Chapter 9: Filesystems

Introduction

Till this point, we have seen how a kernel of an operating system works as a resource manager. Given that both the processor and the main memory are resources in the system, 539kernel manages these resources [1] and provides them to the different processes in the system. Another role of a kernel is to provide a way to communicate with external devices, such as the keyboard and hard disk. *Device drivers* are the way of realizing this role of the kernel [2]. The details of the external devices and how to communicate with them are low-level and may be changed at any time. The goal of a device driver is to communicate with a given device by using the device's own language and the other goal of a device driver is to provide an interface for any other component of the system that wish to use the given device, most probably the low-level details of this given device will be hidden behind the interface that the device driver provides, that means the user of the device drivers doesn't need to know anything about how the device really work.

The matter of hiding the low-level details with something higher-level is too important and can be found in, basically, everywhere in computing and the kernels are not an exception of that. Of course, there is virtually no limit of providing higher-level concepts based a previous lower-level concept, also upon something that we consider as a high-level concept we can build something even higher-level. Beside the previous example of device drivers, one of obvious example where the kernels fulfill the role of hiding the low-level details and providing something higher-level, in other words, providing an *abstraction*, is a filesystem which provides the well-known abstraction, a file.

In this chapter we are going to cover these two topics, device drivers and filesystem by using 539kernel. As you may recall, it turned out that accessing to the hard disk is an important aspect for virtual memory, so, to be able to implement virtual memory, the kernel itself needs to access the hard disk which makes it an important component in the kernel, so, we are going to implement a device driver that communicate with the hard disk in this chapter. After getting the ability of reading from the hard disk or writing to it, we can explore the idea of providing abstractions by the kernel through writing a filesystem that uses the hard disk device driver and provides a higher-level view of the hard disk, that we all familiar with, instead of the physical view of the hard disk which has been described previously in chapter . The final result of this chapter is version `NE` of 539kernel which has as we mentioned a hard disk device driver and a filesystem.

[1] Incompletely of course, to keep 539kernel as simple as possible, only the basic parts of resources management were presented.

[2] At least a monolithic kernel.