ENGINEER-TO-ARCHITECT AND DURABLE THINKING SKILLS

From Developer to Architect

Level: Practitioner • 2 days (expandable to 3) • Virtual, In-person

Overview

There is no clear, well-marked path from senior developer to software architect, which is exactly why so many capable engineers stall at the threshold. The skills that make someone an excellent developer are necessary but not sufficient: architecture is less about writing the best code and more about making sound decisions under uncertainty, reasoning in tradeoffs, and communicating those decisions to people who will live with them for years.

This is a hands-on, practitioner course. It maps that transition in the order the skills build on each other: what architects actually do and how the role differs from senior engineering, how to think architecturally, how to identify and prioritize the architecture characteristics that shape a system, how to evaluate structural styles and their tradeoffs, and how to document, communicate, and defend the decisions you make, closing with a practical, personal roadmap for growing into the role. Rather than survey everything an architect might ever touch, it goes deep on the through-line: a way of thinking, the same durable, paradigm-proof thinking that keeps technologists valuable through every shift in the industry. Every module ends with a lab and builds on the one before.

Who Should Attend

- Senior and lead developers moving toward an architecture role
- Technical leads responsible for design decisions on their teams
- Engineers who want to understand how architects think and work

Prerequisites

- Several years of hands-on software development experience
- Comfort designing and building non-trivial applications
- Familiarity with object-oriented design

What You Will Learn

- Describe what a software architect does and how the role differs from a senior developer
- Think in breadth and in tradeoffs rather than reaching for a single right answer
- Identify, prioritize, and trade off the architecture characteristics that shape a system
- Evaluate common architecture styles and justify decisions through explicit tradeoff analysis
- Document and diagram an architecture, and communicate and defend it as a technical leader
- Build a personal roadmap for growing into the architect role

Course Outline

Day one: how architects think

- The Architect's Role
  - What architects actually do, beyond writing code

  - Architecture versus design; types of architect: application, solution, and enterprise
  - The "no path" problem, and a roadmap through it
  - Lab: map your current role against the architect's responsibilities and identify your gaps
- Architectural Thinking
  - Breadth over depth: the knowledge pyramid
  - Thinking in tradeoffs: why the honest answer is often "it depends"
  - Balancing hands-on coding with architecture work; first-principles reasoning and removing ambiguity
  - Lab: analyze a design request, surface its hidden tradeoffs, and remove its ambiguities
- Architecture Characteristics (the "-ilities")
  - Deriving characteristics from business and technical requirements
  - Explicit versus implicit characteristics
  - Prioritizing, limiting, and trading off characteristics against each other
  - Lab: extract and prioritize the architecture characteristics from a real-world brief

Day two: decisions, communication, and leadership

- Structure: Components and Styles
  - Identifying components and choosing the right granularity
  - Coupling, cohesion, and connascence
  - A survey of architecture styles and their tradeoffs
  - Lab: propose two candidate styles for the same system and compare their tradeoffs
- Making and Communicating Decisions
  - Tradeoff analysis in practice, and common decision traps
  - Architecture Decision Records
  - Diagramming approaches, including the C4 model, without drowning people in detail
  - Lab: capture a decision as an ADR with a C4 diagram that others can act on
- The Architect as Leader, and Your Path Forward
  - Communication, influence, and negotiation with teams, stakeholders, and the business
  - Presenting and defending an architecture
  - Building technical breadth and soft skills deliberately
  - Lab: defend an architecture decision under questioning, then draft your personal twelve-month roadmap

Extended Version

The three-day version keeps the same gradient and adds:

- A running case study carried across all three days
- Deeper hands-on work in tradeoff analysis and characteristic prioritization
- More diagramming practice with feedback
- A capstone of architecture katas: teams design an architecture from a real-world brief and defend their decisions