Feedback comment:
Under OS/Role and purpose of the operating system ... CS Core Topics

Add topic example to Overarching concern of security/protection: Neglecting to consider security at every layer creates an opportunity to inappropriately access resources.

⇒Vulnerabilities in system firmware can provide attack vectors that bypass the operating system entirely

How incorporated:
Added

Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Role and purpose of the operating system ... CS Core Topics

Add topic example to Overarching concern of security/protection: Neglecting to consider security at every layer creates an opportunity to inappropriately access resources.

⇒Improper isolation of virtual machine memory, computing, and hardware can expose the host system to attacks from guest systems

How incorporated:
Added

Why not incorporated:

Date considered: 7/16/23
The operating system may need to mitigate exploitation of hardware and firmware vulnerabilities, leading to potential performance reductions (e.g. Spectre and Meltdown mitigations)

How incorporated:
Added

Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Principles of operating systems ... CS Core Topics

Add topic
Performance costs of context switches and associated cache flushes when performing process switches in Spectre-mitigated environments

How incorporated:
Added

Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Concurrency ... CS Core

Add topic
Multiprocess concurrency vs. multithreading

How incorporated:
Added

Why not incorporated:

Date considered: 7/16/23
Feedback comment:
Under OS/Concurrency ... CS Core

**Update topic wording**
Old - Race conditions, critical sections (role of interrupts if needed)
New - Race conditions, critical regions (role of interrupts if needed)

**Additional comment** - The historical term for concurrent code with potential race conditions (Tony Hoare, Per Brinch Hansen)

**How incorporated:** Made suggested update

Why not incorporated:

**Date considered:** 7/16/2023

Feedback comment:
Update OS/Scheduling ... KA Core Topics

**Add topic**
*Fairness and starvation*

**How incorporated:**
Added

**Why not incorporated:**

**Date considered:** 7/16/23

Feedback comment:
Under Process model - Illustrative Learning Outcomes...KA Core

**Remove outcome(s)**
Create executable using compilers and linkers from source code, shared libraries and object code
Evaluate a software artifact and problem to determine appropriate use dynamic vs static shared libraries
Additional comment: I think these two items fall squarely into userspace issues as opposed to OS. Also, the linking and loading process is dependent on the programming language used for development. Some languages (e.g. Go) insist on static linking. Others that are popular today (e.g. Rust) want all their modules written in one language. The C-based model is still a good one, but it probably belongs in a different course.

How incorporated: Made suggested update

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:
Under OS/Memory Management .. KA Core

Update topic wording
Old - Memory Caching, and cache coherence
New - Memory Caching, and cache coherence, and the effect of flushing the cache to avoid speculative execution vulnerabilities

How incorporated: Made suggested update

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:
Under OS/Protection and Safety ... KA Core Topics

Add topic example to Security methods and devices
⇒ x86_64 rings -1 and -2 (hypervisor and ME/PSP)

How incorporated:
Added

Why not incorporated:
Feedback comment:
Under OS Virtualization ... KA core
Add topic
⇒ Popek and Goldberg requirements for recursively virtualizable systems
How incorporated:
Added
Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Virtualization ... KA Core Topics
Add topic Example to Containers
⇒ Emphasizing that containers are NOT virtual machines, since they do not contain their own operating systems [where operating system is pedantically defined as the kernel]
How incorporated:
Added
Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Virtualization ... Non-core Topics
Update Topic Wording
Old - Hypervisors
New - Hypervisors - hardware virtual machine extensions

rather than examples of:

- Hypervisor monitor w/o a host operating system
- Host OS with kernel support for loading guests, e.g. QEMU KVM

Additional comments: The distinction between Type 1 and Type 2 hypervisors starts to break down when the system is evaluated more closely. In practice, even hypervisors classified as Type 1 (Xen and VMWare come to mind) contain what is effectively a minimal operating system running directly on the metal. Understanding that a hardware-provided hypervisor mode can improve performance by allowing recursive virtualization of otherwise non-virtualizable instruction set architectures (Popek and Goldberg requirements) is more critical than whether or not the bare metal is also running meaningful userspace code.

How incorporated: Made suggested update

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:

Under OS/Virtualization  ... Non-core Topics
Add to illustrative outcomes under Noncore

=Dynamic recompilation as an intermediary between full emulation and use of hardware hypervisor extensions on non-virtualizable ISAs whenever the guest and host system architectures match

How incorporated:

Added

Why not incorporated:

Date considered: 7/16/23

Feedback comment:
Under OS/Social Ethical and Professional...KA Core

Add Topic

⇒Privacy implications of using proprietary operating systems/operating environments, including telemetry, automated scanning of personal data, built-in advertising, and automatic cloud integration

How incorporated:

Added

Why not incorporated:

Date considered: 7/16/23

Feedback comment:

Update opening statement of preamble to focus on the most important OS function. Two examples given are:

1. OSes multiplex untrusted applications onto shared hardware
2. OS manages a computer's resources on behalf of users and their applications.

How incorporated: Made suggested update

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:

Preamble: Update Overarching themes that are reused at many levels in computer systems are well illustrated in operating systems (e.g. polling vs interrupts, caching, flexibility costs overhead, similar scheduling approaches to processes, page replacement, etc.).

to

Overarching themes should focus on how those concepts apply in other areas of CS - trust boundaries, concurrency, persistence, safe extensibility,

How incorporated: Made suggested update
Why not incorporated:

Date considered: 7/16/2023

Feedback comment:
Discussion of mental models in preamble is not well-motivated and may be irrelevant

How incorporated:
Added sentence to contextualize this statement

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:
In terms of core hours, the document is internally inconsistent - it says that the scheduling, the process model, and memory management are core and not KA, but it only lists KA topics and learning outcomes in the main body.

How incorporated:
Fixed the table in Gamma version to indicate KU are KA Core

Why not incorporated:

Date considered: June 25, 2023

Feedback comment:
Under OS/Role and purpose of the operating system ... CS Core Topics

Topic under consideration: Universal operating system functions
Example concepts:
  ● Creation and execution of application specific software

Comment -“creation and execution of application specific software“ as an OS concept?”
How incorporated:
Removed

Why not incorporated:

Date considered: 7/16/2023

Feedback comment:
Under OS/Role and purpose of the operating system ... CS Core Topics

Topic under consideration: Universal operating system functions
Example concepts:
Trade offs between error checking and performance, flexibility and performance, and security and performance"

Comment: Design issues where we "trade off" correctness for performance?

How incorporated:

Why not incorporated:
There are instances where OSs do not check for loops in symbolic links or for deadlocks

Date considered: June 25, 2023

Feedback comment:
Under OS/Role and purpose of the operating system ... CS Core Topics

Topic under consideration: Extended or specialized OS functions?

Comment: These are not core topics.

How incorporated:

Why not incorporated:
The intent here is to introduce the existence of extended or specialized operating systems
Feedback comment:
Under OS/Principles of operating systems... CS Core Topics

Topic under consideration: Operating system software design and approaches such as Monolithic, Layered, Modular, Micro-kernel models and Unikernel

Comment: Why is OS structure central? We have in practice only monolithic OSes, so why is any of this core?

How incorporated:

Why not incorporated:
Monolithic operating systems are not a desirable design structure. Although there may be few “pure” instances of microkernel architecture, many operating systems lean more into microkernel structure than monolithic structure.

Date considered: June 25, 2023

Feedback comment:
Under OS/Principles of operating systems ... CS Core Topics

- Concept of system calls and links to application program interfaces (APIs)
  Example concepts:
  - Many system calls must be invoked to accomplish program application program requests
  - APIs (Win32, Java, Posix, etc) bridge the gap between highly redundant system calls and functions that are most aligned with the requests an application program would make
  - Approaches to syscall ABI (Linux "perma-stable" vs. breaking ABI every release).

Comment: Concepts of system calls are important - how to safely cross a protection boundary is a vitally important topic - but I didn't understand any of the three sub-bullets (Many system calls must be invoked?)

How incorporated:
**Why not incorporated:**

The examples are concepts, priorities or ideas that help to the idea that a topic was intended to communicate. They have been collected from other reviewers to have added to them as a way to illustrate the significance of a topic. A curriculum designer may want to consider these when they teach each topic.

However, they are tagged as examples because the relevance of these examples may change over time. They list algorithms or approaches that will be superseded by new algorithms or approaches. The curriculum scope does not change to leave them out. However, it may help to make the topics more concrete.

In this case, the idea is to compare the difference between a system call (which a student is probably unfamiliar with) and an API (which they should be very familiar with). The topics can be explained without any reference to the examples. The examples may help someone that is unfamiliar with operating systems to understand the motivation for discussing the link between APIs and system calls.

**Date considered:** June 25, 2023

---

**Feedback comment:**

Under OS/Principles of operating systems ... CS Core Topics

Comment - Much better would be to spend your time here on the basics of the OS API - the process concept, interprocess communication, file I/O, network I/O, at the system call abstraction.

**How incorporated:**

**Why not incorporated:**

The system call abstraction is important for the security mechanisms it enables so it is included in the CS Core in the Principles of Operating Systems knowledge unit. The distribution of the topics between CS Core and KA Core accommodates programs that have a dedicated course in operating systems and those that create a higher-level, combined systems course that includes architecture and other systems topics.

Although CS Core is the absolute minimum, it is expected that topics from the KA Core will also be included in the combined systems course.

**Date considered:** June 25, 2023

---
Feedback comment:
Under OS/Principles of operating systems ... CS Core Topics

Illustrative outcome-
Understand the advantages and disadvantages of using interrupt processing in enabling multiprogramming

Comment - As an example, under outcomes - understand the disadvantages of using interrupt processing in enabling multiprocessing? No widely used OS does it any other way, so what are we getting at here?

How incorporated:

Why not incorporated:
This is an example of how low-level functions impact the correct functioning of parallel applications. It also illustrated why parallel applications present additional challenges.

Date considered: 7/16/2023

Feedback comment:
Under OS/Principles of operating systems ... CS Core Topics

Illustrative outcome-
Analyze for potential threats to operating systems and the security features designed to guard against them

Comment - Analyze potential threats to OSes - ok, but the more fundamental question is threats to user data!

How incorporated:
Suggested update: Analyze for potential vectors of attack via the operating systems and the security features designed to guard against them

Why not incorporated:

Date considered: June 25, 2023

Feedback comment:
Under OS/Concurrency … CS Topics/KA Topics

- **CS Core**
  - Thread abstraction relative to concurrency
  - **Race conditions, critical sections** (role of interrupts if needed)
    - Deadlocks and starvation
  - Multiprocessor issues (spin-locks, reentrancy)

- **KA Core Topics**
  - Thread creation, states, structures
  - Thread APIs
  - **Deadlocks and starvation** (necessary conditions/mitigations)
  - Implementing thread safe code (semaphores, mutex locks, cond vars)
  - Race conditions in shared memory

I believe condition variables are essential. Without them, you can build a basic parallel program, but even BSP is impossible, and concurrent execution is impossible. That section also is quite confusing by listing the same topics in both CS Core and KA Core.

**How incorporated:**

**Why not incorporated:**

Topics of race conditions, deadlocks and starvation would be discussed in more depth in KA Core than time would allot in the CS Core. Therefore all CS students would get at least an introduction.

**Date considered:** June 25, 2023

---

**Feedback comment:**

**OS/Protection and Safety (Overlap with Security TBD)**

- **CS Core Topics**
  - Overview of operating system security mechanisms
  - Attacks and antagonism (scheduling, etc)
  - Review of major vulnerabilities in real operating systems
  - Operating systems mitigation strategies such as backups

- **KA Core Topics**
  - Policy/mechanism separation
  - Security methods and devices
    - Example concepts:
      - Rings of protection (history from Multics to virtualized x86)
  - Protection, access control, and authentication
This seems to be envisioned as where we might talk about threats to subverting the OS kernel. The more basic topics are the definition of ACLs and the access control matrix. There is not an overlap with security - we're a critical mechanism for security.

**How incorporated:**

**Why not incorporated:**

The intent is to cover the critical mechanisms that the OS provides such as services that utilize protected mode, virtual memory addressing, etc in the principles of operating systems, memory, processes and concurrency knowledge units.

**Date considered:** June 25, 2023

Feedback comment:

**OS/File Systems: API and Implementation** (Historical significance but may play decreasing role moving forward)

Comment: I don't think there's any reason to think these are only of historical significance, and if so, why are you moving them into the core?

**How incorporated:**

**Why not incorporated:**

Updated outdated comment in introduction that puts file systems topics into the CS Core. They are currently in the KA Core.

**Date considered:** 7/16/2023