DIGILEAF INC.

Leading Excellence Among Fellows

Course Outline

Enterprise Architecture Learning Track

Java Application Architecture

Modularity is critical in software architecture. It fills a gap that has existed in developing enterprise software systems in Java. This course discusses that gap and explores how modularity is an important intermediary technology that fills that gap. This course covers software architecture as applied to Java applications. It highlights modularity patterns. Participants will learn the importance and the techniques of modularity patterns in software architecture, which is used to increase modularity of Java-based applications to easily scale up large application systems. This course shall provide the essential elements needed to incorporate modular design thinking into Java development initiatives. This course shall present why modularity is a critical tool in your arsenal of design tools. A discussion of the OSGi framework is also covered to provide a holistic view of software architecture

Training Objectives

At the end of the course, the participants will be able to:

- Use modularity in software architectures to control the increasing complexity of software systems.
- Describe the different modularity patterns that can be used in software systems.
- Apply the right pattern based on the system's architectural needs.
- Examine the advantages and disadvantages of each pattern for a given architectural problem.
- Apply POMA and OSGi frameworks and use them in Java-based solutions.

Duration 4 day(s)

DIGILEAF INC.

Leading Excellence Among Fellows

Course Outline

Enterprise Architecture Learning Track

Java Application Architecture

Topics

Part 1 (Day 1 & 2)

I. Introduction
   a) Logical versus Physical Design
   b) Pattern Form
   c) Pattern Catalogue
II. The Case of Modularity
   a) Defining a Module
III. Facets of Modularity
IV. Enterprise Continuum and Architecture Reuse
   a) Runtime Model
   b) Development Model
   c) Today's Modularity
V. Architecture and Modularity
   a) Architecture Defined
   b) A Software Architecture Story
      i. The Ivory Tower
      ii. Turtles and the Tower
   c) The Goal of Architecture
   d) Modularity: The Missing Ingredient
VI. Complexity
   a) Enterprise Complexity
   b) Technical Debt
   c) Design Rot
   d) Cyclic Dependencies – The Death Knell
VII. Realizing Reuse
   a) The Use/Reuse Paradox
   b) The Reuse Disclaimer
VIII. Modularity and SOA
   a) Structural Flexibility
   b) Granularity – Architecture's Nemesis
   c) An Alternative View
IX. Reference Implementation
   a) Why No OSGi?
   b) Refactoring

Part 2 (Day 3 & 4)

I. The Patterns
II. Base Patterns
   a) Manage Relationships
   b) Module Reuse
   c) Cohesive Modules
III. Dependency Patterns
   a) Acyclic Relationships
   b) Levelize Modules
   c) Physical Layers
   d) Container Independence
   e) Independent Deployment
IV. Usability Patterns
   a) Published Interface
   b) External Configuration
   c) Default Implementation
   d) Module Facade
V. Extensibility Patterns
   a) Abstract Modules
   b) Implementation Factory
   c) Separate Abstractions
VI. Utility Patterns
   a) Colocate Exceptions
   b) Levelize Build
   c) Test Module
VII. POMA and OSGi
VIII. OSGi and Scala
IX. OSGi and Groovy
X. Future of OSGi