MACH DISTRIBUTED OPERATING SYSTEM

Start

GROUP MEMBERS

1. Hammad Ansari (2018450002)
2. Maulika Arekar(2018450003)
3. Diksha Bandagale (2018450004)
4. Riya Barve (2018450005)
5. Siddhesh Chaughule (2018450008)
6. Anuradha Date (2018450011)
7. Nilam Khatal (2018450026)
8. Pradnyot Morwekar (2018450033)
9. Pratik Pawar (2018450038)
10. Mantasha Shaikh [2018450047]
11. Prashika Walke [2018450059]
12. Chaitanya Yadav(2018450060)

CONTENT

1. Features and Goals
2. System Model & Kernel Model
3. System Architecture
4. IPC mechanism
5. Use of RPC
6. Objects and their Component.
7. Load Balancing
8. Process Management
9. Memory management/Device Management
10. Can this OS be used for implementing cloud -Cloud Example
11. Deployment model
12. Delivery model

WHOA!

Let's get started

Go!

MACH IS A MICROKERNEL-BASED OPERATING SYSTEM

DEVELOPED AT CARNEGIE-MELLON UNIVERSITY

BY RICHARD RASHID

WITH THE SUPPORT OF DARPA

FEATURES AND GOALS

Go!

HAMMAD ANSARI

Open-System Architecture

Compatibility with BSD UNIX

Network Transparency

Flexible Memory Management

Flexible IPC

High Performance

Simple Programmer Interface

OPEN-SYSTEM ARCHITECTURE

- Main Design Goal: Providing a base for building new operating systems and emulating the existing ones

COMPATIBILITY WITH BSD UNIX

- Full compatibility with UN1X was an important goal of MACH. So, it could receive global acceptance.

NETWORK TRANSPARENCY

- A higher level network wide naming system is used in Mach.

FLEXIBILITY MEMORY MANAGEMENT

- Mach provides an elaborate virtual-memory system that is implemented in terms of fixed-size pages.

FLEXIBLE IPC

- Mach uses a message-based IPC that is based on ports, which are kernel objects that hold messages.

HIGH PERFORMANCE

- MACH has performance penalty because of context switching between serving process and the microkernel. After all of these it still performs better.

SIMPLE PROGRAMMER INTERFACE

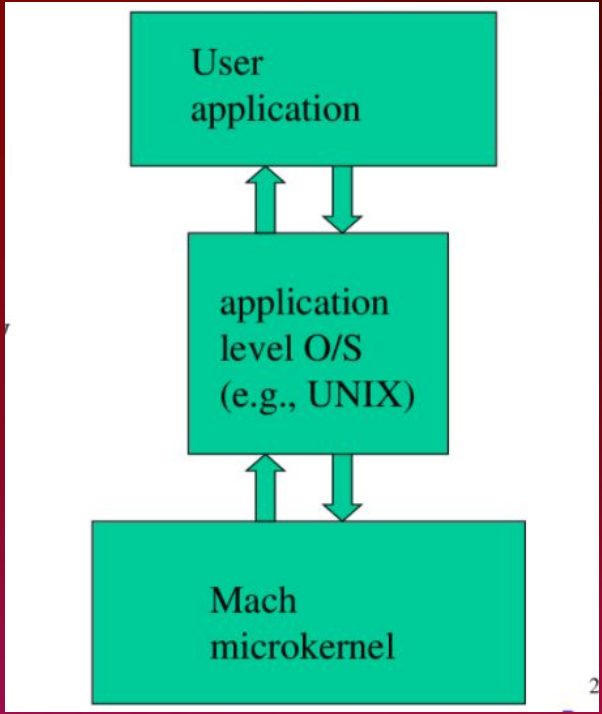
- Mach provides simple interface for programmers which is Mach Interface Generator (MIG)
- Mach Interface Generator allows the application programmers to use a service by simply making a procedure call, rather than making a system call or writing code for sending and receiving messages

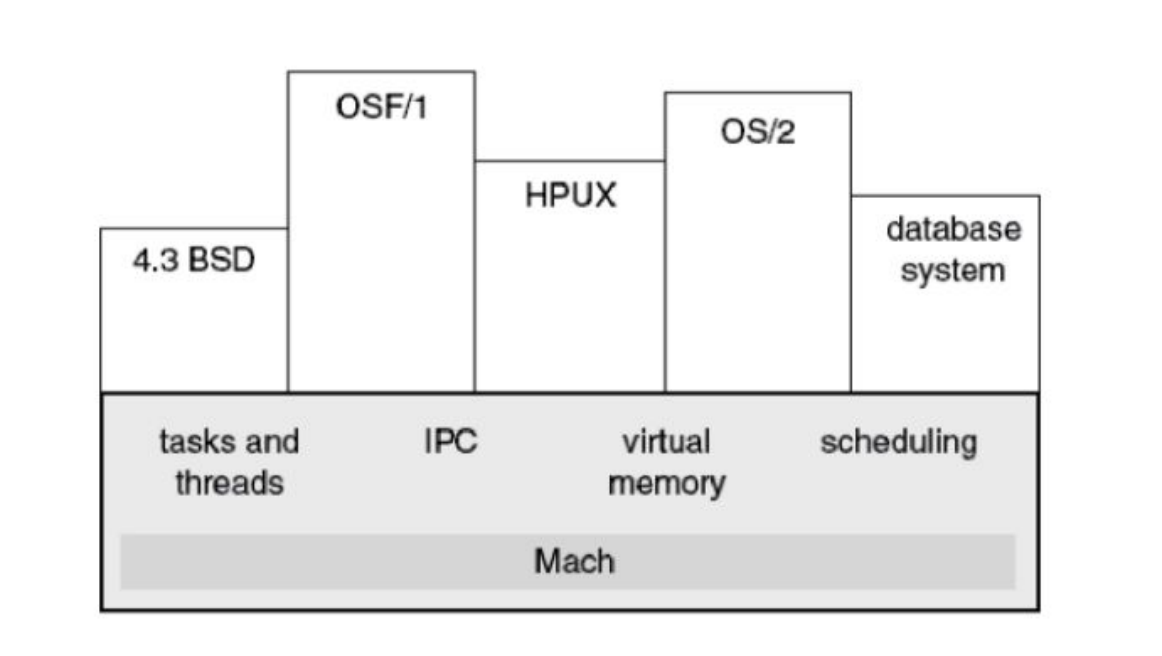
SYSTEM MODEL & KERNEL MODEL

Go!

MAULIKA AREKAR

SYSTEM MODEL OF MACH





FIVE MAIN ABSTRACTION IN MACH KERNEL

- → TASKS
- → PROCESSES
- → THREADS
- → PORTS
- → MESSAGES

WHAT IS MACH MICROKERNEL?

- The Mach kernel is an example of modular layered approach to operating system design.
- It provides flexible execution environment
- Kernel provides only fundamental services
- These services are basic but powerful enough to be used on wide range of architectures.

MACH MICROKERNEL FEATURES:

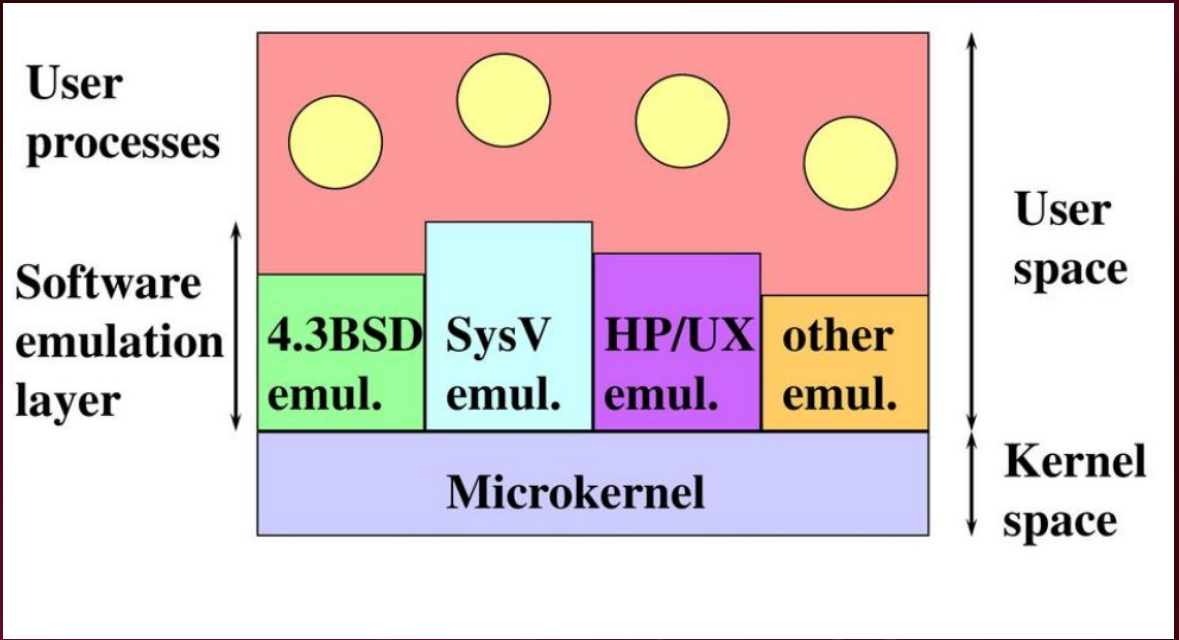
- Task and thread management
- Interprocess communication
- Memory object management
- System call redirection
- Device support

SYSTEM ARCHITECTURE

Go!

PRADNYOT MORWEKAR

What is Mach microkernel system architecture



Key Aspects of Mach System Architecture

Tailorability

Portability

Network accessibility

Multiprocessor support

Security

IPC MECHANISM

Go!

PRATIK PAWAR

IPC MECHANISM

- Inter-process communication (IPC) can be defined as set of techniques used for exchanging data among multiple threads in one or more processes.

- The two components of Mach IPC are ports and messages.

Techniques available for IPC

- Shared Memory
- Messages
- Synchronization through IPC

USES OF RPC

Go!

RIYA SACHIN BARVE

RPC?

- A RPC is an interprocess communication technique that is used for client-server based applications.
- Also known as a subroutine call or function call.

USES

Exception-Handling

Remote Access

NetMsgServer

Programmer Interface

OBJECTS AND THEIR COMPONENTS

Go!

CHAITANYA YADAV

OBJECTS

- The data and the operations that manipulate the data are encapsulated into an abstract object.
- MACH has a C-based object-oriented programming package. (Which is integrated with the inter-process communication facility)
- This provides the means to dynamically specify classes and objects.
- Multiple inheritance, automatic object locking, garbage collection of objects, class/superclass hierarchy are all enabled due to this package.
- Only the operations of the object are able to act on the entities defined in it.
- The details of how these operations are implemented are hidden, as are the internal data structures.

COMMUNICATION COMPONENTS

- **Task** is an execution environment that provides the basic unit of resource allocation.
- **Thread** is the basic unit of execution and must run in the context of a task (which provides the address space).
- **Port** is the basic object-reference mechanism in Mach and is implemented as a kernel-protected communication channel.
  (Task must have a port right to send a message to a port.)
- **port set** is a group of ports sharing a common message queue.
  (thread can receive messages for a port set and thus service multiple ports.)

- **Message** is the basic method of communication between threads in Mach. It is a typed collection of data objects.
- **Memory object** is a source of memory. Tasks can access it by mapping portions of an object.
- memory object can be any object for which memory-mapped access makes some sense.

LOAD BALANCING

Go!

NILAM KHATTAL

LOAD BALANCING

- Distribution of load to the processing elements
- It' a strategy make every processor equally busy & finish work at same time
- Load balancing operation

Location Rule Distribution Rule Selection Rule

- Improves performance of each node
- Reduces job idle time
- Quality depends on no of steps & extent of load

TYPES OF LOAD BALANCING

1. Static Load Balancing
    a. Processes assigned at compile time.
    b. No change or reassignment
    c. Probabilistic algorithm
2. Dynamic Load algorithm
    a. Job/reassigned done at runtime
    b. Communication over heads

PROCESS MANAGEMENT

Go!

DIKSHA BANDAGALE
AND
SIDDHESH CHAUGHULE

BASIC ABSTRACTION

The two basic abstractions used for process management in Mach are task(Process) and thread.

Process and Thread States

At any instance of time, a thread may be in one of the following states:

- Running.
- Suspended.

OPERATIONS ON PROCESSES AND THREADS

- Each process owns a process port and each thread owns a thread port.
- Process management primitives provided in Mach include those for creating a process, killing a process, suspending or resuming a process
- In Mach, threads are managed by the kernel..

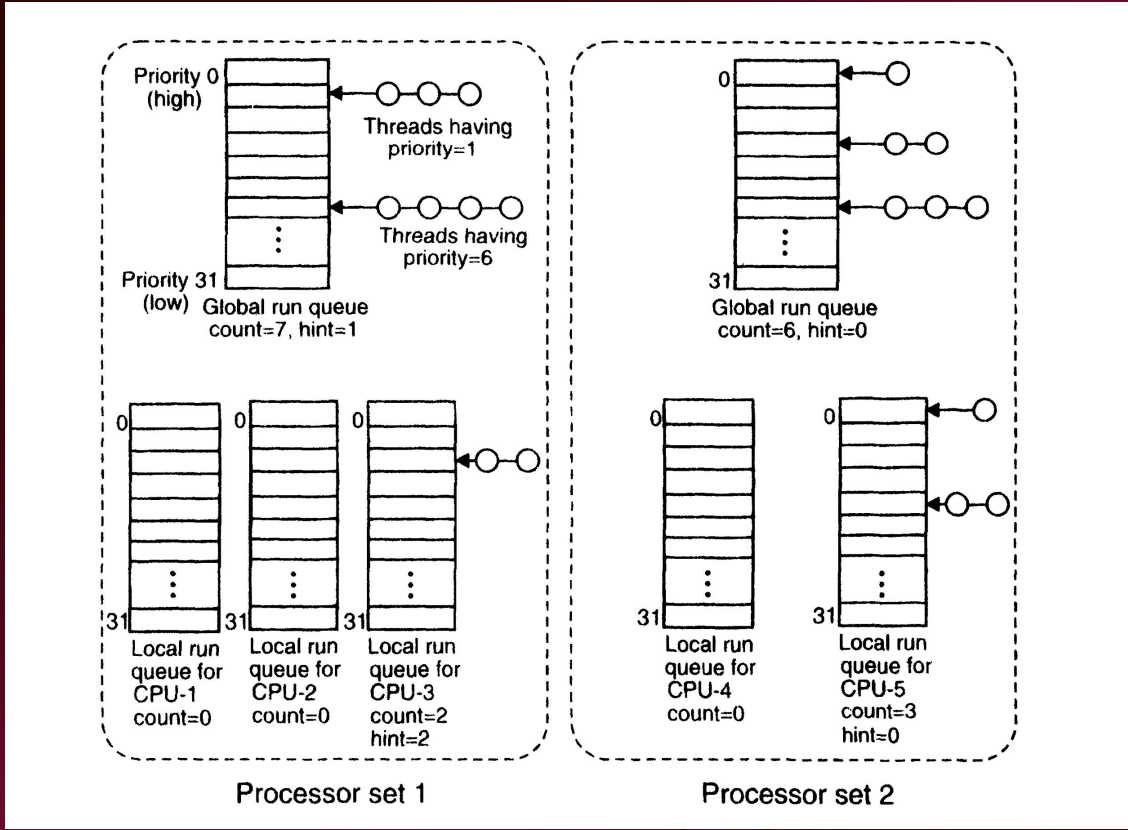
THE C-THREADS PACKAGE

The C-threads package allows the programmers to use the kernel thread primitives in a simple and convenient manner.

The routines for direct thread manipulation are given below:

- Cthreadfork
- Cthreadexit
- Cthreadjoin
- Cthread_detach
- cthreadyield

THREADS SCHEDULING



SCHEDULING ALGORITHM

- When a thread blocks, exits or uses up its quantum, CPU looks on local run queue to see if there are any active threads.
- If it is nonzero, search for highest priority thread.
- If empty, same algorithm applied to global run queue.
- If no thread to run, a special idle thread is run until some become ready

SCHEDULING ALGORITHM COTD.

- If a runnable thread found, scheduled and run for one quantum.
- After one quantum, check local and global queue for any runnable thread.
- If found, thread switch occur
- If not found, thread run for another one quantum

MEMORY MANAGEMENT

Go!

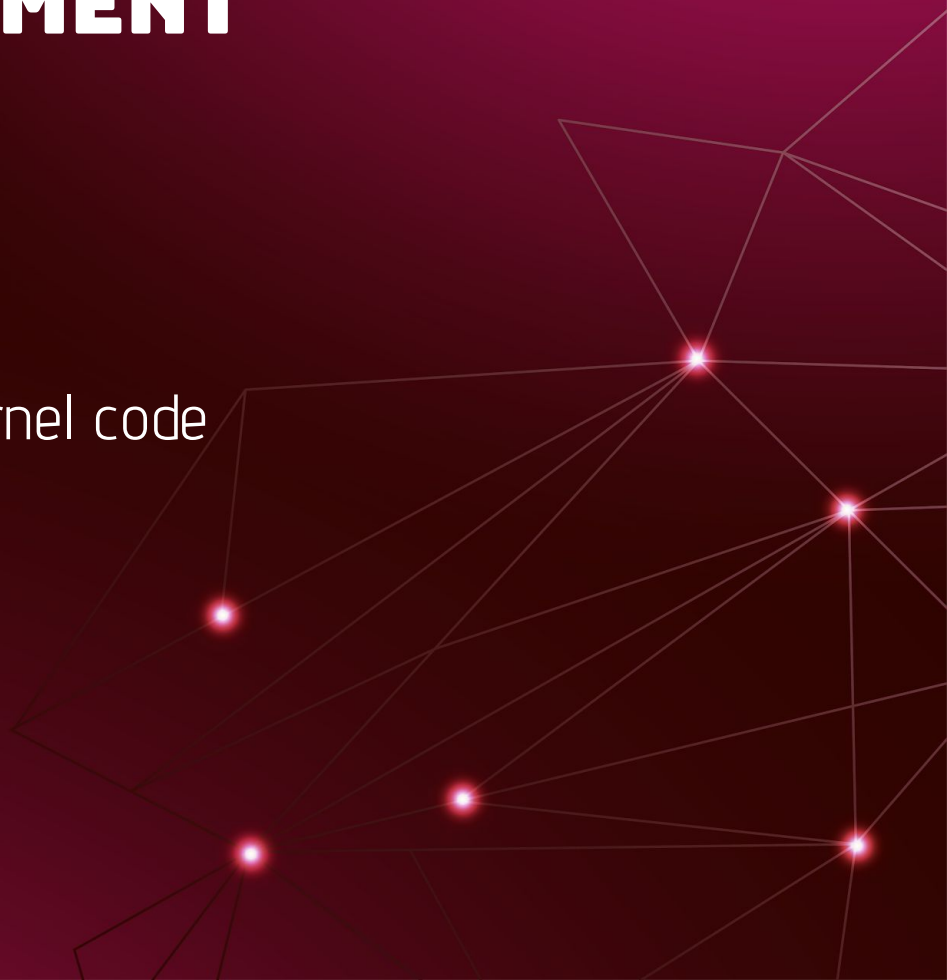
ANURADHA DATE

MEMORY MANAGEMENT

- Memory objects are used to manage secondary storage
- They represent the files ,pipes etc
- User-level Memory Managers
  - Back up the Memory object
  - Responsible for consistency
- System Calls
  - Allocate , deallocate and return information
- Shared Memory
  - Supported only when execution is on processors that share memory
  - Reduces complexity

MEMORY MANAGEMENT

- Is split in 3 parts
  - Pmap module
  - Machine independent kernel code
  - The memory manager



CLOUD EXAMPLE

Go!

MANTASHA SHAIKH

CLOUD EXAMPLE

- Yes, we can use Mach as a cloud operating system.
- In 1989, OSF a consortium of computer vendors, selected Mach.
- OSF has important companies like IBM, H.P.& DEC.
- NeXT workstation also used Mach.
- It is compatible with BSD Unix.
- Although, cloud platforms uses their own flavor of linux.

DEPLOYMENT MODEL

Go!

DEPLOYMENT MODEL

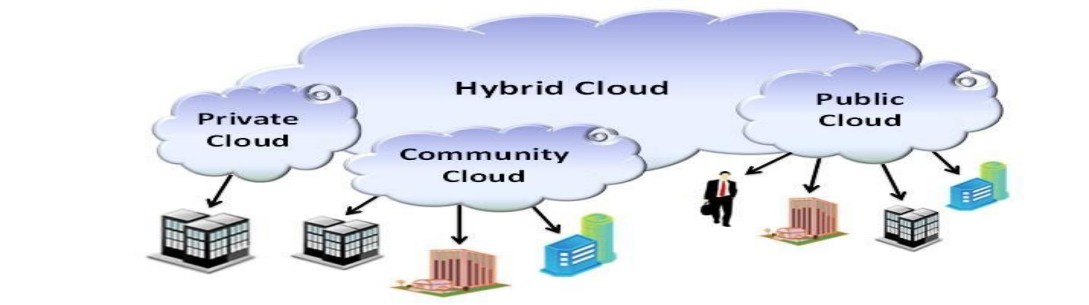
Public Cloud

- Services are rendered by third-party providers over a network open for public use.
- Provide an access control mechanism for their users.
- No responsibilities over the management of the cloud – you only use it to store your data, and pay as you go.

Private Cloud

- A private cloud refers to a cloud deployment model operated exclusively for a single organization.
- Mach can be deployed using private cloud deployment model.

DEPLOYMENT MODEL



Hybrid Cloud

- Hybrid Cloud is a type of Cloud Computing which is integrated into more than two cloud servers.
- For example, in case of high-volume, less sensible data that doesn’t require strong security layers.

DELIVERY MODEL

Go!

PRASHIKA WALKE

DELIVERY MODEL

Infrastructure-as-a-service(IaaS)

- The IaaS delivery model represents a self-contained IT environment that can be accessed and managed via cloud service-based interfaces.
- This environment can include hardware, network, connectivity, operating systems, and other "raw" IT resources.
- The general purpose of an IaaS environment is to provide cloud consumers with a high level of control and responsibility over its configuration and utilization.
- Mach can be delivered using IaaS delivery model.

Platform-as-a-service(PaaS)

- The PaaS delivery model represents a predefined "ready-to-use" environment typically comprised of already deployed and configured IT resources.
- Specifically, Paas relies on (and it's primarily defined by) the usage of a ready-made environment.
- That establishes a set of pre-packaged products and tools used to support the entire delivery lifecycle of custom applications.

DELIVERY MODEL

Software-as-a-service(SaaS)

- The SaaS delivery model is typically used to make a reusable cloud service widely available to range of cloud consumers.
- An entire marketplace exits around SaaS products that can be leased and used for different purposes and via different terms.
- A cloud consumer is generally granted very limited administrative control over a SaaS implementation.
- It is most often provisional by the cloud provider, but it can be legally owned by whichever entity assumes the cloud service owner role.

REFERENCES

1) PK Sinha - Distributed Operating System
2) OS Book - https://www.os-book.com/OS9/appendices-dir/b.pdf

THANKS!

Any Questions?