

LINUX



LINUX



Corporate Trainer's Profile

Corporate Trainers are having the experience of 4 to 12 years in development , working with TOP CMM level 5 companies (Project Leader /Project Manager) qualified from NIT/IIT/IIM and work exp in USA and UK.



CMM (Capability Maturity Model) level Project Standard:-

The Capability Maturity Model (CMM) is a method for evaluating the maturity of organizations on a scale of 1 to 5.
Get the Opportunities to work on Client Projects Of US/UK, which follow the all standard of CMM level 5 Company.

Linux Kernel & Device Drivers Development

Course Description

The Linux Device Drivers Development course provides engineers with a fast, cost-effective way to acquire the knowledge necessary to build, load, and debug Linux device drivers in a cross-development environment.

After this course, students will be able to:

- Understand the Linux boot process and role of the boot loader and BSP
- Prepare to develop Linux I/O drivers, serial drivers, and network drivers
- Debug Linux device drivers

Who Should Attend

- New and experienced device driver writers
- Application Programmers interested in seeing what goes on at the hardware level
- Senior engineers who want to decide on a final production image of their product
- Experienced developers interested in the interfaces between drivers and Linux

Prerequisite Skills

- Functional knowledge of Linux
- One year programming, including experience with structures, pointers, pointers to structures, typedefs, macros, and bitwise operators.
- Experience using binary and mutual exclusion, semaphores, message queues, pipes, and managing multiple tasks
- Experience debugging target code
- Good understanding of OS fundamentals like Processes, tasks, scheduling, Process Synchronization etc

Linux Kernel and Driver Development:

- Review of Linux boot process and the role of the boot loader
- Building and accessing modules
- Linux Device driver debugging techniques
- Using Timer services and memory management techniques for developing device driver
- Overview of interrupt handling when implementing a Linux device driver
- Steps necessary to develop character and network device drivers
- Techniques for Debugging Device Drivers
- Overview of hot-plug support, power management and use of sysfs

Course Content:

Linux OS Architecture

- Overall Linux Architecture
- User/Kernel mode division

- Kernel Components
- Linux Kernel Scheduling
- Linux System Process Priorities and System Calls

Linux Driver Architecture

- Need and Use of Drivers
- Different Types of Linux Drivers

Char Driver

- Basic Concept
- Major/Minor Numbers
- Device registration
- Driver Compilations, Make files
- Loading of Kernel Drivers
- Hello World Kernel Driver
- Parameterized Hello World Kernel Driver

File Operations on Char Driver

- File Operation Concept
- Open/Close
- Read/Write
- User/Kernel Mode Memory Transfers
- IOCTL Operation

Driver Debugging Techniques

- Prints
- Procs File Systems
- Other Debugging Techniques

Kernel Driver Synchronization

- Basic of Synchronization
- Semaphores
- Mutexes
- Spin Locks
- Kernel Memory Allocations
- Basic of Synchronization

Kernel and Modules

- Kernel Configuration Tools
- Building and Configuring the Kernel
- Loading the Kernel
- Adding a Driver
- Building and Installing Modules
- Loading and Unloading Modules
- Passing Parameters to Modules
- Kernel Versioning

User Space to Kernel Space Interfaces

- Data Flow
- Using /procfs file system
- Using relayFS

Memory Management

- Linux Memory Model
- Kernel Address Space
- Caching

- Allocating Memory in User and Kernel Space
- Mapping a Device

IOCTL Usage

- Need and Use of IOCTLs
- Implementation of IOCTLs

Blocking IO Mechanism

- Blocking in Driver
- Blocking vs Non Blocking I/O
- Wait_Event based Implementation
- Sleep
- Race Conditions and Locks
- Using Barriers
- Semaphore Usage

Interrupt Handling

- Linux and Interrupts
- IRQ Action Table
- Register, Enabling and Disabling Interrupts
- Minimizing Interrupt Latency

Network Device Drivers

- Linux Network Driver Overview
- Network Device Structure
- Network Driver Functions
- Network Stack Functions
- Socket Buffers
- Transmit and Receive Logic Flow

Asynchronous IO Mechanism

- Asynchronous usages
- Poll and Select Based Mechanism

Leftovers

- Linux Interrupt Handling
- Kernel Timer Concepts
- DMA and Memory Accesses

Global Reach of Miracle Education Services

- 7,000 students per year
- 800 classes delivered per year
- 40 instructors worldwide
- Access to 125 Technical Domain Experts
- On-site courses are conducted at your location and include the use of preconfigured laptops and target boards, plus shipping and travel costs.