Agile Development

Jeffery Payne
CEO, Coveros, Inc.
jeff.payne@coveros.com

Agenda

- Introductions & Expectations
- What is Agile?
- Agile Development Planning
- Agile Development Iterations
- Wrap Up

About Coveros

coveros

- Coveros helps organizations accelerate the delivery of secure, reliable software

- Our consulting services:
  - Agile software development
  - Application security
  - Software quality assurance
  - Software process improvement

- Our key markets:
  - Financial services
  - Healthcare
  - Defense
  - National security

Corporate Partners

ARXAN

FORTIFY SOFTWARE

Telos

SQE TRAINING

Introductions

Instructor – Jeffery Payne

Jeffery Payne is CEO and founder of Coveros, Inc., a software company that helps organizations accelerate the delivery of secure, reliable software. Coveros uses agile development methods and a proven software assurance framework to build security and quality into software from the ground up. Prior to founding Coveros, Jeffery was Chairman of the Board, CEO, and co-founder of Cigital, Inc. Under his direction, Cigital became a leader in software security and software quality solutions, helping clients mitigate the risk of software failure. Jeffery is a recognized software expert and popular speaker at both business and technology conferences on a variety of software quality, security, and agile development topics. He has also testified before Congress on issues of national importance, including intellectual property rights, cyber-terrorism, Software research funding, and software quality.

Class Attendees

Expectations

- What are your expectations for this class?
- What do you wish to learn?
- What questions do you want answered?

Objectives

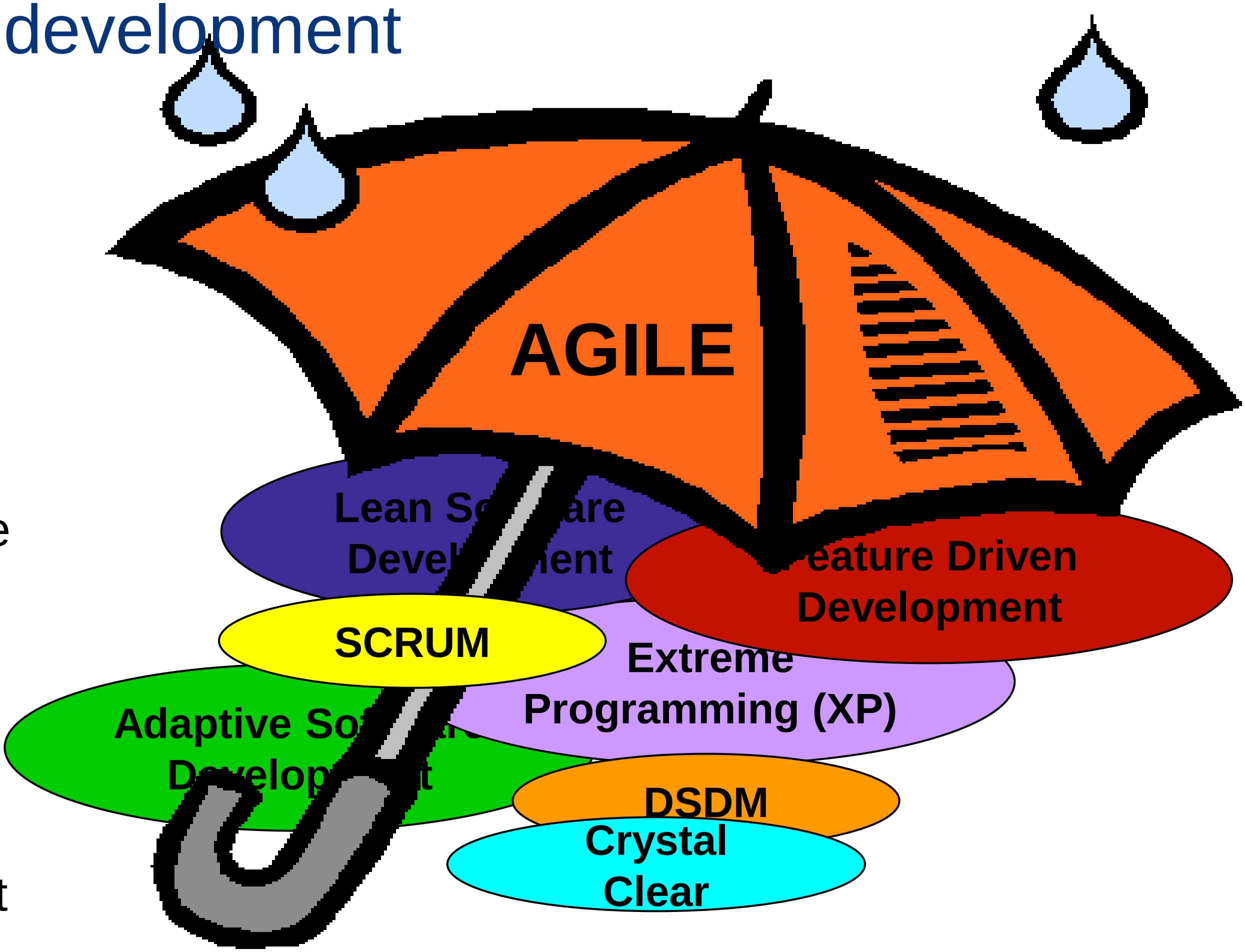
The primary objectives of this course are to:

- Introduce you to Agile software development
- Outline the major steps required to successfully plan and execute an Agile software project.
- Provide an overview of the leading Agile development best practices

What is Agile?

The agile movement began as a set of ideas for improving software development

- Close collaboration between programmers & business people
- Face-to-face communication
- Frequent delivery of deployable business value
- Self-organizing teams
- Crafting code & environment to support requirements changes
- The most important output of a project is working software


AGILE
Lean Software Development
Feature Driven Development
SCRUM
Extreme Programming (XP)
Adaptive Software Development
DSDM
Crystal Clear

http://www.agilemanifesto.org

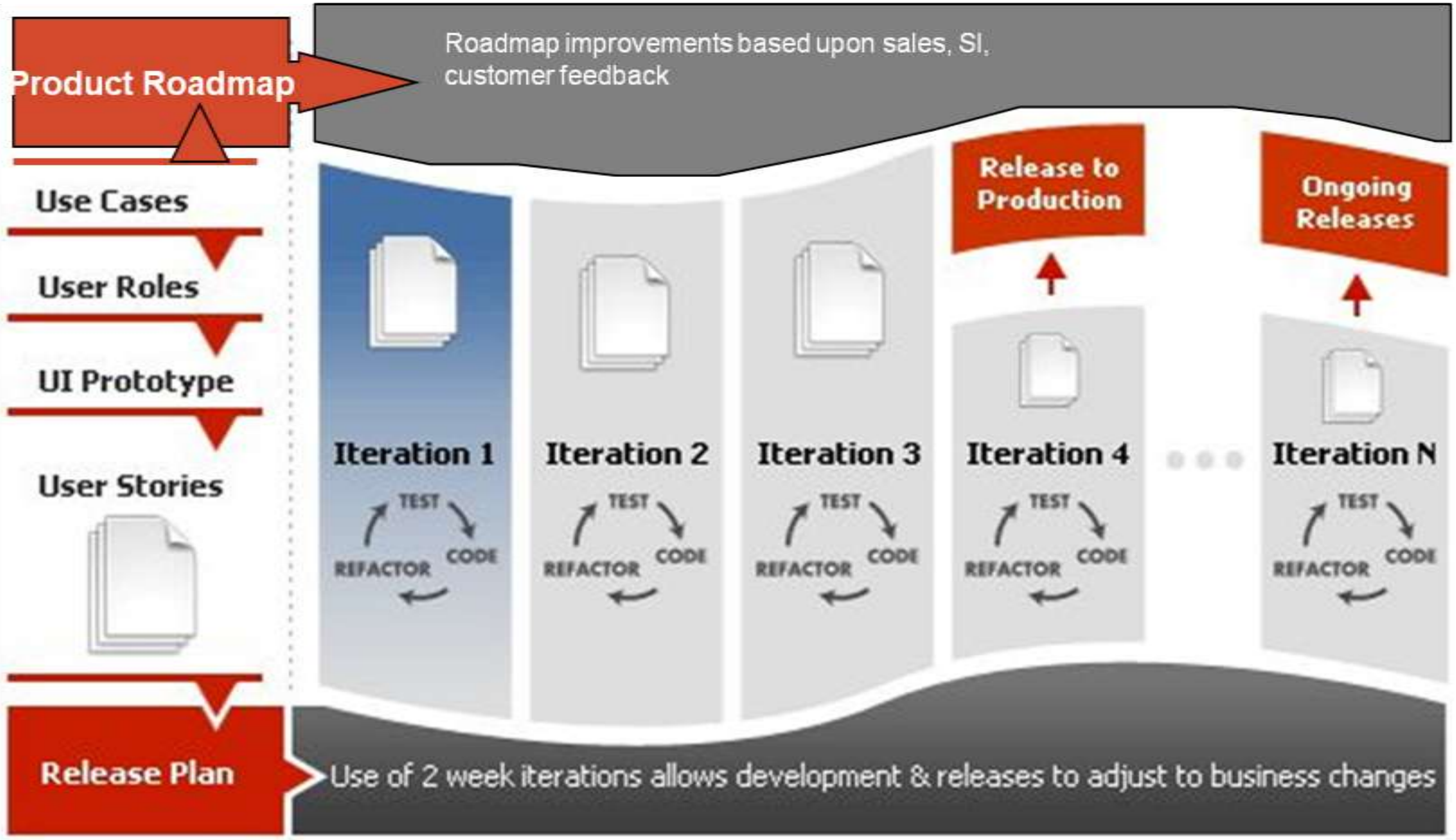
What is Agile?

All agile methodologies adhere to some basic principles

- Early and continuous delivery of valuable software
- Welcome changing requirements, even late in development
- Deliver working software frequently
- Business people and developers work together daily
- Build projects around motivated individuals and trust them to get the job done.
- Frequent conversation to convey information efficiently
- Working software as the primary measure of progress
- Sustainable development
- Continuous attention to technical excellence and good design
- Simplicity—maximizing the amount of work not done
- The best architectures, requirements, and designs emerge from self-organizing teams
- At regular intervals, the team reflects on, tunes, and adjusts its behavior

Agile Product Development Process

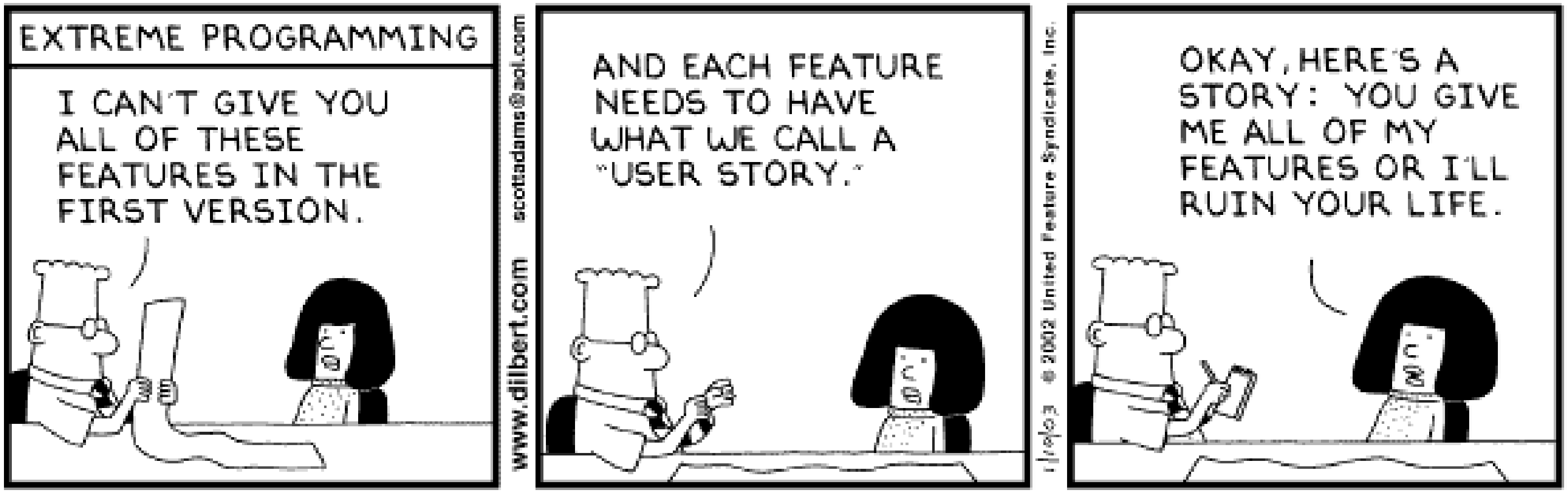
Initial Planning On-going Planning, Implementation & Release



- Incremental product delivery process that encompasses all aspects of the organization
- Team-oriented with day-to-day interactions between all functions

Agile Development Planning

Agile Development Planning


EXTREME PROGRAMMING
I CAN'T GIVE YOU ALL OF THESE FEATURES IN THE FIRST VERSION.
scottadams@aol.com
www.dilbert.com
AND EACH FEATURE NEEDS TO HAVE WHAT WE CALL A "USER STORY."
© 2002 United Feature Syndicate, Inc.
1/10/03
OKAY, HERE'S A STORY: YOU GIVE ME ALL OF MY FEATURES OR I'LL RUIN YOUR LIFE.

Copyright © 2003 United Feature Syndicate, Inc.

Agile Planning Process

Product
Strategy
Sales
Customer
Feedback
Market
Engineering

Product wish list

Initial Release Planning (common artifacts below)

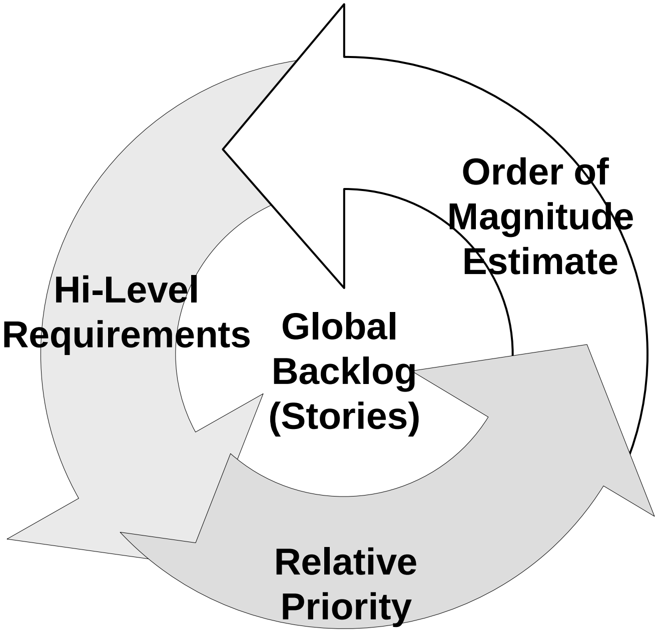
Initial Architecture

UI Wire Frames

Detailed User Stories

Test Strategy

Release Plan



Iterative Planning (during Sprints)

- Review output from User Acceptance Tests (UATs)
- Review changes in priority
- Update stores for next Sprint
- Update release plan

Creating a Global Backlog

- A Global Backlog encompasses all items from the Wish List that Product Management deems the highest priority for inclusion in upcoming releases.
- Backlog items are prioritized and assigned an Order of Magnitude (OOM) estimate
- OOM can be used for budgeting purposes BUT ONLY IF THE TOP END VARIANCE ESTIMATE IS USED
- A Global Backlog contains User Stories

Creating a Global Backlog

User Stories

- A story represents some small slice of visible, usable functionality—typically something a user can do with the system
- A well-written story possesses the following characteristics:
  - Understandable
  - Testable
  - Valuable to the customer
  - Independent of each other
  - Small enough to build a handful each Sprint
- Stories are written during initial planning or during a Sprint planning meeting once the project has begun.
- Although the idea for a story will most likely originate from the business stakeholders, many team members may have a hand in authoring the story card, including project managers, tech leads, analysts, and testers.

Initial Release Planning

- Planning within Agile is iterative
- Regardless, some planning must be done up front for a variety of reasons:
  - Build a release plan the organization can plan around
  - Resolve upfront architectural tradeoffs so implementation conforms to an overall architectural vision
  - Prototype / wireframe a UI to get early feedback from stakeholders on the requirements
  - Prepare development & testing platforms accordingly
- Detail out initial Sprint(s) and projected releases for more detailed budgeting purposes

Roles of Core Planning Team Members

- *Business Stakeholder(s)* – Own the product vision and helps make sure the requirements meet the needs of the end customer
- *Project Manager* – Responsible for the end-to-end planning process
- *Lead Architect* – Responsible for the initial architecture and scoping
- *Lead Business Analyst* – Acts as SME and documents requirements
- *QA Lead* – Responsible for overall test strategy
- *Web Designer* – Optional depending upon whether UI prototyping is involved

Others participate in the process as required and necessary

Building a Release Plan

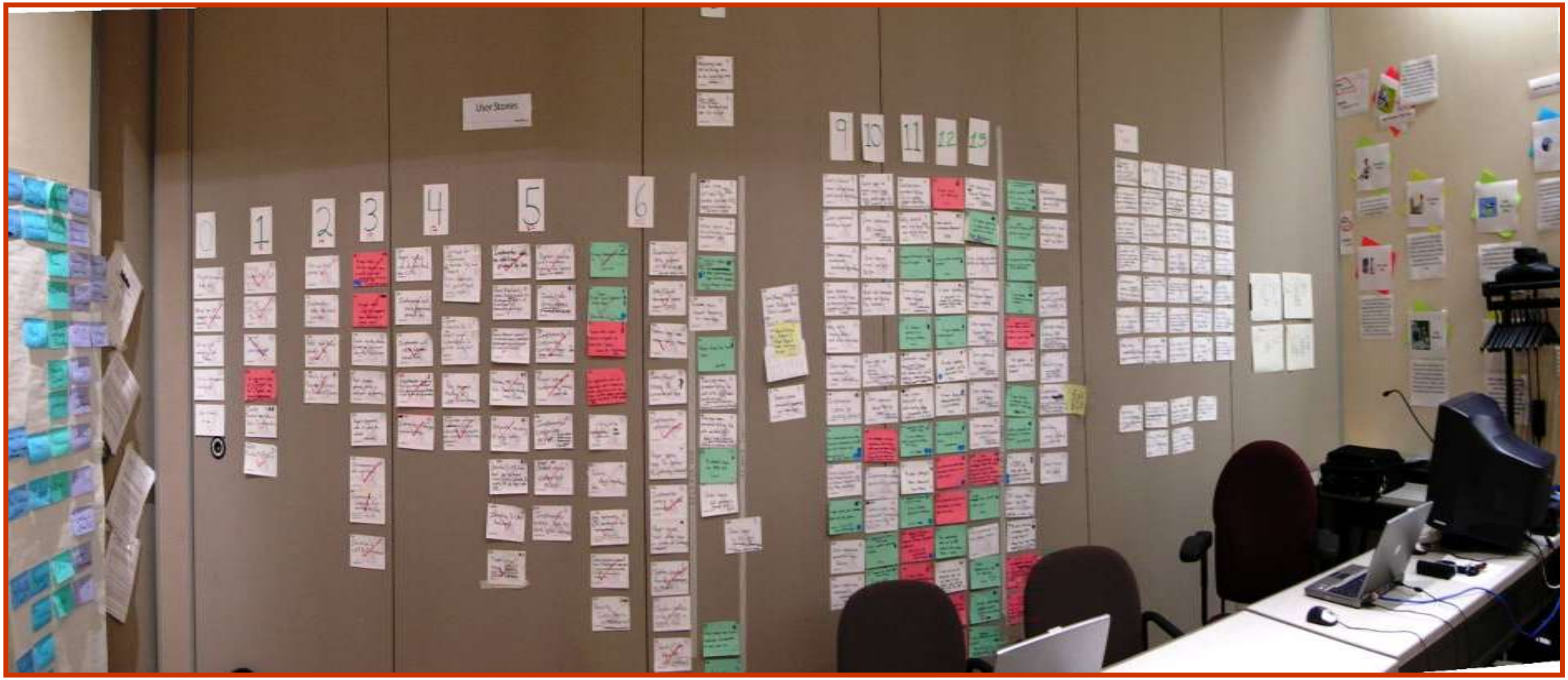
- Release plans that include Sprints* are built using:
  - Prioritized stories that include Points estimates
  - Team size
  - A decision around Sprint duration
  - A decision around how much functionality is enough to justify a release

- Sprint duration
  - Tradeoff between cost of change and organization’s agility
  - Typically 2 – 4 weeks in duration

- Release decision
  - Tradeoff between cost of deployment/release and market dynamics
  - Releases vary from daily to 3 months (huge variation!)
  - Releases are now a business decision

**A Sprint is a development iteration in SCRUM terminology*

The Team Room Approach to Release Plan Mgmt



Pros – very visible and tangible, great for co-located teams, easy to modify
Cons – not under version control, harder for distributed teams to visualize, takes space

Agile Development

Goals of Development Sprints

- Construct a piece of software that fully meets the demands of the stakeholders.
- Ensure that the software is high quality and production ready.
- Have a code base that is well architected, commented and easily maintainable.
- Establish a sustainable development process that matches the skills and work habits of your development organization.
- Establish continuous integration infrastructure in time for production deployment.

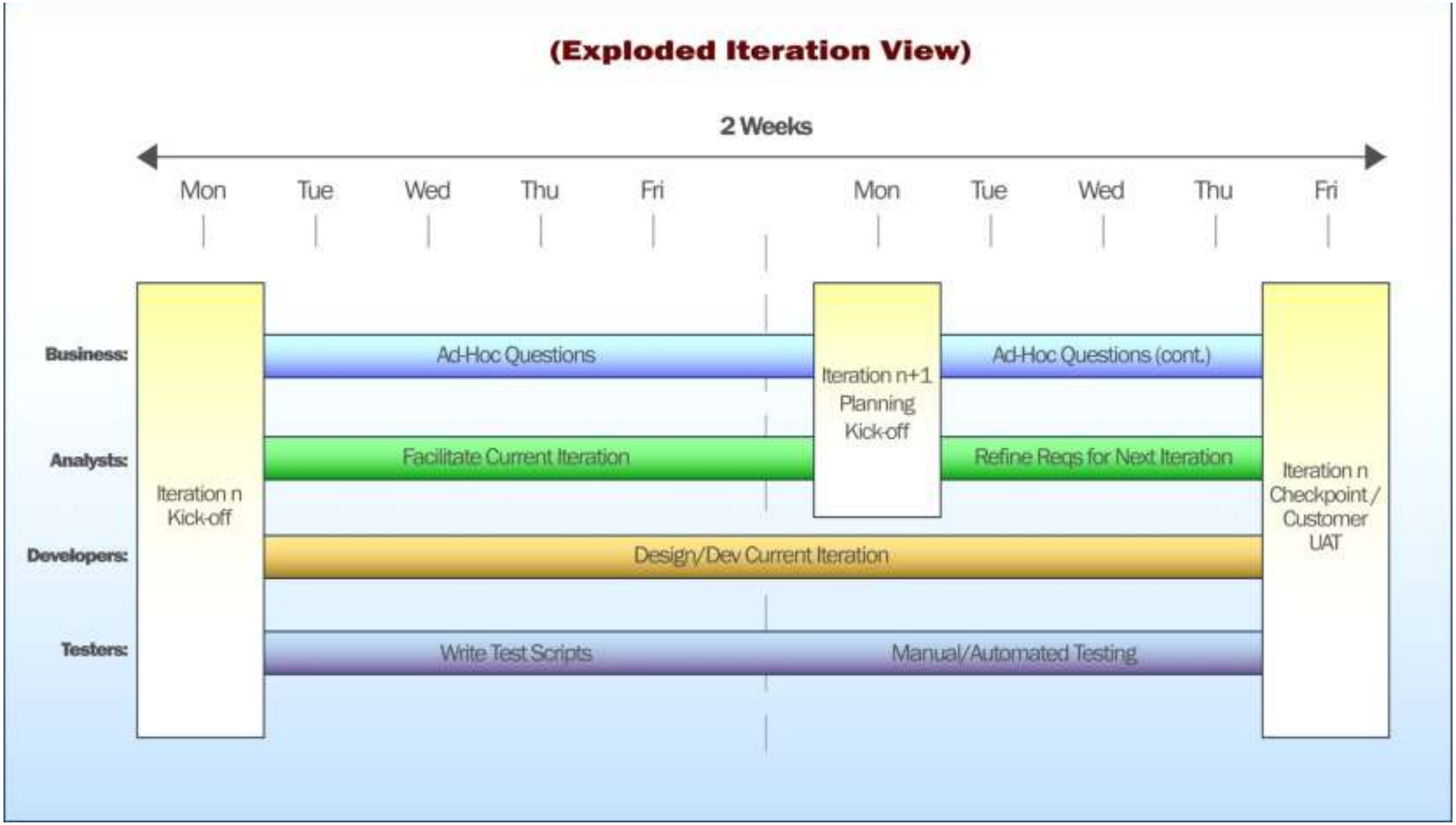
Am I Ready to Begin?

- Have Initial Planning deliverables been accepted by stakeholders?
- Are there at least one Sprints scheduled to full capacity?
- Is the management and development infrastructure established sufficiently to begin?
- Have the product stakeholders been identified and given full authority on the direction of the software to be developed?
- Is a dedicated development team identified?

Typical Roles

- Project Manager – responsible for the day to day functional delivery of the software, managing project schedule and priorities, and working with stakeholders to resolve any project issues
- Architect – responsible for coding, design and architecture standards review and compliance, solutions definition and overall performance characteristic of the software
- Analyst – supports project manager in the proper definition of requirements
- Development Lead – responsible for day to day technical implementation of the software and technical management of developers
- Developers – responsible for technical implementation of the software
- QA/Test Lead – responsible for the day to day testing, verification and validation of the software, compliance, management of the testers and automation of the test cases
- Testers – responsible for the testing, verification and validation of the software and the automation of the test cases
- Business stakeholder or proxy – available when needed to answer questions regarding the product, market, customer needs

Iterative Development Process



(Exploded Iteration View)
2 Weeks
Mon
Tue
Wed
Thu
Fri
Mon
Tue
Wed
Thu
Fri
Business:
Analysts:
Developers:
Testers:
Iteration n Kick-off
Ad-Hoc Questions
Facilitate Current Iteration
Iteration n+1 Planning Kick-off
Ad-Hoc Questions (cont.)
Refine Reqs for Next Iteration
Design/Dev Current Iteration
Write Test Scripts
Manual/Automated Testing
Iteration n Checkpoint / Customer UAT

Team Communication during Agile Development

- Effective communication between all team members is absolutely critical to a successful Agile project
- A meeting rhythm should be established to assure communication happens at least at key Sprint junctures
- Important team meetings include:
  - Sprint kickoffs
  - Daily standups
  - User acceptance testing
  - Retrospectives

Sprint Kickoff Meeting

- Occurs at the beginning of a new Sprint. May require as little as a couple of hours or a whole day.

- The purpose of the Sprint kickoff is to:
  - Allow the business team to communicate the very latest understanding of the scheduled stories.
  - Provide the development team with a chance to ask questions about the stories to the business team.
  - Allow the development team to break down the stories into tasks.
  - Enable the development team to refine the initial story estimates based on the tasks.

Daily Standup Meetings

- The daily standup occurs at the start of each day. Intended to last no more than 15 minutes.
- Participants are encouraged to actually ‘stand up’ during the meeting so that the meeting stays short.
- The purpose is to allow each participant to quickly communicate:
  - What did I do yesterday?
  - What do I plan to do today?
  - What obstacles are standing in the way of achieving my goals today?
- One intent of this meeting is to identify potential issues as soon as possible.

Sprint Planning Meeting

- Sprint planning occurs sometime after the Sprint kickoff. In a two week Sprint, planning is ideally completed between the end of the first week and the beginning of the second week.
- The purpose of Sprint planning is to:
  - Analyze and discuss the stories that are scheduled for the next Sprint.
  - Adjust the list of scheduled stories based on various feedback channels, such as UAT.
  - Provide enough detail in the requirements so that the stories can be broken down into tasks during the next Sprint kickoff.

User Acceptance Testing (UAT)

- User Acceptance Testing usually occurs on the last day of the Sprint.
- This meeting:
  - Allows the stakeholders, in a hands on way, to use the features newly developed during the Sprint.
  - Allows the stakeholders to provide feedback on the features.
  - Identify bugs that may have been missed during development.
  - Typically spurs ideas for new features which go onto the Wish List

Retrospectives

- The retrospective takes place at the end of each Sprint, usually after the UAT.
- This meeting allows the team to talk about what went right and what went wrong during the Sprint.
- This meeting can often follow a fixed format (such as SAMOLO). But it's more important that it's conducted in a manner that encourages the participants to provide honest feedback.
- It is intended that the lessons learned in the retrospective are applied in future Sprints.
- Team members are held accountable for action items assigned during retrospective discussions.

SAMOLO

Common format for conducting a retrospective

- Same As (SA) – What should we keep doing the way we are doing it?
- More Of (MO) – What should we do more of than we’ve done in the past?
- Less Of (LO) – What should we do less of than we’ve done in the past?

- Other similar formats
  - Keep doing, Start doing, Stop doing
  - Thorns and Roses

The Role of Project Manager

- The Project Manager has final authority on all project decision
- During each Sprint, the PM:
  - Leads Sprint kickoff
  - Leads Daily Stand-ups
  - Leads UATs
  - Leads Retrospectives
  - Leads Iterative Planning Process
  - Removes Hurdles / Barriers from Team
- On small projects, a PM may also:
  - Participate in requirements definition
  - Participate in testing efforts

The Role of Business Analysts

- Business Analysts are responsible for assuring that requirements meet the needs of the customer

- During each Sprint, BAs:
  - Detail requirements for the next Sprint(s)
  - Provide subject matter expertise to development and testing on product requirements and customer needs
  - Review test plans for completeness

- On small projects a BA may also:
  - Participate in testing

The Role of Software Developers

- Software developers are responsible for software development
- During each Sprint, software developers:
  - Perform team-based design
  - Implement the application
  - Developer testing
  - Refactoring
  - Setup / maintain Continuous Integration environment
- On smaller projects a software developer may also:
  - Participate in software testing of functionality

Team-Based Design

- In team-based design, developers invest as little as possible in upfront design.
- They do not anticipate problems down the road that may or may not happen.
- Assume that the simplest design will work until proven otherwise.
- Involve all members of the team in system design. Multiple perspectives will ensure that as many potential issues are identified and addressed as early as possible.

Implementation – Coding Standards

- At the start of the project, the team should discuss which coding standards they agree to adopt. This can be a prickly issue.

- Some developers feel very strongly about how code should be written. Some goals of adopting coding standards:
  - Avoid petty disagreements during pair programming
  - Improve code readability
  - Enhance refactoring productivity (reducing the cost of change)
  - Enhance code maintainability

- Some of the most common coding standards address:
  - Naming conventions
  - Code organization and layout
  - Use of code comments
  - Avoidance of language specific problem constructs

Implementation – Pair Programming

- In pair programming, one developer works at the keyboard while the other follows along.
- The typing developer is focused on the code mechanics while the other is thinking at a broader level about what to do next. The second developer is also better able to spot bugs before they are deployed.
- This practice helps to spread domain and technical knowledge across the various members of the development team.
- Most teams prefer a balance between pair and individual programming that works best for them.

The Role of Software Developers

- The following benefits can be realized from effective pair programming:
  - Continuous code review
  - Cross pollination of developer knowledge
  - Increased code quality

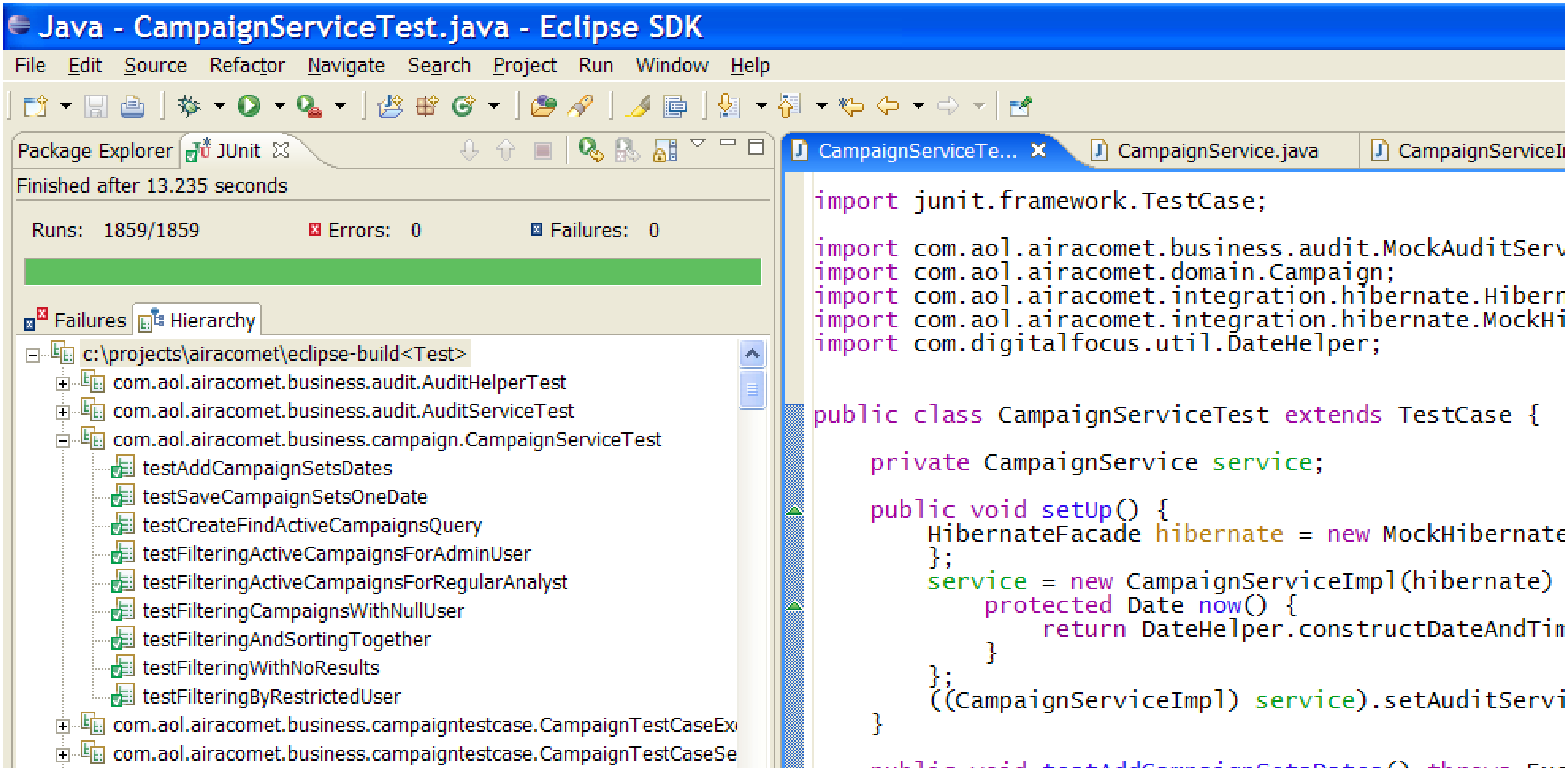
Developer Testing – Test Driven Development

- Test Driven Development (TDD) is a software development technique in which developers consider tests as part of their specification when building software
  - Test First Development – creates tests before creating code
  - Test Influenced Development – outlines positive and negative test scenarios while thinking through implementation
- Tests are written to:
  - Test the functionality of units
  - Test interfaces between implemented components
  - Validate bug fixes
  - Validate refactoring
- Tests are automated for use during code build / test cycles

Developer Testing – Unit Testing

- A unit test is a piece of software that validates the correctness of a small unit of production code in a well-defined, repeatable manner.
- The adoption of unit tests in an agile environment hinges very much on the early availability of continuous integration tools.
- This allows the tests to be run continuously and gives the developers confidence to make changes to production code.
- Unit tests also serve as part of the documentation of the code.

Developer Testing – Automated Unit Testing



Java - CampaignServiceTest.java - Eclipse SDK
File Edit Source Refactor Navigate Search Project Run Window Help
Package Explorer
JUnit
CampaignServiceTe...
CampaignService.java
CampaignServiceI
Failures
Hierarchy

Finished after 13.235 seconds

Runs: 1859/1859 ☒ Errors: 0 ☒ Failures: 0

```
c:\projects\airacomet\eclipse-build<Test>
    com.aol.airacomet.business.audit.AuditHelperTest
    com.aol.airacomet.business.audit.AuditServiceTest
    com.aol.airacomet.business.campaign.CampaignServiceTest
        testAddCampaignSetsDates
        testSaveCampaignSetsOneDate
        testCreateFindActiveCampaignsQuery
        testFilteringActiveCampaignsForAdminUser
        testFilteringActiveCampaignsForRegularAnalyst
        testFilteringCampaignsWithNullUser
        testFilteringAndSortingTogether
        testFilteringWithNoResults
        testFilteringByRestrictedUser
    com.aol.airacomet.business.campaigntestcase.CampaignTestCaseEx
    com.aol.airacomet.business.campaigntestcase.CampaignTestCaseSe
```

```
import junit.framework.TestCase;

import com.aol.airacomet.business.audit.MockAuditServ
import com.aol.airacomet.domain.Campaign;
import com.aol.airacomet.integration.hibernate.Hibern
import com.aol.airacomet.integration.hibernate.MockHi
import com.digitalfocus.util.DateHelper;


public class CampaignServiceTest extends TestCase {

    private CampaignService service;

    public void setUp() {
        HibernateFacade hibernate = new MockHibernate
        };
        service = new CampaignServiceImpl(hibernate)
            protected Date now() {
                return DateHelper.constructDateAndTim
            }
        };
        ((CampaignServiceImpl) service).setAuditServi
    }
```

The Role of Software Developers

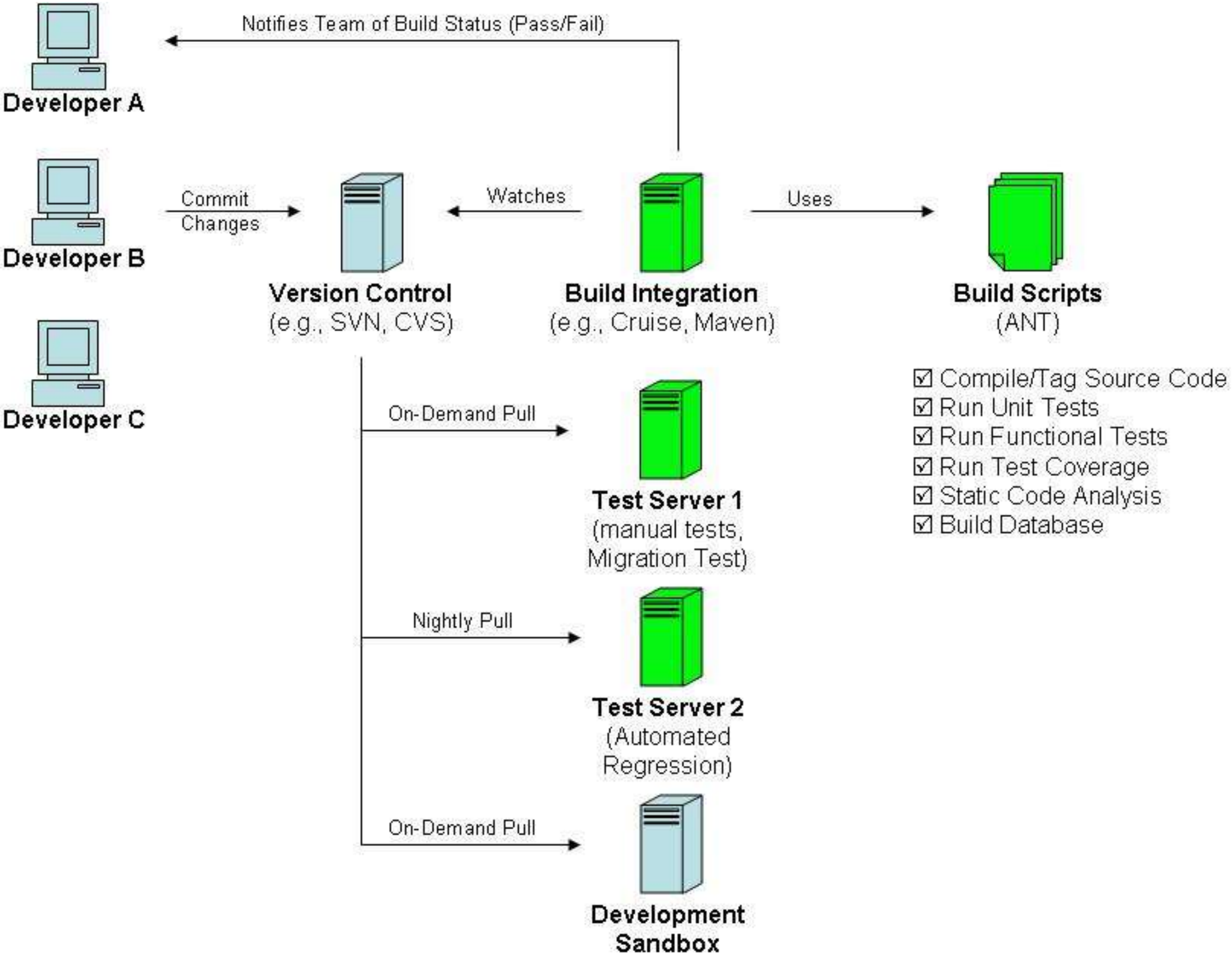
Refactoring

- As stories are completed, the code moves in directions that the development team did not anticipate.
- Sometimes code gets duplicated or unnecessarily complicated.
- If a developer is about to implement a story by increasing the amount of inefficient code, this is the appropriate time at which to refactor the code.
- Refactoring should be done in small amounts in order not to adversely impact the delivery schedule. Larger refactors should be broken up over one or more Sprints.

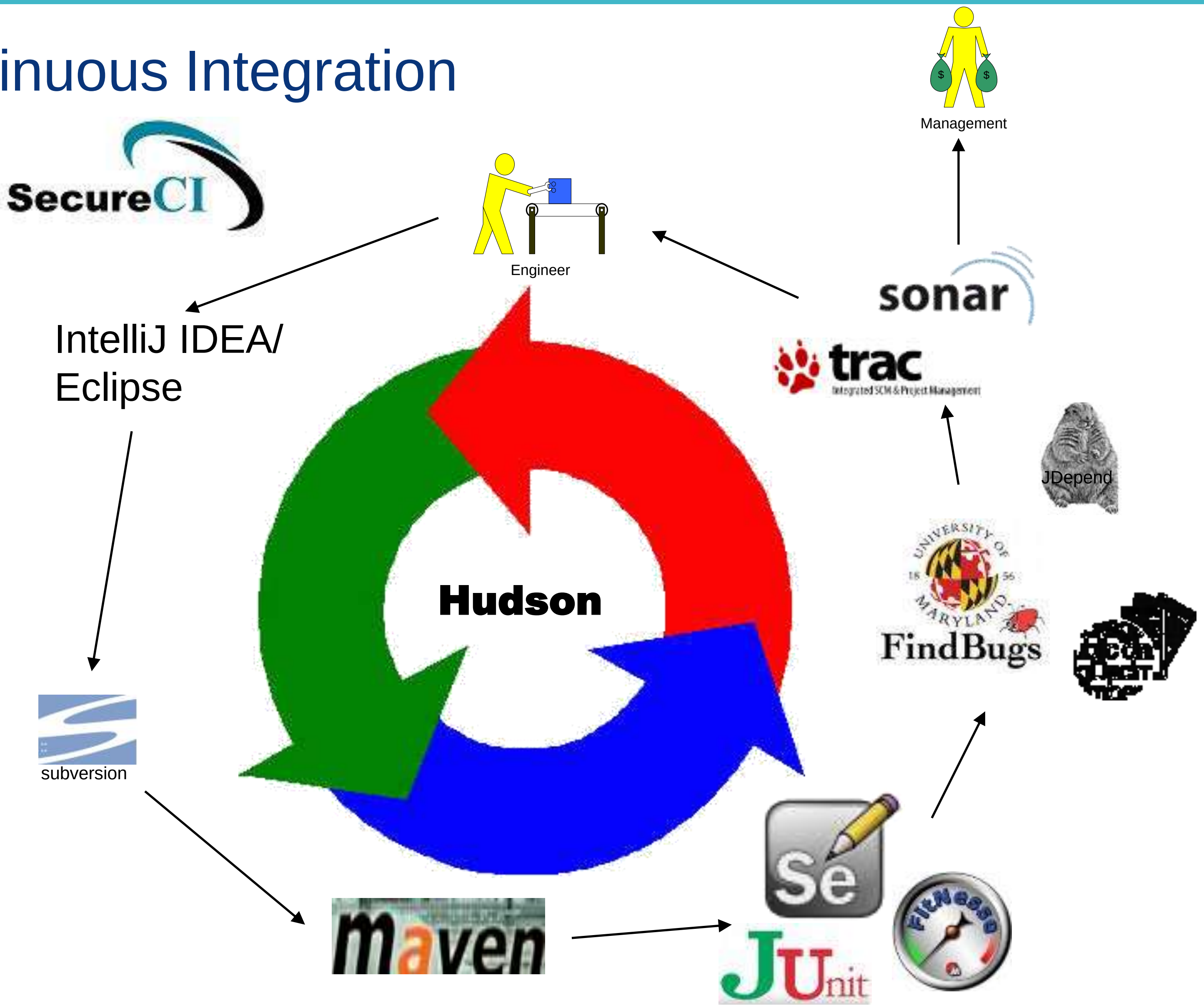
Continuous Integration

- Continuous Integration is the practice of frequently integrating the source code for a project or group of related or dependent projects
- The purpose of Continuous Integration is to keep code and build quality high and make delivery of the application or system easier because the build is performed enough to keep it clean and to work out any problems
- In a Continuous Integration process, after a successful build, a set of tests are run against the resulting software
- These tests range from unit and functional tests to integration, performance and security tests
- After the tests are run, many Continuous Integration processes apply code analysis tools to the code base in order to find code and security defects not detected by the tests.

Continuous Integration



Continuous Integration



Automated Deployments

- The product team must carefully consider whether or not CI should be utilized to deploy to production.
- The team may choose not to use CI for production deployment for the following scenarios:
  - Large or complicated applications
  - Companies with policies that prevent auto deployment
  - Applications where the exact timing of the release must be strictly controlled
- The team might choose CI deployments to production under the following scenarios:
  - Applications with smaller user bases
  - Internal applications
  - Applications developed and owned by smaller companies

The Role of Software Testers

- Software testers are responsible for testing the application above and beyond developer testing

- During each Sprint, software testers:
  - Test software
  - Automate tests
  - Test plan for future Sprints
  - Analyze requirements for testability

- On smaller projects a software tester may also:
  - Participate in business analysis

Software Testing – Agile Testing

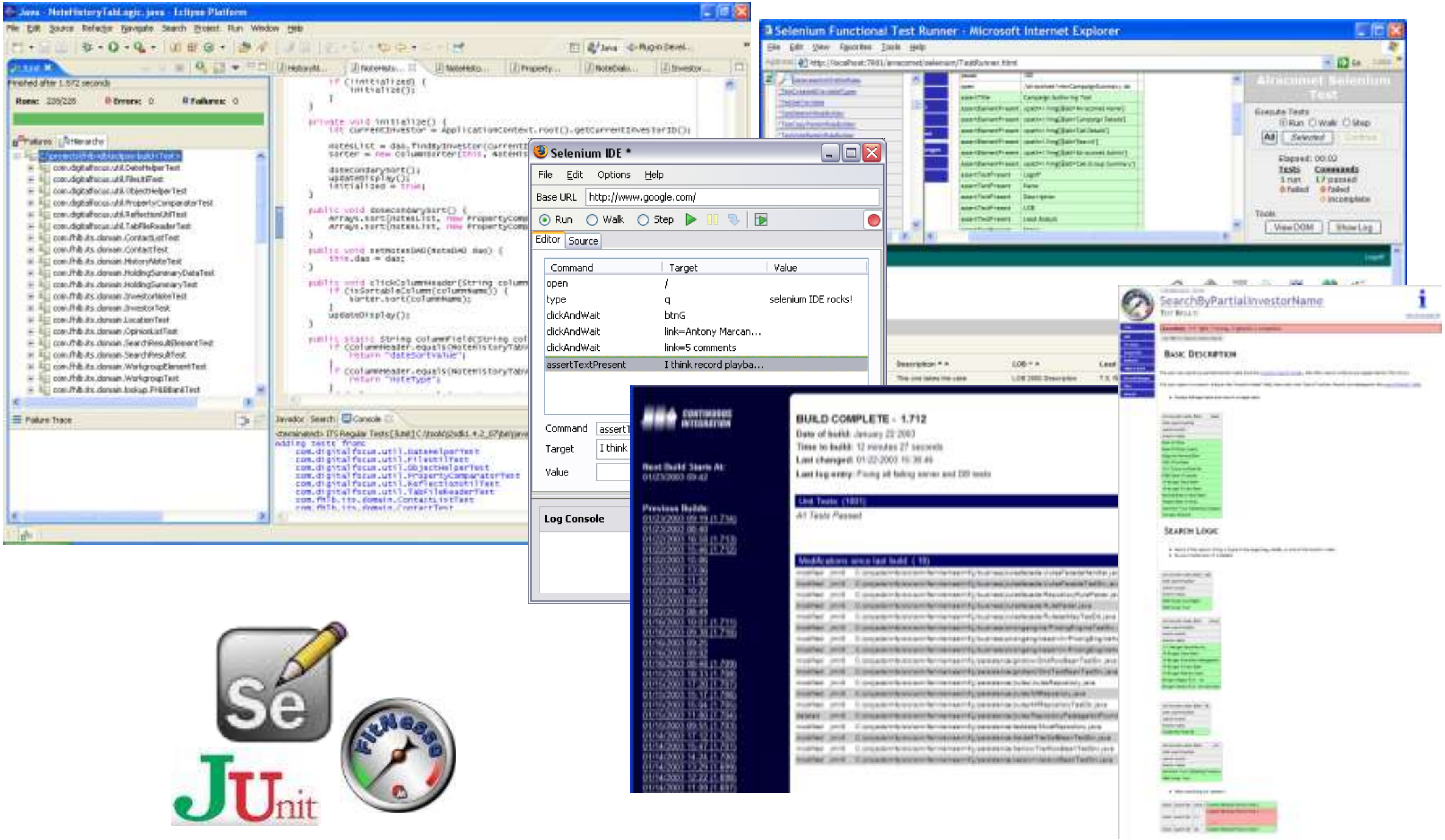
- An essential part of Agile is continuous testing. Rather than delivering large amounts of untested code at the end of a Sprint or release, it is essential to test on a daily basis as the code is being developed.
- Software testing performed by software testers
  - Testing of key components, end-to-end stories, use cases, and feature sets for each Sprint
  - Testing of non-functional requirements (load/performance, security, fault-tolerance, etc.) for releases
  - Coordinate User Acceptance Testing and capture results
- Software testers work very closely with software developers on all testing tasks

Software Testing – Test Automation

- Effective test automation is achieved by:
  - Applying automation only where there is a clear ROI for doing so
  - Often times NOT test execution -> automating test setup, test results validation, test cleanup are often highly effective
  - A test must be run 3 – 10x unchanged before there is a return for automating it
- Structuring and treating test automation scripts as software that must be designed, developed, tested, and maintained
- Leveraging test automation infrastructure (both off-the-shelf and custom) across all appropriate development projects

The Role of Software Testers

Software Testing – Test Automation Tools



Selenium IDE *

Base URL http://www.google.com/

Command	Target	Value
open	/	
type	q	selenium IDE rocks!
clickAndWait	btnG	
clickAndWait	link=Antony Marcan…	
clickAndWait	link=5 comments	
assertTextPresent	I think record playba…	

Command assertT

Target I think

Value

BUILD COMPLETE - 1.712

Date of build: January 22 2003

Time to build: 12 minutes 27 seconds

All Tests Passed

Software Testing – Which tests to automate?

- Part of the test planning process is deciding which tests or parts of tests to automate. Some criteria that should be considered:
  - Are the tests easy to automate? What makes a test easy to automate is the ability to script not only the behavior but also the analysis of the results to determine if the test passed or failed.
  - How often is the functionality or API point, used by the users or consumers of the product? - The more popular, prominent or useful the functionality under consideration is the more benefit in automating it.
  - How risky is the functionality? –No matter what the definition of risk, the goal in automating that functionality is to help mitigate the risk. One definition of risk is what features are hardest to implement correctly. The added assurance of automated tests can help mitigate the risk by having the tests run more frequently than they would without automation.
  - Is the cost of automating the functionality less than the cost of manual testing the functionality though the life of the project?

The Role of Business Customer / Proxy

- Represents the customer base on the project team

- During each Sprint, the business customer:
  - Answers ad-hoc questions on the product and its requirements
  - Participates in User Acceptance Testing

- Sometimes the appropriate business customer isn’t available to be involved in the project. A business “proxy” acts on the business customer’s behalf
  - Must have the authority to make decisions

Wrap-Up

Agile books we recommend

- Beck, Kent, “Extreme Programming – Embracing Change”, Addison-Wesley Professional, 2004
- Cohn, Rob, “User Stories Applied”, Addison-Wesley Professional, 2004
- Cohn, Rob, “Agile Estimating & Planning”, Prentice Hall PTR, 2005
- Crispin, Lisa, “Agile Testing – A Practical Guide for Testers & Agile Teams”, Addison-Wesley Professional, 2009
- Duvall, Paul, “Continuous Integration: Improving Software Quality and Reducing Risk”, Addison-Wesley Professional, 2007

Questions?

- Contact information:
  - Jeffery Payne, Coveros Inc.
  - 703-431-2920
  - jeff.payne@coveros.com