferred members get 15% discount on basic service
Verify that preferred members get 25% discount on extended services and reservation priority over other members
Verify that past Christmas customers get reservation priority
Verify that declines get email with discount coupon for future services

Code Freeze Dynamics still Apply!

- Try and front-load major features and high priority work
- Develop milestones for coalescing your code towards a freeze point
 - Enter your hardening Sprints with a specific freeze target
 - During hardening have a coding halt target
- May need layered freezes
 - UI versus Back-end Database layers
 - Database deployment script development
- Still need to triage bugs to see what 'fits'...usually in daily release Scrum of Scrums

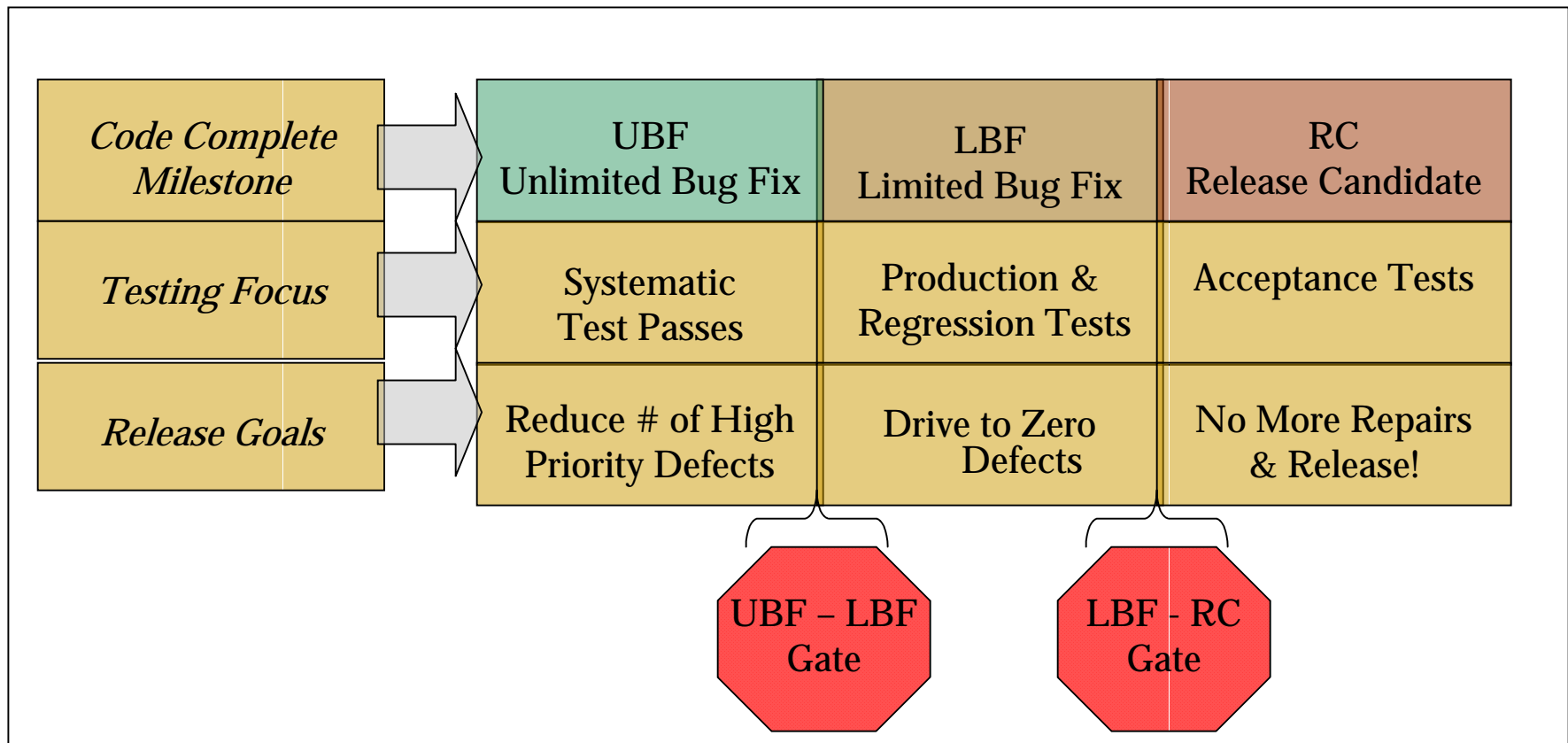
Microsoft - Code Complete Model

- Microsoft employs a “code complete” strategy as defined below –



Schoor – Example of Repair Code Freeze

- Bruce Schoor has an extension to the Microsoft “code complete” model as defined below –



Iteration / Goal Acceptance

- Each Scrum Sprint has a Product Owner determined goal
- Usually sprint success is not determined by the exact number of completed stories or tasks
- Instead, what most important is meeting the spirit of the goal

Deliver a 6 minute demonstration of the software that demonstrates our most compelling value features and achieves venture capital investment interest

Traditional Entry & Exit Criteria

■ Entry Criteria

- Conditions that must be met prior to SQA / Testers beginning their testing efforts
- Usually some sort of change log, content position
- Smoke Testing (manual and/or automated) is a form of entry criteria – tied to execution / passing of focused testing

■ Exit Criteria

- Conditions that must be met prior to testing SQA / Testers completing testing on a specific deliverable or iteration
- Normally includes coverage (test cases run, features completed)
- Also includes quality attributes (pass rates, acceptable defect levels)

Traditional Smoke Testing

Automated “Entry” Criteria

- A set of tests that are run prior to SQA “accepting” a release for testing
 - Typically automated and “connected” to the build system
 - Intended to prevent wasted effort by SQA on broken releases (basic operations and core features)
 - Focus (tests) can / should change release over release
 - Programmatic form of release criteria
 - Usually defined collaboratively with and owned by the development team

Release Criteria

- Goals and objectives for the entire project release
- Usually they are multi-faceted, defining a broad set of conditions
 - Required artifacts & levels of detail
 - Testing activities or coverage levels
 - Quality or allowed defect levels
 - Results or performance metrics achievement levels
 - Collaboration with other groups – dependency management
 - Compliance levels
- That **IF MET** would mean the release could occur.

Release Criteria – Johanna Rothman

- Johanna Rothman – 5 Steps for definition
 1. Define Success
 2. Learn What's Important for This Project
 3. Draft Release Criteria
 4. Make the Release Criteria SMART
 - Specific, Measurable, Attainable, Realistic and Trackable
 5. Gain Consensus on the Release Criteria

- Binary interpretation – (pass or fail) (go or no-go) intended to drive release decision-making

- Changed infrequently – holding to your initial goals!

Release Criteria – Rothman Example

Example Release Criteria

1. The code must support both Windows Vista and Windows XP
2. All priority P0, P1 and P6 defects will be repaired / addressed
3. For all documented bugs – on-line help, release notes and formal documentation will contain relevant updates
4. All QA tests run at 100% of expected coverage
5. No new P0 – P3 defects found within the last 3 weeks
6. Release target – GA of April 1, 2008

To the left is a sample set of release criteria intended to guide activity for a given release. They speak to –

- ✓ Platforms
- ✓ Defects allowed
- ✓ Defect actions
- ✓ Coverage
- ✓ Maturity
- ✓ Performance
- ✓ Release date

Release Criteria – Another Format

An alternative approach is High / Low or
Do / Don't Do Contrasting Charts

High Priority / Focus

- Existing functionality can not be affected by new changes - (functional regression testing)
- Existing performance may not be degraded by new changes - (performance regression testing, we also lacked a performance benchmark)
- New functionality must work
- Component interoperability w/o performance regression

Low Priority / Focus

- Interfaces beyond 10/100/1000 Ethernet and ATM are lower priority
- Existing performance may not be degraded by new changes - even when running multiple components, don't get hung up on improvement
- New functionality must work, except new reports that do not map to older reports
- Component interoperability across all permutations, we can identify (n) key

Release Criteria – Another Format

Will Cover (In Bounds)

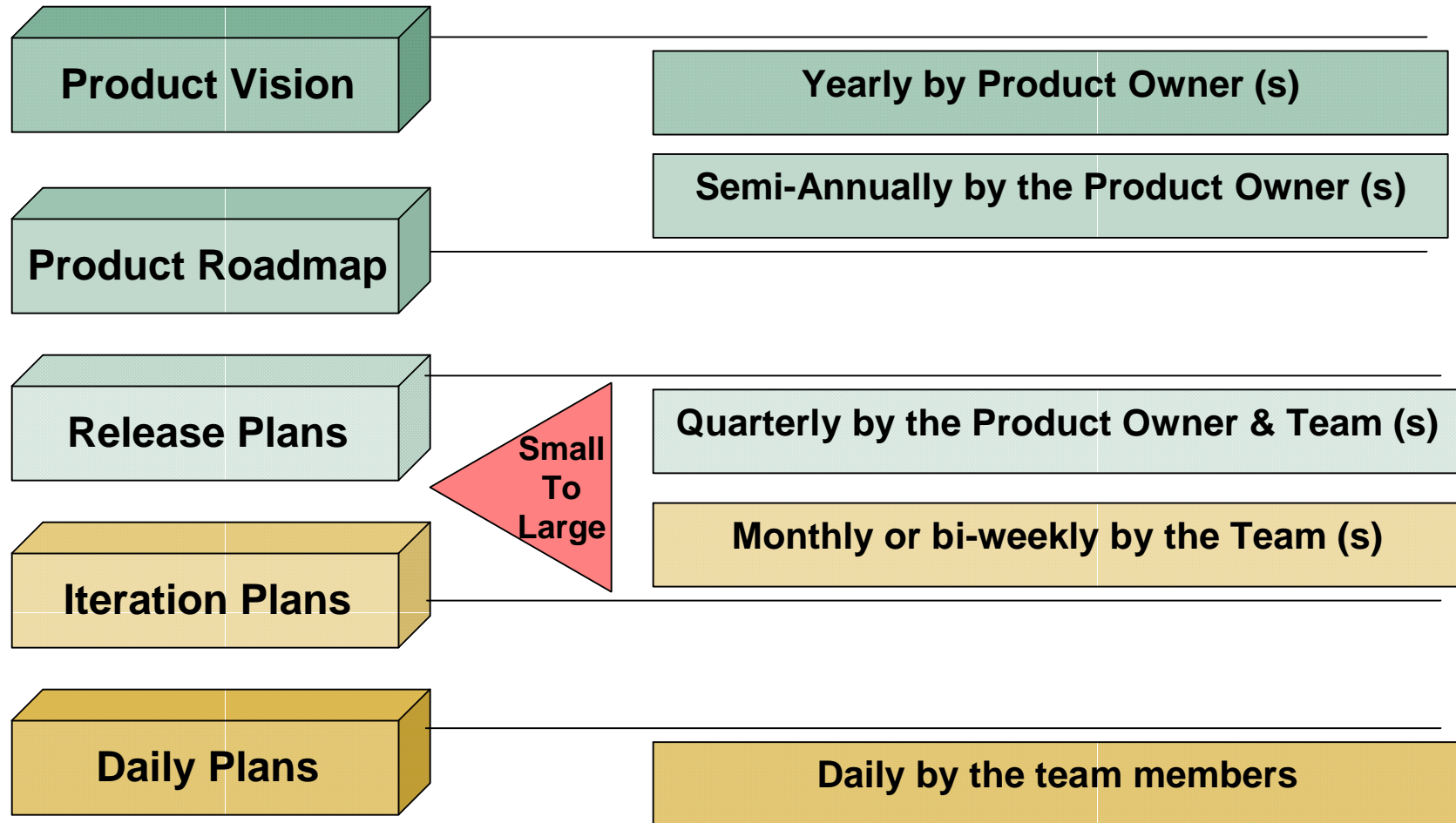
- Will run 100% of the existing regression suite prior to final release, must have > 95% pass rate of tests
- Will test all new features associated with higher performance network interfaces (GE 10, 100, 1000)
- Of the reporting options, will test the customer highest priority (most used) 50 for correctness.
- Performance will be tested at the 10Gb., 50% throughput level, with filtered recording for 10 hours.
- Will work with customer to demonstrate & run 100% of the acceptance tests.

Will Not Cover (Out-of-Bounds)

- Regression will not contain features presented in the last iteration and these will only be tested on a risk (< 20% of the tests) basis.
- Will not regress nor feature test on slower interfaces – less than 10 Gb.
- Of the remaining 250 reporting options, will only sample a few via exploratory testing.
- All other interface performance will not be tested. Also, the requirement is for 24 hours of record time and, at best, we can extrapolate from 10.
- There will be no usability testing nor support of the planned pilot releases.

Product Owner

Planning Levels in Large-Scale Agile Projects



Release Train Management

- **Iterative model with a release target**
 - Product centric
 - Focused on a production push/release
- **Synchronized Sprints across teams**
 - Some teams are un-synchronized, but leads to less efficient cross-team (product) interactions
- **Continuous Integration is the glue**
 - Including automated unit and feature tests; partial regression
- **Notion of a “Hardening Sprint”**
 - Focused more on Integration & Regression testing
 - Assumption that it's mostly automated
 - Environment promotion
- **Define a final Hardening Sprint where the product is readied for release**
 - Documentation, Support, Compliance, UAT, Training

Release Train Management

“Motivational” Driving Forces

■ Vision

- Schedule for delivery
- Feature set(s)
- Goals for each team

■ Themes

- Which teams will be working on what components
- Packages of User Stories
- Prioritized by business value & need

■ End-to-End Use Cases

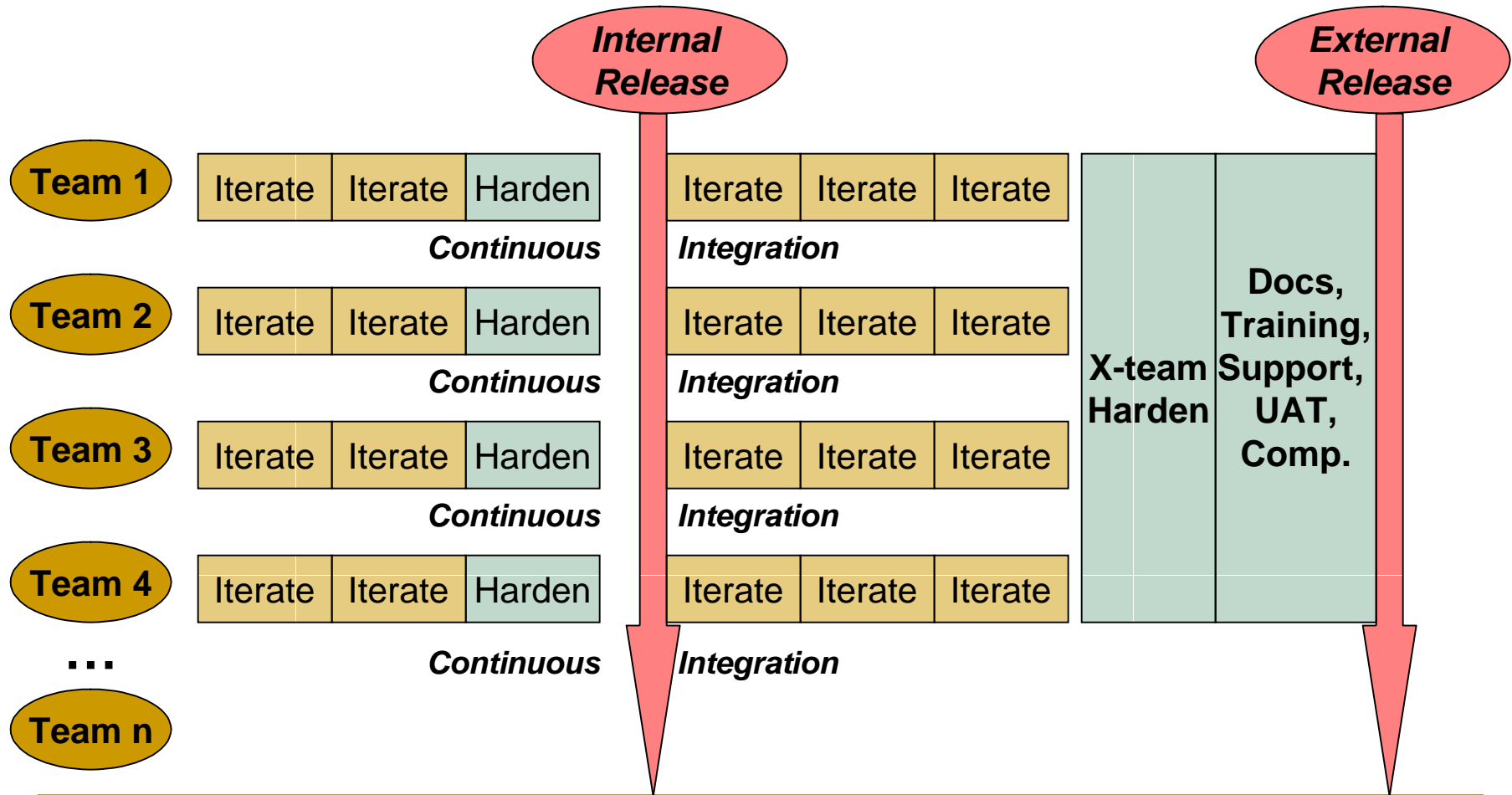
- Integration focused execution

Release Train Management

“Internal” Driving Forces

- Customer’s ability to “accept” the release timing
- Value being delivered within the release – purely scope
- Hardening Sprint (*production readiness “reality”*)
 - Time, Complexity & Size
 - Levels of Test Automation
 - Compliance & Industry
- Internal team readiness
 - Customer support
 - Sales & Marketing readiness
 - Overall documentation & training

The Agile Release Train Synchronized



Release Goal Setting

A Key for Coordination

- As you scale, each planning level should create criteria (Sprint Goals) that are –
 - Interrelated and cohesive
 - Focused towards the end product release and not simply on each teams deliverables
 - Identify dependencies and overall workflow
- The traditional notion of Chartering also applies at the higher levels, with Charters defined as:
 - Goals, Objectives & Scope
 - Clearly measurable view to “Done” – Release Criteria
 - Multi-faceted view towards quality (defects, coverage, non-functional requirements)
 - Allowing for team scope trade-offs

Levels of Criteria

Activity	Criteria	Example
Basic Team Work Products	Done'ness criteria	Pairing or pair inspections of code prior to check-in; or development, execution and passing of unit tests.
User Story or Theme Level	Acceptance Tests	Development of FitNesse based acceptance tests with the customer AND their successful execution and passing. Developed toward individual stories and/or themes for sets of stories.
Sprint or Iteration Level	Done'ness criteria	Defining a Sprint Goal that clarifies the feature development and all external dependencies associated with a sprint.
Release Level	Release criteria	Defining a broad set of conditions (artifacts, testing activities or coverage levels, results/metrics, collaboration with other groups, meeting compliance levels, etc.) that IF MET would mean the release could occur.

Exercise

- Gather in small groups; better that you are from the same company/organization OR those with similar characteristics
- In this section, we discussed the various levels of criteria that are important within agile teams.
 - 1) Amongst your team, discuss how you've established goals & criteria in your Agile (or non-Agile) teams
 - 1) What levels are the most important?
 - 2) What levels don't matter as much?
 - 2) What part does the team play in definition? Should play?
 - 3) Does defining what "done" means really matter? How?
 - 4) If you had only one to pick, which would it be? And why?

Wrap-up

- Self-directed Agile teams are “directed” by their Goals
- There are 4 Levels:
 1. Professional / Done-ness
 2. Feature / Story
 3. Iteration / Sprint
 4. Release Criteria

of goals driving Excellence in any agile team.

In Waterfall you get what you plan for. In Agile, you get what your Goals drive you towards...

Questions?

Thank you!

Contact Info

Scrum Product Ownership – Balancing Value From the Inside Out published by RGCG in 2009.

Software Endgames: Eliminating Defects, Controlling Change, and the Countdown to On-Time Delivery
published by Dorset House in 2005.

Bob Galen

RGalen Consulting Group, L.L.C. Go to www.rgalen.com for purchasing / for order info, misc. related presentations, and papers.

Agile focused training, coaching & consulting

PO Box 865, Cary, NC 27512
919-272-0719

www.rgalen.com
bob@rgalen.com

