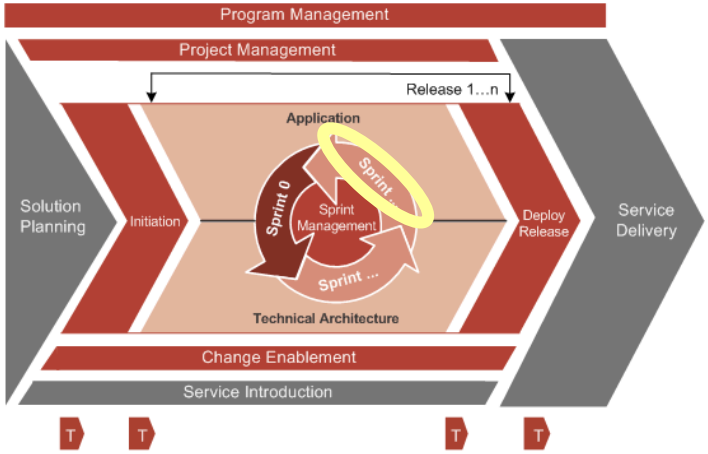
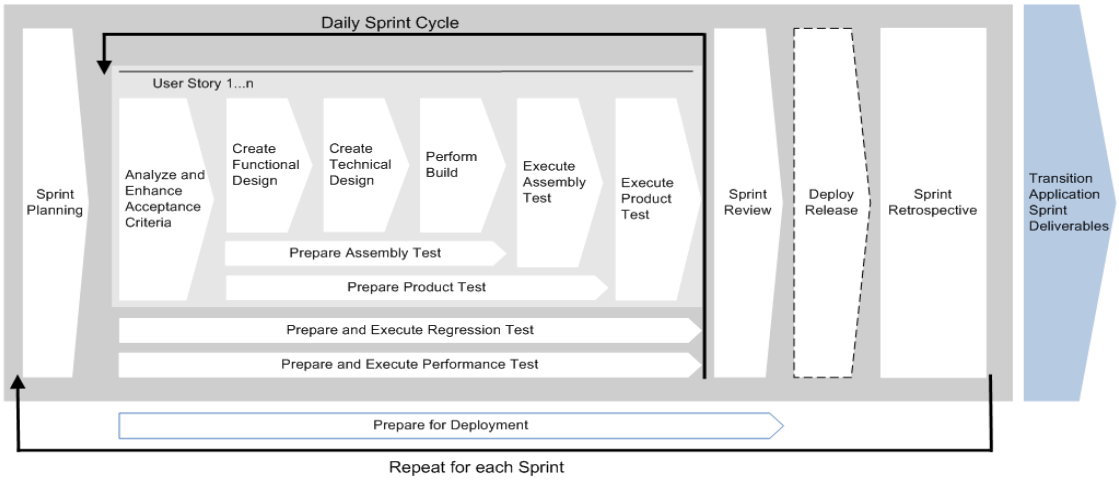

accenture technology
Agile University: Delivery School
Sprint Application & Management
High performance. Delivered.
Strategy | Digital | Technology | Operations

Agenda

- Sprint Application
- Sprint Management
- Testing in Agile Vs. Traditional Approach
- Roles
- Work Products
- Practices

Sprint Application - Overview





Sprint in ADM involves:

- **Design**, **build** and **test** of the User Stories selected in Sprint planning
- **Assembly**, **Regression** and **Application Product tests** of the stories
- Preparation for Deployment
- **Sprint Review** with the Project Stakeholders and **Sprint Retrospective** with the Project Team

Sprint Planning

Sprint Planning meeting will be held in two parts: '**What**' and '**How**'

What: focuses on what the team is being asked to build

- **Identify** and **commit** the **User Stories** to the current Sprint

How: focuses on how the team plans to build desired functionality

- Create/update the Sprint Backlog
- Decompose items into tasks and estimate for the tasks
- Update the Release plan

Together for the two parts, duration can be 4-8 hours

Sprint Execution – 2 Weeks Sprint

Mon	Tues	Wed	Thurs	Fri
1	2	3 **Sprint Start** Sprint Planning User Story Tasks breakdown User Story Tasks effort estimation	4 Build Start Sprint Low Level Design, DB Setup and Build Sprint QA Test Planning and Test Scripts Daily Stand-Up	
8	9	10	11	12 Product Backlog Refinement for Next Sprint Build End
15 Story Point Estimation for Next Sprint Sprint QA Test, Regression Test Execution Daily Stand-Up	16 **Sprint End** Sprint Demo Sprint Retrospective Sprint Planning for Next Sprint	17	18	19

Sprint Low Level Design, Build and UT

Sprint QA Test, Regression Test Execution

Daily Stand-Up

Agenda

Sprint Application

Sprint Management

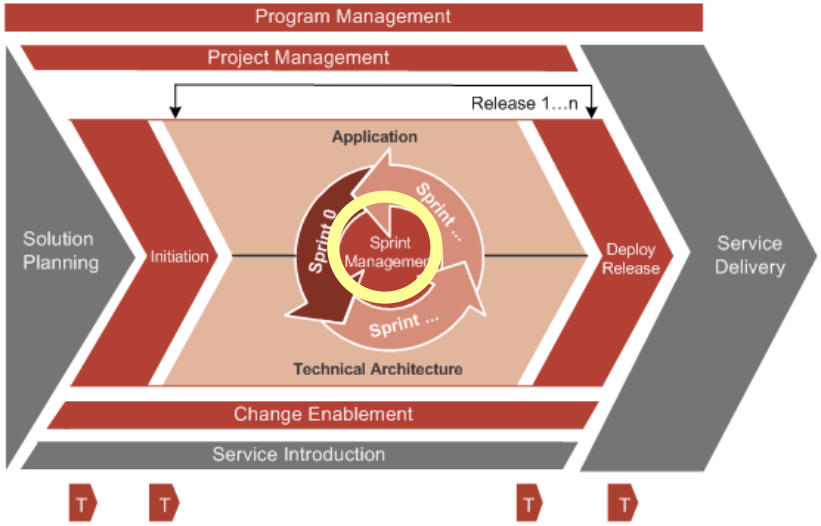
Testing in Agile Vs. Traditional Approach

Roles

Work Products

Practices

Sprint Management – Overview



- Conduct Sprint Activities
- Conduct Scrum of Scrums
- Conduct High Frequency Transition
- Measure Sprint Progress
- Manage Impediments
- Maintain Product Backlog
- Cancel Sprint

Sprint Management

- **Sprint Activities**
- **High-frequency transitions**
- **Sprint progress**
- **Backlog**
- **Impediments**
- **Cancelling a Sprint (if the Sprint is not viable)**

Agenda

Sprint Application

Sprint Management

Testing in Agile Vs. Traditional Approach

Roles

Work Products

Practices

How is Testing in Agile different?

Traditional Testing	Agile Testing
Testing occurs at the end	Testing occurs iteratively within each Sprint
Communication is formal between the groups.	Communication cannot always be formal as interaction is required between the groups.
Test Automation is optional	Automation is highly recommended
Testing from Requirements perspective	Testing from Customers perspective
Detailed Test Plan exists	High Level Test Plan exists

Challenges of Testing in Agile

- Changes in **requirements**:
  - Requirements evolve over time and testing teams have to keep up with the changes
- Different **skills** for testers
  - Emphasis on Automation
- Lack of Detailed **Test planning**
  - Requirements and plan evolve over time, so the Test Plans have to keep up with the changing overall plans.
- Increased **Regression** Cycle
  - The Regression testing scope increases with each sprint
- **Reduced perspective**
  - Testers may lose overall system perspective while testing from sprint to sprint

Testing Strategies in Agile

- Consider using **Risk-Based testing**
  - Test cases can be added, removed and changed based on the requirements priority
- Create smart **automated** test scripts
  - Create automated scripts in terms of re-usable steps rather than one monolithic script
- **Plan** for the test activities
  - Allocate time for re-factoring test scripts
  - Add testing perspective to the selection of User Stories
- Re-use **Unit** as well as **Functional** tests for Regression
- Have a separate **Test Phase** just before the release ends

Agenda

Sprint Application

Sprint Management

Testing in Agile Vs. Traditional Approach

Roles

Work Products

Practices

Key Roles

- **Product Owner**
  - Presence mandatory in **Sprint Planning** to clarify the User Stories and priorities
  - Updates the Product Backlog
    - ✓ Enter new stories as they are discovered
    - ✓ Refines Epics as more details emerge
    - ✓ Re-prioritizes the Product Backlog Items when required
  - Availability as required throughout the Sprint to **clarify** User Story details
  - Provide **feedback** during the Sprint Review

- **Scrum Master**
  - Setup of the sprint processes and guides the team from the process perspective
  - Conducts daily stand-up meetings, notes down the **impediments** identified by team and works toward eliminating them

Key Roles

- **Team**
  - Estimates work required for User Stories selected by the Product Owner, from the Product Backlog, for inclusion in Sprints
  - During sprint planning, the Team decides the **scope** and also **estimates** and **plans** the sprint
  - Designs, builds and tests the User Stories
  - Transitions deliverables to other teams
  - Updates Sprint Backlog
  - Prepares for the **demo** and obtains feedback on the User Stories completed
  - Participates actively in the **sprint retrospective** and drives **continuous improvements**

- **Project Manager**
  - Liaison with the stakeholders to manage expectations
  - Metrics collection and publishing
  - Measures Sprint progress in terms of metrics captured at Project, Release and Sprint level
  - Day-to-Day operations
  - Conducts Scrum of Scrums meeting between clusters of teams

Agenda

Sprint Application

Sprint Management

Testing in Agile Vs. Traditional Approach

Roles

Work Products

Practices

Key Work Products

- **IE211 Sprint Backlog**
  - Key **output** of the Sprint Planning Meeting
  - Contains **Tasks, Estimates and Owners** for each of the User Stories included in the Sprint
  - Updated by the Team daily with estimated time to complete User Stories within an in-flight Sprint
  - Generates **Sprint Burn-down** chart

- **IE212 Sprint Estimate –** This is the estimate for the individual tasks obtained after breaking down the User Stories into smaller tasks.

- **AP496, AP493 Source Code and Executable**
  - Source Code designed, built and tested by the team for the agreed User Stories
  - **Build scripts** and **common test data** required to execute the source code

- **MG119 Sprint Metrics**
  - **Metrics** collected to measure the health of the project
  - Contains **Agile** specific as well as Managed Delivery Metrics
  - The measures and the metrics can be "**In-Progress**" (which are measured during the Sprint) or "**Outcome**" (which are measured at the end of the Sprint).

Key Work Products

- **IE217 Process Improvement Log**
  - Key output of the **Sprint Retrospective Meeting**
  - Log of **improvements** in processes / practices identified by the **team**

- **IE215 Impediment Log -** Log used by the Scrum Master to maintain and track the impediments or obstacles that come in the way of the Scrum's progress

- **AP215 Fit/Gap Analysis ****
  - Key **output** of Sprint 0
  - Contains all **potential gaps**
  - Used as a guiding point to evolve the **Product backlog** requirements during **Sprints**

- **AP322 Configuration Design ****
  - Repository for all **configurations** done in the **packaged software**

**** Packaged development Projects**

Agenda

Sprint Application

Sprint Management

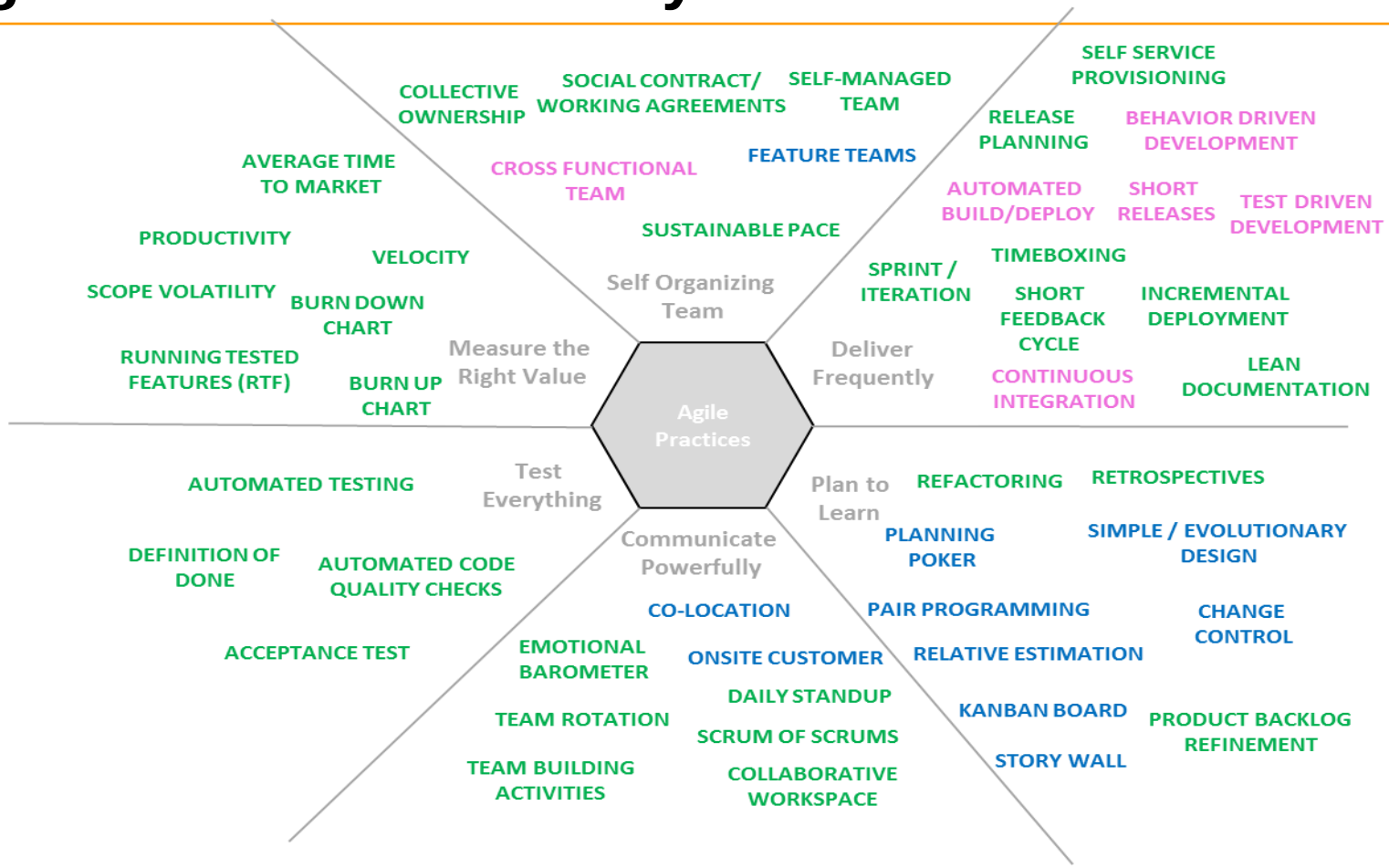
Testing in Agile Vs. Traditional Approach

Roles

Work Products

Practices

Agile Practices Summary

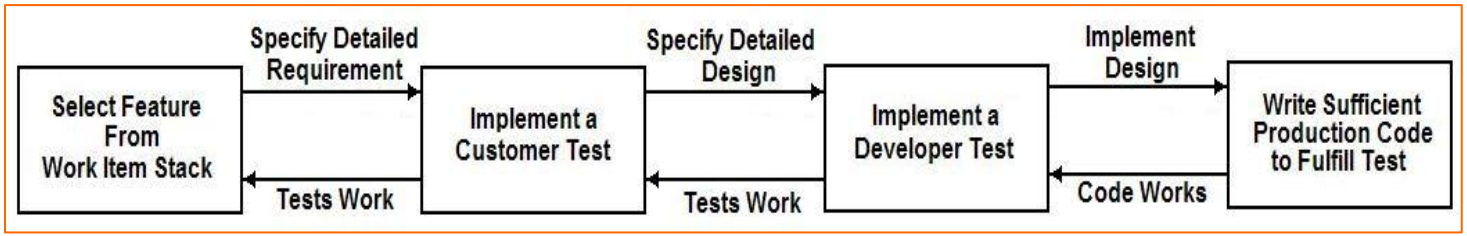


GREEN – Easily integrated in off-shore Package Agile engagements and add value.
BLUE – Guidelines will be provided but these practices are difficult to implement in offshore and should not be mandated
PURPLE – These practices are difficult to implement in Package development, except for E-Commerce where all these would be GREEN

Fig: Agile Practices for Custom and Packaged Development

Test Driven Development

- TDD is a programming practice that has both **specification** and **validation** aspects
- TDD is the combination of **test-first development** (TFD) and **re-factoring**
- **Test First Development** – Write the test first, then write the code that passes that test Process of implementing a new feature via a test-driven approach



Source: http://www.drdobbs.com/architect/205207998?cid=Ambysoft

Test Driven Development

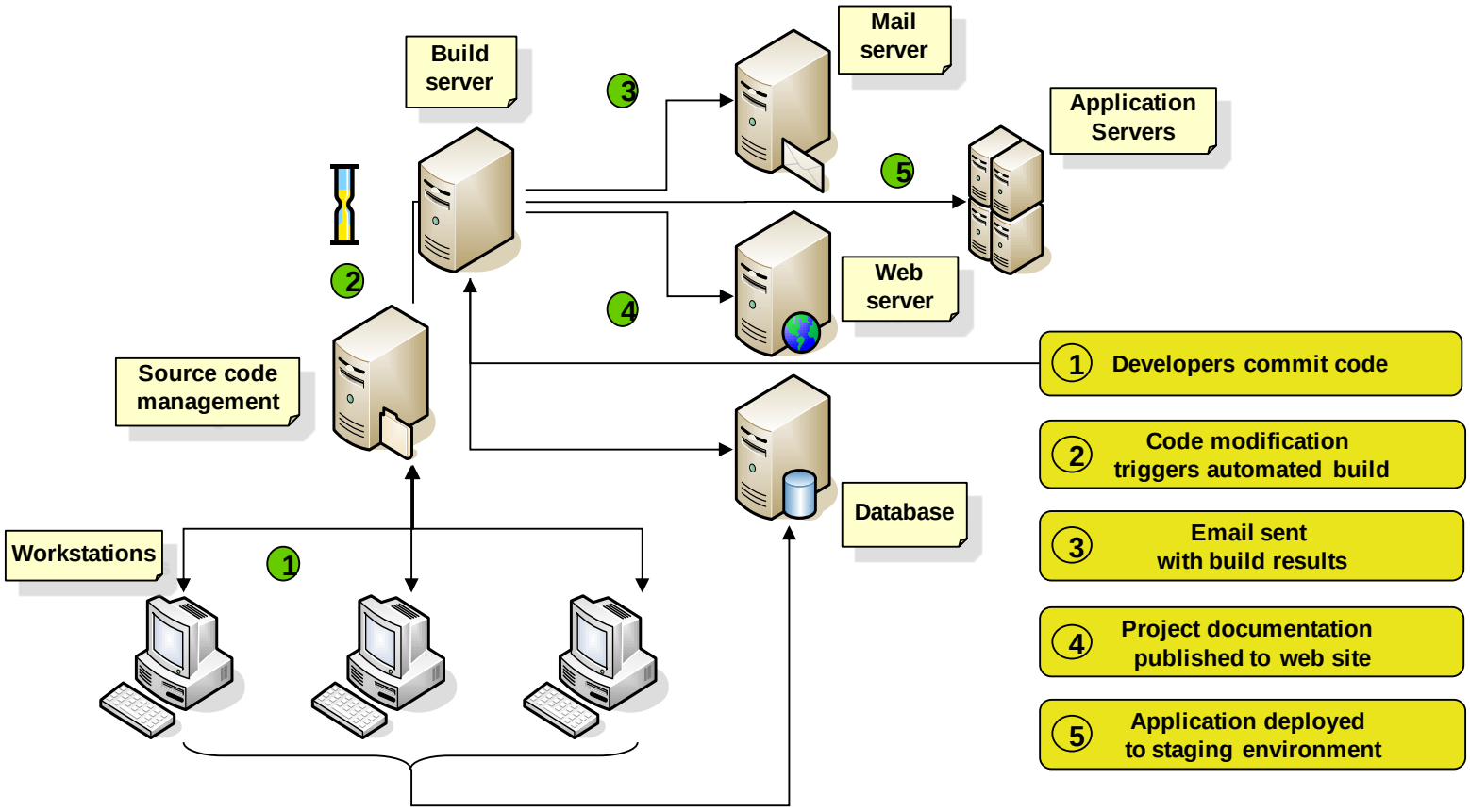
- Three types of test cases:
  - Positive
  - Negative
  - Exception test cases

- Advantages of Test Driven Development
  - Provides significant **code coverage**
  - Helps **improve the contract/interface** of the classes
  - Provides disciplined approach to coding and makes the code more robust
  - Helps identify the problems in code quickly

Test Drvien Development[TDD]:
https://search.accenture.com/search.aspx?aid=IKN&k=tdd&origin=kxHome&suborigin=Home

Source: http://www.drdobbs.com/architect/205207998?cid=Ambysoft

Continuous Integration Process



Source: https://collaboration.accenture.com/display/ADSJ/Continuous+Integration

Continuous Integration Servers:
https://kx.accenture.com/repositories/ContributionForm.aspx?path=C25/41/4&mode=Read&fn=Continuous_Integration_Servers_POV

Advantages of Continuous Integration

- Detect Errors Rapidly
  - Reduces Development Downtime
- Full Potential Achieved with Test-Driven Development
  - Automatically Execute Unit Tests at Check-in
- Runs Integration Tests on Production Mirror
  - Build Server ≈ Production Server
- Runs Application Builds Outside IDE
  - Automated and Consistent
- Repeatable and Automated Process

Agile Packaged Software Project Challenges

	Challenges	Benefits
Implementation Projects Greenfield Implementations	• Extensive dependencies (organization wide) • Heavy analysis and design phase	• Early visibility of business value • Ability to reprioritize scope and correct project course • Increased visibility in project status
Upgrade Projects Upgrade existing package implementation to new version	• Late realization of customer value • Define and validate tangible sprint/ release outcome criteria	• Identify risks early • Increased team collaboration • Increased visibility in project status • Ability to reprioritize scope and correct project course
Additional Function Implementation New implementation on existing package	• Dependency on existing implementation	• Fast time-to-market • Increased team collaboration • Easily implement Agile best practices
Enhancement Projects Enhancement on existing systems integration engagement	• Limited client availability • Limited automation support • Complexities in data integration	• Early visibility of business value • Address high requirement volatility • Easily implement Agile best practices

Q & A