Software Diagnostics and Troubleshooting Training Course

26 – 30 July 2027
Kuala Lumpur

AGILE LEADERS
Training Center

AGILE LEADERS
Training Center

Software Diagnostics and Troubleshooting Training Course

Ref: 1709_228438 **Date:** 26 – 30 July 2027 **Location:** Kuala Lumpur **Fees:** 5200 Euro

Overview

Software Diagnostics and Troubleshooting Training Course is a one-week course for support engineers, application administrators, developers, quality engineers, and operations teams, who leave with a Software Diagnostic Evidence Pack. The course connects reproducible symptoms, logs, metrics, traces, debugging, dependency checks, configuration comparisons, performance profiling, fault isolation, causal hypotheses, and remediation validation across test and production environments. Agile Leaders Training Center presents the course on software diagnostics and troubleshooting.

Who Should Attend

- Teams responsible for application support and incident triage
- Functions responsible for software development and debugging
- Teams responsible for testing, defect evidence, and release quality
- Functions responsible for application administration and configuration
- Teams responsible for service monitoring and technical operations

The course assumes participants investigate software behavior at work and leaves out introductory programming, platform-specific certification preparation, malware analysis, and hardware repair.

Departments and Industries

The course supports software fault diagnosis across digital services and technology-enabled operations.

- Software engineering and application support departments
- Quality engineering and release management teams
- Site reliability and technical operations functions
- Financial services and telecommunications technology teams
- Healthcare and education application operations
- Manufacturing and logistics software teams

Learning Objectives

By the end of this course, participants will be able to:

- Apply a reproducible symptom-capture method
- Analyze logs, metrics, traces, and exceptions
- Compare dependencies and configuration states
- Use profiling and dump evidence for isolation
- Evaluate causal hypotheses against observations
- Validate remediation and incident closure evidence

Course Agenda

Day 1: Diagnostic Foundation

- Symptom and Impact Capture Template
- Reproduction Condition Decision Tree
- Black-Box and White-Box Evidence Map
- Diagnostic Timeline and Change Register
- Evidence Preservation Checklist

Day 2: Telemetry and Context

- Structured Log and Exception Correlation Matrix
- Metric Baseline and Deviation Worksheet
- Distributed Trace and Request Path Map
- Dependency Health and Timeout Register
- Configuration State Comparison Method

Day 3: Debugging and Profiling

- Debugger Hypothesis and Breakpoint Plan
- CPU Sampling and Hot-Path Profile
- Memory Allocation and Leak Evidence Sheet
- Process Dump Triage Checklist
- Concurrency and Resource Contention Map

Day 4: Isolation and Validation

- Fault-Domain Isolation Matrix
- Controlled Change and Experiment Log
- Root-Cause Hypothesis Evidence Table
- Remediation Risk and Rollback Checklist
- Fix Verification and Regression Record

Day 5: Software Diagnostic Practice

- Symptom Reproduction and Timeline Exercise
- Log Metric and Trace Correlation Exercise
- Performance Profile and Dump Triage Exercise
- Fault Isolation and Fix Validation Exercise
- Software Diagnostic Evidence Pack Capstone Exercise

Practical Exercises

The course uses suggested activities to practice evidence-based software diagnosis.

- Suggested activity: reproduce a failure and build a timeline from user symptoms, changes, and environmental conditions.
- Suggested activity: correlate logs, metrics, traces, exceptions, dependencies, and configuration differences.
- Suggested activity: review CPU, memory, dump, and contention evidence to isolate a fault domain.
- Suggested activity: test causal hypotheses and record remediation, rollback, verification, and regression evidence.

FAQs

Who suits software diagnostics and troubleshooting training and what does it assume?

Software diagnostics and troubleshooting training suits application support, development, quality, administration, and technical operations teams. It assumes practical exposure to software failures and access to operational or test evidence.

How does software diagnostics and troubleshooting differ from software testing training?

Software diagnostics and troubleshooting investigates why an observed failure occurs using telemetry, debugging, profiling, isolation, and experiments. Software testing training focuses on designing and executing checks that reveal defects and verify expected behavior.

Which evidence supports software diagnostics and troubleshooting?

Software diagnostics and troubleshooting uses reproducible symptoms, timelines, logs, metrics, traces, exceptions, dependencies, configuration states, profiles, process dumps, experiments, and fix-verification records.

AGILE LEADERS
Training Center

How do logs, metrics, and traces work together in software diagnosis?

Logs explain recorded events, metrics show changes in system behavior, and traces connect activity across request paths. Correlating their timestamps and identifiers helps teams move from a symptom toward a testable causal hypothesis.

How is a root-cause hypothesis validated in software troubleshooting?

A root-cause hypothesis is validated by predicting observable evidence, running a controlled test, ruling out competing explanations, applying a bounded remediation, and confirming the symptom disappears without unacceptable regression.

Conclusion

Participants take a Software Diagnostic Evidence Pack back to work. It connects symptom records, timelines, telemetry, dependency maps, configuration comparisons, profiles, dumps, hypotheses, experiments, remediation decisions, and verification evidence. It changes ad hoc troubleshooting into reproducible investigations, traceable decisions, controlled fixes, and reviewable incident closure.