1 - THE

**Mainpoints: abstracted processes and hierarchy structure**

- hierarchy

- CPU (process scheduling)

- memory management, page -> segment

- console

- communication to peripheral devices

- user program

main benefits of the system:

- debug & correctness

Pro:

Handle higher complexity

Con:

Low efficiency

Have to take loop responsibility

hard to design

MUTEX +> TENUX -> UNIX -> LINUX

semaphore

P && V

mutual exclusion && synchronization

2 - Nucleus

**Mainpoints: kernel and message buffering**

- that is, tiny kernel, which multiple kinds of OSs are built on

(motivation: existing OS structures inhibit freedom of design)

- [ general & flexible ] —> contrary to some specializations of some companies.

The system structure: nucleus, and user-level

The work of nucleus: communications, scheming, manipulation

internal processes: program execution; external processes: input/output

Message buffering compared to semaphores(shared memory)

Pro:

- less deadlock, high reliability

Con:

- less efficiency

Storage allocation: parent-children

The limit on the number of outstanding messages a process can send: to avoid a process using up all the common pool to lead to a system block.

3 - Hydra

**Main point: capabilities, separate mechanism and policy**

## mechanism and policy

About how(mechanism, like a tool) and what(upper-level decides on when and what to apply that mechanism on)

Benefit: provide flexibility

## capability && ACL

ACL: like more in storage system(the face recognition)

Capability: transferable to other procedures(the key)

## right amplify

Procedure A ——->(call). Procedure B

Procedure B may own more rights than procedure A, however A can have the right of whether it calling B, which means that A takes a trust on B.

Trust model: trust the procedure will do the right thing when calling a procedure.

4 - tenex

**Mainpoints: rich time-sharing system, rich virtual memory and support for backwards compatibility**

## BBN pager

page table.

virtual memory to physical memory.

two goals: 1) simple memory management; 2) easy sharing between processes.

## processor modifications

JSYs instructions -> micro-coded -> help implement the processor instruction set.

Old system calls are trapped to a compatibility lib which is in user space.

## virtual machine

virtual machine in this paper: kernel

5 - protection

[Protection.docx](https://docs.google.com/document/u/1/d/1OQAfz9bvcKkgzf5Hn5BruGJ8sLSs9nQl/edit)

6 - Multics

[multics.docx](https://docs.google.com/document/u/1/d/18mjeKssHjFdUGxrOG8brmicjJQFtyOGg/edit)

7 - UNIX

[unix](https://docs.google.com/document/u/1/d/1MKpA27vkDbKx0sY_Wd14AALTY_OCc4lpU2JLfyFWz1E/edit)

8 - PLAN9

[plan9](https://docs.google.com/document/u/0/d/1_RvUz57euFcY5tNXPxiJFe82Ap2mYVcudMmO7Spabi4/edit)

9 - Medusa

[medusa](https://docs.google.com/document/u/1/d/1mIzFKW1oV7qrcuVwrXlf3nWrRIR5MNl2JH2JudcqtS0/edit)

10 - Pilot

[pilot](https://docs.google.com/document/u/1/d/1GNjer1raqgAmWjGOPOXEMJrZJxM8JkBkpbOtRmhVTAQ/edit)

Topics

1 - The term of “objects”

In HYDRA:

three object types: procedure, LNS, and process.

In Protection:

those things which need to be protected

In Multics:

four kinds of objects: segments; Message queues (experimental implementation); directories (called catalogs in some systems); removable media descriptors (not yet implemented).

In PLAN9:

2 - Protection machinism

In Multics:

combination of ACL and capabilitites.

ACL: owner, project, compartment

Comparison between two kinds of mechanisms:

ACL:

* easy to set up and understand
* slow to check

Capabilities:

* hard to revoke
* efficient to check and transfer