# **Desarrollo de Aplicaciones** en Red

José Rafael Rojano Cáceres http://www.uv.mx/rrojano

#### Design aspect in DS

- Transparency
- Flexibility
- Reliability
- Performance
- Scalability

## Transparency

- From the point of view of the user exist just ONE system.
  - Two levels of transparency
    - User.- the result is the important thing, not how it was reach.
    - Program.- it more difficult to hide the details of call system for a programmer
- · In transparency aspect there is kind of levels:

### **Transparency considerations**

Tipo	Significado
Transparencia de	Los usuarios no pueden indicar la localización
localización	de los recursos
Transparencia de	Los recursos se pueden mover a voluntad sin
migración	cambiar sus nombres
Transparencia de réplica	Los usuarios no pueden indicar el número de copias existentes
Transparencia de	Varios usuarios pueden compartir recursos de
concurrencia	manera automática
Transparencia de	Las actividades pueden ocurrir en paralelo sin e
paralelismo	conocimiento de los usuarios

- 1, it is not allow to specifies the resource Ex. \\UNC 2, It is related to how directories do not change their names for accessing them just by moving from one location to another Ex. NFS
- 3, The aspect of where o when are replicates files is not inform to the user
- 4, It is related to how to manage the access to share resources, Ex. exclusively
- 5. The hardest one, because it considers that hardware and software knows how to use specific resource in function to one problem

### Flexibility

- There are two points of view respect this aspect:
  - . Monolithic Kernel
  - Microkernel
- Monolithic kernel is the current centralized OS approach, it includes networking capabilities
- Microkernel provides 4 services:
- 1. Interprocess communication mechanism
- 2. Memory administration
- 3. Scheduling and administration of low level
- 4. I/O of low level

# **Flexibility considerations**

- Monolithic kernel
  - Better performance because there is not message passing between servers.
- Microkernel
  - New services can be implemented to user level (it does not need new kernels)
  - There are well defined interfaces for services (message that server understand)
  - It can be deployed in different systems (unix and dos)

#### Reliability

- This aspect implies a system that always work!
- Two consideration applies

#### - Availability

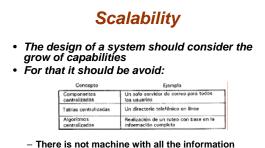
- Time in which server is available
- Redundancy
  - Data coherence because of redundancy

#### – Fail-over

· Transparency for recovering to the users

### Performance

- Here, communication aspect it's very important! (Message passing)
  - Grain aspect, how big should be the process
    - Parallelism of fine grain (small computation) • Parallelism of coarse grain (big computation)
- Fail-over is an aspect that should be considered in performance because of the number of messages



- complete
- Decisions are taken just with local information - The fail of a node does not crash the system
- There is not a global clock



### **Cluster Design**

- Master node
  - better characteristics that the other nodes
- Operating System
  - Support for kernel 2.4.x
- · File system - MFS, maintains cache consistency
- Software for administration
  - User Lan tools, manage process
  - OpenMosix view, monitor of process

## **Deployment of cluster**

- Slave nodes
  - Install Openmosix kernel
  - User Lan tools
  - It can be configured to start at level 3 (without graphics mode)

# **Deployment of cluster**

- Installation of the
   #rpm-hiv openmosix-kernel-2.4.26-openmosix1.i386.rpm
   #rpm-hiv OpenMosix-kernel-source-2.4.24-OpenMosix2.i386.rpm
- Compile the source
   # cd /usr/src/linux-2.4.24-OpenMosix2
   # make mrproper
   # make xconfig
   # make comf
   # make rpm
- Install the new kernel # rpm -hiv /usr/src/packages/RPMS/i586/kernel-2.4.24OpenMosix2-1.i586.rpm # mkinitrd



# **Deployment of cluster**

#### Install the monitor

# cd /root/srcCluster/toolsOM # rpm -ivh openmosixview-1.5-redhat90.i386.rpm

# **Configure Openmosix**

- Configure the node that form part of the cluster
  - # vi /etc/openmosix.map 1 192.168.1.1 1 2 192.168.1.2 1 3 192.168.1.3 1 4 192.168.1.4 1

### **Proving the cluster**

· vi prueba.sh

#!/bin/bash #

- # DESCRIPCION: Script que lanza 10 tareas simultaneas. # Cada tarea itera en un ciclo doble de 10000 unidades. # El objetivo es probar que el cluster esta migrando las # tareas a los demás nodos
- #-----

for x in 1 2 3 4 5 6 7 8 9 10 do

awk 'BEGIN {for(i=0;i<10000;i++)for(j=0;j<10000;j++);}' &

• To see the migration use the command mosmon # mosmon

### References

 Sistemas operativos distribuidos, Tanenbaumm, 1996
 Implementación de un Cluster openmosix para cómputo científico en el instituto de ingeniería, José Castilo Castillo, 2006