Implementing a Real File System

Peichao ZHANG

Li HAN

History

- 2007 and before: there was no such phase
  - same as Berkeley: thread, userprog, vm, network
- 2008: Kang Zhang (ACM05) imported the file system from C version
- 2009: network phase was discarded because of TeNeT
- 2010: Peichao Zhang (ACM07) got the framework completely rewritten

Overview

- *StubFileSystem*
  - fake, using Java File IO methods
- *RealFileSystem*
  - the goal of this phase
  - with minimum use of the file system of the host operating system
  - support basic operations and advanced operations

Originality Wanted

- Feel free to develop your own framework
- Underlying design and implementation is open
- Demonstrate your innovation in the report
  - not required, but recommended

Foundation

- *nachos.machine.Disk* – a hard disk without synchronization
- *nachos.machine.SynchDisk* - provide synchronous accesses to the disk
- *nachos.filesys* – a basic framework to help you to start with

Framework

- INode
  - contain metadata of a file
  - organized as a list
- File
  - extended from OpenFile
  - basic IO operation

Framework

- Folder
  - special file to support hierarchical structure
  - fixed location for the INode of the root folder
- FreeList
  - special file for the disk space allocation
  - fixed location

Tips

- Easier than previous phases, but more tedious
- May need to deal with several corner cases in various operations
  - Relative path
- *FilesysKernel.format = true* formats the disk and copy all files in the test directory into your '/'

Tips

- Read code carefully
- Test with virtual memory (Phase 3)
  - Heavy page faults help debugging