Project Proposal

Course Title	:	Operating system.
Course Code	:	CSE - 309
Project Covering	:	Tightly Coupled Distributed Memory Architectures.

Submitted to

Sarnali Basak
Lecturer, CSE, JU.

Submitted by

1. Romena Afroz	- 03
2. Md. All Rabbi	- 25
3. Junayed Bhuiyan	- 32

Table of Contents

1. Project Description

The aim is to design a distributed OS with Tightly coupled memory architecture which provides the essential services and functionality required of an OS but adds attributes and particular configurations to allow it to support additional requirements such as increased scale and availability.

2. Problem Statement

Research and write a term paper on Puma and relatives (Sandia National Laboratory). The Puma operating system targets high-performance applications on tightly coupled distributed memory architectures. It is a descendant of SUNMOS.

3. Problem Analysis

3.1 Distributes OS

A distributed operating system is a software over a collection of independent, networked, communicating, and physically separate computational nodes. Each individual node holds a specific software subset of the global aggregate operating system. Each subset is a composite of two distinct service provisioners. The first is a ubiquitous minimal kernel, or microkernel, that directly controls that node's hardware. Second is a higher-level collection of system management components that coordinate the node's individual and collaborative activities. These components abstract microkernel functions and support user applications

3.2 Tightly coupled memory architecture

While both are very fast accessed memories, cache stores dynamically data/code which has been lately used in order to improve access speed, compared to standard memory connected to the global Avalon matrix. Every time a memory access is required, the processor checks if the required data is already present in the cache or must be newly fetched from memory; in the meantime, old unused cache data is being continuously replaced with new data. Tightly coupled memory is also a fast access memory, since it exploits a dedicated port, but it has static content

4. Multiprocessing Handling

4.1 Processor symmetry

In a multiprocessing system, all CPUs may be equal, or some may be reserved for special purposes. A combination of hardware and operating system software design considerations determine the symmetry (or lack thereof) in a given system. For example, hardware or software considerations may require that only one particular CPU respond to all hardware interrupts, whereas all other work in the system may be distributed equally among CPUs; or execution of kernel-mode code may be restricted to only one particular CPU, whereas user-mode code may be executed in any combination of processors. Multiprocessing systems are often easier to design if such restrictions are imposed, but they tend to be less efficient than systems in which all CPUs are utilized.

Systems that treat all CPUs equally are called symmetric multiprocessing (SMP) systems. In systems where all CPUs are not equal, system resources may be divided in a number of ways, including asymmetric multiprocessing (ASMP), non-uniform memory access (NUMA) multiprocessing, and clustered multiprocessing.

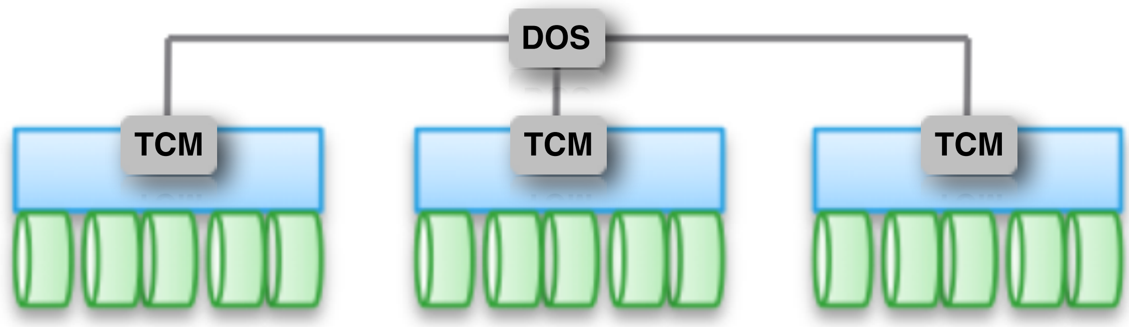
4.2 Instruction and data streams

In multiprocessing, the processors can be used to execute a single sequence of instructions in multiple contexts (single-instruction, multiple-data or SIMD, often used in vector processing), multiple sequences of instructions in a single context (multiple-instruction, single-data or MISD, used for redundancy in fail-safe systems and sometimes applied to describe pipelined processors or hyper-threading), or multiple sequences of instructions in multiple contexts (multiple-instruction, multiple-data or MIMD).

4.3 Processor coupling

Tightly coupled multiprocessor systems contain multiple CPUs that are connected at the bus level. These CPUs may have access to a central shared memory (SMP or UMA), or may participate in a memory hierarchy with both local and shared memory (SM)(NUMA). The IBM p690 Regatta is an example of a high end SMP system. Intel Xeon processors dominated the multiprocessor market for business PCs and were the only major x86 option until the release of AMD's Opteron range of processors in 2004. Both ranges of processors had their own onboard cache but provided access to shared memory; the Xeon processors via a common pipe and the Opteron processors via independent pathways to the system RAM.

5. Procedures and Methods



5.1 System Design

Chip multiprocessors, also known as multi-core computing, involves more than one processor placed on a single chip and can be thought of the most extreme form of tightly coupled multiprocessing. Mainframe systems with multiple processors are often tightly coupled.

5.2 Expected Challenges

Designing a distributed system does not come as easy and straight forward. A number of challenges need to be overcome in order to get the ideal system. The major challenges in distributed systems are listed below

5.2.1. Heterogeneity

The Internet enables users to access services and run applications over a heterogeneous collection of computers and networks. Heterogeneity (that is, variety and difference) applies to all of the following:

Hardware devices: computers, tablets, mobile phones, embedded devices, etc.
Operating System: Ms Windows, Linux, Mac, Unix, etc.
Network: Local network, the Internet, wireless network, satellite links, etc.
Programming languages: Java, C/C++, Python, PHP, etc.
Different roles of software developers, designers, system managers
Different programming languages use different representations for characters and data structures such as arrays and records. These differences must be addressed if programs written in different languages are to be able to communicate with one another.

5.2.2. Transparency

Transparency is defined as the concealment from the user and the application programmer of the separation of components in a distributed system, Some terms of transparency in distributed systems are:

Access	Hide differences in data representation and how a resource is accessed
Location	Hide where a resource is located
Migration	Hide that a resource may move to another location
Relocation	Hide that a resource may be moved to another location while in use
Replication	Hide that a resource may be copied in several places
Concurrency	Hide that a resource may be shared by several competitive users
Failure	Hide the failure and recovery of a resource
Persistence	Hide whether a (software) resource is in memory or a disk

5.2.3. Concurrency

Both services and applications provide resources that can be shared by clients in a distributed system. There is therefore a possibility that several clients will attempt to access a shared resource at the same time. For example, a data structure that records bids for an auction may be accessed very frequently when it gets close to the deadline time. For an object to be safe in a concurrent environment, its operations must be synchronized in such a way that its data remains consistent. This can be achieved by standard techniques such as semaphores, which are used in most operating systems.

5.2.4. Security

Many of the information resources that are made available and maintained in distributed systems have a high intrinsic value to their users. Their security is therefore of considerable importance. Security for information resources has three components:

5.3. Failure Handling

Computer systems sometimes fail. When faults occur in hardware or software, programs may produce incorrect results or may stop before they have completed the intended computation. The handling of failures is particularly difficult.

6. Project Plan

6.1 Risks

There are special factors of risk in distributed systems. Existing distributed systems offer significant opportunities for the introduction of insecure or malicious software. They also permit hacking and browsing. Even those distributed systems which are intended to support a low or medium risk area of business still have to be careful today not to leave themselves wholly unprotected.

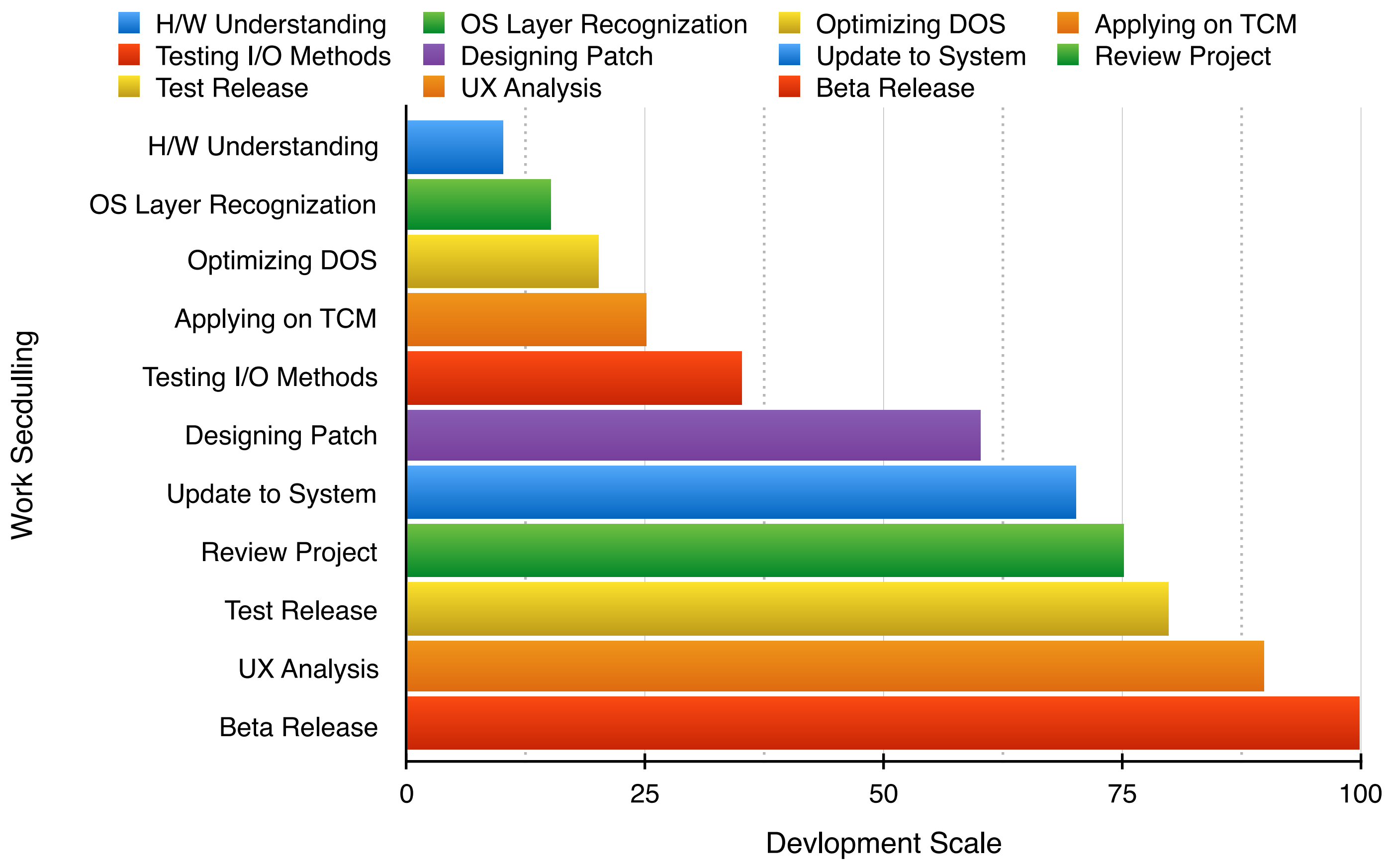
6.2 Resources Required

The main requirements for the project will be computing devices that are running the latest operating systems and Linux is better to use. In terms of development nit much is required besides a set of Computers with the latest IDE, which can be installed on the devices. During the User testing and evaluation stages of the project money for incentives would also be needed.

6.3 Milestones

Project proposal making	19 May
Presentation of project proposals	22 May
Initial project sample presence	30 June

6.4 Gantt Chart



7. References

- **ACM Digital Library** : http://dl.acm.org/citation.cfm?id=6074&dl=ACM&coll=DL&CFID=765354488&CFTOKEN=62978686
- **Oxford Reference :** http://www.oxfordreference.com/view/10.1093/oi/authority.20110803095722609
- **Reliability analysis in distributed systems :** http://ieeexplore.ieee.org/document/2173/citations
- **Memory access ordering :** https://community.arm.com/processors/b/blog/posts/memory-access-ordering---an-introduction
- **Wikipedia** : https://en.wikipedia.org/wiki/Distributed_operating_system