Design Patterns Thinking

SUMMARY

Design Patterns Thinking helps participants learn to use design patterns to support Agile development. Design patterns enable the emergent design that is needed on Agile projects. Although patterns at one time were thought of as a "design up-front technique, they can be used in Agile projects to encapsulate variations that are discovered over time. For development organizations to thrive in an Agile environment, it is critical that the systems they create are not vulnerable to changing requirements, business priorities, technologies, or market pressures.

This course addresses key questions in modern development such as how to design systems that have changeability as a fundamental quality and how the team can communicate effectively when a design is in a constant state of change.

DESCRIPTION

This course uses patterns in a new way. Historically, patterns have been thought of as part of an "up-front design" style of development. Now, we realize they are powerful tools in an Agile environment, where much is assumed to be unknown at the beginning of a project, and where the design is expected to emerge through the iterative process.

Patterns are examples of following the wisdom that guides good design. They emphasize the importance of creating encapsulation and using delegation to segregate systems to protect them from cascading change. When they were first suggested, computers were fundamentally slower, and technology resources (memory, disk space) were far more expensive than they are today. The style of design suggested by patterns is far more realistic. Patterns are far more useful as collection of best practices rather than simply "reusable solutions."

Participants learn to use patterns as part of a thought process that guides analysis, using "pattern orientation" as a way to understand an ever-changing problem domain.

Practical Experience Applying Lean Thinking

All Net Objectives consultants have hands-on practical experience with applying Lean thinking to the delivery of software technology solutions. Our approach is to show how Lean applies to an incremental realization of software features and systems and how to apply practices that improve the team's ability to deliver.

Participants also learn to consider patterns in the context of testability (specifically unit testing), commonality-variability analysis, and a technique called "refactoring to the open-closed." By combining these techniques into an overall paradigm of design, developers and teams learn to move a design forward confidently even when there is an expectation that the sand beneath their feet will shift.

LEARNING OBJECTIVES

In this course, you will learn:

- How to add functionality to designs while minimizing complexity
- What design patterns really are
- Twelve specific design patterns
- What code qualities you need to maintain to keep code flexible
- How to use design patterns to keep code quality high

COURSE OUTLINE

Day 1: Theory

- Examination of typical causes of project failure
- Example of a failed design
- Qualities of changeable code
- Testability as a Trim-Tab in development
- The principles and practices of professional development
- Wisdom from our field

CONTACT US
info@netobjectives.com
1.888.LEAN-244 (1.888.532.6244)

LEARN MORE
www.NetObjectives.com
portal.NetObjectives.com

Day 2: Application

- Commonality Variability Analysis (CVA)
- The Template Method as an example of CVA
- Using CVA to derive the Bridge Pattern
- Patterns in context: Adapter and Façade Patterns
- Re-solving the problem from Day 1 in a new, better way
- Emergence through encapsulation and patterns: Refactoring to the Open-Closed
- Group design exercise, Part 1

Day 3: Expansion

- Group design exercise, Part 2
- Aspects of flexibility
- The Analysis Matrix and the Abstract Factory
- Separation of use and creation
- Encapsulating construction
- The Singleton Pattern
- The Proxy Pattern
- The Decorator Pattern
- The Chain of Responsibility Pattern

COURSE LEVEL

Intermediate

TARGET AUDIENCE

Experienced developers, testers, designers, architects, technical leads, and those who manage development teams. They should have intermediate to advanced object-oriented experience. This course can also be delivered to teams using non-Agile methods. In this case it is tailored for their current methods while opening up the possibility of incremental design.

Less experienced developers will find value and should be included if part of an experienced team. In this case we provide pre-course material to help bring them up to speed. If the entire team is inexperienced in Object-Orientation, they will be better served with our Effective Object-Oriented Analysis and Design Course.

Project managers have often attended this course to learn the approach of their teams and have found that to be useful.

ATTENDEE MATERIALS

Course materials will be provided at the start of the class.

ROOM SETUP AND EQUIPMENT

No computer equipment is needed for this course.

Students need to sit at tables, 4-6 students per table.

One flip chart per table and a flip chart or whiteboard for the instructor.

A projector with screen.

COURSE LENGTH

3 days

PDU CREDITS

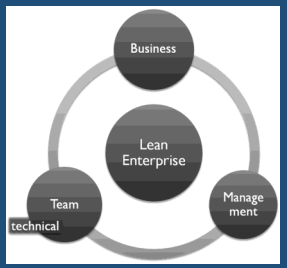
21 PDU Category B Credits

MAXIMUM NUMBER IN CLASS

24

NET OBJECTIVES

We are committed to delivering the principles, practices, and perspectives that businesses must know in order to maximize their return on their technology solution and software development efforts. We combine our experience and a time proven approach based on lean thinking to continuously extend the capability of what is possible in creating effective technology delivery organizations (IT or product). We provide these learned methods to our clients to assist them in achieving their goals and in assisting them in making their organizations more successful.



Full course descriptions may be found at
www.NetObjectives.com/training

Lean ● Agile ● Kanban Patterns ● TDD ● ATDD ● Assessments ● Consulting Training ● Coaching