Developing File Systems for Windows®

This four (4) day seminar deals with the issues of developing both file systems and file system filter drivers in the Windows environment, with both a discussion of broader operating systems issues for developers as well as practical issues related to current development. Upon completion of this course, you will have the basic information and technical understanding on how file systems work in the Windows environment. This seminar is taught only by experienced systems developers with experience developing file systems for Windows and other operating systems platforms.

Details

Length: 4 days

Format: Lecture

Target Audience
Students with a need to understand or develop file systems or file system filter drivers on Windows.

Prerequisites
Students must have an understanding of Windows kernel mode drivers, either via direct experience developing kernel mode drivers (including file system drivers) and/or by taking our *Writing WDM Kernel Mode Drivers* seminar or *Windows Internals and Software Driver Development* seminar. While not required, those students who have prior experience developing file systems and file system filter drivers will get a great deal out of this course because it will aid them in putting everything into context. Indeed, students who have previously taken this course prior to embarking upon their project have found it useful a second time after engaging in substantial development because it helps them "fill in the blanks".

Seminar Outline

- **Review of the OS**
  The course starts with a standard overview of the operating system and its components from a file system centric perspective.

- **Advanced I/O Manager Concepts**
  This section reviews the I/O Manager and then focusing on advanced I/O Manager concepts essential to an understanding of Windows file systems. Topics include: a review of I/O Manager operations, how the I/O Manager handles IRP completion processing, thread context issues, using worker threads, and the construction of IRPs.

- **Naming Concepts**
  This section delves into the arena of naming issues and incorporates a discussion of the Windows Object Manager, how names are actually parsed by the I/O Manager and Object Manager and thus how names are presented to the file systems. It also covers issues relating to mount points, drive letter maps, network providers, reparse points and symbolic links.

- **The Virtual Memory System**
  Because Windows file systems are tightly integrated with the VM system it is imperative for any course of study to cover the VM System. This section describes in detail the two key components of the VM system, the Memory Manager and the Cache Manager, and how they interact with file systems.

- **File System Introduction**
  Having reviewed the other components of the OS essential to an understanding of file systems, the course turns its attention to a high level overview of how file systems on Windows are constructed. This section includes a description of the types of file systems as well as file servers and filter drivers, both of which are closely related to Windows file systems.

- **File System Fundamentals**
  The balance of this course is constructed to cover, in detail, how file systems are constructed and operate within Windows. The focus here is very pragmatic in an attempt to facilitate the development of file systems by those attending the class. This section covers the basic data structures that are essential to file systems. It describes various objects (File, Section, Device, and Driver) which are of interest to Windows file systems as well as data structures (IRP, Common Header, VPB) used throughout Windows file systems. It ties these concepts together by reviewing actual code within the IFS Kit that shows these key data structures and functions.

- **File System Core Functions**
  This section of the course covers the core file system functions, of interest to all file system developers. This includes creation of new file object, and their subsequent cleanup and close operations as well as read, write, directory operations, delete, rename, and various other ancillary operations necessary for the proper implementation of a file system. This section includes review of concepts critical to understanding IFS relevant sections of the Windows Driver Kit.

- **Security**
  File systems may optionally support the Windows security model, or implement their own security model. This section describes security in the context of the Windows security model using discretionary access control lists, system access control lists, owner, and group information. It also describes the process of validating access using the security reference monitor, and interfacing with auditing of I/O operations.

- **File System Auxiliary Functions**
  This section discusses various "auxiliary" file system functions that are implemented by some file systems. This includes support for extended attributes, quotas, byte range locks, name tunneling, sparse files, transactions, etc. This section includes review of the relevant sections of the IFS Kit.

- **Fast I/O Operations**
  Microsoft has increasingly implemented a synchronous procedure call interface into certain portions of the OS, of which File Systems are a notable example. The Fast I/O operations were initially intended to improve the overall performance of read and write I/O operations but have since been used to target specific performance improvements for Lan Manager Server (which is a kernel-based file server on Windows). This section describes these operations and their basic implementation.

- **Filter Drivers**
  With the introduction of the filter manager several years ago, Microsoft created a new model for building file system filter drivers ("mini-filters") and we discuss both mini-filters as well as legacy file system filter drivers in this section. This section discusses important issues involved in the development of both legacy and mini file system filter drivers. Many developers prefer to "skip the file systems" material and learn just about filtering, but as we discuss in this section it is critical that file system filter drivers understand not only their own specific issues, but also the behavior of the components they are actually filtering.