Operating–System Structures

Exercises

2.9 The services and functions provided by an operating system can be divided into two main categories. Briefly describe the two categories, and discuss how they differ.

2.10 Describe three general methods for passing parameters to the operating system.

2.11 Describe how you could obtain a statistical profile of the amount of time a program spends executing different sections of its code. Discuss the importance of obtaining such a statistical profile.

2.12 What are the advantages and disadvantages of using the same system-call interface for manipulating both files and devices?

2.13 Would it be possible for the user to develop a new command interpreter using the system-call interface provided by the operating system?

2.14 Describe why Android uses ahead-of-time (AOT) rather than just-in-time (JIT) compilation.

2.15 What are the two models of interprocess communication? What are the strengths and weaknesses of the two approaches?

2.16 Contrast and compare an application programming interface (API) and an application binary interface (ABI).

2.17 Why is the separation of mechanism and policy desirable?

2.18 It is sometimes difficult to achieve a layered approach if two components of the operating system are dependent on each other. Identify a scenario in which it is unclear how to layer two system components that require tight coupling of their functionalities.

2.19 What is the main advantage of the microkernel approach to system design? How do user programs and system services interact in a microkernel architecture? What are the disadvantages of using the microkernel approach?

2.20   What are the advantages of using loadable kernel modules?

2.21   How are iOS and Android similar? How are they different?

2.22   Explain why Java programs running on Android systems do not use the standard Java API and virtual machine.

2.23   The experimental Synthesis operating system has an assembler incorporated in the kernel. To optimize system-call performance, the kernel assembles routines within kernel space to minimize the path that the system call must take through the kernel. This approach is the antithesis of the layered approach, in which the path through the kernel is extended to make building the operating system easier. Discuss the pros and cons of the Synthesis approach to kernel design and system-performance optimization.