5-DAY COURSE

Software Engineering

Engineering successful software and software-intensive systems

Why Software Engineering?

Software now underpins almost every modern system, product, and enterprise capability, yet software projects continue to experience high rates of cost overruns, delays, technical debt, and outright failure. As projects increase in complexity and scale, many organizations struggle to consistently deliver software that satisfies stakeholder needs while remaining cost-effective, maintainable, and reliable over time.

This course provides a practical immersion in software engineering best practice across the entire software life cycle. Participants learn how proven engineering principles, effective development approaches, and systems thinking combine to improve software project outcomes. The course addresses software development from both a software technology and systems perspective, helping participants better understand the technical, organizational, and management contributors to software success and failure.

Participants will gain insight into how stronger software engineering practice can help:

- Improve delivery confidence and predictability
- Reduce rework, defects, and downstream project risk
- Strengthen software architecture and detailed design decisions
- Improve integration, verification, and validation outcomes
- Enhance collaboration across technical and management teams

Earn CE/CPD Credit

PMI Talent Triangle® Suggested PDUs

- Ways of Working - 35
- Business Acumen - 3
- Power Skills - 2

INCOSE Certified Systems Engineering Professional (CSEP)

- 40 Continuing Education PDUs

What You Will Learn

Participants develop a practical understanding of the principles, methods, and management practices needed for successful software and software-intensive system development. The course explores the software life cycle in depth, including requirements, architecture, design, construction, integration, verification, validation, deployment, maintenance, and support.

You will learn how to apply sound engineering methods to improve software quality, delivery performance, and stakeholder outcomes.

Areas of learning include:

- Agile, incremental, and other iterative development approaches, and when to use them
- Software requirements capture, validation and authoring for quality
- Software architecture and detailed design practices
- Software design and code verification and validation
- Software project planning, estimation, and risk management
- Configuration/change management
- Effective software engineering team practices
- The role of AI in software engineering.

Who Should Attend and Why

This course is designed for professionals involved in specifying, acquiring, developing, evaluating, supporting, or managing software and software-intensive systems. It is particularly valuable for organizations seeking to improve software project performance, strengthen engineering practices, and reduce avoidable delivery risk.

It is especially relevant for professionals seeking a broader and more disciplined approach to software development.

The course will be valuable to:

- Software engineers and software team leaders
- Programmers and developers
- Systems engineers and business analysts, especially those who create software requirements
- Project managers of software-intensive projects
- Verification and validation personnel
- Configuration and risk managers
- Software maintainers and procurement managers
- Reliability, safety, and quality professionals.

Training Methods and Materials

The course is delivered through a balanced mixture of formal presentation, guided discussion, practical exercises, and collaborative group activities designed to reinforce learning through application. Throughout the course, there is a strong emphasis on interaction, variety, and integrating new knowledge with participants' existing software development experience.

The emphasis is on practical learning that strengthens both technical understanding and engineering judgment. Participants receive comprehensive course materials together with numerous supplementary descriptions, forms, charts, and reference resources that can be applied immediately within professional software engineering environments.

PPI Training Review

"It was great learning about the total scope of software engineering. Interesting discussions on aspects and experiences of real projects of the presenter and participants."

— Course participant, Europe

Why PPI?

Trusted Worldwide
PPI delivers outstanding training and consulting to many hundreds of enterprises worldwide, from Fortune 100 companies (presently 19% of them) to small startups. PPI is a truly international company, with personnel based in eight countries, and clients across six continents benefiting from our work.

PPI Presenters
PPI's presenters are internationally recognized systems engineering practitioners and consultants who bring decades of real-world experience, ensuring every concept taught is value-adding, practical, relevant and immediately implementable.

NEC SIEMENS BAE SYSTEMS AIRBUS THALES

MITSUBISHI TNO babcock BHP

Software Engineering 5-Day Course – Course Outline

1. Introduction and Overview (Day 1)

- Presenter background
- Overview of the learning methodology for the course
- General introduction into software engineering
- History of software development, recent trends, the current state of the practice and looking beyond the current state
- An engineering approach to software development including concurrent engineering, systems methodology and systems thinking
- Life cycle characteristics and typical stages, introduction into sequential versus incremental and iterative development models approaches
- Lean software development and value-driven design
- Introduction into tailoring and process improvement principles
- Case study on how applying Systems Engineering could have saved a Software Project

2. Technical Processes (Day 1 – 3)

- Requirement Analysis
  - Requirement fundamentals
  - System requirements capture, system boundaries, hierarchy, and subsystems
  - Requirement specifications
  - Agile documentation
  - Quality attributes and other fundamentals
  - Requirement documentation, natural language, models and diagrams, unified modeling language (UML), and storyboards
  - Requirements parsing
  - Common pitfalls in performing requirements analysis (RA)
- Software Design
  - Design fundamentals
  - Architectural styles and patterns
  - Architectural tactics, views and documentation
  - Architectural frameworks
  - Design methods
  - Design notation
  - Architectural description quality factors
- Software Construction
  - Structure and complexity
  - Standards for coding
  - Coding Process
  - Assessing quality
- System and Software Integration
  - Integration strategies
  - Integration issues
  - Pitfalls in system and software integration
- Validation and Verification
  - Fundamentals of verification and validation
  - Formal, informal, technical, design, code, requirements and other types of reviews
  - Software and systems testing
  - Other verification and validation (V&V) methods
- Transition (transfer of responsibility between different entities)
- Operations, Maintenance, and Support
- Disposal or Retirement

3. Project Processes (Day 3 – 4)

- Project management introduction, themes and frameworks
- Project vs. product focus
- Waterfall vs agile project processes
- Project definition, planning and estimation
- Assessment and control
- Project crashing (Shortening the project timeline)
- Resources planning
- Project communication
- Risk management
- Cost management
- Agile project management
- Project plans and software development plans
- Decision management
- Configuration management
- Information management
- Measurement
- Quality assessment and control
- Release and deployment

4. Agreement Processes (Day 4)

- Overview of agreement processes and contract models
- Acquisition
- Supply

5. Enterprise-Level Processes (Day 4 – 5)

- Life cycle model management in particular sequential, iterative and agile models, frameworks and methodologies - their advantages, disadvantages and application
- DevOps, continuous integration, continuous delivery, lean, Kanban, TDD, FDD, ATDD, BDD
- Scaling agile development for large projects
- Program and portfolio management
- Human elements, organizational structure, types of teams, integrated product teams, leadership and other factors
- Organizational quality management

6. Tailoring
(Day 5)

- Adaptation of processes both at organizational as well as project level

7. Specialty Fields and Parking Lot
(Day 5)

- Engineering of trusted/high integrity systems
- Software life cycle cost analysis
- Interoperability
- Usability analysis and human system integration

8. In Closing
(Day 5)

www.ppi-int.com

systems/product engineering training & consulting for project success ...