

Introduction

I233 Operating System

Satoshi UDA <zin@jaist.ac.jp>

Logistics

- Language:
 - Class&Handouts: Japanese
 - Resume&Exam: Japanese (+English)
- Syllabus: described later
- Textbook:
 - Modern Operating Systems, 5th Edition, Global Edition / Andrew S. Tanenbaum, Herbert Bos. Pearson, 2022. (ISBN: 978-0137618828)
- Questions & Discuss:
 - “Q & A forum” on LMS
- Prerequisite knowledge:
 - Programming Language: C
- Some knowledge of machine language
- Computer architecture: Instruction, Program Counter (PC), Arithmetic Logic Unit (ALU), Memory cycle, Devices, Interruption
- Schedule
 - 20mins rule
 - Tutorial Hour
- Evaluation Criteria
 - Contribution to class: 10%
(click #presence on LMS each day)
 - Assignment (Survey report): 15%
 - Assignment (Programming Project): 30%
 - Exam: 45%

Syllabus

1. Introduction

File systems

12. Files & Directories

13. File-system Implementation

14. Example File-systems

Processes

2. Processes & Threads

3. InterProcess Communication

4. Scheduling

Memory Management

7. Virtual memory, Paging and Page Tables

8. Page replacement algorithm

9. Design Issues & Segmentation

Deadlocks

5. Resources and Deadlocks

6. Deadlock detection, recovery, avoidance and prevention

Input/Output

10. I/O hardware

11. I/O software

Overview of Computer Hardware

- Processor
- Memory
- I/O devices
- Bus

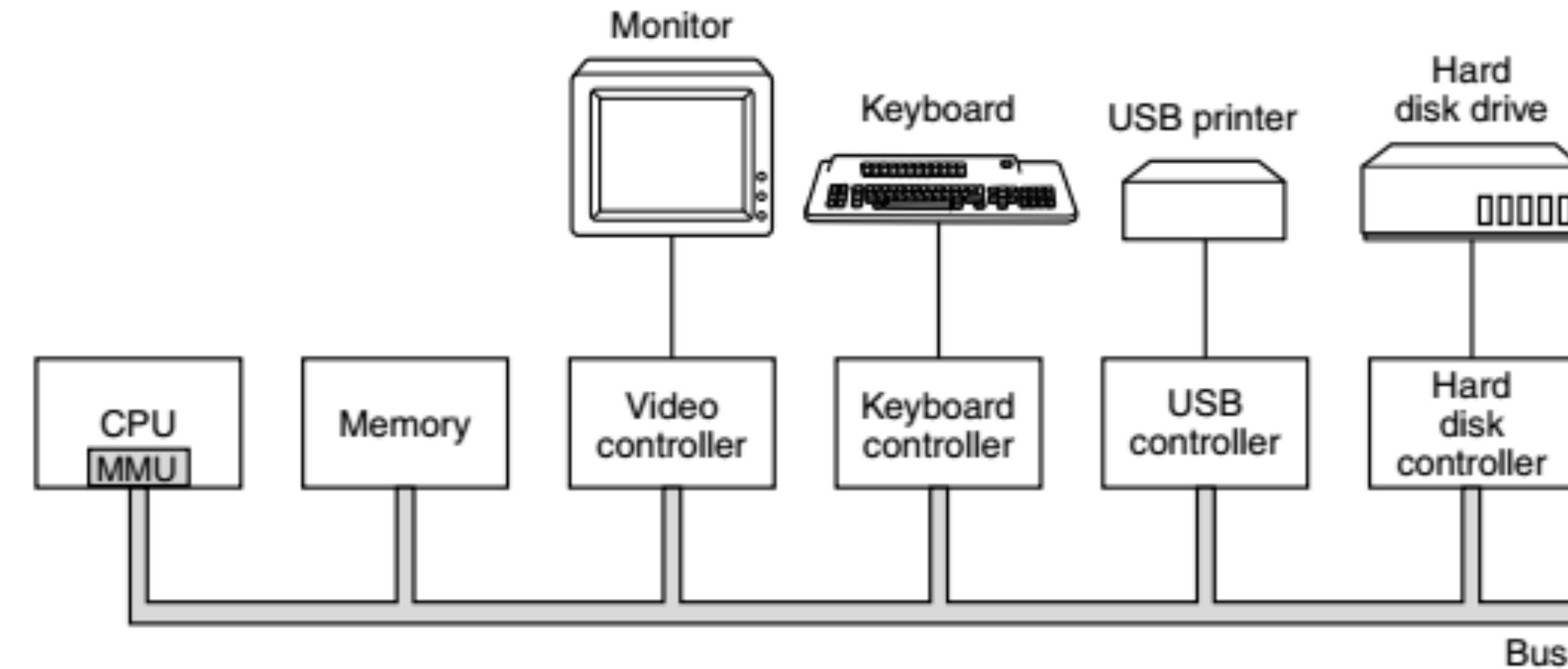


Figure 1-6. Some of the components of a simple personal computer.

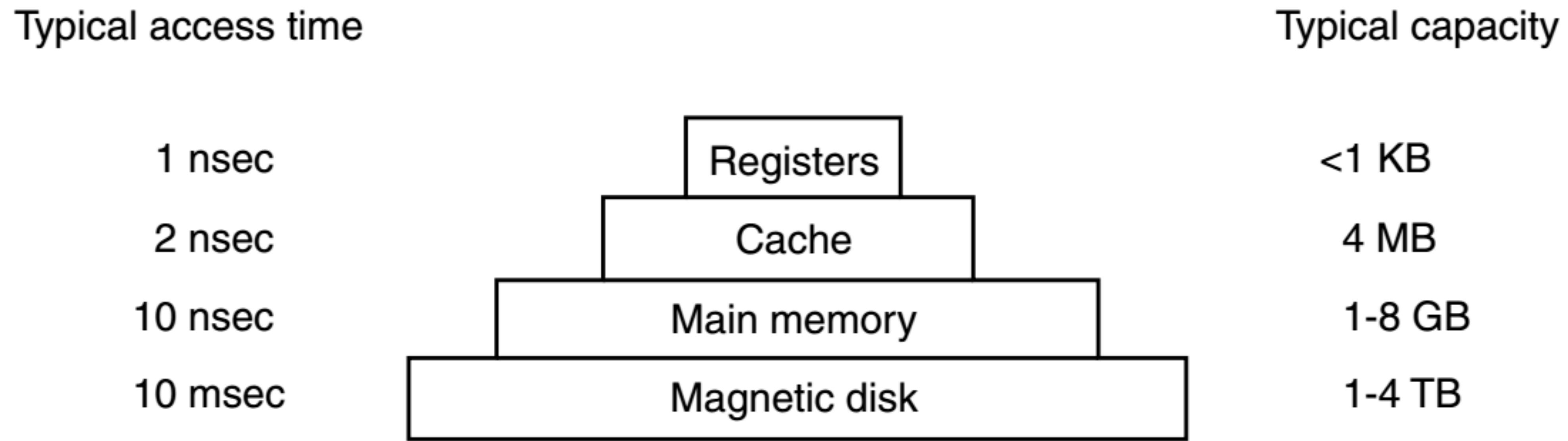
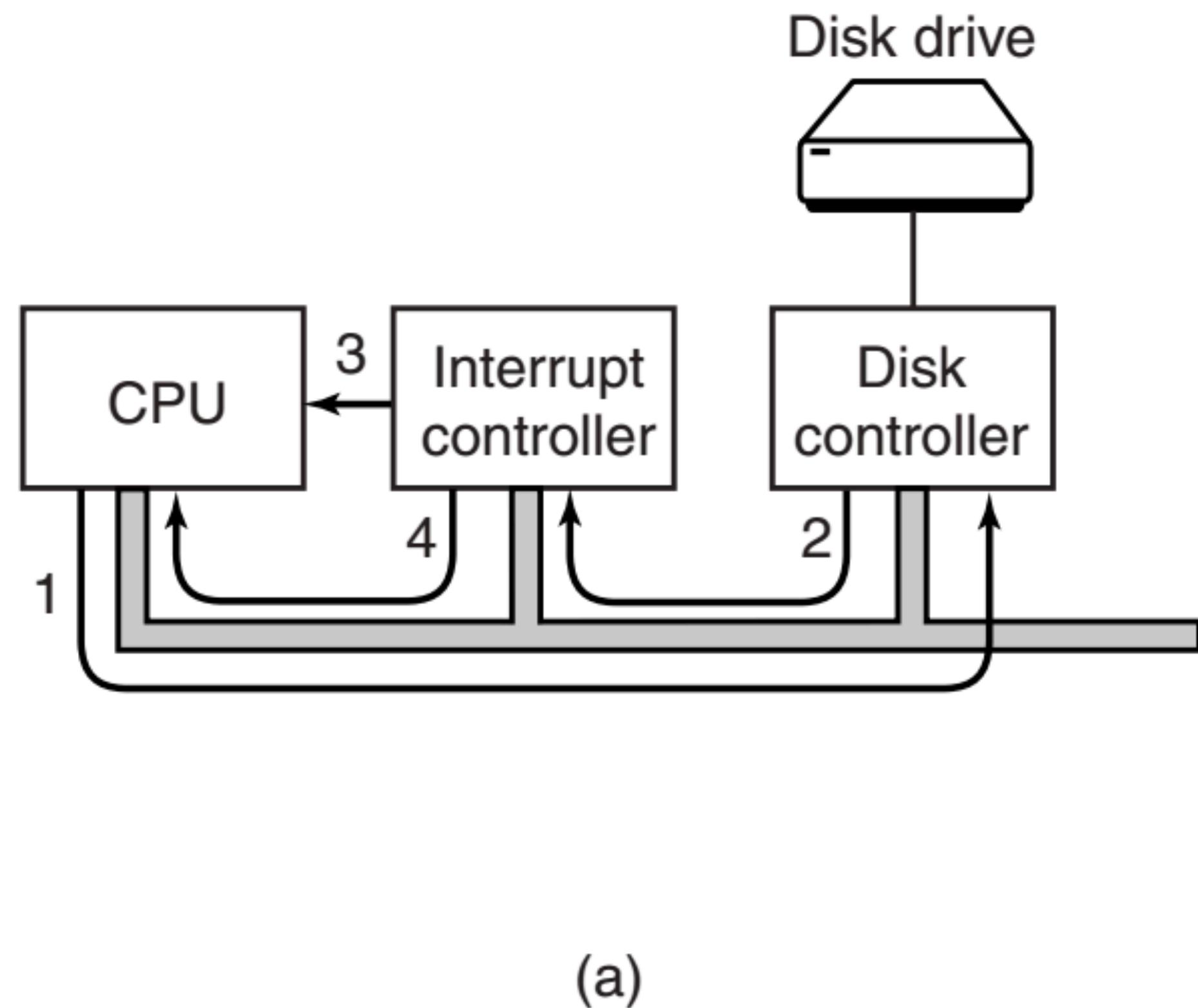
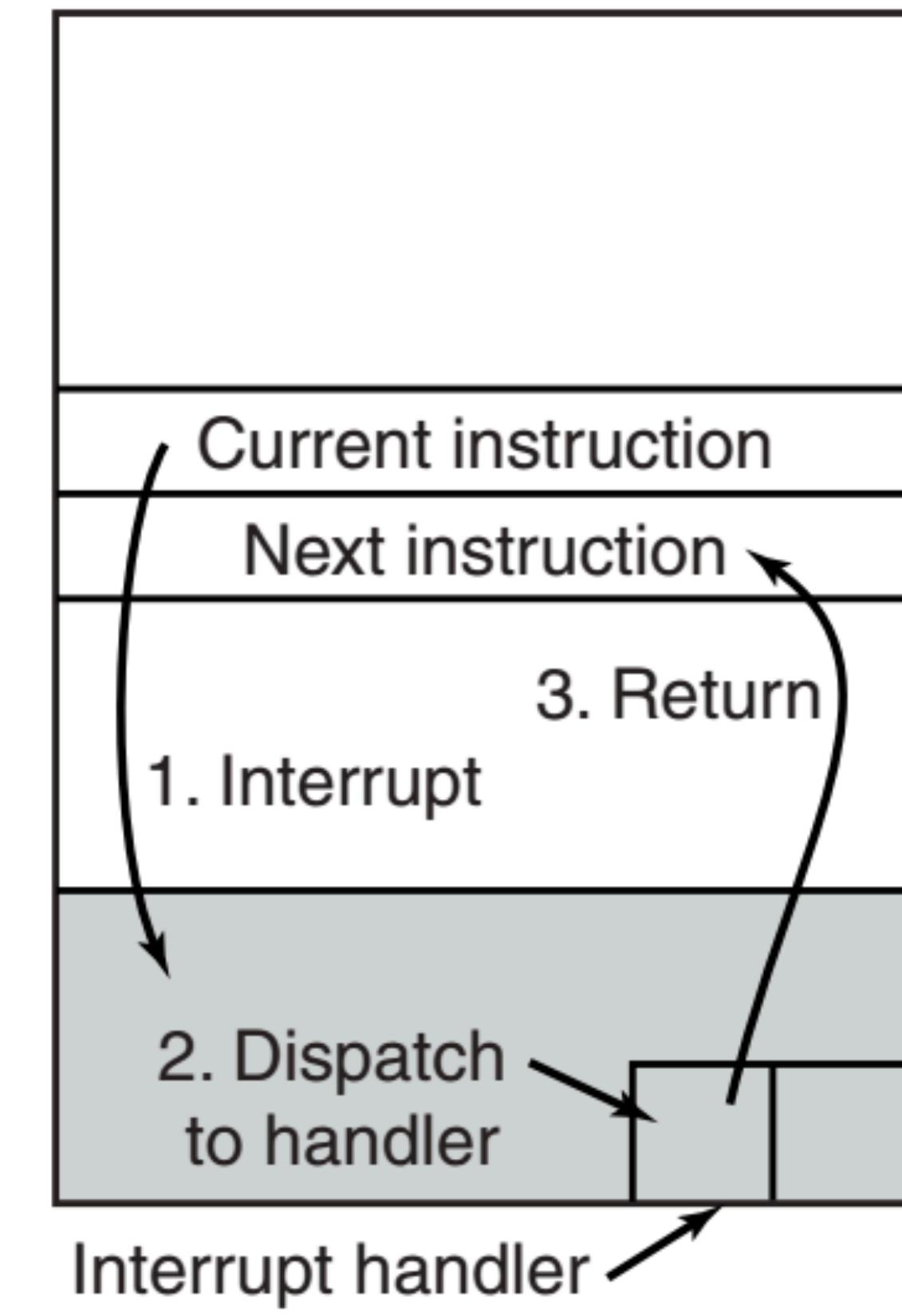


Figure 1-9. A typical memory hierarchy. The numbers are very rough approximations.



(a)



(b)

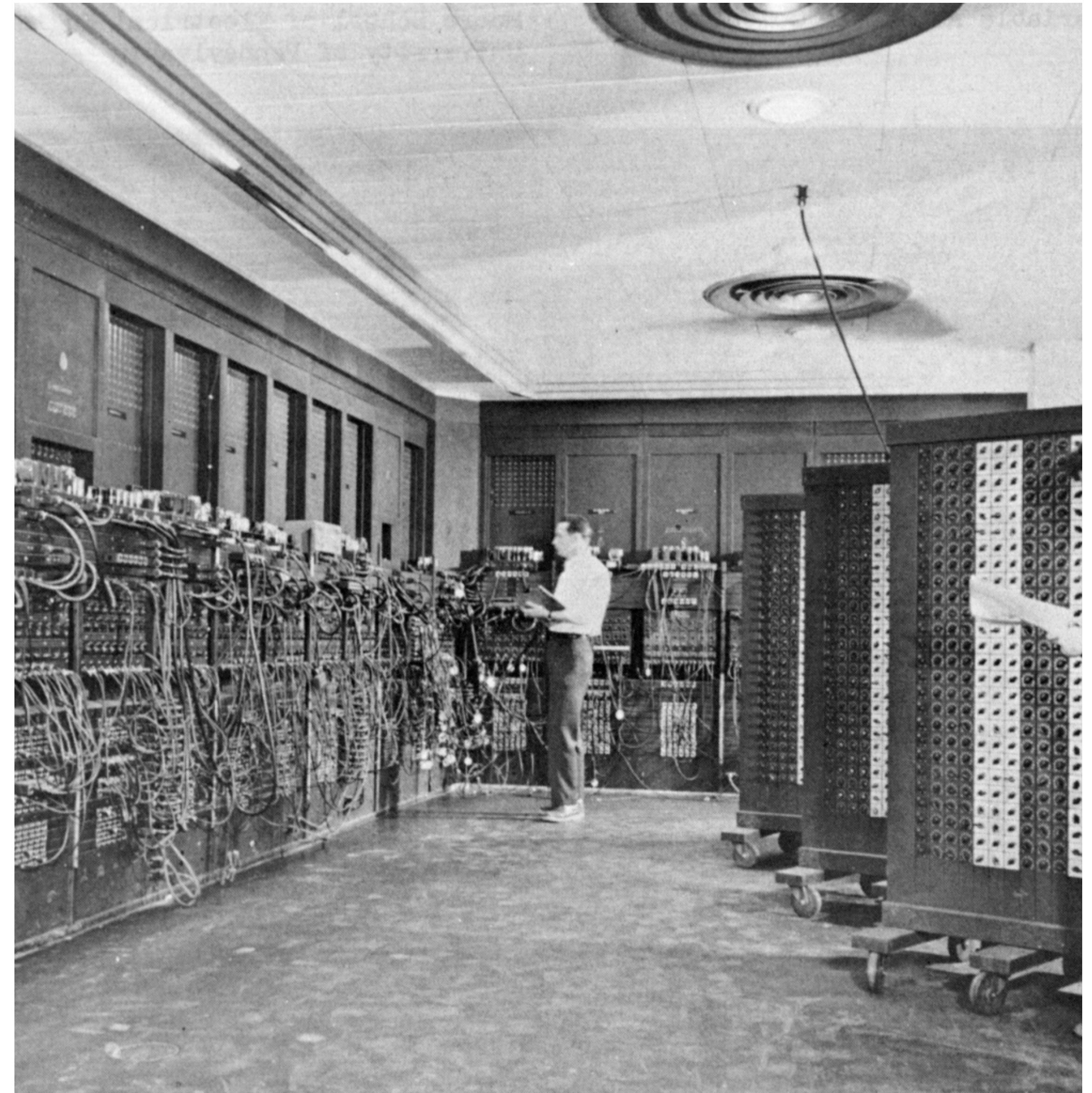
Figure 1-11. (a) The steps in starting an I/O device and getting an interrupt. (b) Interrupt processing involves taking the interrupt, running the interrupt handler, and returning to the user program.

History of Hardwares and OSs

- 1st Gen. (1945-55): Vacuum Tubes
- 2nd Gen. (1955-65): Transistors and Batch Systems
- 3rd Gen. (1965-80): ICs and Multiprogramming
- 4th Gen. (1980-): Personal Computers
- 5th Gen. (2005-): Cloud distribution and WebCentric

ENIAC - Electric Numerical Integrator and Computer

- 18,000 vacuum tubes; 7,200 crystal diodes; 1,500 relays; 70,000 resistors; 10,000 capacitors.
- Size: 2.4m(H)×0.9m(D)×30m(W), Weight: 30t
- 5,000 add|sub ops/s, 385 multiplication/s, 40 division/s.
- Electronically emulate the operation of the digit wheels of a mechanical adding machine.
- Not the stored-program computers that exist today,



IBM7090



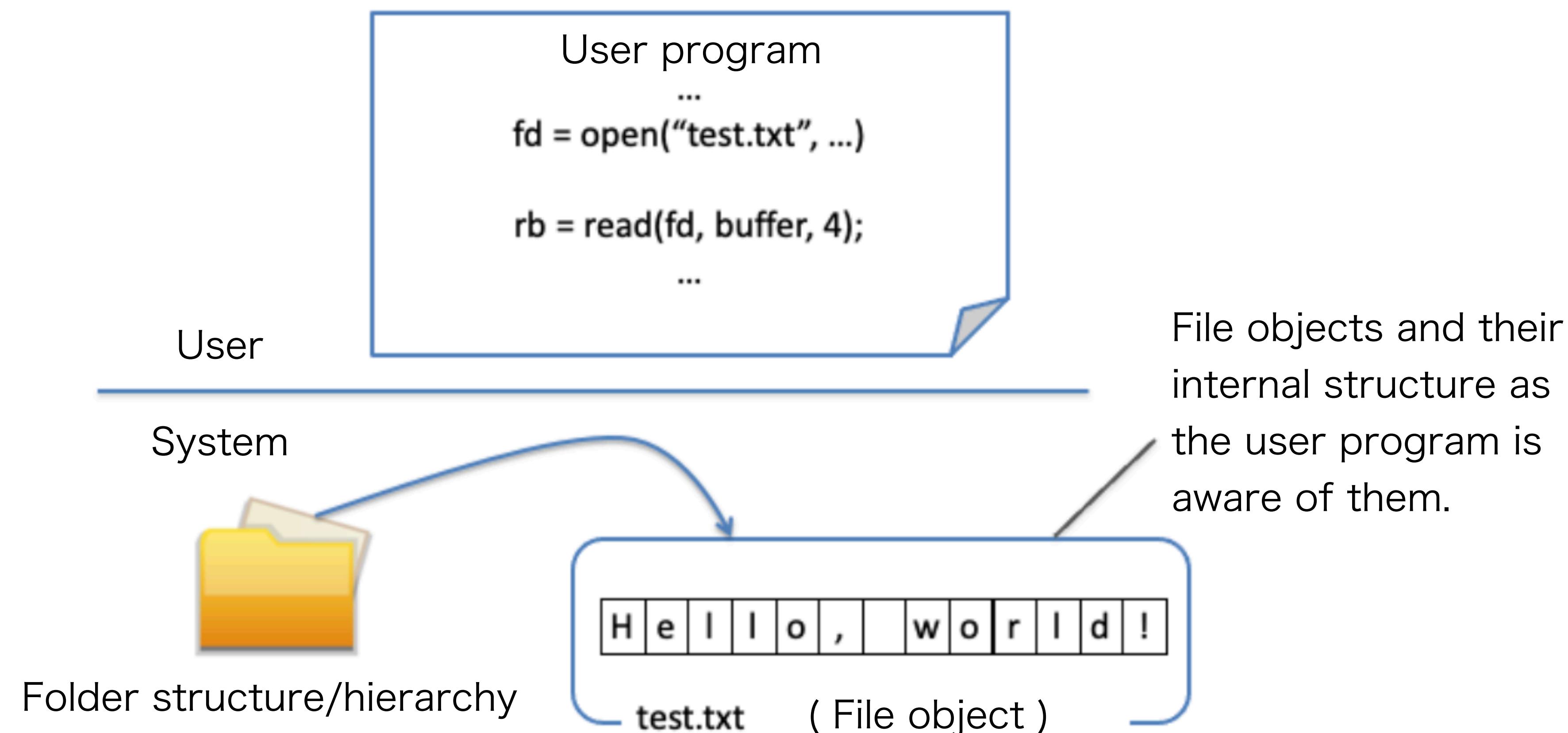
Source: <https://thisdayintechhistory.com/11/30/ibm-7090-delivered/>

What is an operating system?

- As an **Extended Machine**
 - It hides bare hardware and provides an “Extended Machine” with more advanced features. = Useful Computer
 - This is a technique called “abstraction”.
 - Bare hardware, OS, system libraries, system utilities, system components such as GUI libraries, etc. are the “layers of abstraction”.
- As a **Resource Manager**
 - Allocate all the resources that make up the computer to the many programs that are going to use those resources to do their work, under “a policy”.
 - Resources (Hardware level): Processor, Memory, Timer, I/O devices (Disks, Mouse, Keyboard, Network access and bandwidth)
 - Multiplexing (Sharing) resources
 - Exclusive control of resource usage

Abstraction — Example on file handling (1)

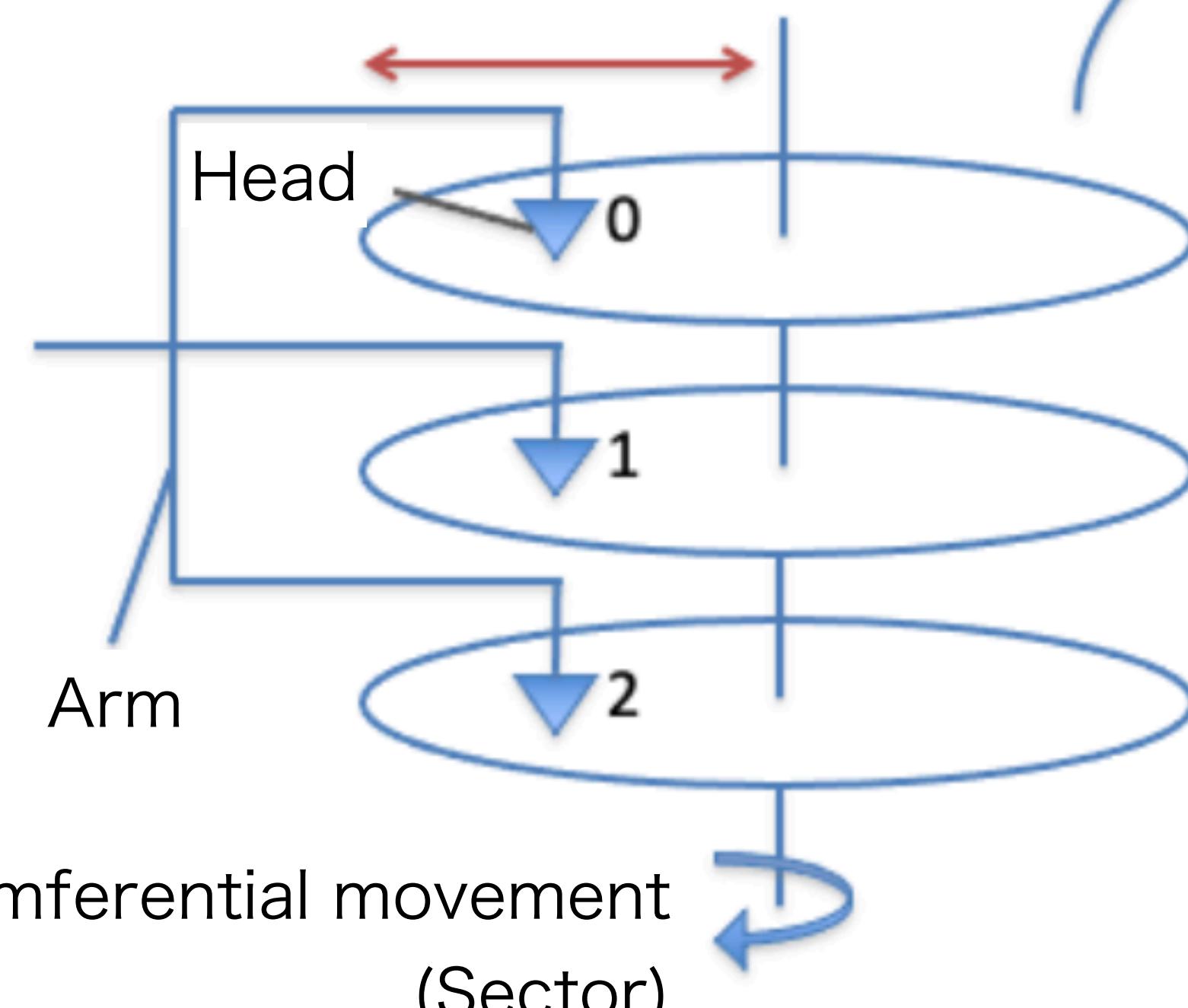
A program that read first 4 characters from a file named “test.txt” to memory.



Abstraction — Example on file handling (2)

Structure and function of HDD device

Movement in the diameter direction (Cylinder)



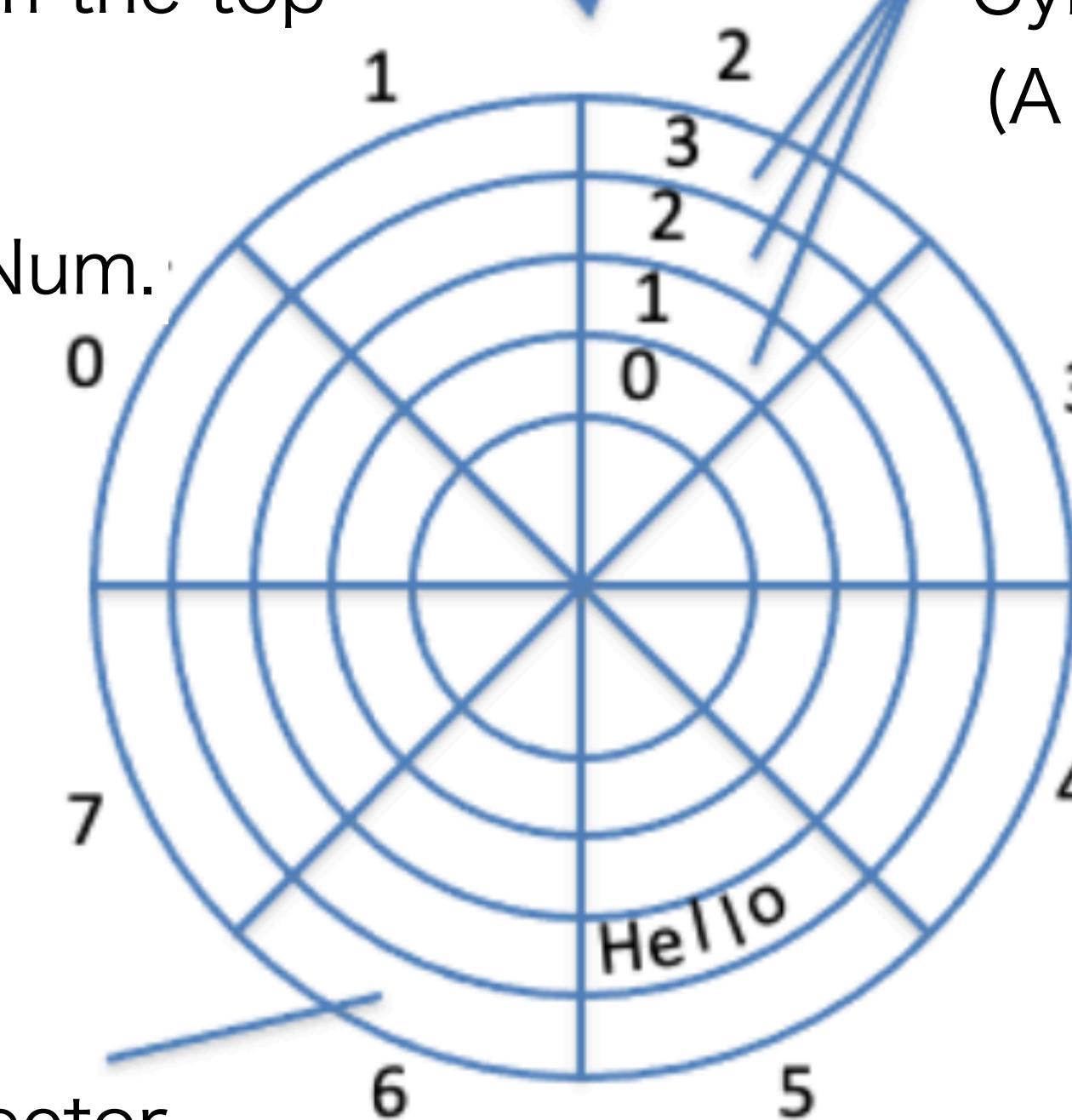
Circumferential movement
(Sector)

View
from the top

Sector Num.

A sector
(Minimal Unit)

Truck
or
Cylinder
(A ring)



Physical Location is Identify by (Head num., Cylinder num., Sector num.)

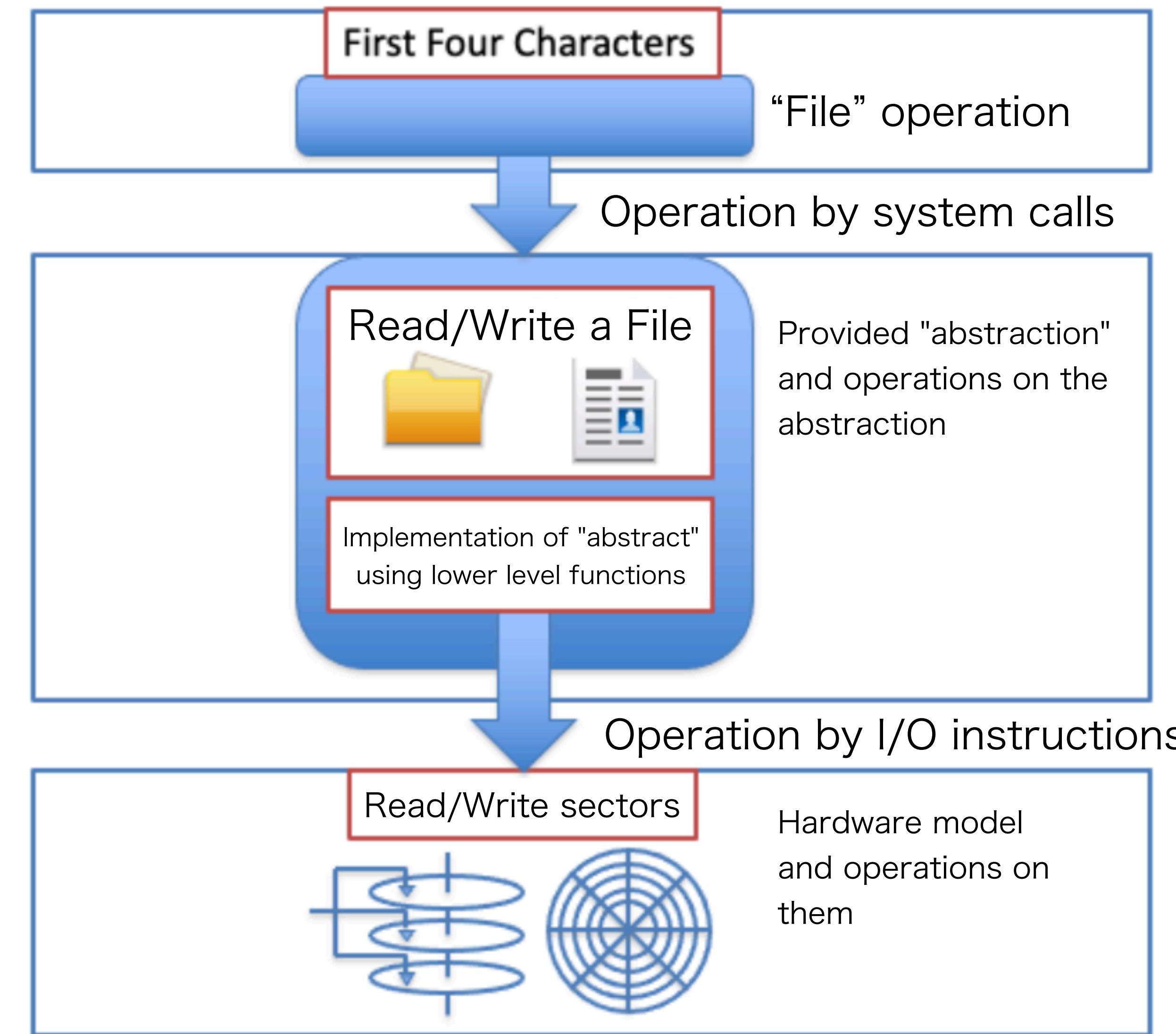
Services provided by HDD device = Read/Write sectors

Abstraction — Layering of Abstraction

User program

OS
(Filesystem,
Device Driver)

Hardware
(HDD device)

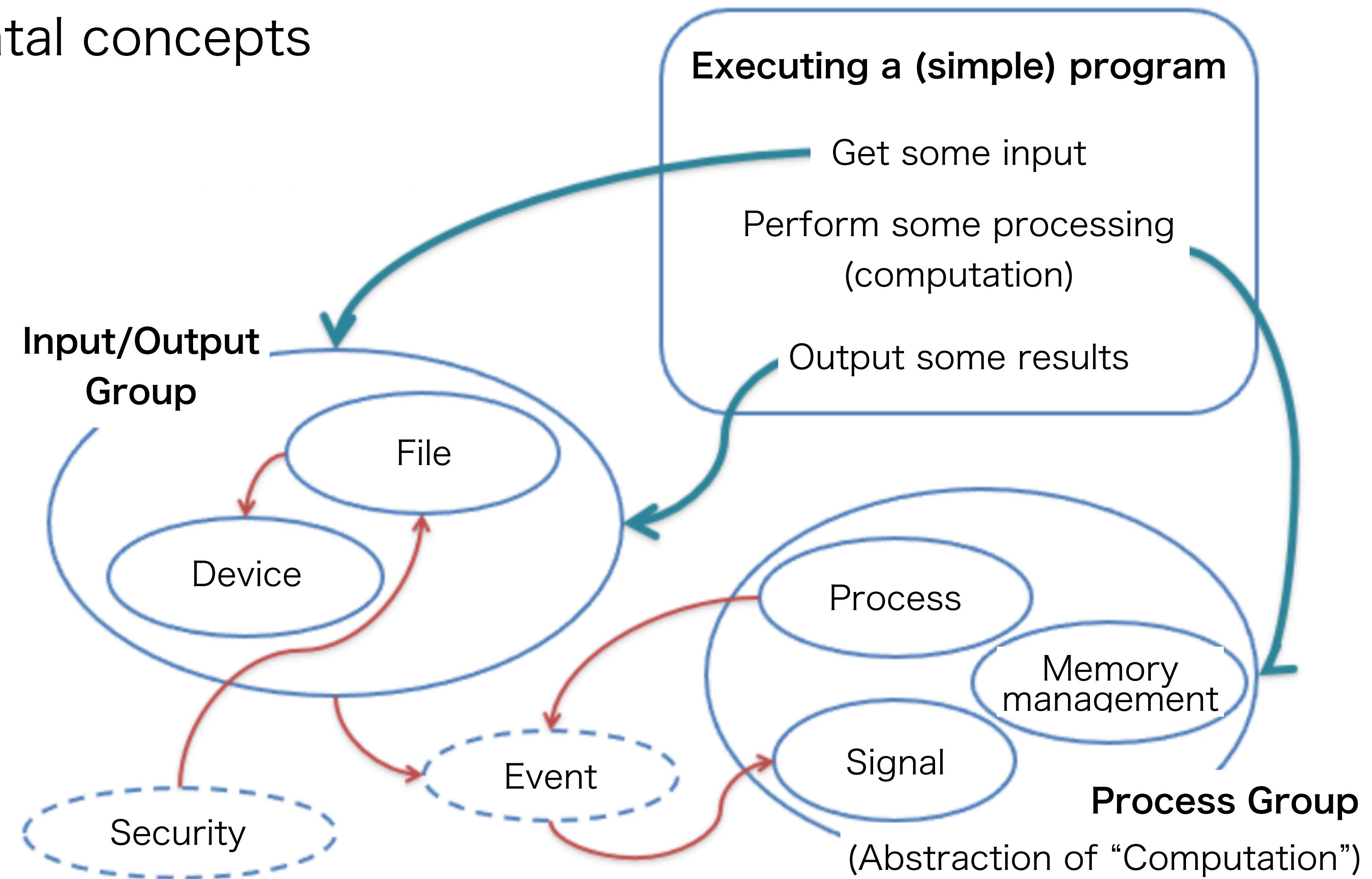


Resource Manager

- Resources for computation
 - Time
 - Space (memory)
 - Devices
- Managing the volume of resources (of things)
 - Various algorithms + Tables
 - Simultaneously, realizing and managing the "thing" aspect of "abstraction".
- Managing the timing of resource use — Sharing and Exclusive use
 - Sharing
 - Time multiplexing
 - Space multiplexing

The concepts (abstractions) that compose an OS are not many!

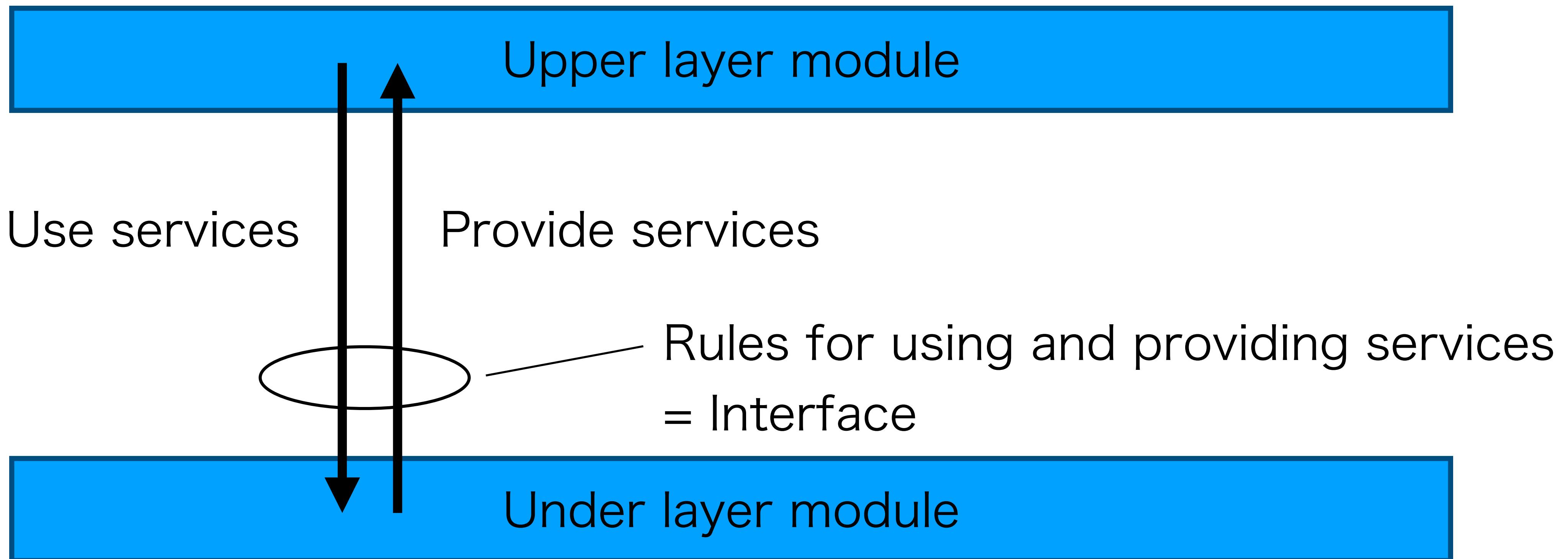
- Two fundamental concepts
 - Process
 - I/O system



Concepts and Abstractions that composing an OS

- Process
- Memory management
- Input / Output
- File
- Security
- (Shell — Execute users' programs)

Layer structure of Software (1)

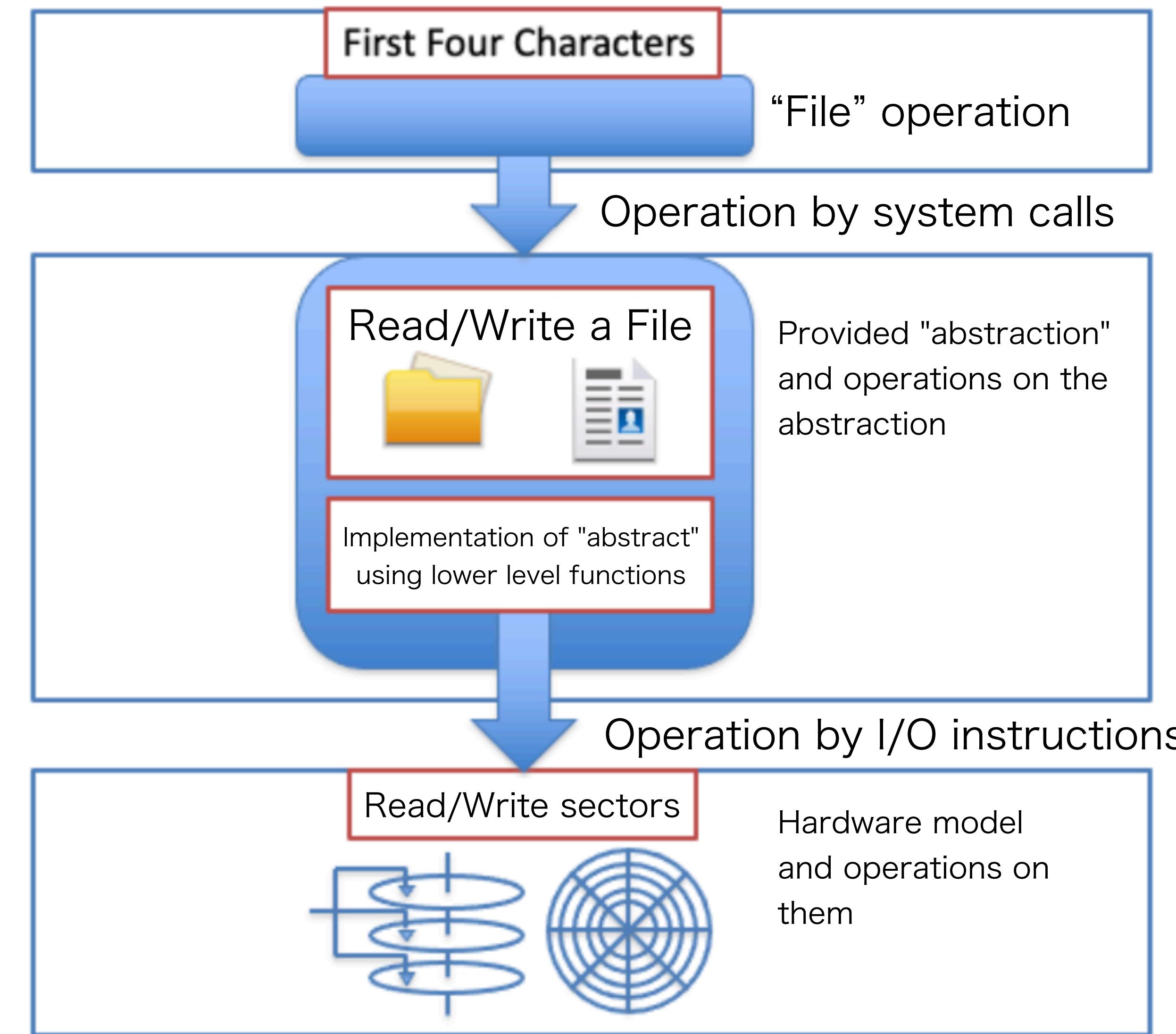


Abstraction — Layering of Abstraction (shown before)

User program

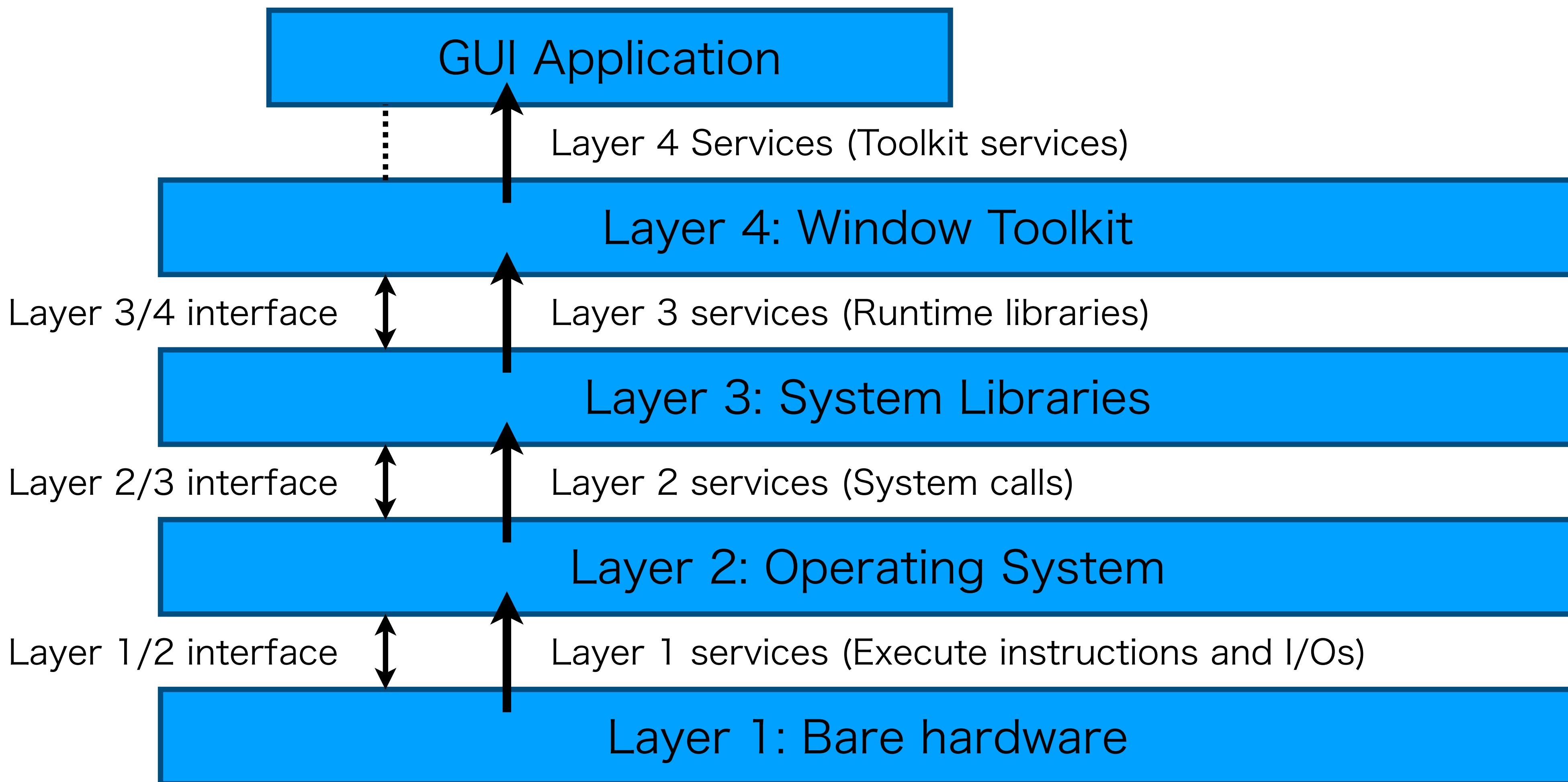
OS
(Filesystem,
Device Driver)

Hardware
(HDD device)



Layer structure of Software (2) — Example on OS

Implement more abstract and advanced services using the services provided by the lower layers, and provide them to the higher layers.



System calls

- Instructions for the "Extended machine" provided by the OS.
- Invoked by special machine instructions (system interrupt, Exception instruction, Trap instruction).
 - Normal call instruction = Only the execution position changes without CPU state transition
 - Trap instruction = With CPU state transition (User mode → System mode)
 - It means that the system call "looks like a normal machine instruction."

System trap instructions and subroutine call instructions on some CPU architectures

CPU arch	TRAP instruction	Subroutine CALL instruction
x86	INT (INTerrupt)	CALL
SPARC	ta (Trap All)	call
ARM	swi (SoftWare Interrupt) svc (SuperVisor Call)	bl (Branch with Link)
R4000	syscall (SYStem CALL) txx (Trap on condition)	jal (Jump And Link)

Execute a system call “read(fd, buffer, bytes)”

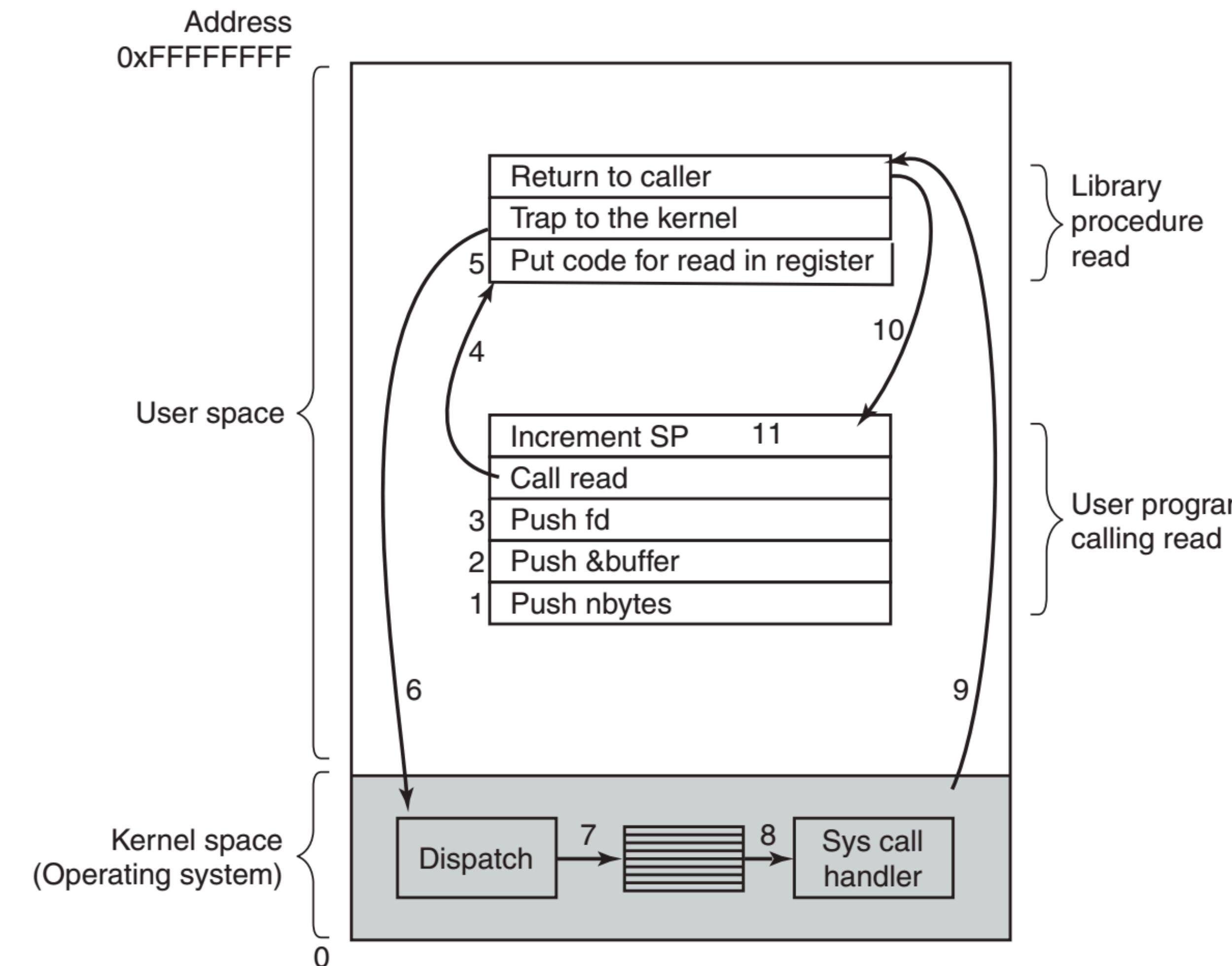


Figure 1-17. The 11 steps in making the system call `read(fd, buffer, nbytes)`.

Organizing OS concepts from the perspective of system calls

- Process management
 - fork
 - waited
 - execve
 - exit
- File management
 - open / close
 - read / write
 - lseek
 - stat
- Directory & File system management
 - mkdir / rmdir
 - link / unlink
 - mount / umount
- Others
 - chdir
 - chmod
 - kill
 - time

Operating System structure (1)

- Monolithic
- Layered Systems
- Virtual Machines
- Microkernels
- Client-Server Model

Operating System structure (2)

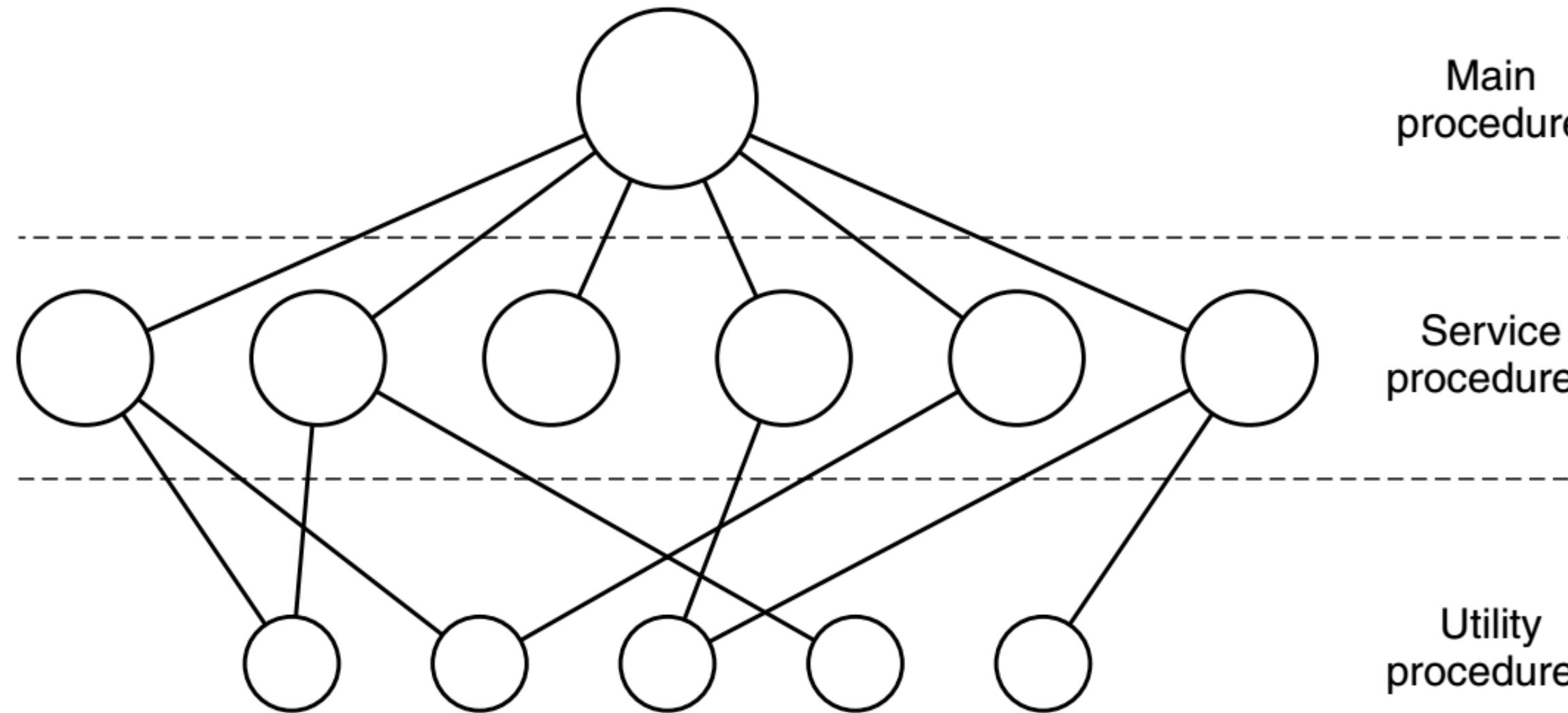


Figure 1-24. A simple structuring model for a monolithic system.

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Figure 1-25. Structure of the THE operating system.

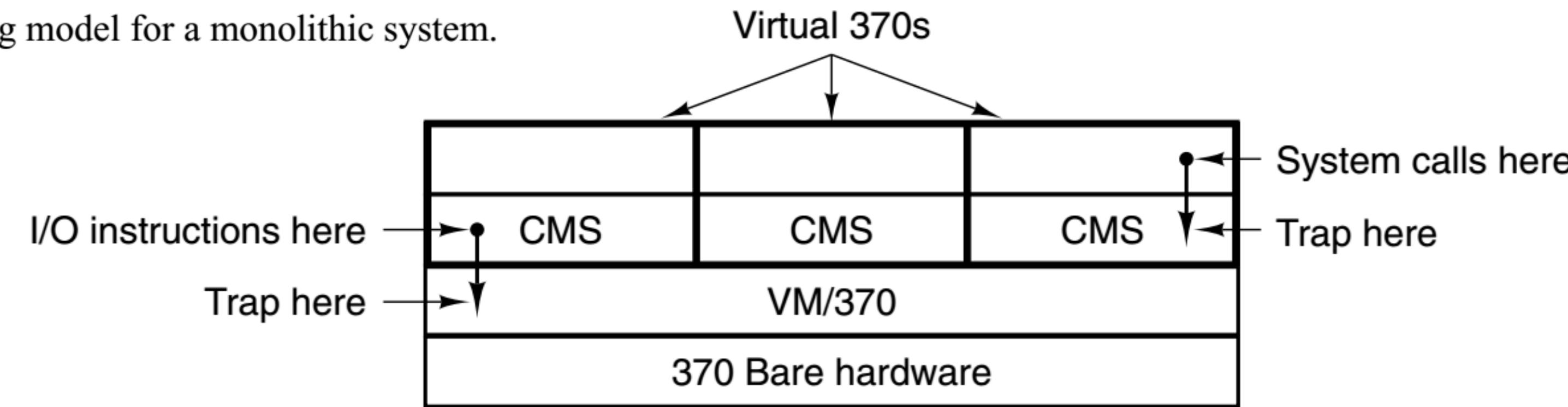


Figure 1-28. The structure of VM/370 with CMS.

Operating System structure (3)

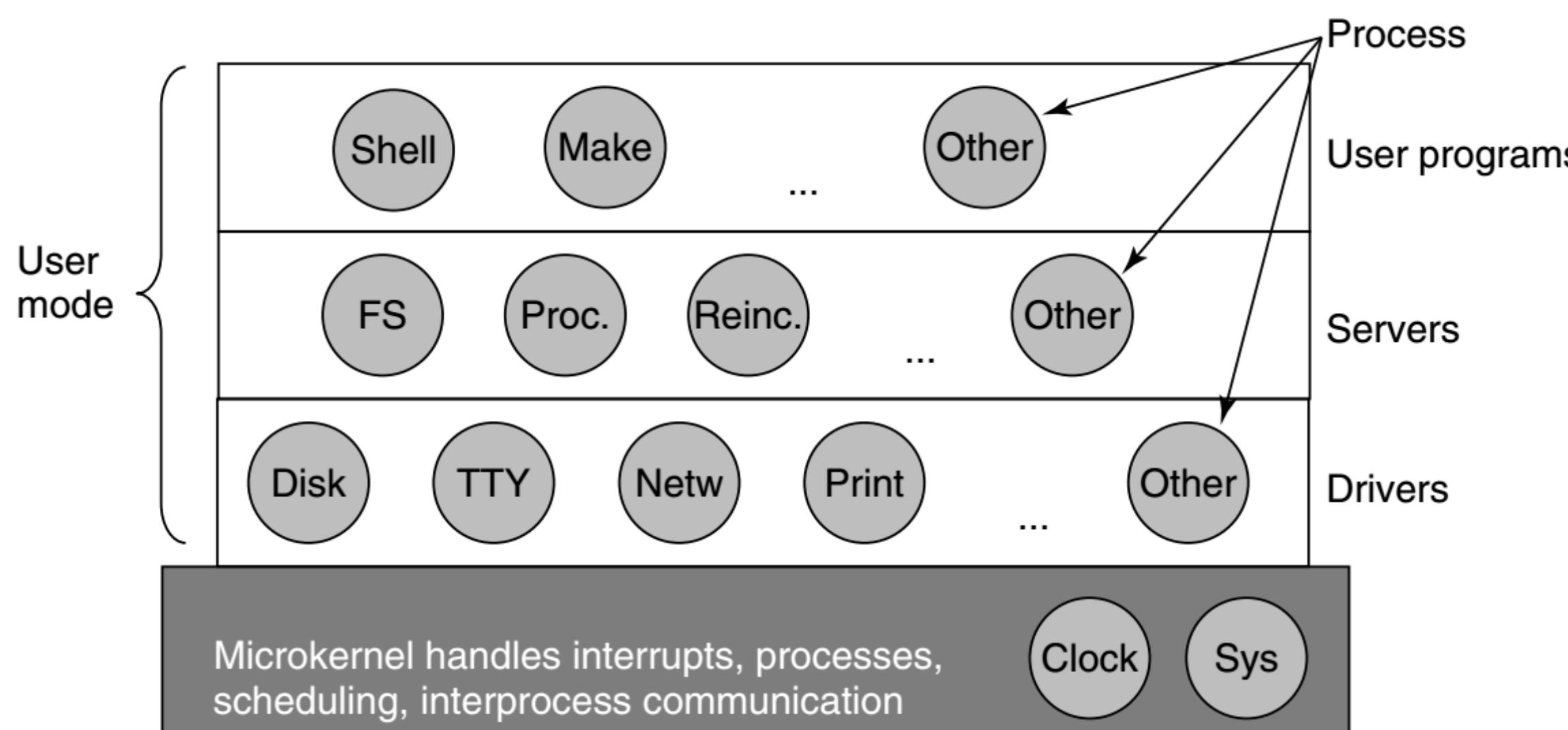


Figure 1-26. Simplified structure of the MINIX system.

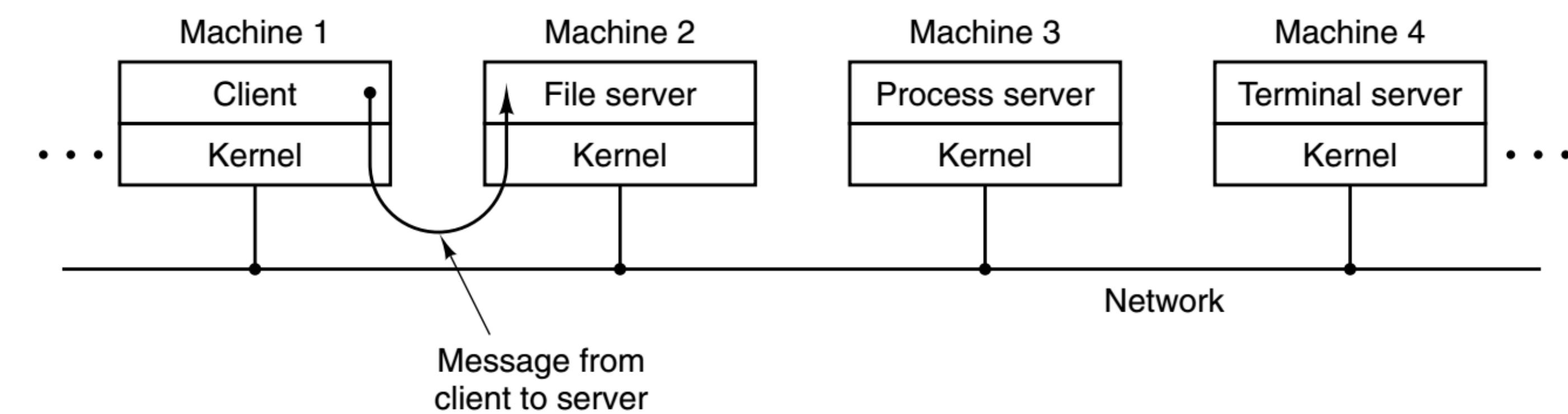


Figure 1-27. The client-server model over a network.

OSs for a variety of environments and usages

- Personal Computer Operating Systems
- Server Operating Systems
- Multiprocessor Operating Systems
- Real-Time Operating Systems
- Embedded Operating Systems
- Smart Card Operating Systems
- Mainframe Operating Systems
- etc.

日本語資料

Logistics

- 使用言語:
 - 講義と資料: 日本語
 - レポートと試験: 日本語 (+英語)
- シラバス: 講述
- 教科書・参考書:
 - Modern Operating Systems, 5th Edition, Global Edition / Andrew S. Tanenbaum, Herbert Bos. Pearson, 2022. (ISBN: 978-0137618828)
 - 邦訳: モダンオペレーティングシステム (第2版), ピアソン・エデュケーション・ジャパン, 2004
- 質問や議論:
 - LMS 上にフォーラムを設けます
- 前提知識:
 - プログラミング言語 (C言語)
 - マシン語に関する若干の知識
 - コンピューターアーキテクチャ (命令・プログラムカウンタ・演算機構・メモリサイクル・デバイス・割り込み)
 - スケジュール
 - 20分ルール
 - チュートリアルアワー
 - 評価
 - 講義(クラス)への貢献: 10% (LMS 上の #presence を毎回クリックして)
 - レポート (調査): 15%
 - レポート(プログラミング・プロジェクト): 30%
 - 期末試験: 45%

講義内容

1. イントロダクション・概要

ファイルシステム

12. ファイル、ディレクトリ

13. ファイルシステムの実装

14. ファイルシステムの例

プロセス

2. プロセス、スレッド

3. プロセス間通信、同期

4. スケジューリング

メモリ管理

7. 仮想記憶、ページング、ページテーブル

8. ページ置き換えアルゴリズム

9. 設計時の課題、セグメンテーション

デッドロック

5. リソース、デッドロック

6. デッドロックの検出、回復、回避、防止

入出力

10. 入出力1 (I/O ハードウェア)

11. 入出力2 (I/O ソフトウェア)

コンピュータハードウェアの概要

- ・プロセッサ
- ・メモリ
- ・入出力デバイス
- ・バス

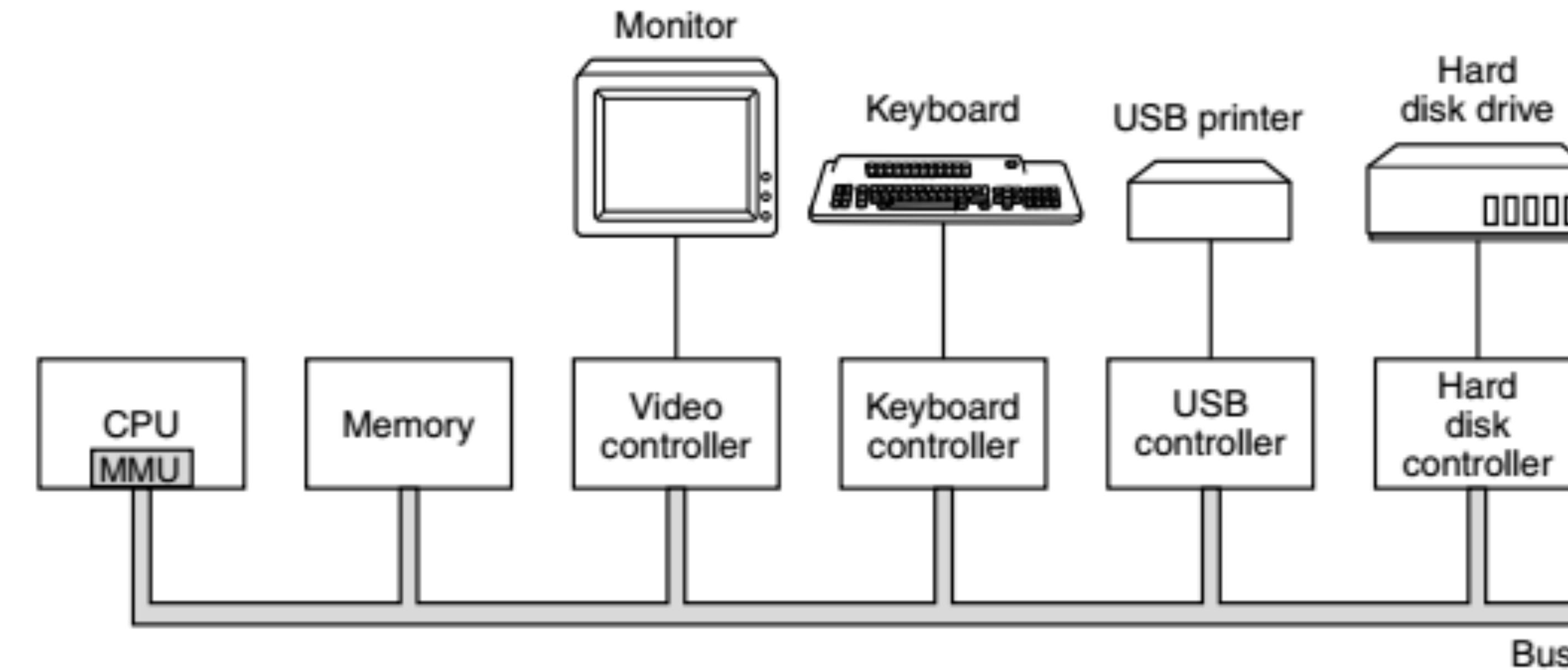


Figure 1-6. Some of the components of a simple personal computer.

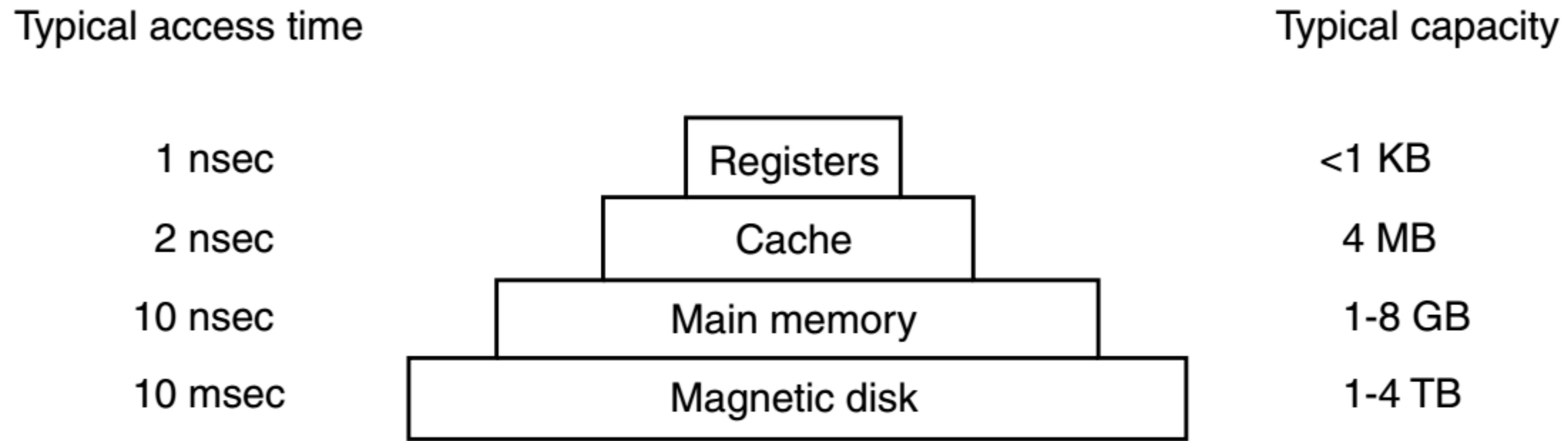


Figure 1-9. A typical memory hierarchy. The numbers are very rough approximations.

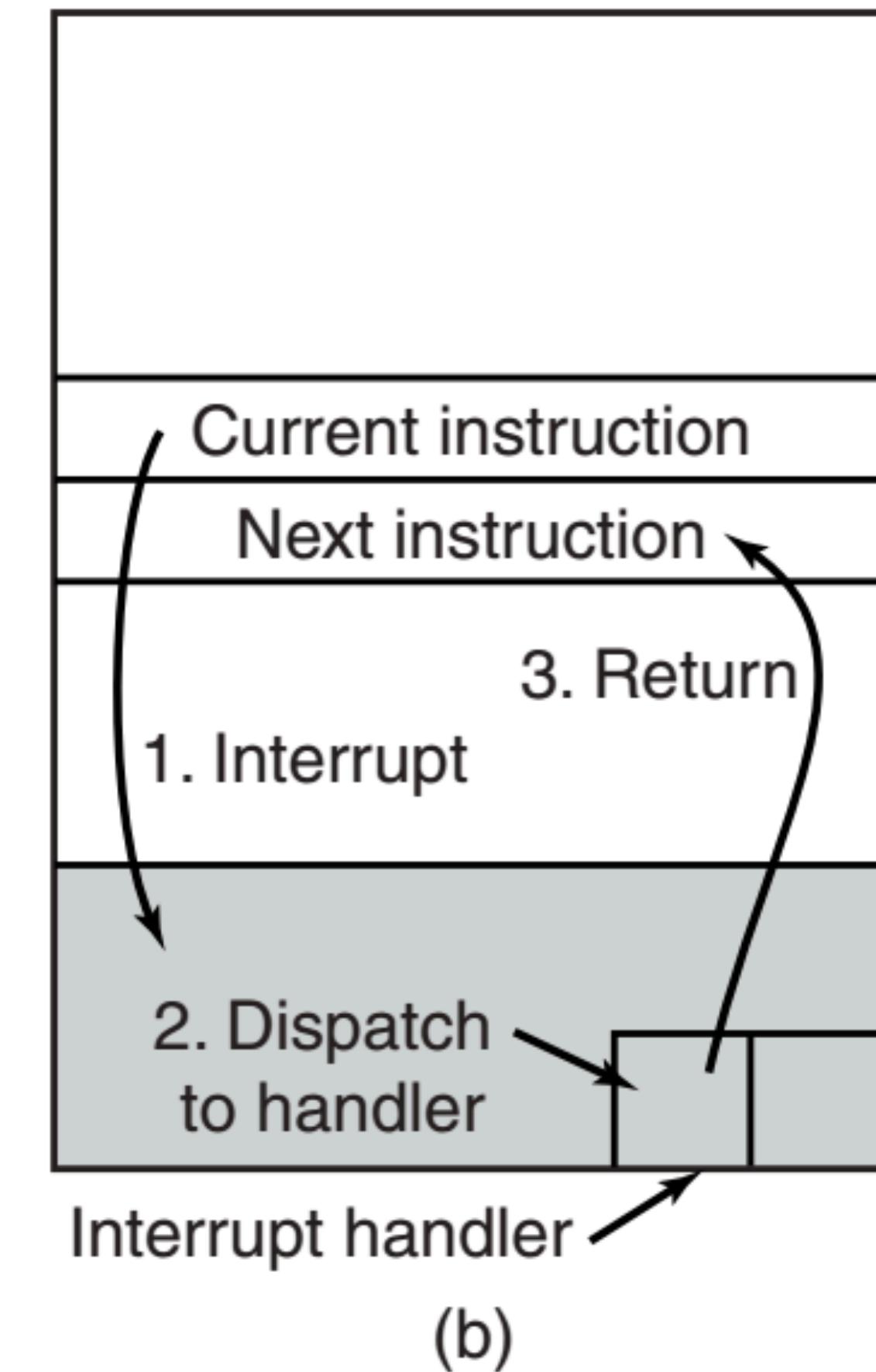
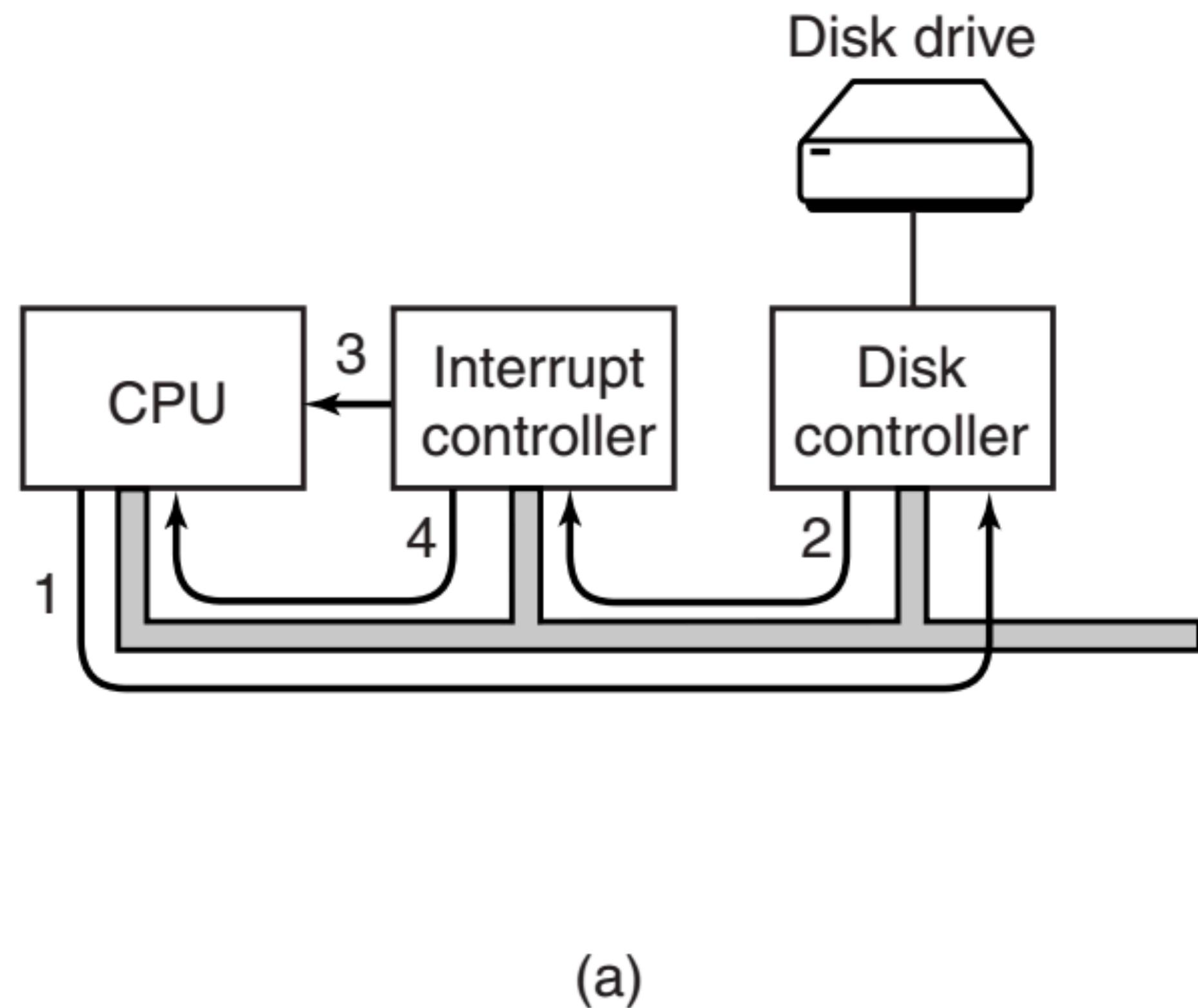


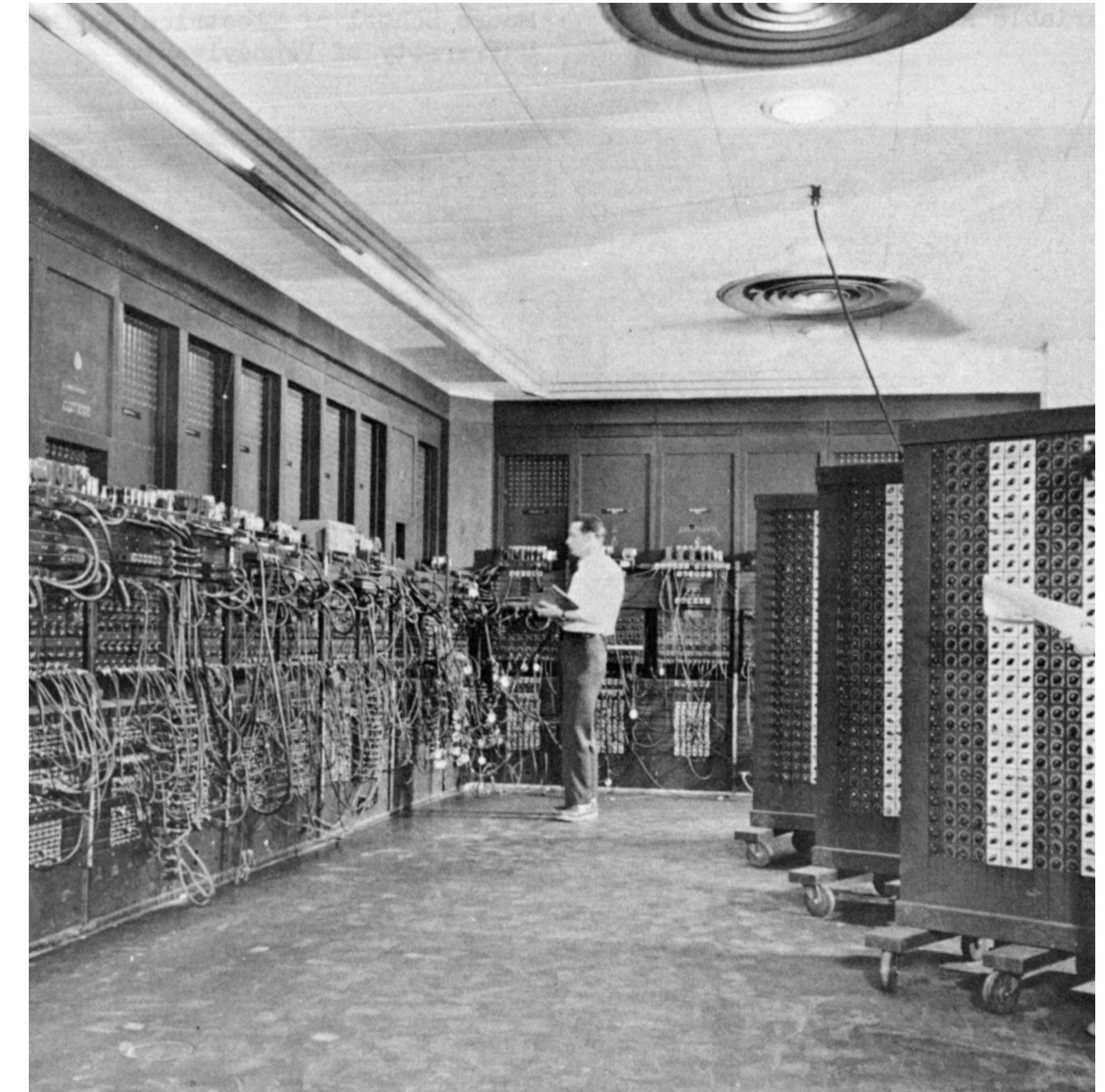
Figure 1-11. (a) The steps in starting an I/O device and getting an interrupt. (b) Interrupt processing involves taking the interrupt, running the interrupt handler, and returning to the user program.

ハードウェアとOSの歴史

- ・ 第一世代 (1945-55): 真空管と配線盤
- ・ 第二世代 (1955-65): トランジスタとバッチシステム
- ・ 第三世代 (1965-80): ICとマルチプログラミング
- ・ 第四世代 (1980-): パソコン
- ・ 第五世代 (2005-): Webセントリックとクラウド分散

ENIAC - Electric Numerical Integrator and Computer

- ・ 真空管 18,000 本、ダイオード 7,200 個、リレー 1,500 個、抵抗器 70,000 個、コンデンサ 10,000 個.
- ・ 大きさ: 2.4m(H)×0.9m(D)×30m(W), 重量: 30t
- ・ 加算 5,000 ops/s, 乗算 385 ops/s, 除算 40 ops/s.
- ・ 機械的加乗算器の電気的エミュレーション
- ・ プログラム内蔵式ではなかった



IBM7090



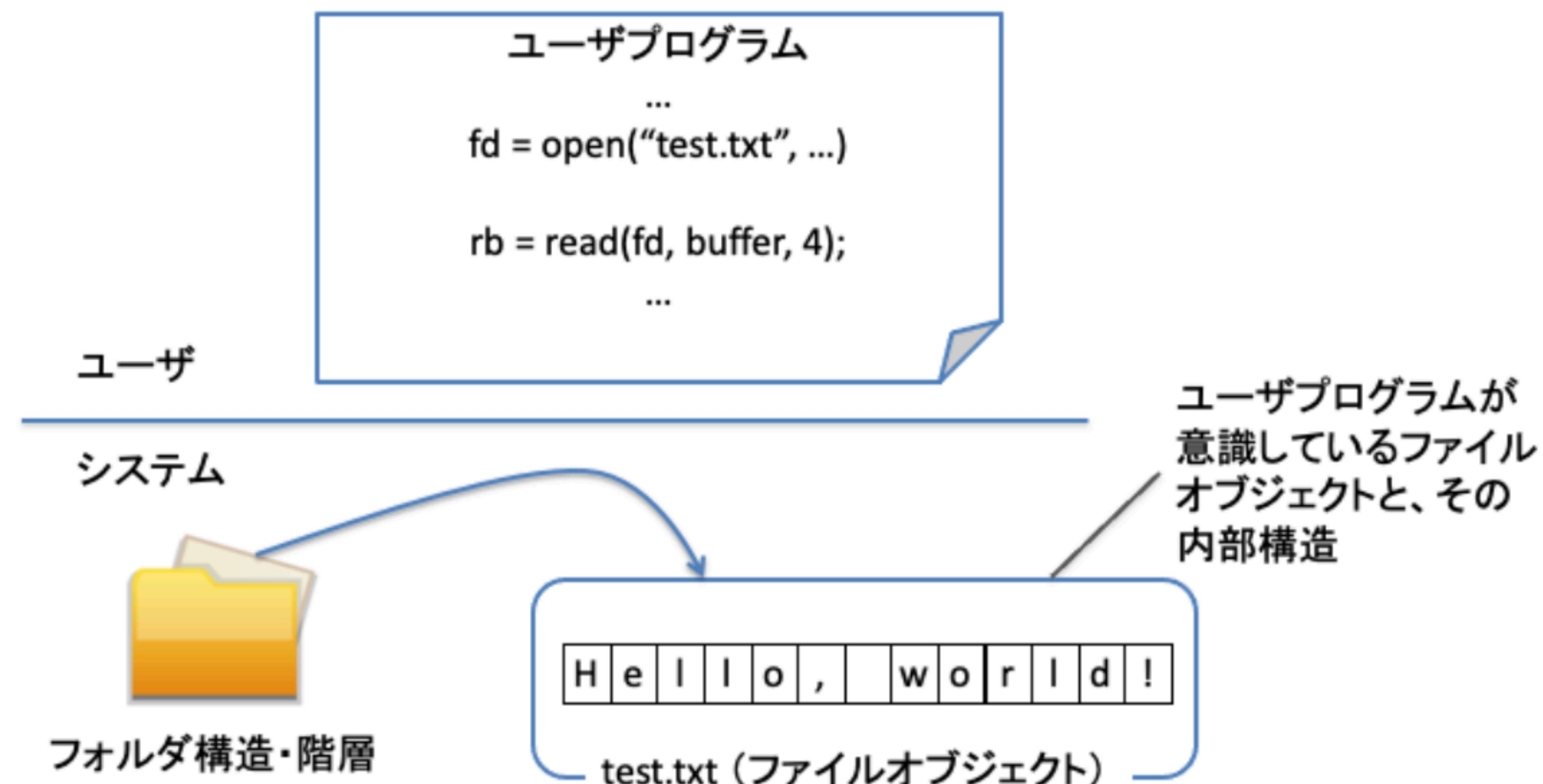
Source: <https://thisdayintechhistory.com/11/30/ibm-7090-delivered/>

オペレーティングシステム(OS)の本質

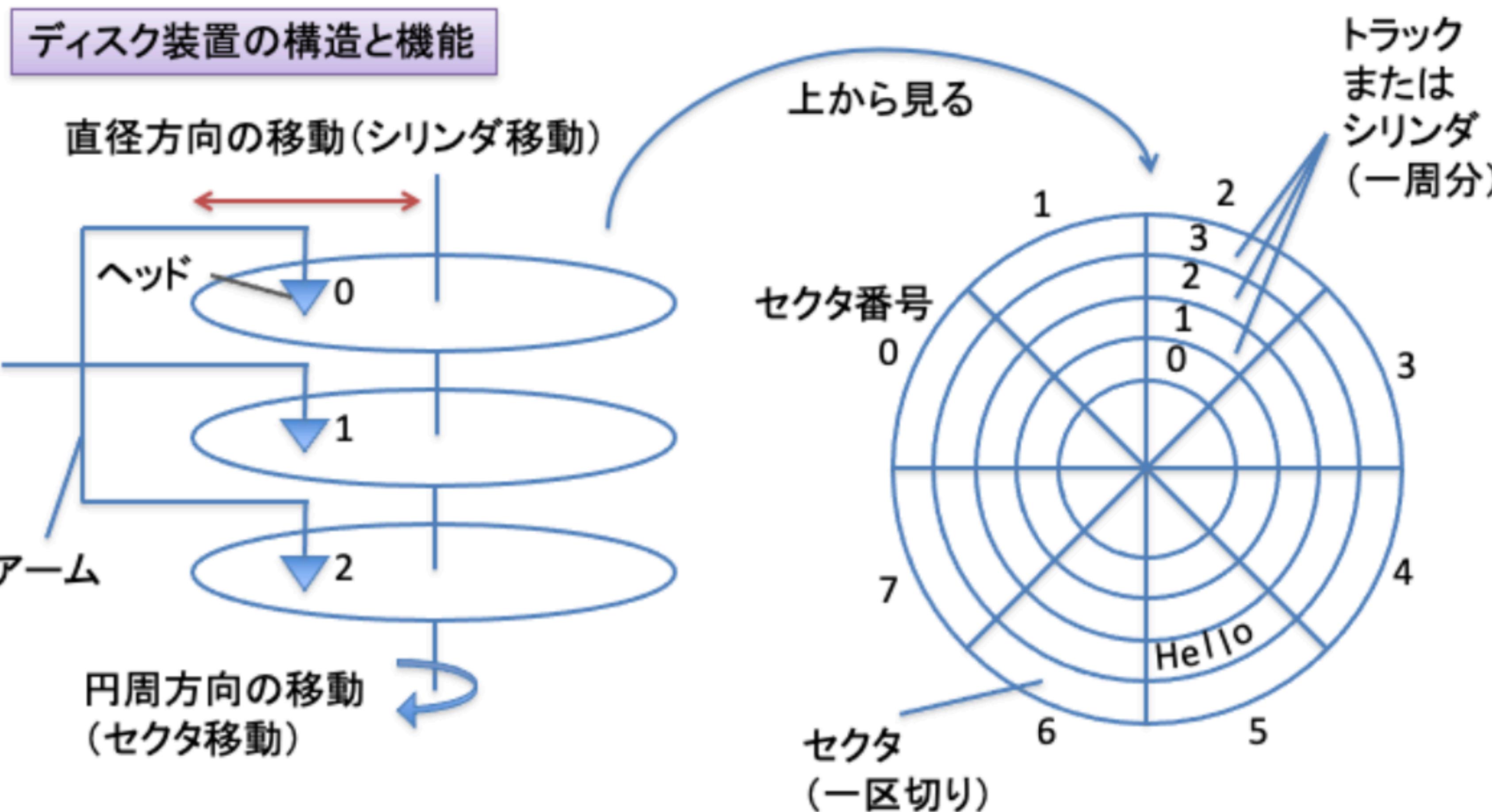
- ・ 拡張マシン
 - ・ 裸のハードウェアを隠蔽し、より高度な機能を持つ「拡張マシン(計算機)」を提供する
= 使い勝手の良い計算機。
 - ・ これは「抽象化(アブストラクション - abstraction)」と呼ばれる操作である。
 - ・ 裸のハードウェア、OS、システムライブラリ、システムユーティリティ、GUIライブラリ等のシステムコンポーネントは、「抽象化の系列(層)」を成す。
- ・ リソーススマネージャ
 - ・ 計算機を構成するすべての資源(リソース)を、それらの資源を利用して仕事を行おうとしている多くのプログラムに、「ある方針」のもとに割り当てる。
 - ・ 資源(ハードウェアレベル): プロセッサ、メモリ、タイマ、入出力デバイス(ディスク、マウス、キーボード、ネットワークへのアクセスと帯域)
 - ・ 資源利用の多重化(共有)を行う。
 - ・ 資源利用の排他制御を行う。

抽象化とは — ファイル処理を例に (1)

“test.txt” という名前のついたファイルの、先頭4文字をメモリに読み出すプログラム



抽象化とは — ファイル処理を例に (2)

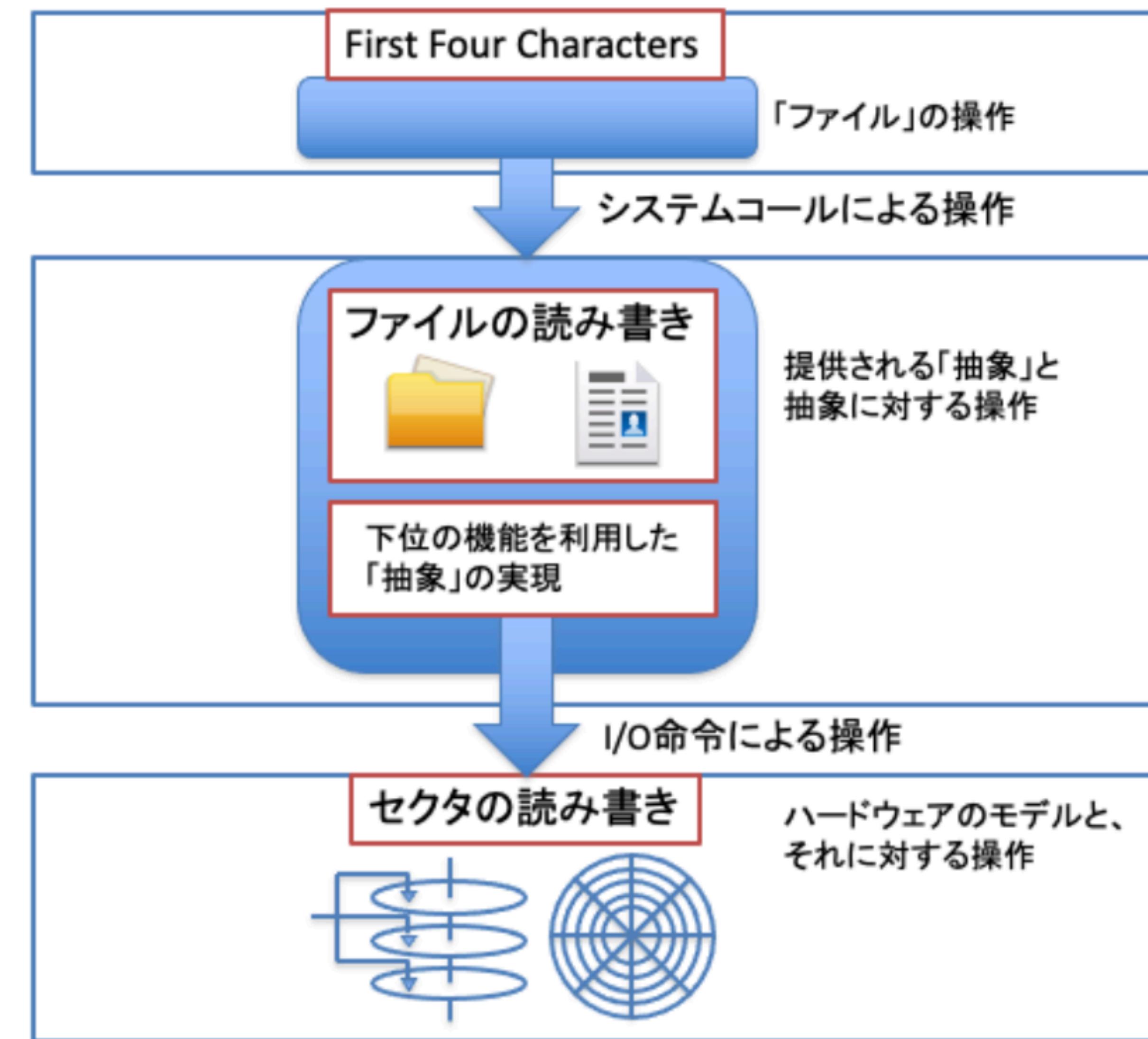


セクタ指定 = ヘッド番号 + シリンダ番号 + セクタ番号

ハードディスクの提供するサービス = セクタの読み書き

抽象化とは — 抽象化のレイヤリング

ユーザプログラム
OS
(ファイルシステム,
デバイスドライバ)
ハードウェア
(ハードディスク装置)

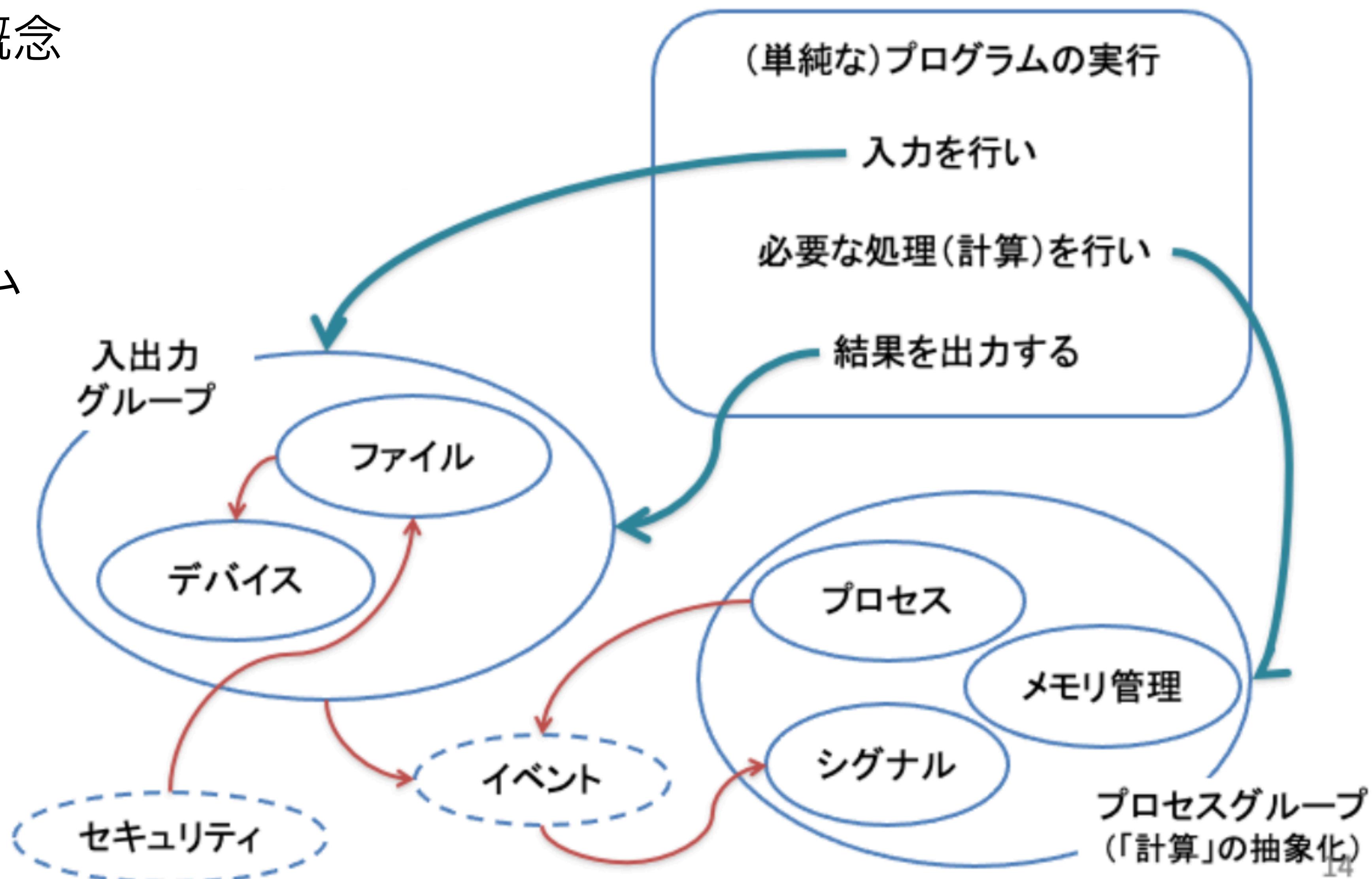


リソースマネージャ (資源管理)

- ・ 計算に関わる資源
 - ・ 時間
 - ・ 空間 (メモリ)
 - ・ デバイス
- ・ 資源の(モノの)量の管理
 - ・ 各種アルゴリズム + テーブル
 - ・ 同時に「抽象」の「モノ」の側面を実現・管理する
- ・ 資源の使用タイミングの管理 — 共有と排他的利用
 - ・ 共有
 - ・ 時間分割
 - ・ 空間分割

OSを構成する概念(抽象)は多くない!

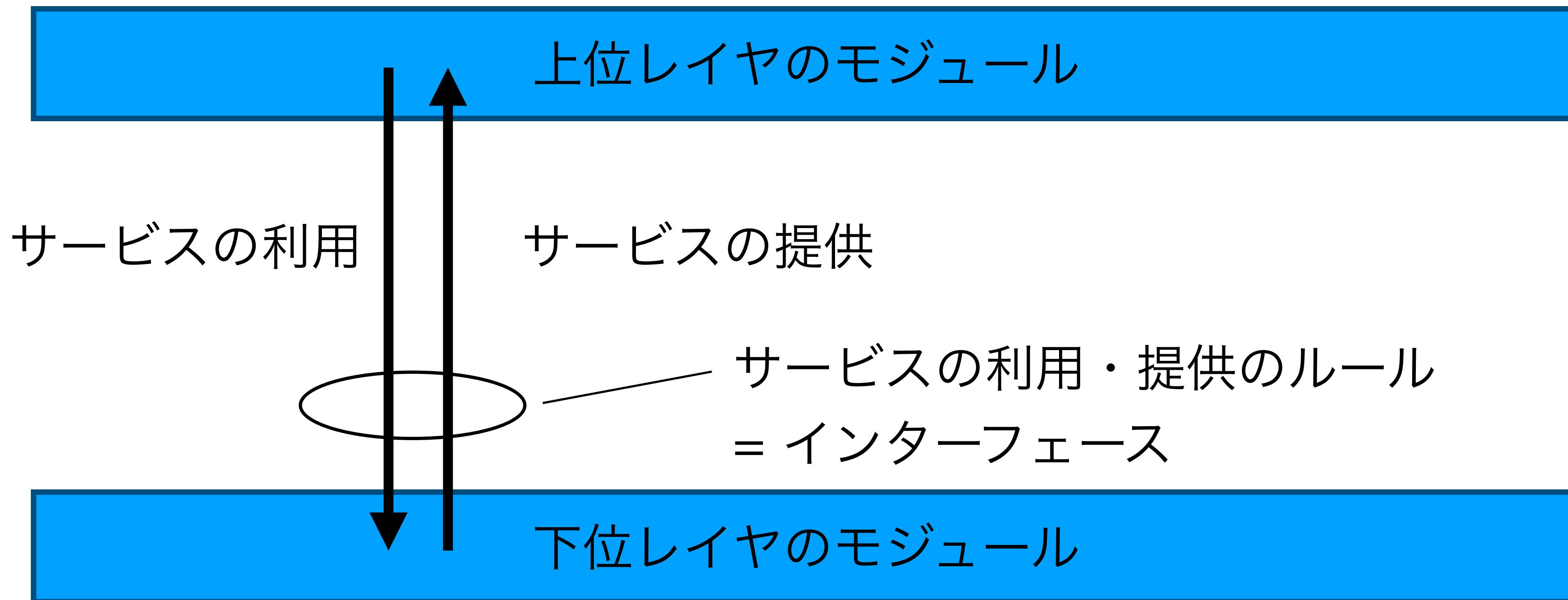
- 基本的な2つの概念
 - プロセス
 - 入出力システム



OSを構成する概念・抽象

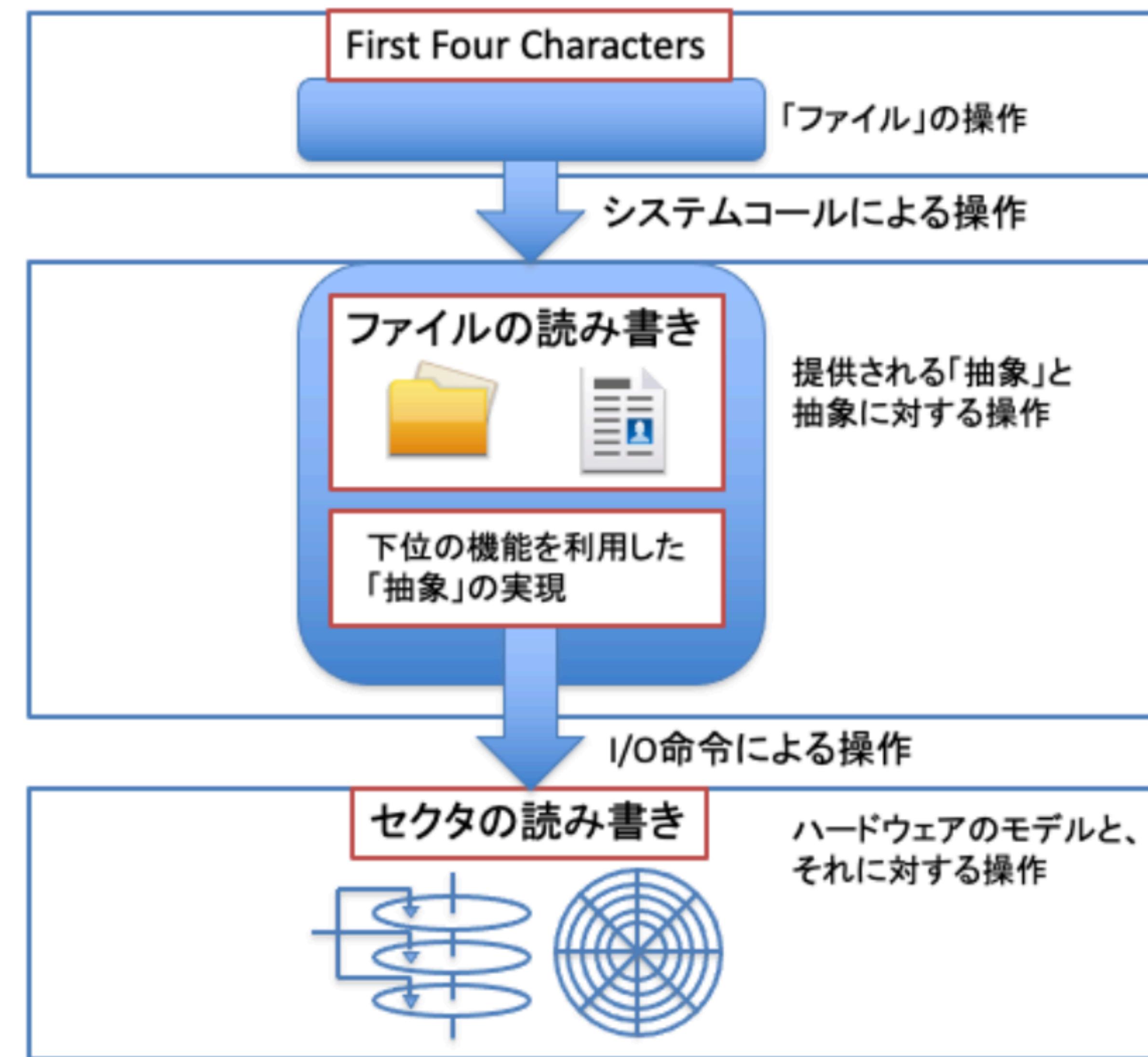
- プロセス
- メモリ管理
- 入出力
- ファイル
- セキュリティ
- (シェル – ユーザプログラムの起動)

ソフトウェアのレイヤ構造 (1)



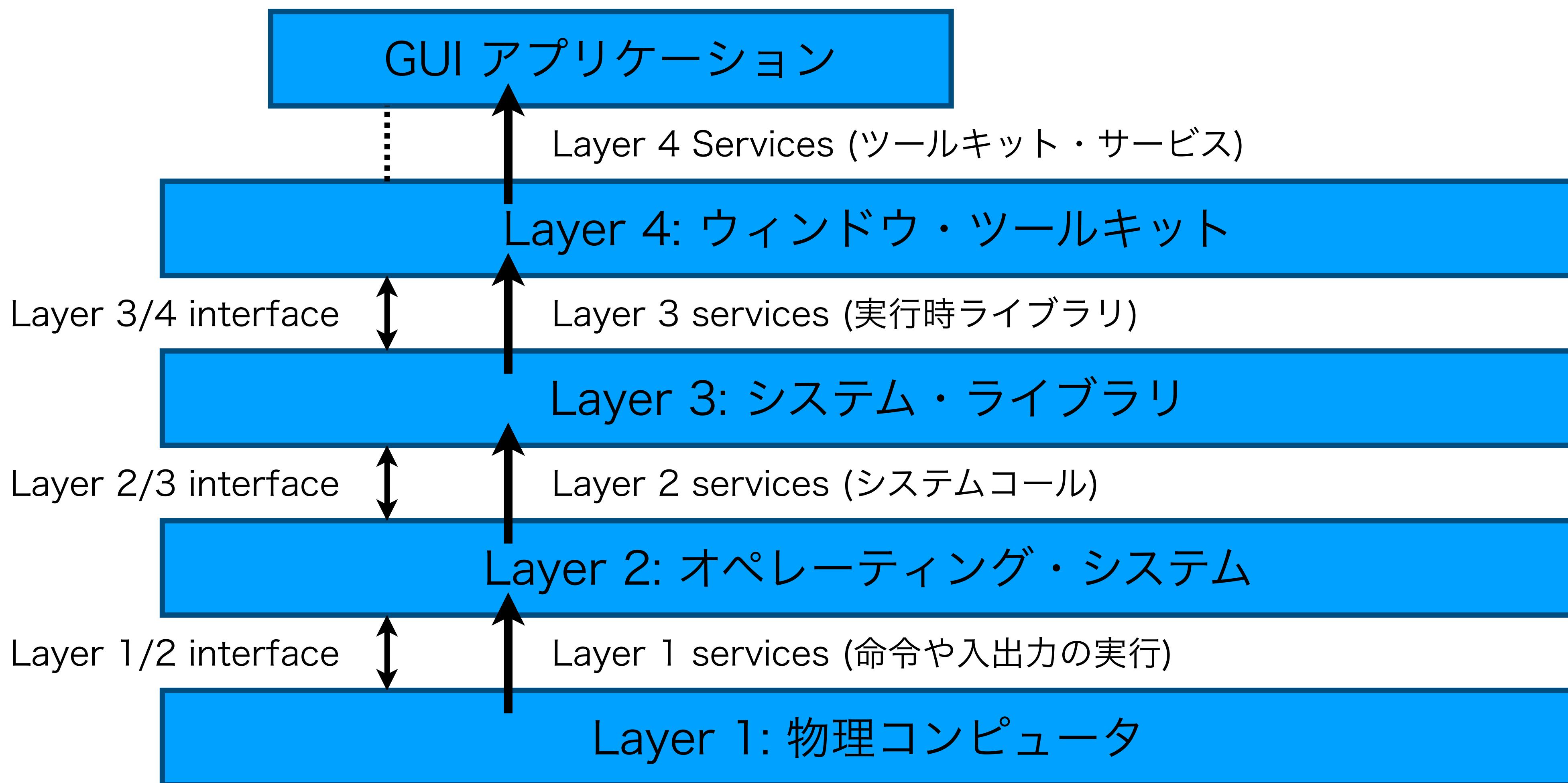
抽象化とは — 抽象化のレイヤリング (再掲)

ユーザプログラム
OS
(ファイルシステム,
デバイスドライバ)
ハードウェア
(ハードディスク装置)



ソフトウェアのレイヤ構造 (2) — OSにおける例

下位のレイヤが提供するサービスを利用して、より抽象度の高い高度なサービスを実現し、それを上位のレイヤに提供する。



システムコール

- ・ OS の提供する「仮想マシン」の命令に相当。
- ・ 特殊なマシン命令(システム割り込み・割り出し命令・トラップ命令)で起動される。
 - ・ 通常のコール(CALL)命令 = CPU の状態遷移を伴わないで、実行位置のみが変化する。
 - ・ トラップ(TRAP)命令 = CPU の「状態遷移(ユーザ→システム)」を伴う。
 - ・ システムコールが「通常のマシン命令のように見える」ことを意味する。

いくつかの CPU アーキテクチャにおけるシステムトラップ命令とサブルーチン呼び出し命令

CPU arch	システムトラップ命令	サブルーチン呼び出し命令
x86	INT (INTerrupt)	CALL
SPARC	ta (Trap All)	call
ARM	swi (SoftWare Interrupt) svc (SuperVisor Call)	bl (Branch with Link)
R4000	syscall (SYStem CALL) txx (Trap on condition)	jal (Jump And Link)

システムコール “read(fd, buffer, bytes)” の実行

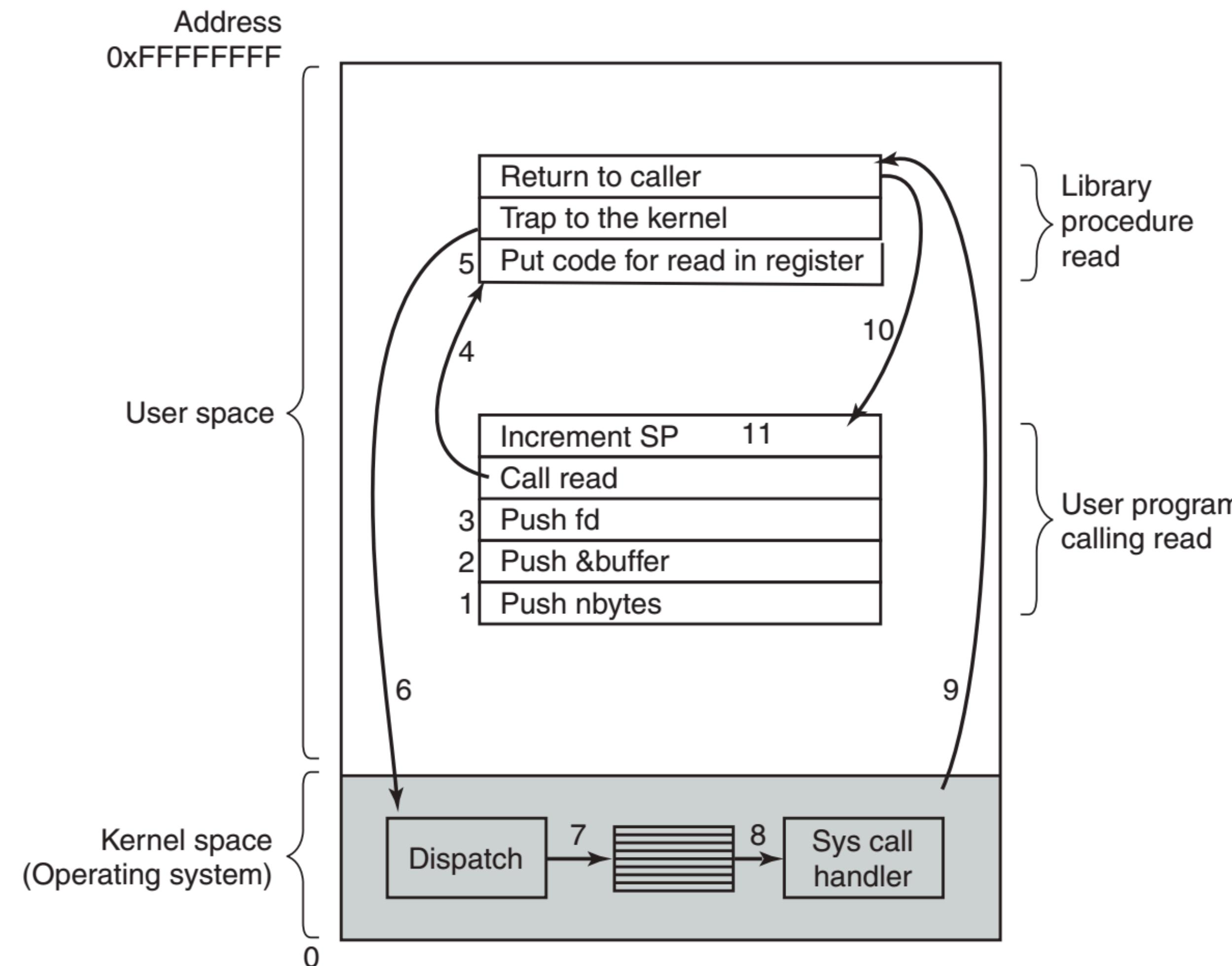


Figure 1-17. The 11 steps in making the system call `read(fd, buffer, nbytes)`.

システムコールから見た OS の概念の整理

- ・ プロセス管理
 - ・ fork
 - ・ waited
 - ・ execve
 - ・ exit
- ・ ファイル管理
 - ・ open / close
 - ・ read / write
 - ・ lseek
 - ・ stat
- ・ ディレクトリ・ファイルシステム管理
 - ・ mkdir / rmdir
 - ・ link / unlink
 - ・ mount / umount
 - ・ その他
 - ・ chdir
 - ・ chmod
 - ・ kill
 - ・ time

OS の構造 (1)

- ・ モノリシック
- ・ 階層システム
- ・ 仮想マシン
- ・ 外部カーネル (ハイパーバイザ型)
- ・ クライアント・サーバモデル

OS の構造 (2)

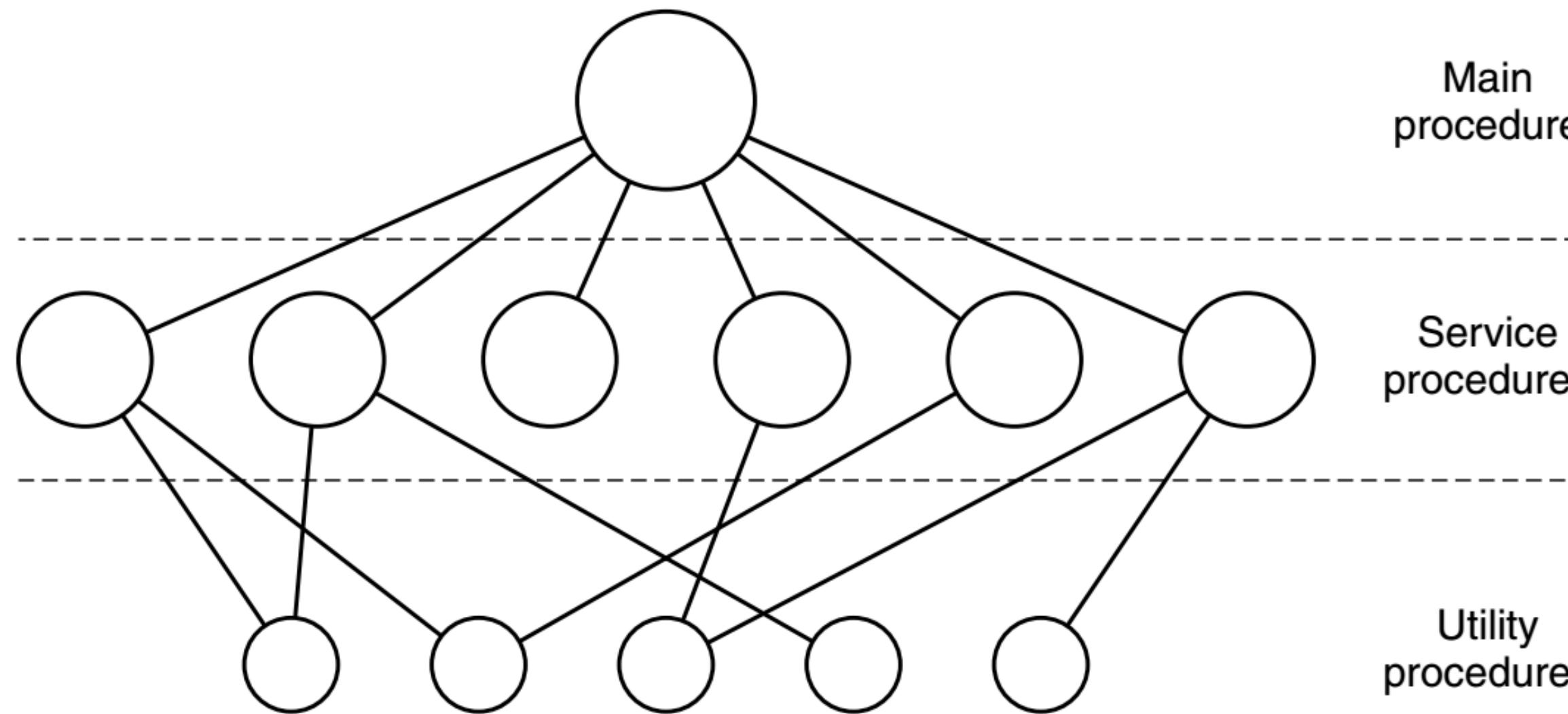


Figure 1-24. A simple structuring model for a monolithic system.

Layer	Function
5	The operator
4	User programs
3	Input/output management
2	Operator-process communication
1	Memory and drum management
0	Processor allocation and multiprogramming

Figure 1-25. Structure of the THE operating system.

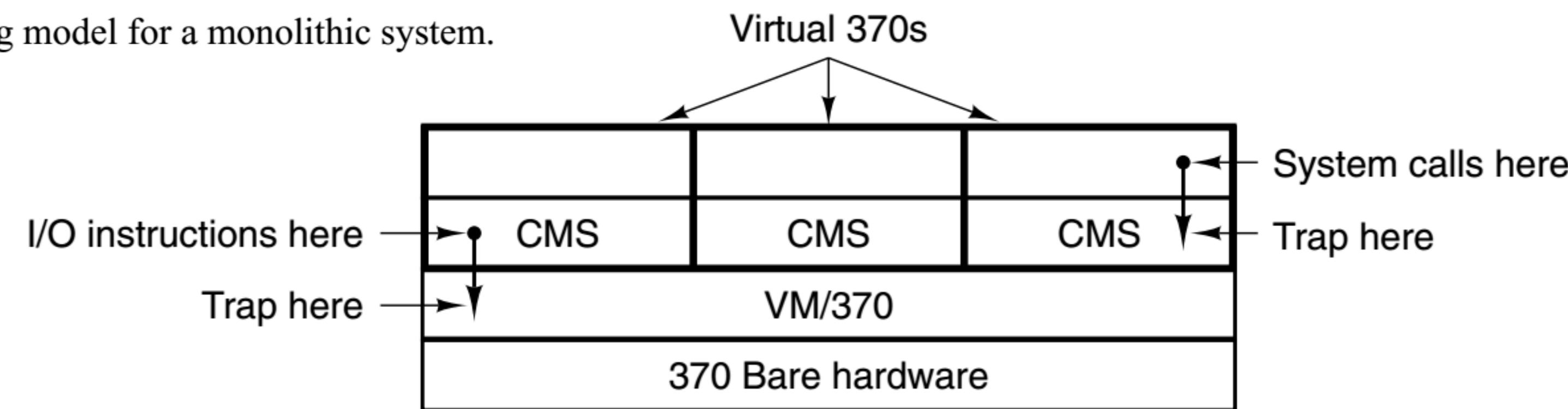


Figure 1-28. The structure of VM/370 with CMS.

OS の構造 (3)

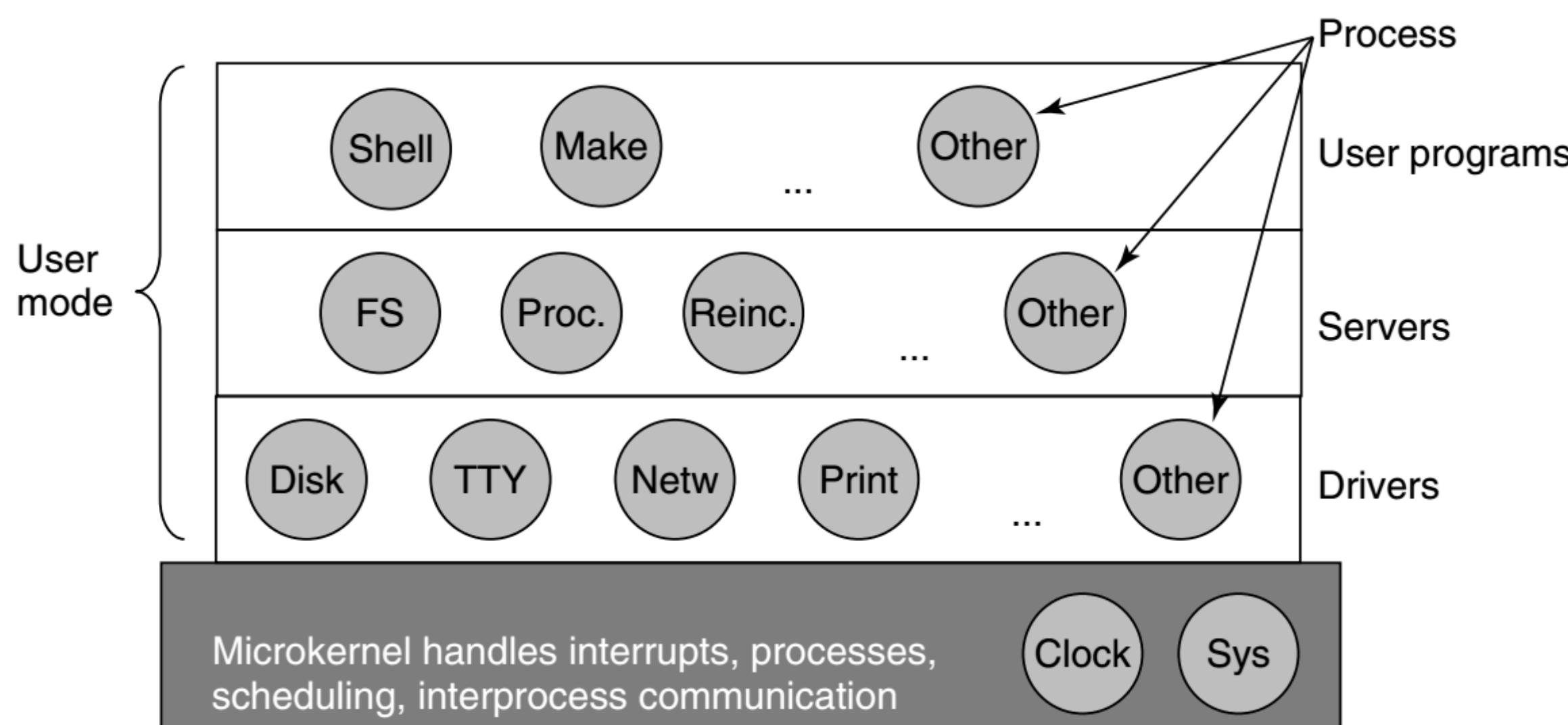


Figure 1-26. Simplified structure of the MINIX system.

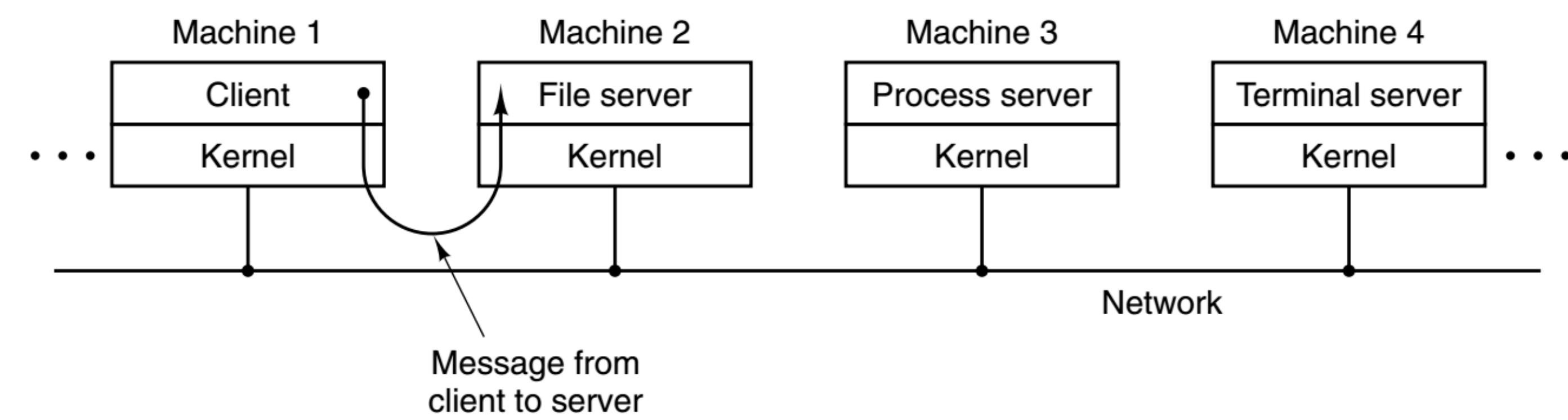


Figure 1-27. The client-server model over a network.

さまざまな環境・用途向けの OS

- ・ パソコン用 OS
- ・ サーバ用 OS
- ・ マルチプロセッサ用 OS
- ・ リアルタイム用 OS
- ・ 組込み用 OS
- ・ スマートカード用 OS
- ・ メインフレーム用 OS
- ・ etc.