The Mach System

lecture-27

Some arguments

Microkernel:

- Is a layered – structure a partial microkernel idea???
- Not so:
- A microkernel is physically divided into separate modules. It may consist of 1 or more layers – but only logically.
- A layered kernel is physically divided into layers, but logically – it might consist of one or more modules.
- A microkernel may be (and often is) logically single layered because many layered kernel is again a ??

Some Issues

- How do you deal with hardware in UNIX?
  - Operating systems provide interfaces and management of hardware resources
  - E.g., interrupts and I/O devices.
- A microkernel seems to optimize operating system design
  - So, should make operating system (lower level) easier to modify
  - Layered approach– so, seems good in principle
- Is the UNIX (or other “user application” O/S) really a User Application?
  - Are users going to write additional operating systems?

Solution – Microkernel.

- Microkernel designs put a lot of OS services in separate processes to build modular operating systems.
  - Kernel's functionality is reduced and put into user servers.
- This architecture is actually a client-server model.
  - Clients call other OS services through microkernel.
- The central processes that provide the process management, file system etc are frequently called the servers.
- Microkernels are often also highly multithreaded.
  - Each thread has a different service to perform.
  - What happens with speed of IPC?

What is Mach?

- Mach
  - Transparent multiprocessing – Avoiding issues in BSD.
  - Protected message passing – Better than Unix message messaging.
  - “extensible” Microkernel
  - Multiple levels of operating system
    - Other O/S’s implemented as “applications”
  - Basis for NeXT O/S, Mac X O/S, OSF/1

Design Goals of Mach

- Full support for multiprocessing.
- Exploit other features of modern hardware architectures that were emerging at that time.
- Supports transparent distributed operation.
- Reduce the number of features in the kernel, and therefore make it less complex, giving the programmer a very small number of abstractions to work with.
- The abstractions are just general enough to allow several operating systems to be implemented on top of Mach.
- Full compatibility with UNIX BSD.
- Address the shortcomings of previous systems such as Accent.

Approach:

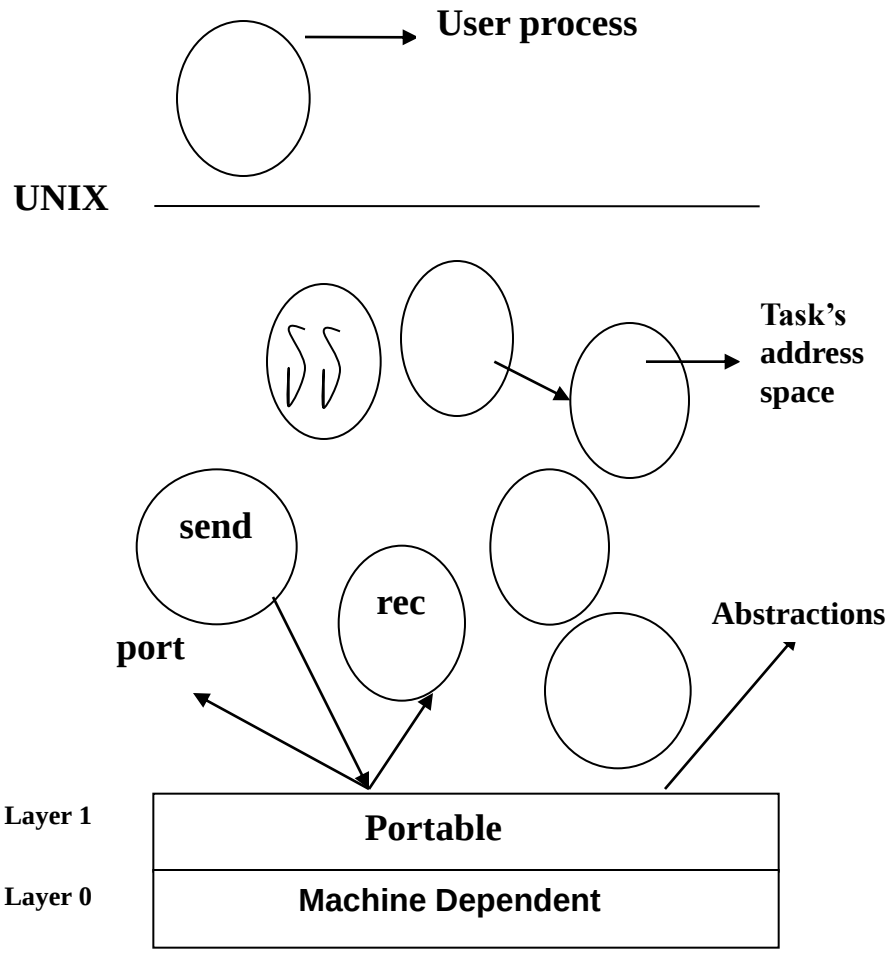
- a small, extensible system kernel which provides scheduling, virtual memory and interprocess communications
- and several, possibly parallel, operating system support environments which provide the following two items:

1) distributed file access and remote execution

2) emulation for established operating system environments such as UNIX.

Overall Mach

- IPC – RPC messages.
  - Send and receive.
- When the message queue is full the senders block; when it is empty, the receivers block.
- Indirect communication.
- Heavy weight context switching.
- Speed is compromised ; but protection is ensured.



Mach's abstractions

- A task is an execution environment and is the basic unit of resource allocation.
  - Includes a paged virtual address space (potentially sparse)
  - protected access to system resources (such as processors, port capabilities and virtual memory).
- A thread is the basic unit of execution. A thread executes in the context of a single task. A UNIX Process = Task + thread.
- A port is a simplex communication channel -- implemented as a message queue managed and protected by the kernel.
  - Basic object reference mechanism in MACH.
  - Ports are used to refer to objects; operations on objects are requested by sending messages to the ports which represent them.

Contd..

- A port set is a group of ports, implemented as a queue combining the message queues of the constituent ports.
  - A thread may use a port set to receive a message sent to any of several ports.
- A message is a typed collection of data objects used in communication between threads.
  - Can by of any size and may contain inline data, pointers to data, and capabilities for ports.
- A memory object is a secondary storage object that is mapped into a task's virtual memory.
  - memory object is treated like any other object.

Differences

- Ports are a protected entity that can only be addressed by the Mach microkernel,
- Port rights are attached to a given task and describe the operations that they can provide on a port,
- Port names are the identifiers that tasks must use to request some operations on this ports.
- This looks similar to - Files, files access rights and file descriptors in a traditional UNIX system.

ASSIGNMENT

- Q: What is MACH?