HOMEWORK – TRANSLATION OF TECHNICAL TEXT

Date: 13/5/2026
Topic: Operating System Model

TEXT FOR TRANSLATION (ENGLISH → SERBIAN)

1. Operating System Model

In most multiuser operating systems, applications are separated from the operating system itself. The operating system kernel code runs in a privileged processor mode (referred to as kernel mode in this book), with access to system data and to the hardware. Application code runs in a nonprivileged processor mode (called user mode), with a limited set of interfaces available, limited access to system data, and no direct access to hardware.

When a user-mode program calls a system service, the processor traps the call and then switches the calling thread to kernel mode. When the system service completes, the operating system switches the thread context back to user mode and allows the caller to continue.

Windows is similar to most UNIX systems in that it is a monolithic operating system in the sense that the bulk of the operating system and device driver code shares the same kernel-mode protected memory space. This means that any operating system component or device driver can potentially corrupt data being used by other operating system components.

---

2. Is Windows a Microkernel-Based System?

Although some claim it as such, Windows is not a microkernel-based operating system in the classic definition of microkernels, where the principal operating system components (such as the memory manager, process manager, and I/O manager) run as separate processes in their own private address spaces, layered on a primitive set of services provided by the microkernel.

For example, the Carnegie Mellon University Mach operating system, a contemporary example of a microkernel architecture, implements a minimal kernel that comprises thread scheduling, message passing, virtual memory, and device drivers. Everything else, including APIs, file systems, and networking, runs in user mode.

However, commercial implementations of the Mach microkernel typically run at least file system, networking, and memory management code in kernel mode. The reason is simple: the pure microkernel design is commercially impractical because it is too inefficient.

---

3. Reliability and Kernel Mode

Does the fact that so much of Windows runs in kernel mode mean that it is more susceptible to crashes than a true microkernel operating system? Not necessarily.

Consider the following scenario. Suppose the file system code of an operating system has a bug that causes it to crash from time to time. In a traditional operating system, a bug in kernel-mode code such as the memory manager or the file system would likely crash the entire operating system.

In a pure microkernel operating system, such components run in user mode, so theoretically a bug would simply mean that the component process exits. However, in practical terms, the system would still likely crash because recovering from the failure of such a critical process is difficult or impossible.

---

4. Protection and Stability

All operating system components are fully protected from errant applications because applications do not have direct access to the code and data of the privileged part of the operating system. They can, however, call system services.

This protection is one of the reasons that Windows is considered both robust and stable as an application server and workstation platform, while also being fast and efficient in core operating system services such as virtual memory management, file I/O, networking, and file and print sharing.

---

5. Object-Oriented Design in Windows

The kernel-mode components of Windows also embody basic object-oriented design principles. For example, they generally do not access each other's internal data structures directly. Instead, they use formal interfaces to pass parameters and access or modify data structures.

Despite its use of objects to represent shared system resources, Windows is not an object-oriented system in the strict sense. Most of the operating system code is written in C for portability and because C development tools are widely available. C does not directly support object-oriented constructs such as dynamic binding, polymorphism, or inheritance.

---

6. Architecture Overview

With this brief overview of the design goals and packaging of Windows, let us take a look at the key system components that make up its architecture.

The architecture is divided into user mode and kernel mode. User-mode processes include system support processes, service processes, user applications, and environment subsystems. Each of these runs in its own private process address space.

Kernel-mode components include the Windows executive, Windows kernel, device drivers, the hardware abstraction layer (HAL), and the windowing and graphics system.

(SOURCE: Mark Russinovich, David Solomon, Alex Ionescu – *Windows Internals*, 2008, p. 34)

---

7. User-Mode Process Types

The four basic types of user-mode processes are:

- Fixed (or hardwired) system support processes, such as the logon process and Session Manager.
- Service processes that host Windows services such as Task Scheduler and Print Spooler.
- User applications (Windows 32-bit, Windows 64-bit, Windows 16-bit, MS-DOS, POSIX).
- Environment subsystem server processes that implement operating system personalities.

---

8. Kernel-Mode Components

Kernel-mode components include:

- Windows executive (memory, process, security, I/O, networking, IPC)
- Windows kernel (scheduling, interrupts, synchronization)
- Device drivers (hardware, file system, network drivers)
- HAL (hardware abstraction layer)
- Windowing and graphics system (GUI: USER and GDI)

INSTRUCTIONS FOR STUDENT

- Translate the entire text into Serbian.
- Maintain technical accuracy.
- Preserve terminology consistency (kernel mode, user mode, HAL, etc.).
- Pay attention to clarity and precision.
- Do not simplify the technical content.
- Focus on correct rendering of system architecture terminology.

SECTION 2
TOPIC: SQUEEZING THEOREM (SANDWICH THEOREM)
DATE: _____

PROOF (SQUEEZING THEOREM)

Let f, g, and h be three functions with the same domain, and let a be an accumulation point of that domain.

Assume that for all x in that domain:
g(x) <= f(x) <= h(x)

and that:
lim (x -> a) g(x) = lim (x -> a) h(x) = L

Then:
lim (x -> a) f(x) = L

PROOF STEPS:

• Assume that a is an accumulation point of the common domain of f, g, and h.

• Assume:
lim (x -> a) g(x) = L
lim (x -> a) h(x) = L

• Assume:
g(x) <= f(x) <= h(x) for all x in the common domain.

• Let epsilon > 0 be given.

• Then there exists delta1 > 0 such that:
for all x with 0 < |x - a| < delta1,
|g(x) - L| < epsilon

• There exists delta2 > 0 such that:
for all x with 0 < |x - a| < delta2,
|h(x) - L| < epsilon

• Let delta = min(delta1, delta2).

• Then for all x with 0 < |x - a| < delta:
|g(x) - L| < epsilon
|h(x) - L| < epsilon

• Since:
g(x) <= f(x) <= h(x)

it follows that:
-L - epsilon < g(x) - L <= f(x) - L <= h(x) - L < epsilon

• Therefore:
|f(x) - L| < epsilon

• Hence:
lim (x -> a) f(x) = L

This proof is taken from Jonathan Kane, *Writing Proofs in Analysis* (2010).

Translation: English → Serbian
Source: Kim Stanley Robinson – *Down and Out in the Year 2000*

TEXT FOR TRANSLATION

Lee shook his head distastefully at the idea.

“Delmont, you know any weed I can buy? I need a finger baggie for twenty.”

“Not so easy to do, Robbie.” Delmont hemmed and hawed and they dickered for a while, then he sent Lee over to Jim Johnson, who made the sale under a cheery exchange of the day’s news, over by the chess tables.

After that Lee bought a pack of cigarettes in a liquor store and went up to the little triangular park between 17th, S, and New Hampshire, where no police or strangers ever came. They called it Fish Park for the incongruous cement whale sitting by one of the trash cans.

He sat down on the long broken bench, among his acquaintances who were hanging out there, and fended them off while he carefully emptied the Marlboros, cut some tobacco into the weed, and refilled the cigarette papers with the new mix. With their ends twisted he had a dozen more joints.

They smoked one and he sold two more for a dollar each before he got out of the park.

But he was still anxious, and since it was the hottest part of the day and few people were about, he decided to visit his plants. He knew it would be at least a week till harvest, but he wanted to see them. Anyway, it was about watering day.

East between 16th and 15th he hit no-man’s-land. The mixed neighborhood of fortress apartments and burned-out hulks gave way to a block or two of entirely abandoned buildings.

Here the police had been at work, and looters had finished the job. The buildings were battered and burnt out, their ground floors blasted wide open, some of them collapsed entirely into heaps of rubble.

No one walked the broken sidewalk; sirens a few blocks off and the distant hum of traffic were the only signs that the whole city wasn’t just like this.

Little jumps in the corner of his eye were no more than that; nothing there when he looked directly.

The first time, Lee had found walking down the abandoned street nerve-racking; now he was reassured by the silence, the stillness, the no-man's-land smell of torn asphalt and wet charcoal, the wavering streetscape empty under a sour-milk sky.

INSTRUCTIONS FOR STUDENTS

- Translate the text into Serbian.
- Maintain narrative tone and informal spoken dialogue where appropriate.
- Preserve names and street references as given.
- Pay attention to colloquial expressions and urban slang.
- Do not omit any part of the text.
- Ensure consistency in tense and narrative voice.