Introduction to Agile and DevOps Technical Practices

Agile software development is the process of software development where the application requirements and desired solutions evolve through the collaboration of small and cross functional teams involving the customer as well. This technique advocates adaptive planning, evolving solutions in iterative way, frequent delivery of usable application in increments and continuous improvement with testing and refactoring. This leads to rapid and flexible response by the development team to the change requirements at any stage.
DevOps is a software engineering practice and culture that aims at bridging the gap between the Development and Operations team by collaborative efforts so that the customers get quick releases, increased deployment frequency with more useful releases of the product in close alignment with business objectives.
The DevOps practice strongly recommends automation and monitoring at all steps of software development and release cycles from unit testing,
build, integration, acceptance testing, infrastructure management and releasing to deployment.

This course introduces agile philosophy and applying the agile practices technically from development to deployment and monitoring with the usage of tools and techniques.

Prerequisites
The participants with any of the following prerequisites

- Working on Java web application development
- Working on Testing(QA)
- Participants from build/release management
- Project Manager/Architect
- Operation team

Training Methodology
The theoretical topics are discussed interactively and practical coverage is done by practical demonstrations for each topic and hands on practical exercises which strengthens the concepts learned.
The course material includes pdf presentations, documents and sample programs with case study exercises.

Session Machine
Intel dual core compatible CPU with minimum 4GB RAM and 500GB HDD with Windows 7/8/10 64 bit OR Ubuntu Linux 16.04 64 bit. Adobe pdf reader, JDK 1.8 64 bit, Eclipse-Jee-Photon 64 bit, Apache Ant, Apache Tomcat 8.0,Jenkins 2.250 and Firefox browser to be installed.

The participants have to install this on their session machines for practical hands on work.

Live internet connection with reasonable speed and download permissions should be available in the training room.
The participants must have admin rights on their systems.

Course Duration: Two days.

Instructor: Prakash Badhe.

Course Plan
This day wise course plan is based on the discussions with the Oracle team.

Day1
SDLC Overview

- Waterfall model disadvantages
- Manage the requirement change in Waterfall model
- Delivery to the customer
- Fault detection and rectification

Agile Methodology

- Why Agile?
- What is Agile?
- When Agile?
- Agile techniques
- Forming 'The Agile Team'

The next topic is the theoretical discussion about Agile methodologies and practices.

Bird's Eye View about Agile Practices

- Being Agile
- Extreme programming practices
- The Dev and Ops combined in a team
- Small team size
- Pair programming
- Develop and Build in smaller increments
- Getting the quick feedback and correcting
- Iterative development with smaller increments
- Test Driven development - Test First approach.
- Refactoring the code and regression testing
- Continuous Testing
- Continuous Integration
- Acceptance Testing
- Continuous Delivery and deployment
- Collaboration between the Dev and Ops teams.

- Build automation

The next topics are the practical oriented discussion and demonstration about applying Agile practices in technical environment.

Test-Driven Development

- Why and what is TDD..?
- How to Adopt TDD
- Where to Begin TDD
- The Test First Approach
- The TDD cycle : Red-Green-Refactor
- Implement a case study in TDD way
- BDD and ATDD overview.

Application development in TDD way

- Case study introduction and overview
- Understand the requirements
    - Start with lowest minimum component.
    - Write first test case and let it fail first time.
    - Write minimum code to make the first test success.
    - Write more test cases and let them fail first time one by one.
    - Write minimum code to satisfy the test cases one by one.
    - Verify success with all the tests
    - Evolve the application in TDD way.
- Unused/dead code?
- Code coverage

Testing with Mock objects

- Code Dependencies
- Isolate dependencies during testing
- Testing in isolation with mock objects
- The Mocking process
- Assert and Verify test Invocations
- Record-Play-Verify
- The stub and proxy behavior
- Testing the Database objects in absence of real data access

Code Refactoring

- Why and when to refactor?
- Change the code structure without affecting functional behavior
    - Make it maintainable.
    - Make it more reusable and flexible.

    - Apply Design Patterns in refactoring process
    - Loose coupling with code abstractions
    - Re-test the re-factored Code
- Regression testing
- SOLID Design Principles overview

The participants work on the implementation of shared Case study in TDD way.

Day2

Build Management

- Code integration in teams
- Manual build and test execution
- Manual integration challenges and limitations
- Automate the build with Tool
- Build automation tools : Ant,Maven,Gradle overview
- Configure build tasks such as compile, package and invoke test cases
- Execute the build script

Build Management with Continuous Integration

- Source code repository (GIT/SVN)
- Automate the build and integration process
- Continuously Integrate and delivery for application releases
- The Continuous Integration Server : Jenkins
- Jenkins Overview
- Configure Jenkins and related tools
- Configure a new build job
- Configure Trigger for the build
    - Build on demand
    - Periodic build
    - Trigger from repository for change in the code base
    - Dependent jobs
- Pre-build Stages
    - Compile
    - Unit Test
    - Integration Test
    - Package for the deployable artifact
- Monitor the build job execution
- Generate test reports
- Add more Post build activities in the build job

    - Code coverage check
    - Code quality analysis
    - Package the build
    - Execute batch files/shell scripts
    - Notify the collaborators by email about the result of build
    - Trigger another build job
- Parameterized build jobs
- Distributed build with master-slave
- Jenkins clustering overview
- Overview on groovy scripting in Jenkins

The participants work on Jenkins to define new build job and execute.

The participants incorporate a change into the user story to understand the continuous integration process.

Production Environment

- Continuous delivery of the application releases.
- Apache Tomcat 8/Oracle Weblogic server
- Cluster with Load balancer
- Centralized log management
- ELK stacks and Docker containers
- The Kubernetes Cluster overview

The participants commit a small change into the user story implementation and build and deploy it from the CI server to understand the continuous automatic delivery process.

Application Delivery in Manual mode

- Production deployment environment
    - Remote Web Server
    - Docker Container environment
    - Cloud
- Manual delivery tasks
    - Manage the production deployment environment
    - Deploy the artifact to production
    - Invoke the acceptance test cases
    - Generate the acceptance test reports

Application Delivery by Ops Team

- Overview on the provisioning tool (Configuration Management) to create and manage the production environment. E.g. Puppet, Chef, Ansible etc.
- Deliver the application in smaller increments frequently to the production server
- Deployment scenarios

The participants write set of acceptance test scenarios for the application deployed and invoke it as part of post delivery process.

Acceptance Testing Overview

- Define the acceptance test specifications for the application in BDD way
- Cucumber with Selenium for acceptance testing.
- Ops team to execute the acceptance test scenarios post deployment
- Monitoring.

The Ops team will be monitoring the application in production environment for performance, functioning issues, metrics etc. and convey to the proper team for action.

Scaling and updates by Ops team

- Monitor and analyze for optimum performance
- Scale up/down the nodes in the cluster
- Deploy Rolling updates
- Blue-Green deployment
- Monitoring in production

****