UNIVERSIDAD NACIONAL JOSÉ MARÍA ARGUEDAS

FACULTAD DE INGENIERÍA

ESCUELA PROFESIONAL DE INGENIERÍA DE SISTEMAS

INFORME FINAL DE TESIS

INFLUENCIA DE UN CLÚSTER DE COMPUTADORAS DE ALTO RENDIMIENTO EN EL TIEMPO DE RENDERIZACIÓN DE MODELOS 3D FOTOREALISTAS, UNIVERSIDAD NACIONAL JOSÉ MARÍA ARGUEDAS 2013.

PRESENTADO POR : **BACH. EN INGENIERÍA DE SISTEMAS**
CARLOS YINMEL CASTRO BULEJE

PARA OPTAR EL TÍTULO PROFESIONAL DE INGENIERO DE SISTEMAS

ASESOR : **ING. ENRIQUE EDGARDO CONDOR TINOCO**

ANDAHUAYLAS – APURÍMAC

PERÚ

MAYO, 2014

DEDICATORIA

A mi querida abuela María quien supo compartir sus experiencias y enseñarme lo maravilloso de la vida, quien con aliento constante e incesante hizo posible culminar mis estudios de ingeniería.

AGRADECIMIENTOS

- A Dios por haberme dado todo en la vida y permitirme ser parte de este mundo maravilloso.

- A mis padres quienes me apoyaron constante e incesantemente.

- A mi asesor de tesis quien guio el desarrollo del presente trabajo de investigación así como contribuir en mi formación profesional y personal.

- A mi enamorada quien apoyo e inspiró el desarrollo de la presente investigación.

- A mis docentes universitarios quienes contribuyeron en mi formación profesional, personal y abrirme las puertas al maravilloso mundo del conocimiento.

- A mis inolvidables compañeros de clase, hoy algunos colegas de trabajo con quienes compartí maravillosos momentos.

- A la Universidad nacional José María Arguedas mi casa de estudios superior que me abrió las puertas para realizar la presente investigación.

RESUMEN

El presente trabajo de investigación nació a partir de la necesidad de indagar sobre la computación en clúster, un conjunto de computadoras que trabajan unidos como si fuesen un solo equipo para resolver problemas cuya complejidad computacional es muy elevada, los clúster de computadoras permiten dar solución a diversos problemas como son: el procesamiento intensivo de datos, la aceleración de cálculos y la tolerancia a fallos, la computación en clúster está enfocada a solucionar todos estos problemas orientados a ramas mucho más específicas como son: el análisis y diagnóstico de enfermedades a través del procesamiento de tomografías médicas, simulación de diversos sistemas que permiten predecir y realizar estudios futuros, aceleración de búsquedas en internet, renderización de modelos tridimensionales. La presente investigación aborda la aplicación de la computación en clúster a la renderización de escenas en el ámbito fotográfico artificial tridimensional, actualmente para la renderización se utiliza diversas técnicas, una de ellas es conocida como el trazado de rayos el cuales está caracterizado por generar imágenes que poseen un gran realismo, esto debido a la gran cantidad de cálculos que utiliza este procedimiento para simular un ambiente físico casi perfecto, si bien es cierto la aplicación de un clúster de computadoras obviamente mejora los tiempos de renderizado, el estudio que se realiza en la presente investigación consiste específicamente en determinar la influencia de un clúster de computadoras de alto rendimiento en el tiempo de renderización de modelos tridimensionales fotorealistas utilizando la técnica de trazado de rayos, determinando la incidencia de mejora con respecto a la reducción del tiempo de renderizado, para ello el desarrollo de la investigación abarca tres capítulos: en el primer capítulo se aborda el planteamiento metodológico, en este capítulo se define el problema, los objetivos, las hipótesis, las variables, diseño de investigación, población, muestra y los instrumentos de recolección y medición de datos, seguidamente en el segundo capítulo se realiza el desarrollo del marco teórico, el marco teórico comprende los antecedentes de la investigación, el procesamiento en paralelo, los sistemas distribuidos, los clúster de computadoras así como el render y la postproducción. En el tercer capítulo se desarrolla la aplicación de la metodología, este capítulo aborda la instalación y configuración de las soluciones obtenidas de la exploración tecnológica, se desarrolla también la fase de pruebas iniciales y la fase de pruebas finales de la investigación. En el cuarto capítulo se desarrollan los resultados, este capítulo comprende las pruebas estadísticas, las pruebas de hipótesis así como la discusión de los resultados obtenidos. Finalmente se expone las conclusiones y recomendaciones de la investigación.

Palabras clave: Trazado de rayos con clúster

ABSTRACT

This research work was born from the need to investigate the computing cluster, a set of computers that work together as if they were a single computer to solve problems whose computational complexity is very high, the cluster computers allow to solve various problems such as: intensive data processing , the acceleration of calculations and fault tolerance , cluster computing is focused on solving all these oriented branches much more specific problems such as: the analysis and diagnosis of disease through processing of medical scans, various simulation systems for predicting and future studies , rapid internet searches , rendering three-dimensional models . This research addresses the application of cluster computing for rendering scenes in the dimensional artificial photographic field currently for rendering various techniques used , one of which is known as ray tracing on which is characterized by generating images have great realism, this due to the large amount of calculations using this method to simulate an almost perfect physical environment , although the application of a cluster of computers obviously improves rendering times , the study performed in this research is specifically to determine the influence of a cluster of high-performance computers at the time of rendering photorealistic three-dimensional models using ray tracing technique , determining the incidence of improvement on reducing rendering time , for it development of research covers three chapters: the first chapter the methodological approach , this chapter defines the problem , objectives , hypotheses , variables , research design, population , sample and data collection instruments and measurement approaches data , then in the second chapter the development of the framework is done , the theoretical framework includes background research , parallel processing , distributed systems, and computer cluster rendering and postproduction. In the third chapter the application of the methodology is developed, this chapter discusses the installation and configuration of the obtained solutions of technological exploration, it also develops the initial phase of testing and final testing of the investigation. In the fourth chapter the results are developed, this chapter covers the statistical tests, tests of hypotheses and discussion of the results. Finally, conclusions and recommendations of the research is published.

Key words: Ray tracing with cluster

CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE FÓRMULAS

INTRODUCCIÓN

La generación de gráficos por ordenador en la actualidad ha llegado al punto en que forma parte importante de nuestras vidas cotidianas ya que nos da una idea del mundo real y ficticio a través de imágenes y secuencias de videos generadas artificialmente en un computador, prueba de ello son las escenas de películas en tres dimensiones como Avatar, Cars, Sherk, entre otros, sin embargo para producir una simple imagen tridimensional fotorealistas o un video tridimensional de alta definición se necesitan muchos días, semanas y hasta meses de trabajo y ardua labor para diseñar las escenas y secuencias de video, pero no todo acaba en el diseño sino que este tiempo se amplía aún más para procesar las escenas y así de esta manera obtener el producto final, todo este proceso no se lleva en un único ordenador mucho menos en dos ya que si ese fuese el caso estas producciones fotográficas y cinematográficas demorarían muchos años por estar compuesta por miles de millones de cálculos necesarios para interpretar las escenas dándole aspectos realistas a tan alto nivel que no se puede diferenciar una fotografía real de un modelo tridimensional generado artificialmente en un computador, es por ello que para resolver este problema se utiliza la tecnología en clúster, un conglomerado de computadores que funcionan coordinadamente para realizar los cálculos correspondientes a una sola tarea y que permiten tener acceso a capacidades gigantes de procesamiento intensivo de datos y almacenamiento uniendo todos los CPUs (Unidades Centrales de Proceso), memoria RAM(Memoria de Acceso Aleatorio) y unidades de almacenamiento como discos duros, todo esto a través de redes de computadoras de área local, una solución que muchos denominan "*la supercomputadora de los pobres*", evidentemente esta tecnología es utilizada para reducir drásticamente los tiempos de renderización con la utilización de un tipo de clúster en especial, el clúster de alto rendimiento con el que se procesan escenas tridimensionales correspondientes a modelos generados artificialmente en un computador, la aceleración de cálculos es una de las cualidades que brindan los clúster de alto rendimiento, esta cualidad se debe a que todos los cálculos necesarios para la interpretación del modelo son distribuidos entre todo el conglomerado de computadores, en palabras más sencillas se divide el trabajo para lograr la aceleración de cálculos entre todos los nodos (computadoras) conformantes del clúster de alto rendimiento, la planificación de las tareas se realiza desde un nodo del clúster el cual recibe el nombre de *"nodo maestro"* y su trabajo es coordinar la repartición de trabajos así como la recolección y unión de los trabajos encargados.

En el presente trabajo de investigación estudia la tecnología clúster en general así como la síntesis de imágenes fotorealistas con la técnica de trazado de rayos y los tiempos de renderización de escenas correspondientes a un modelo tridimensional determinado con clúster de computadoras de alto rendimiento, con la finalidad de determinar la influencia del clúster en los tiempos de renderización. Si bien es cierto los antecedentes de la presente investigación mencionan la existencia de una investigación similar, ésta está enfocada a medir la reducción de tiempos de

renderización dividiendo la cantidad de franjas de la escena entre los nodos, es también importante mencionar que esta investigación se realiza con solamente una escena por lo cual no hubo mayor profundización acerca del tema y no se evaluaron suficientemente los tiempos, la investigación experimental exige trabajar con muestras mucho mayores y con pruebas estadísticas bastante exigentes, es en ese sentido que la presente investigación pretende profundizar las investigaciones realizadas anteriormente desde un punto de vista distinto y aportando información adicional sobre los porcentajes de reducción en los tiempos de renderización para de esta manera determinar la influencia de un clúster de computadoras de alto rendimiento con trazado de rayos, tomado en cuenta la cantidad de nodos, cantidad de CPUs y Cantidad de Memoria, para ello se esquematiza el desarrollo del tema de la siguiente manera: En el primer apartado se tratará el planteamiento metodológico el cual aborda una serie de sub apartados como son definición del problema, objetivos de la investigación, hipótesis de la investigación, variables, el método de investigación a utilizar, la población y muestra así como las herramientas de medición y recolección de datos.

En el segundo capítulo se abordará todo el marco teórico de la investigación iniciando con los antecedentes de la investigación, un enfoque del crecimiento en el tratado de datos, seguidamente se abordará s los temas de procesamiento paralelo y los sistemas distribuidos así como los clúster de computadoras en su total amplitud, Una parte muy importante de la investigación se halla en el sub apartado de la exploración tecnológica en el cual se investigó sobre las diversas herramientas y plataformas de software libre que nos permiten implementar clúster de computadoras de alto rendimiento, seguidamente se realiza una comparación de dichas herramientas para seleccionar la más adecuada, de similar manera también se procede a realizar una investigación sobre los programas de software libre de trazado de rayos así como los factores a tomar en cuenta para determinar los tiempos de renderización, todo esto inmerso en la exploración tecnológica de las herramientas utilizadas, seguidamente se define toda la teoría sobre las técnicas de renderizado y la postproducción.

En el tercer capítulo se aborda la aplicación de la metodología, este apartado se divide de la siguiente manera: en primer lugar se explica la implementación del sistema Clusterknoppix, una vez finalizada esta fase se procede a explicar la fase de pruebas iniciales que se realizaron con el grupo control seguido de la fase de pruebas finales con los grupos experimentales, todas estas pruebas se desarrollan de manera detallada.

En el cuarto capítulo se desarrolló los resultados de la investigación, con todas las validaciones estadísticas correspondientes al tipo de investigación, se desarrolla en este apartado las pruebas de hipótesis, pruebas estadísticas y la discusión de los resultados de la investigación. Finalmente se culmina la investigación con las conclusiones y las recomendaciones del estudio.

CAPITULO I
PLANTEAMIENTO METODOLÓGICO

1.1. Descripción de la realidad del Problema

La generación de gráficos por computadora es un área de investigación cuyo objetivo es la creación de algoritmos y métodos que permitan generar una imagen sintetizada artificialmente en un computador a partir de un modelo tridimensional, uno de los retos al cual aún se enfrenta a la hora de incorporar estos modelos en un computador es la gran cantidad de cálculos necesarios para determinar la iluminación global, esta es una de las razones por las cuales el objetivo de generar imágenes de manera eficiente aún se mantiene vigente. Por otro lado el uso del paralelismo permite solucionar estos problemas en un tiempo razonable.

En la actualidad, la mayoría de los programas denominados motores de renderizado como Povray, Scaline, Mental Ray, Vray, Brasil, Maxwell, etc. realizan los cálculos necesarios en la interpretación del modelo tridimensional utilizando todo el CPU y el hardware disponible en dicha labor de mano con la memoria RAM que dispone el equipo, de aquí la importancia de contar con elementos de alto rendimiento y capacidad.

Una opción para incrementar el rendimiento y la capacidad es usar un conjunto de computadoras, las cuales se utilizan para renderizar imágenes en tiempos cortos.

"*Conforme la tecnología avanza los recursos de los ordenadores se incrementan también, tales como la cantidad de núcleos de procesamiento, memoria RAM y capacidad de almacenamiento*" "*muchos de estos recursos no son aprovechados en su totalidad, a menudo, el recurso ocioso en los computadores, en promedio, es del 85% durante el día y del 95% durante la noche" (Foster & Kesselman, 2003), además, "se presenta un uso de procesador menor al 5%".* (Gorlatch, Fragopoulou, & Priol, 2008)

En la sede Totoral de la Universidad Nacional José María Arguedas se cuenta con 40 equipos de cómputo distribuidos en el laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas, los cuales no realizan tareas ni procesos complicados y por ende se desperdicia recursos de hardware como antes se mencionó, estos recursos desperdiciados pueden ser utilizados para poder obtener un mejor aprovechamiento con la implementación de un clúster de computadoras de alto rendimiento el cual tenga como propósito estudiar e investigar la incidencia en la reducción de tiempos de renderización de modelos tridimensionales fotorealistas generadas con la técnica de trazado de rayos el cual requiere una gran cantidad de cálculos matemáticos y procesamiento intensivo de datos para simular todos los aspectos físicos de la naturaleza de las imágenes fotorealistas.

1.2. Definición del problema

1.2.1. Problema General

Según lo antes abordado el planteamiento del problema será el siguiente:

¿Cuál es la influencia de un clúster de computadoras de alto rendimiento en el tiempo de renderización de escenas tridimensionales fotorealistas en la Universidad Nacional José María Arguedas en el 2013?

1.2.2. Problemas específicos

- ✓ **¿**Qué herramientas tecnológicas permiten la implementación de un clúster de computadoras de alto rendimiento y que aplicaciones permiten sintetizar imágenes fotorealistas con trazado de rayos**?**

- ✓ ¿Qué factores deben tomarse en cuenta para determinar los tiempos de renderización de modelos tridimensionales fotorealistas con trazado de rayos?

- ✓ ¿Es posible implementar de un clúster de computadoras de alto rendimiento en el laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas que permitan evaluar la cantidad de recursos computacionales del clúster como memoria y procesadores?

- ✓ ¿Qué factores deben tomarse en cuenta para evaluar los tiempos de renderización de modelos tridimensionales fotorealistas utilizando trazado de rayos?

1.3. Objetivos

1.3.1. Objetivo general

Evaluar la influencia de un clúster de computadoras de alto rendimiento en el tiempo de renderización de modelos 3D fotorealistas en la Universidad Nacional José María Arguedas.

1.3.2. Objetivos específicos

- ✓ Realizar una exploración de las herramientas tecnológicas para implementar un clúster de computadoras de alto rendimiento así como las herramientas para la renderización de modelos tridimensionales fotorealistas basadas para el trazado de rayos y seleccionar la más adecuada para la investigación.
- ✓ Realizar un estudio sobre las herramientas y factores para determinar los tiempos de procesamiento en la renderización de modelos tridimensionales fotorealistas utilizando trazado de rayos.

- ✓ Implementar un clúster de computadoras de alto rendimiento en el laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas para así determinar y evaluar los recursos computacionales del clúster como memorias y procesadores.
- ✓ Evaluar los tiempos de renderización de modelos tridimensionales fotorealistas con trazado de rayos.

1.4. VARIABLES

1.4.1. Variables independientes

Se tendrá como variable independiente al ***clúster de computadoras de alto rendimiento.***

1.4.2. Variables dependientes

Tiempos de renderización de modelos tridimensionales fotorealistas usando trazado de rayos.

1.5. Método y diseño de la investigación

El diseño de la investigación es de tipo experimental puro, se manipulará la variable independiente (clúster de computadoras de alto rendimiento), en diversos grados, estos grados son referidos como los indicadores de medida de capacidad del clúster, a su vez éstos son catalogados como la cantidad de núcleos de procesamiento, la cantidad de memoria RAM y la cantidad de nodos conformantes del clúster, estos indicadores son variados para observar la incidencia de mejora en los tiempos de renderización, es importante mencionar que las investigaciones experimentales exigen se deben tratar por lo menos dos grupos, uno de control y los otros experimentales, es por ello que todas las observaciones que se realizaron están enfocadas 6 grupos de escenas, ya que en el laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas se cuenta con 40 equipos de cómputo funcionales, la cantidad máxima que se asignará es de 24 nodos, cada nodo estará conformado por las siguientes características 4 núcleos de procesamiento, 3 GB de memoria RAM, 20 GB de capacidad de disco duro, la cantidad de nodos asignados a los grupos de escenas tridimensionales se ilustra en la tabla 1.

Tabla 1. Grupos de escenas tridimensionales de la investigación

Grupo	Cantidad de nodos asignados
Grupo 1	24
Grupo2	16
Grupo3	8
Grupo4	4
Grupo5	2
Grupo 6	1

Fuente: Elaboración propia

Como se observa en la tabla 1, el grupo 6 solo contendrá un nodo (1 solo equipo de cómputo), esto quiere decir que por sí solo no es un clúster, ya que un clúster está conformado de 2 a más equipos o nodos como se refiere en el contexto, por consiguiente el grupo 6 será el grupo control y los grupos restantes serán los grupos experimentales los cuales se estudiarán para determinar la incidencia de mejora en los tiempos de renderización con respecto al grupo control.

El diseño de la investigación es de subtipo postprueba únicamente y grupo control quedando el diseño de la investigación como se muestra en la figura 1.

G1	→	X1	→	O1
G2	→	X2	→	O2
G3	→	X3	→	O3
G4	→	X4	→	O4
G5	→	X5	→	O5
G6	→	-	→	O6

Figura 1. Diseño de investigación

Fuente: Elaboración propia

Como se observa en la figura 1, se manipulará la variable independiente "X", con ello se realizará las observaciones "O" para cada grupo experimental así como para el grupo control.

1.6. Población y Muestra

1.6.1. Población

Actualmente son muchos los modelos tridimensionales existentes y creados, por lo que la población de estudio es de carácter infinito, los modelos tridimensionales existentes muestran artefactos, esferas, objetos, lugares artificiales, plantas, recipientes entre otros como se muestra en la figura 2.

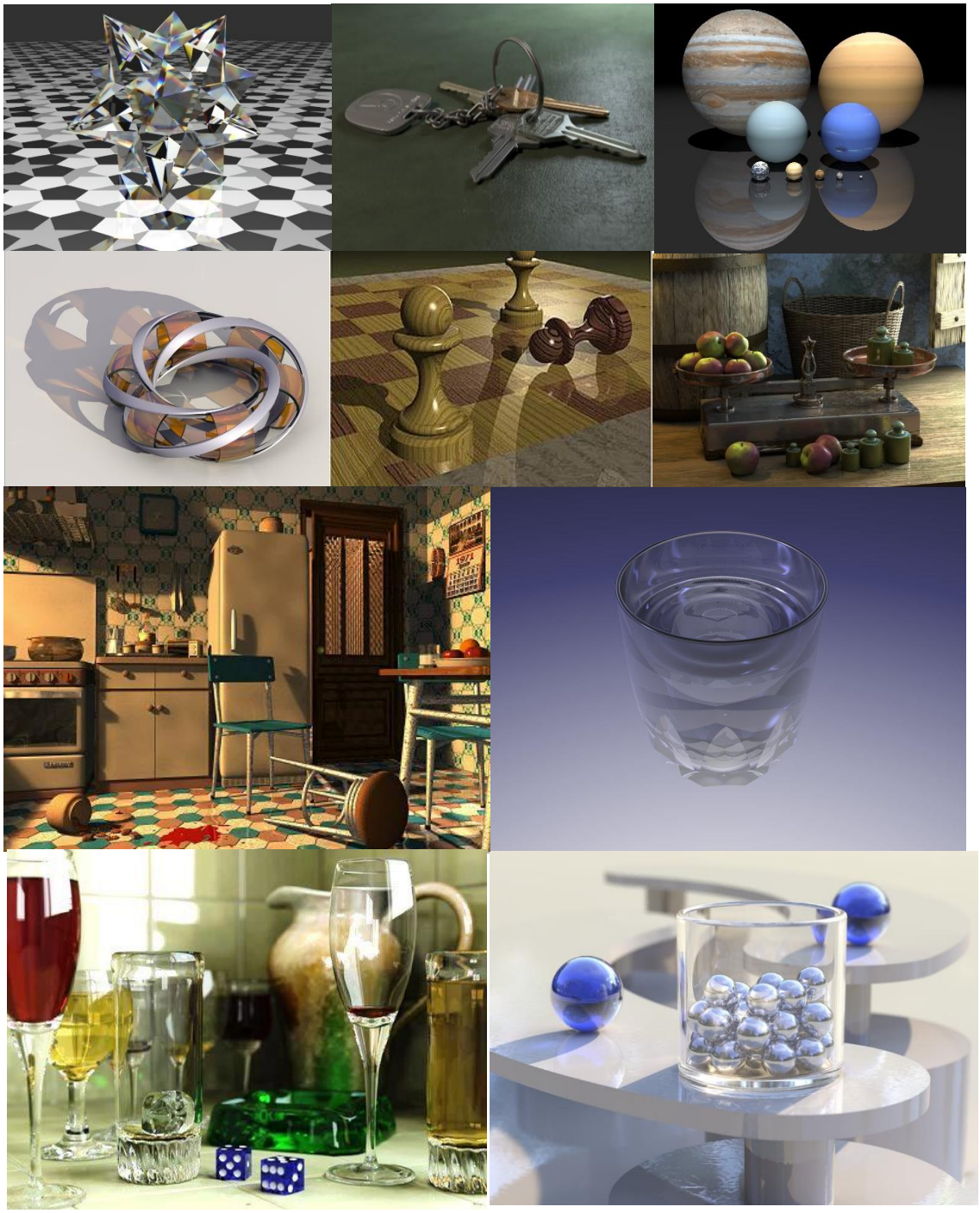


Figura 2. Modelos tridimensionales fotorealistas

Fuente: Salón de la fama, página oficial de Povray, http://www.hof. povray.org, 2013

Por otro lado, existen otros modelos mucho más complejos a los cuales no se tiene acceso directo, estos modelos poseen autores que a su vez los ofrecen por internet bajo pago pero que están dotados de una realidad sorprendente como se muestra en la figura 3.

Figura 3. Modelos tridimensionales complejos

Fuente: Salón de la fama, página oficial de Povray, http://www.hof. povray.org, 2013

1.6.2. Muestra

Como se mencionó anteriormente existen muchos modelos tridimensionales los cuales pudieron haberse utilizado para realizar el estudio, sin embargo ya que el diseño de investigación es netamente experimental se debe cumplir ciertas exigencias, según la revisión de la literatura, la investigación de tipo experimental exige que las muestras seleccionadas sean tomadas totalmente al azar y distribuidas de la misma manera de forma aleatoria entre los grupos experimentales, también nuestras muestras seleccionadas deben poseer las mismas características, por lo tanto sería demasiado complicado e imposible seleccionar modelos al azar que posean las mismas características de entre tanta diversidad de modelos fotorealistas, sin embargo como se define en los antecedentes de la investigación en el apartado 2.1 se ve claramente que en investigaciones anteriores se utilizó solo un modelo tridimensional, por las exigencias del tipo de investigación experimental la selección de las muestras serán realizadas totalmente al azar y tomadas de un único modelo tridimensional, las muestras corresponderán posiciones de cámaras en concreto desde el espacio tridimensional, dichas posiciones de cámaras son conocidas como ***escenas***.

Durante la exploración tecnológica se indagó que el software de Povray cuenta con modelos, estos modelos contienen cierta complejidad, es importante señalar que mientras más compleja sea una escena, mucho mayor será el tiempo de renderización por la inmensa cantidad de cálculos a realizar, multiplicado esto por la cantidad de escenas a renderizar se utilizarían muchas horas de procesamiento, es por ello que el modelo seleccionado para el desarrollo de la presente investigación es archivo Povbenchmark al poseer complejidad mediana.

Es importante mencionar que un modelo tridimensional puede contar con infinitas escenas tomadas desde el espacio tridimensional, éstas escenas serán las muestras de la investigación, así se aprecia una escena del modelo el cual se denomina para caso de estudio el modelo **benchmark** en la figura 4.

Figura 4. Escena del modelo benchmark

Fuente: Página oficial de Povray, http: //www.povray.org/i/benchmark.png, 2013

Ya que se tomó el modelo benchmak, a parir de este modelo se extrajeron escenas tomadas del espacio tridimensional, ya que la población es infinita la can de muestras se establece por la siguiente fórmula.

$$n = \frac{Z_{\alpha.}^2 P.(1-P)}{i^2}$$

Fórmula 1. Muestras para poblaciones infinitas

Donde:

Z= 1.96 i=0.05 P=0.5

Reemplazando los valores se tendrá la siguiente cantidad de escenas:

$$n = \frac{(1.96)(0.5)(1-0.05)}{0.05^2} = 384.16$$

Se tendrá como muestra representativa del modelo tridimensional a 384 escenas del modelo benchmak las cuales serán distribuidas aleatoriamente entre los 6 grupos sujetos a experimento.

La selección de las escenas tridimensionales deben ser de forma aleatoria para ello se procedió a generar en el programa de Microsoft Excel las posiciones en los ejes coordenados X, Y, Z de forma aleatoria con la función "aleatorio", estas escenas aleatorias corresponden a las posiciones de la cámara para realizar el proceso de renderización, dichas posiciones apuntan al centro del modelo benchmark de coordenadas (1, -1, 0.9), la figura 5 muestra las posiciones de las escenas aleatorias.

POSICIONES DE CÁMARAS PARA LA RENDERIZACIÓN DE ESCENAS DEL MODELO BENCHMARK

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Posición de la cámara en el eje X	Posición de la cámara en el eje Y	Posición de la cámara en el eje Z	GRUPO	GRUPOS EXPERIMENTALES	
1	-26.79	18.83	28.84	1	Grupo 1	64
2	-0.22	7.13	13.4	5	Grupo 2	64
3	16.01	10.59	17.82	3	Grupo 3	64
4	11.21	5.78	18.23	3	Grupo 4	64
5	7.68	19.91	18	2	Grupo 5	64
6	0.99	2.26	4.15	2	Grupo 6	64
7	15.21	14.18	4.85	4		
8	1.3	16.89	19.96	4		

Figura 5. Posiciones aleatorias de las cámaras

Fuente: Elaboración propia

En el anexo 1 se pueden hallar las 384 ocurrencias de la población.

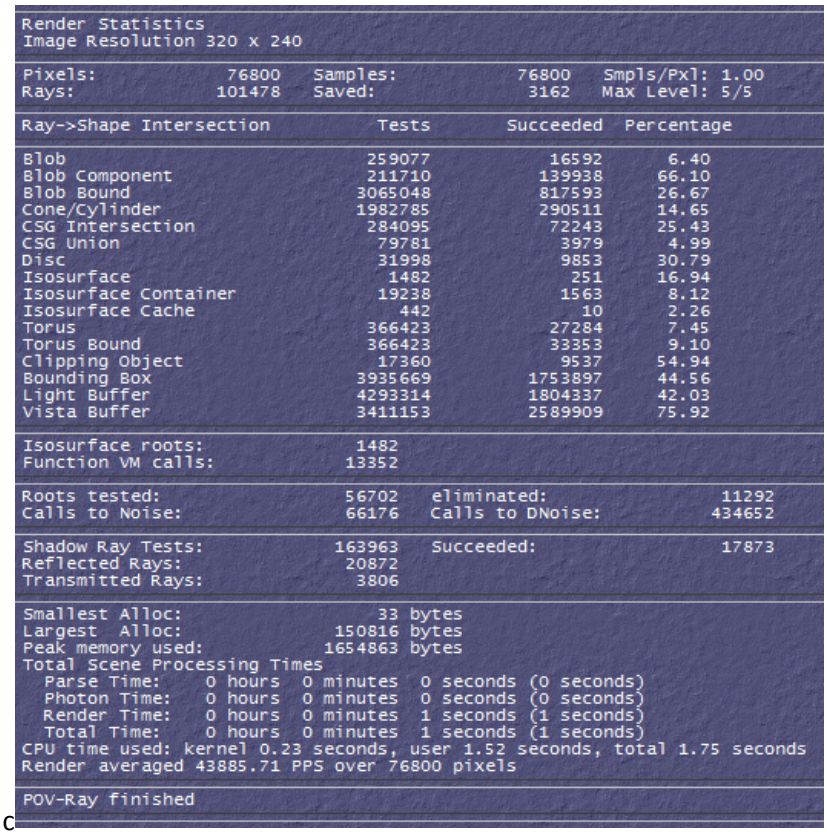
Se realizó la distribución aleatoria de las escenas entre los diversos grupos experimentales, como se puede apreciar en la figura 5 cada grupo está representado por un color, la cantidad de escenas por cada grupo experimental es 64.

1.7. Técnicas e instrumentos de recolección de datos

1.7.1. Instrumentos de medición

Actualmente la cantidad de motores de renderizado con trazados de rayos en el mercado es amplio se pueden hallar programas de software libre como se observa en el apartado 2.5.3, esta amplia gama de programas y paquetes de software incluyen estadísticas e información detallada sobre el proceso de renderización de escenas tridimensionales utilizando la técnica de trazado de rayos, esta información incluye: la cantidad de rayos utilizados, pixeles de la imagen resultante, cantidad de muestras, cantidad de primitivas tridimensionales utilizadas, cantidad de luces, sombras, reflexiones de refracciones de luz así como la cantidad de memoria predestinada al proceso de renderizado, también se halla información sobre: los tiempos de interpretación de las escenas, los tiempos de propagación de los fotones, los tiempos de renderización totales y por último y no menos importante los diversos tiempos de procesamiento como: la cantidad de hilos utilizados para el procesamiento de la escena, el tiempo de utilización del núcleo del sistema operativo y el tiempo de usuario utilizado para el procesamiento de escenas. En conclusión se utilizarán las estadísticas de trazado que nos

brinda el programa Povray como herramienta de medición el cual fue la seleccionada de acuerdo a la comparativa resultante de la exploración tecnológica correspondiente al marco teórico del apartado 2.5, la figura 6 se muestra las estadísticas que genera el programa Povray.



```
Render Statistics
Image Resolution 320 x 240

Pixels:            76800   Samples:           76800   Smpls/Pxl: 1.00
Rays:             101478   Saved:              3162   Max Level: 5/5

Ray->Shape Intersection         Tests      Succeeded  Percentage

Blob                           259077          16592        6.40
Blob Component                 211710         139938       66.10
Blob Bound                    3065048         817593       26.67
Cone/Cylinder                 1982785         290511       14.65
CSG Intersection               284095          72243       25.43
CSG Union                       79781           3979        4.99
Disc                            31998           9853       30.79
Isosurface                       1482            251       16.94
Isosurface Container            19238           1563        8.12
Isosurface Cache                  442             10        2.26
Torus                          366423          27284        7.45
Torus Bound                    366423          33353        9.10
Clipping Object                 17360           9537       54.94
Bounding Box                  3935669         1753897       44.56
Light Buffer                  4293314         1804337       42.03
Vista Buffer                  3411153         2589909       75.92

Isosurface roots:                1482
Function VM calls:              13352

Roots tested:                   56702   eliminated:             11292
Calls to Noise:                 66176   Calls to DNoise:       434652

Shadow Ray Tests:              163963   Succeeded:              17873
Reflected Rays:                 20872
Transmitted Rays:                3806

Smallest Alloc:                    33 bytes
Largest  Alloc:                150816 bytes
Peak memory used:             1654863 bytes
Total Scene Processing Times
  Parse Time:    0 hours  0 minutes  0 seconds (0 seconds)
  Photon Time:   0 hours  0 minutes  0 seconds (0 seconds)
  Render Time:   0 hours  0 minutes  1 seconds (1 seconds)
  Total Time:    0 hours  0 minutes  1 seconds (1 seconds)
CPU time used: kernel 0.23 seconds, user 1.52 seconds, total 1.75 seconds
Render averaged 43885.71 PPS over 76800 pixels

POV-Ray finished
```

c

Figura 6. Estadísticas generadas por el programa Povray 3.6

Fuente: Elaboración propia

Por otro lado existen diversos sistemas operativos que trabajan en entorno distribuido con el cual se puede construir e implementar clúster de computadoras de alto rendimiento, estas soluciones también brindan información que es muy indispensable para la investigación como es: cantidad de memoria total del clúster, cantidad de CPUs del clúster, identificación de los nodos del clúster, numero de procesos, estadísticas de balanceo de carga y estadísticas de migración de procesos, en el apartado 2.5.1 se realizó la exploración tecnológica de las soluciones clúster existentes en el mercado, dicha exploración tecnológica concluyo con la selección del sistema operativo Clusterknoppix basado en la distribución Knoppix del muy conocido sistema operativo GNU/Linux y que trabaja de la mano con clúster Openmosix, software libre en su totalidad, se considera como herramienta de medición a Openmosix View el programa que nos brinda la información esencial para la investigación, en la figura 7 se muestra el programa con las características antes mencionadas.

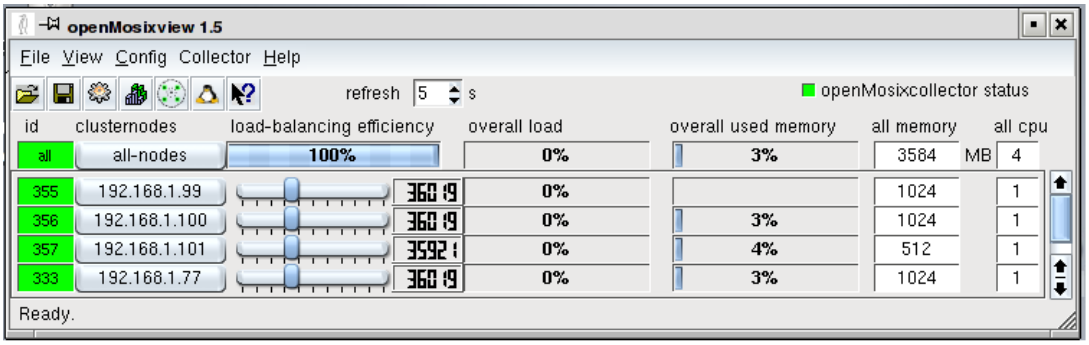


Figura 7. Funcionamiento de Openmosixview

Fuente: Elaboración propia

1.7.2. Instrumentos de recolección de datos.

Como se pudo observar el gráfico 1 se hace referencia a una plantilla diseñada en Microsoft Excel para generar nuestras posiciones aleatorias, de manera similar a esta plantilla se utilizará otras plantillas en las cuales se filtra por grupos experimentales las diversas posiciones de las cámaras para poder recolectar los tiempos de renderización de escenas tridimensionales expresando el tiempo en minutos, segundos, equivalente en segundos totales, cantidad de CPUs , se muestra en la figura 8 la plantilla utilizada.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES	
1	2	36	156	96	96	1		
2	3	59	239	8	96	5		
3	1	46	106	32	96	3		
4	1	48	108	32	96	3		
5	2	6	126	64	96	2		
6	2	29	149	64	96	2		
7	4	38	278	16	96	4		
8	3	59	239	16	96	4		
9	6	20	380	8	96	5		
10	2	32	152	96	96	1		

Figura 8. Plantilla para la recolección de tiempos de renderizado del modelo Benchmak

Fuente: Elaboración propia

En el anexo 2 se muestra completamente las plantillas utilizadas para recolectar los tiempos de renderización de las escenas tridimensionales.

Adicionalmente se utilizó una plantilla para anotar los tiempos de renderización correspondientes a la confiabilidad de los instrumentos de medición, dicha plantilla se muestra en la figura 9.

RESULTADOS DE LAS RENDERIZACIÓN DE POSICIONES DE LAS CÁMARAS						
POSICIONES DE LAS CÁMARAS DEL MODELO MP						
Nº	Minutos	Segundos	Tiempo total en segundos	CPUS		
1			0	2		
2			0	2		
3			0	2		
4			0	2		
5			0	2		
6			0	2		
7			0	2		
8			0	2		
9			0	2		

Figura 9. Plantilla para la recolección de tiempos de renderizado del modelo MP

Fuente: Elaboración propia

En el anexo 3 se muestra la plantilla utilizada.

1.7.3. Confiabilidad de los instrumentos de medición seleccionados

Las herramientas de medición deben ser sometidas a pruebas de confiabilidad para ello se realizaron estudios sobre estas herramientas a fin corroborar el correcto funcionamiento de ellos y que midan lo que deben medir.

Para la prueba de confiabilidad con respecto a la herramienta Povray, se realizaron pruebas en la renderización una escena correspondientes a un único modelo tridimensional en un computador, se verificó que los resultados contengan un grado de variación mínimo, es importante señalar que ningún proceso se guarda en la memoria con la finalidad de acelerar el proceso al ejecutarlo por segunda vez (caché), por lo cual esta prueba goza de aceptación en tal sentido, en la figura 10 se muestra la imagen resultante tras renderizar en 5 ocasiones seguidas el siguiente modelo denominado MP.

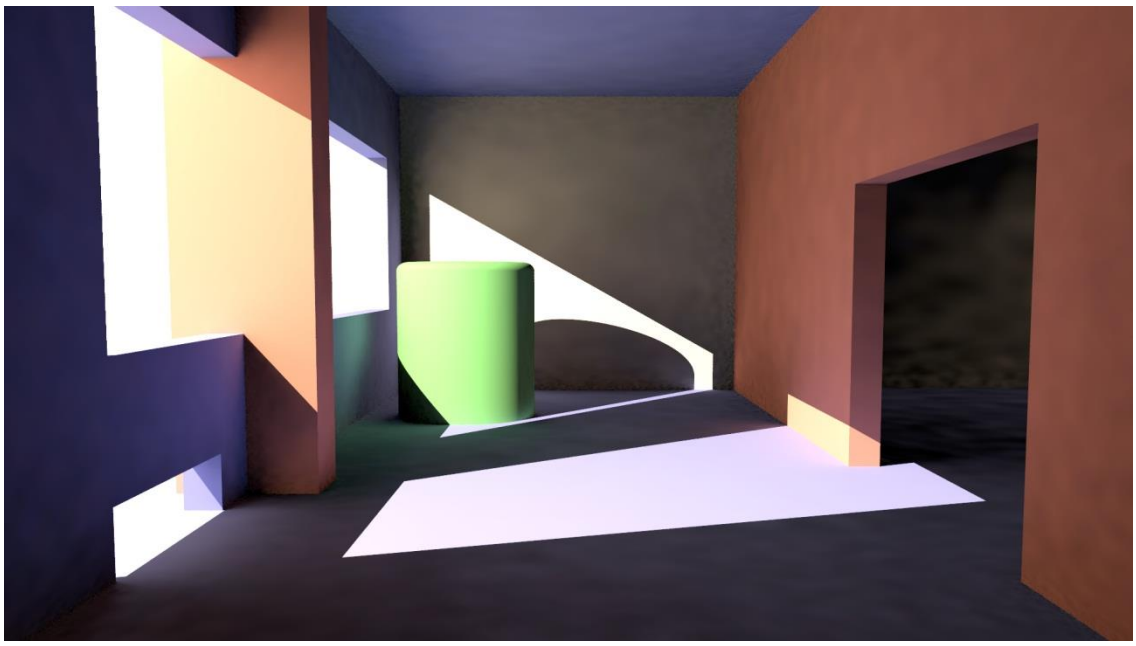


Figura 10. Modelo utilizado para la prueba de confiabilidad

Fuente: Página oficial de Megapov, http: //megapov.inetart.net/demo/radiosity_improvements.pov, 2013

Ya que nuestro estudio está primordialmente enfocado en el análisis de los tiempos de renderización, se muestra el resultado de la prueba 1 el cual cuenta con los siguientes datos: tiempo de análisis de la escena, tiempo de fotón, tiempo de renderización y el tiempo total en la figura 11.

```
Render Statistics
Image Resolution 512 x 384

Pixels:          262145   Samples:          262145   Smpls/Pxl: 1.00
Rays:           5526245   Saved:                0   Max Level: 2/20

Ray->Shape Intersection         Tests      Succeeded  Percentage

Box                          31736411       18162531     57.23
Cone/Cylinder                 1004154         307404     30.61
CSG Intersection              7319335        3456357     47.22
Torus                          126566          59300     46.85
Torus Bound                    126566          68500     54.12
Bounding Box                 82304747       43669401     53.06
Light Buffer                   686021         468133     68.24
Vista Buffer                  2397352        1763896     73.58

Roots tested:                  68500   eliminated:                28942
Calls to Noise:                    0   Calls to DNoise:              10

Shadow Ray Tests:            4827073   Succeeded:               2378881

Radiosity samples calculated:            35094 (13.39 %)
Radiosity samples reused:               226984

Smallest Alloc:                    34 bytes
Largest  Alloc:                149032 bytes
Peak memory used:              6782934 bytes
Total Scene Processing Times
  Parse Time:    0 hours  0 minutes  0 seconds (0 seconds)
  Photon Time:   0 hours  0 minutes  0 seconds (0 seconds)
  Render Time:   0 hours  0 minutes 27 seconds (27 seconds)
  Total Time:    0 hours  0 minutes 27 seconds (27 seconds)
CPU time used: kernel 7.13 seconds, user 19.81 seconds, total 26.94 seconds
Render averaged 7298.67 PPS over 196608 pixels

POV-Ray finished
```

Figura 11. Estadísticas generadas de la prueba 1 del modelo MP

Fuente: Elaboración propia

En el anexo 4 se muestra los gráficos correspondientes a las cinco pruebas con los resultados respectivos.

Se muestra a continuación la tabla 2 que resume los tiempos de renderización de escenas tridimensionales de las 5 pruebas realizadas al modelo MP.

Tabla 2. Resumen de los tiempos de interpretación y renderización de escenas

N° de prueba	Tiempo de interpretación del modelo (segundos)	Tiempo de Fotón (segundos)	Tiempo de renderizado (segundos)	Total de Tiempo (segundos)
1	0	0	27	27
2	0	0	27	27
3	0	0	27	27
4	0	0	27	27
5	0	0	27	27

Fuente: Elaboración propia

Como se puede visualizar en la tabla 2, los tiempos de interpretación del modelo así como los tiempos de renderización no difieren por lo cual la herramienta utilizada es confiable para realizar los estudios.

Con respecto a las pruebas de confiabilidad realizadas al sistema operativo Clusterknoppix específicamente al programa Openmosixview se encendió varios computadores y luego se apagó otros con la finalidad de observar y documentar la detección de los recursos computacionales del clúster tales como la cantidad de memoria RAM así como la cantidad de CPUs.

Dicha observación tuvo como respuesta que un nodo al ser apagado o encendido es detectado automáticamente por los demás nodos gracias al demonio Openmosixcollector, además al realizar dichas pruebas se observó que tanto la cantidad de memoria así como la cantidad de CPUs es detectada casi instantáneamente, las siguientes figuras muestran la detección y la caída de nodos así como el incremento y decremento de los recursos computacionales del clúster con el programa de Openmosixview, se muestra también información detallada del clúster como: identificación de nodos, dirección IP de los nodos, eficiencia en balanceo de carga, carga total, porcentaje de memoria total utilizada, memoria total del clúster en MB y total de unidades centrales de procesamiento CPUs.

Las siguientes pruebas se realizaron con 4 equipos de cómputo, todos ellos con un solo CPU, tres de ellos con 1GB de memoria RAM y uno con 512 MB de memoria RAM, en la figura 12 se puede observar la aplicación Openmosixview con la identificación de nodos.

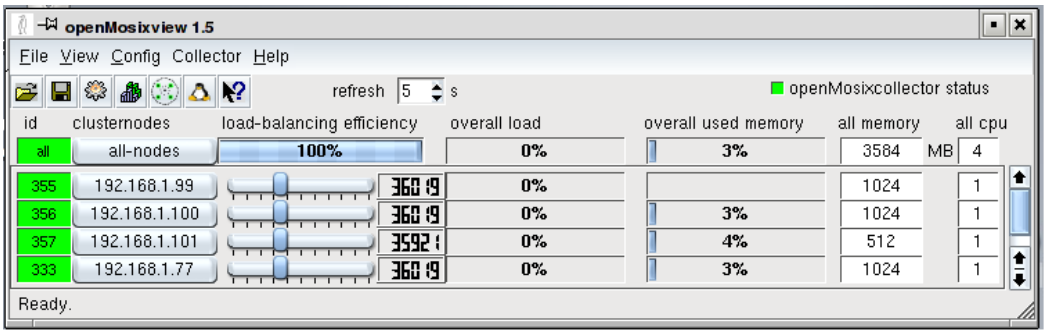


Figura 12. Interface del programa Openmosixview

Fuente: Elaboración propia

Para la primera prueba se apagó el nodo con identificador número 333 y se pudo observar que alrededor de 2 segundos la ausencia del nodo con identificador 333 reduciéndose así la capacidad del clúster tanto en memoria RAM como en unidades de procesamiento CPU, tal y como se puede observar en la figura 13.

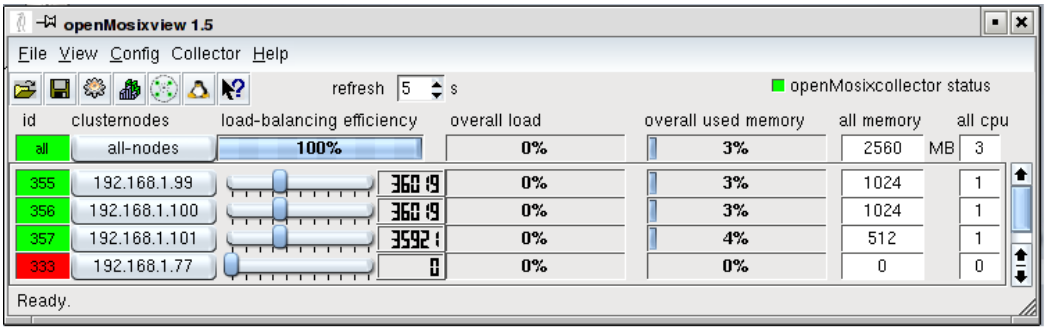


Figura 13. Ausencia de nodo 333 en Openmosixview

Fuente: Elaboración propia

Para la segunda prueba se volvió a encender el nodo con identificador 333 y seguidamente se apagó el nodo 356 como se puede apreciar en la figura 14 los cambios fueron detectados casi al instante y la variación de los recursos computacionales con respecto a memoria RAM y cantidad de CPUs fue la correcta.

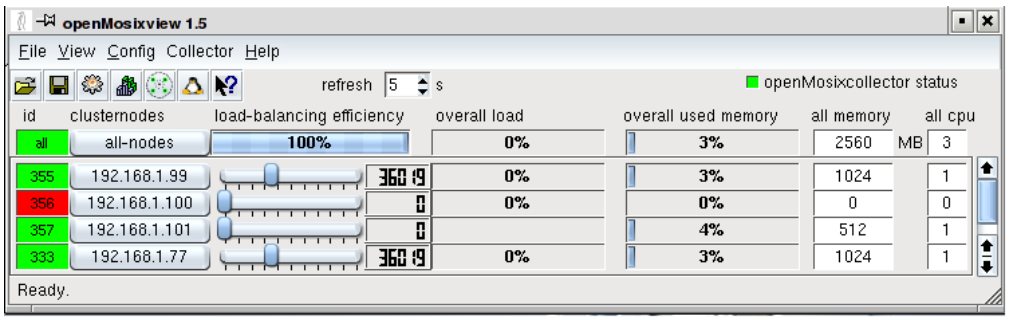


Figura 14. Ausencia del nodo 356 en Openmosixview

Fuente: Elaboración propia

En la tercera prueba se apagaron los nodos con identificadores 358 y 359 a fin de observar la variación en los recursos computacionales, esta variación resultó correcta con respecto a memoria RAM y cantidad de CPUs como se muestra en la figura 15.

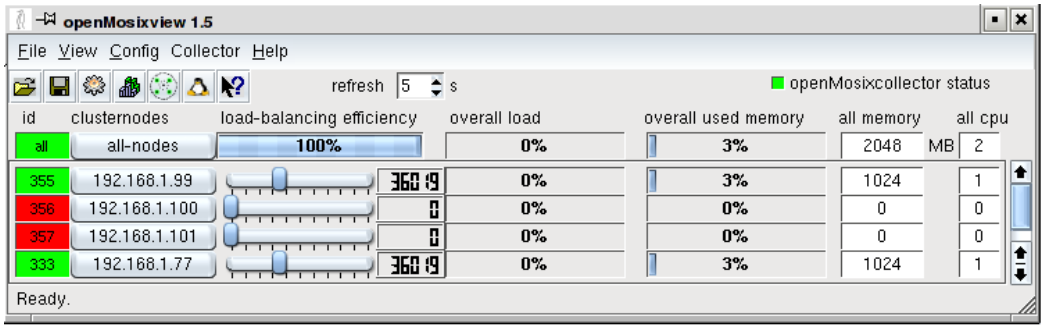


Figura 15. Ausencia de los nodos 356 y 357

Fuente: Elaboración propia

Finalmente para la cuarta prueba se realizaron diversas combinaciones a fin de observar la variación de los recursos computaciones dichas variaciones fueron correctas tal y como se muestra en la figura 16.

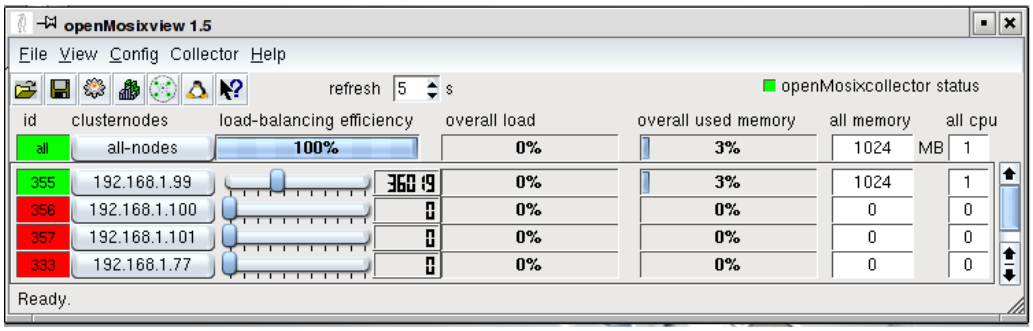


Figura 16. Variación de los recursos computacionales

Fuente: Elaboración propia

1.7.4. Objetividad de los instrumentos de medición

Ya que para la presente investigación se utilizan sistemas y programas previamente calibrados la objetividad de los instrumentos de medición es obvia por ende no se realizaron pruebas adicionales para determinar la objetividad de nuestras herramientas de medición por ser instrumentos verificados y comprobados previamente.

1.8. Justificación de la Investigación

El presente trabajo de investigación tiene como propósito dotar de una potente herramienta de computación distribuida a aquellas personas, empresas, investigadores o alumnos cuyo trabajo esté orientado a la síntesis de imágenes fotorealistas utilizando técnicas de trazado de rayos.

Ya que la síntesis de imágenes fotorealistas suelen aplicarse a diversos rubros como son la creación de secuencias de videos artificiales, tomografías por computador, modelos tridimensionales de estructuras entre otras.

Con las aplicaciones del clúster de computadoras se pretende reducir notablemente los tiempos de renderización de escenas tridimensionales fotorealistas utilizando trazado de rayos, incidiendo así en determinar en qué medida se mejora estos tiempos.

Por otro lado la Universidad Nacional José María Arguedas dispondría del clúster de computadoras para orientarlo a otras áreas de investigación. Según lo tratado en el marco teórico los clúster de computadoras se pueden aplicar a diversas áreas dedicadas a la solución de problemas cuya complejidad computacional suele ser elevada y por ende su tiempo de procesamiento bastante considerable.

CAPITULO II
MARCO TEÓRICO

2.1. Antecedentes de la Investigación

La implementación de clúster de computadoras ha permitido en diversas partes del mundo dar solución a diversos problemas, prueba de ello se hallaron las siguientes investigaciones.

En diciembre del 2002 Leonardi Martin Torrealba realizó una investigación el cual titula "Construcción de un Clúster con Openmosix: Pruebas con Simulación de Halos", el cual se realiza en la Universidad Tecnológica de Mixteca en el que se estudia las simulaciones de Halos para probar y validar el clúster de cómputo científico.

En el 2006 José Castillo Castillo y Ricardo Benjamin Olivera Acosta presentan un trabajo de investigación que titula "Implementación de un Clúster con Openmosix para cómputo científico en el Instituto de Ingeniería" realizado en la Universidad Nacional Autónoma de México el cual aborda problemas científicos y aplica un clúster de computadoras para reducir el tiempo de procesamiento intensivo de datos en investigaciones científicas.

También en el 2006 José Antonio Fernandez Sorribes presenta un proyecto de investigación titulado "Sistema GRID Para el Render de Escenas 3D distribuido: YAFRID" en el cual se crea un sistema vía web que permite la renderización de modelos tridimensionales utilizando múltiples clúster de computadoras reduciendo así drásticamente los tiempos de renderizado y creando a la vez un servicio en línea.

En el 2009 Ignacio Martínez Fernández presenta en su proyecto de fin de carrera titulado "Creación y Validación de un Clúster de Computación Científica Basado en Rocks", dicha investigación se realiza en la Universidad Carlos III de Madrid y tiene como propósito también crear y realizar la validación de un clúster con el propósito de dar solución a problemas complejos utilizando la distribución ROCKS y solucionando problemas matemáticos de diversa complejidad.

Finalmente el Departamento de Sistemas y el departamento de Física y Astrofísica Computacional, Instituto de Física publica un artículo científico que titula "ZeBrA-Core Una Herramienta Para la Generación de Imágenes Fotorealistas Usando POV-Ray en ambientes de Computación Distribuida" en el cual estudia los tiempos de renderización de modelos tridimensionales basados en trazado de rayos, dichas mediciones se realizan sobre una sola imagen con un clúster con capacidad de 22 procesadores y con la división de la imagen en 1, 2, 5,

10, 15 y 20 franjas, este artículo científico tiende a asemejarse al presente proyecto de investigación desde otro punto de vista.

Muchas universidades y centros de investigación hoy en día construyen clúster de computadoras para dar solución a problemas de diversa índole así lo demuestra el ranking de las 500 mejores supercomputadoras del mundo, encabezando esta lista se halla el Tianhe-2, un superordenador desarrollado por la Universidad Nacional de Tecnología de Defensa de China, es el nuevo sistema N º 1 del mundo, con un rendimiento del 33,86 petaflop/s. La lista se anunció el 17 de junio durante la sesión inaugural de la Conferencia Internacional de Supercomputación 2013 en Leipzig, Alemania. (Top 500 Supercomputers, 2013)

En el anexo 5 se muestra la lista de las 5 supercomputadoras del mundo conformantes de este ranking.

2.2. El crecimiento en el tratado de datos

Hoy en día organizaciones y empresas dedicadas a diversos rubros desde industria hasta investigación y educación han incrementado exponencialmente el tratado de sus datos así como su procesamiento.

Según Branch Bedoya & Mesa Munera (2008)
En la actualidad y basándose en los principios de la computación se han planteado algoritmos secuenciales para dar solución a una gran cantidad de problemas. A medida que estos se han ido tornando más y más complejos, las soluciones basadas en este tipo de algoritmos han pasado a ser ineficientes, debido a que su uso implica un consumo de tiempo considerable durante su ejecución.

> Para resolver estos problemas nacieron los denominados supercomputadores, los cuales cuentan con arreglos de microprocesadores que trabajan en sincronía empleando procesamiento paralelo (párr. 5)

Esta solución surge debido a la necesidad de procesar gran cantidad de información y que involucraría demasiado trabajo para un solo equipo de cómputo.

Durante la última década las universidades y centros de investigación al rededor del mundo se han dedicado a construir e implementar sus propios “supercomputadores” al cual se les conoce con el nombre de clúster, estos clúster están basados en los sistemas distribuidos y emplean procesamiento paralelo.

2.2. Procesamiento paralelo

De Dormido, Hernández, Ros & Sánchez (2003)

> La computación paralela o procesamiento en paralelo consiste en acelerar la ejecución de un programa mediante su descomposición en fragmentos que pueden ejecutarse de forma simultánea, cada uno en su propia unidad de proceso. Surge así de forma natural, la idea de la computación paralela, que genéricamente consiste en utilizar n computadoras para multiplicar, idealmente por n la velocidad computacional obtenida de un único computador. Por supuesto esta es una situación ideal que muy rara vez se consigue en la práctica.
>
> Normalmente los problemas no pueden dividirse perfectamente en partes totalmente independientes y se necesita, por tanto una interacción entre ellas que ocasiona una disminución de la velocidad computacional. En este sentido se habla de mayor o menor grado de paralelismo en la medida de que un algoritmo sea más o menos divisible en partes independientes con igual costo computacional.

2.3. Sistemas distribuidos

Para poder entender sobre los sistemas distribuidos, su definición, sus características, su estructura, ventajas, desventajas y aplicaciones nos remontaremos en la historia entre los años 1980 y 1985, años en los cuales se lograron grandes avances tecnológicos en la computación, entre ellos destacan la invención de los procesadores de 8, 16, 32 y 64 bits, la creación de redes de área local (Local Area Network - LAN), los cuales permitieron la interconexión de computadoras pudiendo transferir e intercambiar datos entre ellas en periodos de tiempo muy cortos. Posteriormente aparecen las redes de área amplia (Wide Area Networks - WAN), las cuales eran redes mucho más grandes construidas a partir de redes de área local.

Según Tanenbaum (1996)

> Con la significancia de estos avances tecnológicos hoy en día es posible utilizar y construir sistemas de cómputo compuestos por un gran número de computadoras interconectadas por medio de una red de alta velocidad y con la utilización de software especial. Estos sistemas reciben el nombre de **Sistemas Distribuidos**.

2.3.1. Definición de un sistema distribuido

Existen multitud de definiciones acerca de los sistemas distribuidos, se toma dos de las definiciones más citadas y aceptadas:

"Un sistema distribuido es una colección de computadoras independientes que aparece ente los usuarios como una sola computadora". **(Tanembaum, 1996)**

Según Silberschatz, Galvin, & Gagane (2006)

> Un sistema distribuido es una colección de procesadores que no comparten ni memoria ni una señal de reloj. En lugar de ello, cada procesador tiene su propia memoria local. Los procesadores se comunican entre sí a través de varios tipos de líneas de comunicaciones, como buses de alta velocidad o líneas telefónicas. (p. 557)

Un sistema distribuido tiene como objetivo fundamental coordinar las actividades y compartir recursos como memoria, procesador y también periféricos como impresoras, scanner, cámaras, etc. entre los diversos equipos que conforman el sistema, cabe resaltar dos aspectos fundamentales al hablar de un sistema distribuido, uno de ellos es el **Hardware,** en el sentido en que los equipos son autónomos y trabajan independientemente, y el otro es el **Software** ya que si el sistema posee un buen diseño da la apariencia de que se trata de un solo ordenador.

Además los equipos que conforman parte del sistema distribuido pueden ser dispositivos de mano como equipos móviles, tabletas, también pueden estar conformadas por computadoras personales, estaciones de trabajo y grandes sistemas informáticos que incluyen servidores y otros clúster.

2.3.2. Estructura de un sistema distribuido

Silberschatz, Galvin, & Gagane (2006) conceptualizan un sistema distribuido de la siguiente forma:

> Un sistema distribuido es una colección de procesadores débilmente acoplados interconectados a través de una red de comunicaciones. Desde el punto de vista un procesador en concreto de un sistema distribuido, el resto de los procesadores y sus respectivos recursos son remotos, mientras que sus propios recursos son locales.
>
> Los procesadores de un sistema distribuido pueden variar en cuanto a tamaño y a función. Pueden incluir pequeños microprocesadores, estaciones de trabajo, minicomputadoras, y grandes sistemas informáticos de propósito general. Estos procesadores se designan mediante una serie de nombres, como *sitios, computadoras, máquinas y hosts,* dependiendo del contexto en el que se los mencione. Se utilizará principalmente *nodo* para indicar la ubicación de una máquina y host para referirnos a un sistema específico en un determinado *nodo,* generalmente uno de los hosts de un nodo, el *servidor,* dispone de un recurso de otro *host* en otro nodo, el cliente (o

usuario) desea utilizar. En la figura 17 se muestra la estructura general de un sistema distribuido (p.558).

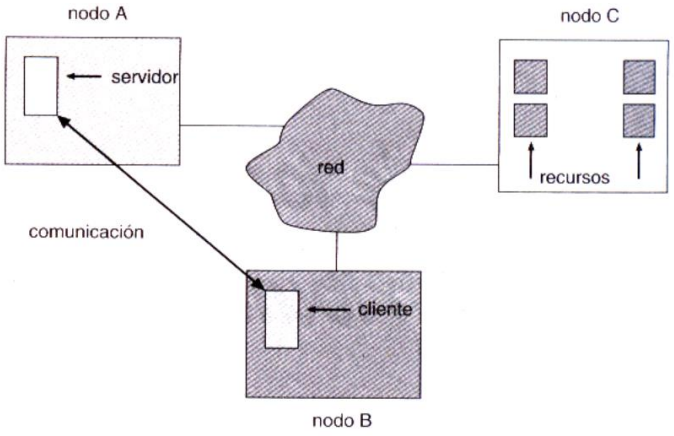


Figura 17. Estructura de un sistema distribuido

Fuente: Silberschatz, Galvin, & Gagane, Sistemas Operativos distribuidos, 2006

2.3.3. Razones para construir un sistema distribuido

"Hay cuatro razones principales para construir sistemas distribuidos: ***compartición de recursos, aceleración de cálculos, fiabilidad*** *y* ***comunicación"***. (Silberschatz, Galvin, Gagne, 2005, p.558)

- ✓ **Compartición de recursos**

Si hay varios nodos (con capacidades diferentes) conectados entre sí, un usuario en uno de los nodos puede utilizar los recursos disponibles del otro. Por ejemplo en la figura 6 un usuario en el nodo A podría estar utilizando una impresora ubicada en el nodo B. Mientras tanto, un usuario en B podría acceder a un archivo que residiera en A. En general, ***La compartición de recursos*** en un sistema distribuido proporciona mecanismos para compartir archivos en sitios remotos.

- ✓ **Aceleración de cálculos**

Si un determinado cálculo puede dividirse en una serie de cálculos de menor envergadura que puedan ejecutarse en forma *concurrente*, un sistema distribuido nos permitirá distribuir esos sub cálculos entre los diversos nodos; los sub cálculos podrán ejecutarse de manera concurrente, permitiendo obtener ***una aceleración de cálculos***. Además, si un nodo en concreto está actualmente sobrecargado de trabajo, parte de estos trabajos pueden desplazarse a otros sitios que estén experimentando una menor carga. Esta transferencia de trabajos se denomina *compartición de carga.* La compartición

automática de carga, en la que el sistema operativo distribuido transfiere automáticamente los trabajos, no resulta todavía común en los sistemas comerciales.

- ✓ **Fiabilidad**

Si falla uno de los nodos de un sistema distribuido, los nodos restantes pueden continuar operando, proporcionando al sistema mayor fiabilidad. Si el sistema está compuesto de múltiples instalaciones autónomas de gran tamaño(es decir computadoras de propósito general), el fallo de una de ellas no deberá afectar a las restantes. Sin embargo, si el sistema está compuesto de máquinas pequeñas, cada una de las cuales es responsable de alguna función crucial del sistema, el único fallo puede detener la operación de todo el sistema, en general si existe suficiente redundancia (tanto en hardware como en datos), el sistema podrá continuar operando incluso aunque fallen todos los nodos. El fallo de un nodo debe ser detectado por el sistema y puede que sea necesario llevar a cabo las acciones necesarias para recuperarse del fallo. El sistema no debe continuar utilizando los servicios de ese nodo. Además, la función del nodo fallido puede ser asumida por otro nodo, el sistema debe garantizar que la transparencia de esa función se produzca correctamente. Finalmente, cuando se recupere o repare el nodo fallido, deben existir mecanismos para volver a integrarlo con suavidad en el sistema.

- ✓ **Comunicación**

Cuando hay varios sitios conectados entre sí mediante una red de comunicaciones, los usuarios de los diferentes nodos tienen la oportunidad de intercambiar información, en los sistemas distribuidos la comunicación puede llevarse a cabo a través de grandes distancias. Dos personas situadas en nodos geográficamente diferentes pueden, por ejemplo, colaborar en un proyecto. Transfiriendo los archivos del proyecto, conectándose a los sistemas remotos de otra persona para ejecutar programas e intercambiando mensajes de correo electrónico para coordinar el trabajo, los usuarios minimizan las limitaciones inherentes al trabajo a distancia. Las ventajas de los sistemas distribuidos han hecho que todo el sector se vea sometido a una tendencia hacia la *reducción de escala (downsizing)*. Muchas empresas están sustituyendo sus computadoras del tipo *mainframe (Computadoras de capacidades de cálculo sorprendentes para procesamiento de datos especiales)* por redes de estaciones de trabajo o computadoras personales. Las empresas obtienen así una mayor funcionalidad por un mismo coste, más flexibilidad a la hora de ubicar los recursos y expandir las instalaciones (escalabilidad), mejores interfaces de usuarios y mantenimiento mucho más sencillo.

2.3.4. Características de los sistemas distribuidos

Un sistema distribuido posee las siguientes características (Tanembaum, 1996)

- ✓ ***Transparencia:*** Una de las principales metas de un sistema distribuido es la transparencia, es decir la manera de hacer que las múltiples computadoras interconectadas que forman un sistema distribuido aparezcan ante los usuarios como si se tratase de una sola máquina, un sistema que logre este objetivo se le conoce como transparente.

- ✓ ***Heterogeneidad:*** Debido a que las computadoras constituyen un sistema distribuido pueden ser máquinas heterogéneas, es decir computadoras de diferentes arquitecturas.

- ✓ ***Flexibilidad:*** La flexibilidad es una de las características más importantes tanto en los sistemas distribuidos como en otros sistemas computacionales ya que por características propias un software nunca es perfecto. Por tal motivo existen dos razones principales para que un software sea flexible: ***que pueda modificarse fácilmente*** causando los mínimos cambios posibles en todo el sistema ya que en ocasiones se encuentran errores (bugs) en el diseño o los usuarios tienen nuevos requerimientos, q**ue pueda mejorarse,** para hacerlo cada vez más robusto, más fácil de usar y la forma de hacerlo es mantener abiertas las opciones de poder corregir.

- ✓ ***Confiabilidad:*** Uno de los objetivos originales de la construcción de sistemas distribuidos es hacerlos más confiables que los sistemas con un procesador, debido a la vulnerabilidad que existe entre las diferentes máquinas remotas que forman el sistema distribuido. Un sistema ampliamente confiable debe ser siempre muy disponible, debido a que los datos confiados al sistema no deben perderse o mezclarse de manera alguna. Otro aspecto importante en un sistema confiable, es la seguridad en los archivos y otros recursos que deben ser protegidos contra el uso no autorizado.

- ✓ ***Desempeño:*** La construcción de un sistema distribuido transparente, flexible y confiable no será bueno si es muy lento. En particular, cuando se ejecuta una aplicación en un sistema distribuido no debe aparecer peor que su ejecución en un sistema centralizado, al menos debe ser igual.

  La métrica estándar para medir el desempeño de un sistema es el tiempo de respuesta mediante la ejecución de un determinado número de trabajos y la capacidad de la red consumida.

  Una de las desventajas de un sistema distribuido con respecto a un sistema de un solo procesador, es el tiempo consumido en el envío y recepción de mensajes a través de la red entre las diferentes máquinas.

En particular, los trabajos que necesitan de un gran número de pequeños cálculos y que necesitan en gran medida interactuar con otros, en general son la causa de algunos problemas en los sistemas distribuidos con una comunicación lenta.

- ✓ ***Escalabilidad:*** En un sistema distribuido, la escalabilidad se refiere a la capacidad del sistema para adaptarse a la creciente demanda de conectar más computadoras si así se requiere. La escalabilidad es algo natural e inevitable, debido a que las necesidades de los usuarios crecen con el paso del tiempo, por eso al diseñar un sistema distribuido se debe tomar en cuenta este aspecto para evitar problemas de comunicación o pérdida de información.

- ✓ ***Tolerancia a fallas:*** Otra de las características más importantes de los sistemas distribuidos es la capacidad para tolerar fallas en los procesadores individuales y continuar trabajando.

  Cuando un sistema distribuido es tolerante a fallas significa que puede seguir trabajando aun cuando fallen algunas partes del sistema.

  Los diseños tolerantes a fallas son importantes sobre todo en *sistemas críticos* en los que quizá no puedan intervenir los seres humanos para solucionar los problemas, también en sistemas en donde las consecuencias de las fallas son demasiado graves que podrían sobrevenir tan rápido que los seres humanos no podrían intervenir a tiempo.

  Existen algunas técnicas para fomentar la tolerancia a fallas:

  - Se deben mantener varias copias de los datos críticos para el sistema y para los diferentes procesos.

  - El sistema operativo debe estar diseñando de forma tal que pueda trabajar de manera eficiente con la configuración máxima del hardware y con subconjuntos del hardware cuando ocurran fallas.

  - Las funciones de detección y corrección de errores en el hardware deben estar implantadas de tal forma que se efectúen comprobaciones exhaustivas.

  - Se debe aprovechar la capacidad de los procesadores ociosos para tratar de detectar fallas potenciales antes que ocurran. (p. 22-31).

2.3.5. Ventajas y desventajas de un sistema distribuido

Las principales ventajas de los sistemas operativos distribuidos con respecto a los sistemas centralizados son: ***economía***, ***velocidad***, ***distribución inherente***, ***confiabilidad*** y ***crecimiento por incrementos***.

En los sistemas distribuidos los datos no son lo único que se puede compartir, también se comparten diferentes dispositivos periféricos, con todo ello se puede mencionar las siguientes ventajas sobre las computadoras independientes: ***datos compartidos***, ***dispositivos compartidos***, ***comunicación y flexibilidad.***

En contraparte los sistemas distribuidos también posen desventajas, algunas de ellas son las siguientes:

- ✓ ***El software:*** En el sentido de que hay cosas aun por investigar como los lenguajes apropiados para crear un sistema operativo distribuido, funciones específicas de un usuario y del sistema distribuido en sí.

- ✓ ***Las redes:*** dado que un sistema distribuido está conformado por computadoras interconectadas por una red, puede darse el caso de que la red esté saturada , entonces cuando un sistema operativo distribuido llega a depender de las características físicas de la red, la pérdida o saturación de mensajes puede negar algunas de sus ventajas.

- ✓ ***La seguridad:*** otro de los problemas que poseen presencia es la seguridad de los datos en el sentido de que hay mucha facilidad de compartición de recursos en un sistema distribuido y por ende existen permisos de cualquier parte del clúster.

2.3.6. Hardware de un sistema distribuido

Ya que todos los sistemas distribuidos constan de varios computadores, existen diversas formas de organizarlas, en particular, la forma de interconectarlas y comunicarlas entre sí.

A través de los años se han propuesto diversas taxonomías de clasificación para sistemas distribuidos. Aunque ninguna ha sido adoptada de manera amplia, la taxonomía más citada y aceptada es la de Flynn.

Flynn (1972) escogió dos características consideradas esenciales ***el número de flujo de instrucciones*** *y* ***el número de flujo de datos*** (Deitel, 1987, p. 22)***.*** De este modo se tiene, cuatro tipos de arquitecturas como se muestra en la figura 18.

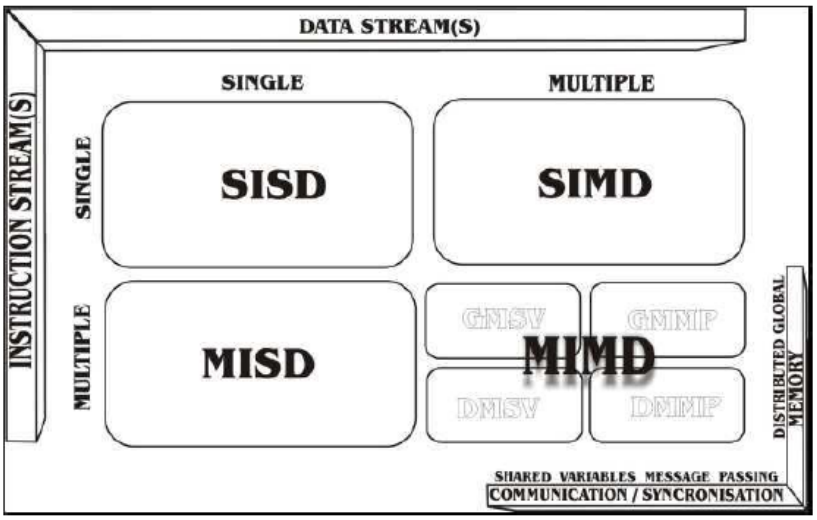


Figura 18. Taxonomía de Flynn

Fuente: Deitel, 1987

- ✓ **Arquitectura SISD** (*Single Instruction, Single Data*) es el modelo tradicional de computadora secuencial donde la unidad de procesamiento recibe una sola secuencia de instrucciones que opera en una secuencia de datos, estos sistemas no trabajan de forma paralela y constan de un solo procesador como se muestra en la figura 19.

Figura 19. Modelo SISD (Single Instruction Single Data)

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

- ✓ **Arquitectura SIMD** (*Single Instruction, Multiple Data)*

En este modelo se tienen múltiples procesadores que sincronizadamente ejecutan la misma secuencia de instrucciones, pero diferentes datos como se observa en la figura 20, el tipo de memoria que éstos sistemas utilizan es distribuida, por lo que diferentes datos pueden ser utilizados en cada procesador, las máquinas con arreglos de procesadores como CRAY1 y CRAY2 son ejemplos de arquitecturas SIMD.

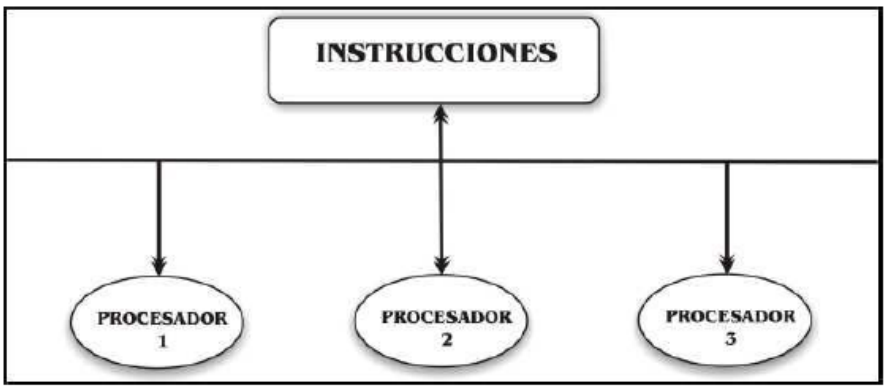


Figura 20. Modelo SIMD (Single Instruction Multiple Data)

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

- ✓ **Arquitectura MISD** (*Multiple Instruction, Single Data*) éste modelo no posee una aplicación real, en este modelo secuencias de instrucciones pasan a través de múltiples procesadores, diferentes operaciones son realizadas en diversos procesadores como se muestra en la figura 21.

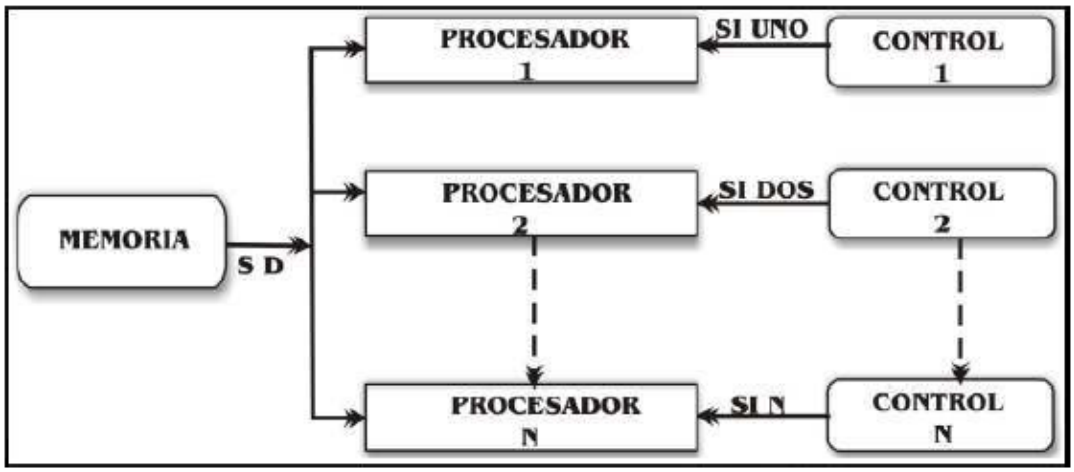


Figura 21. Modelo MISD (Multiple Instruction Single Data)

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

En este modelo el paralelismo es alcanzado dejando que los procesadores realicen cosas al mismo tiempo en el mismo dato.

- ✓ **Arquitectura MIMD** (*Multiple Instruction, Multiple Data*) En este modelo se hallan aquellas computadoras paralelas al igual que el modelo SIMD, la diferencia en estos sistemas es que MIMD es asíncrono: es decir no tiene un reloj central, se muestra la estructura del modelo MIMD en la figura 22.

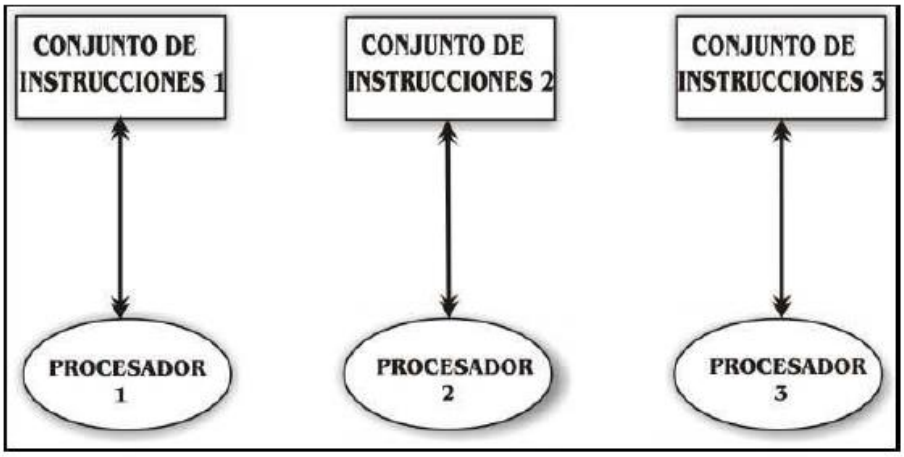


Figura 22. Modelo MIMD (Multiple Instruction Multiple Data)

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

Cada procesador en un sistema MIMD puede ejecutar su propia secuencia de instrucciones y tener sus propios datos: esta característica es la más general y más poderosa de las arquitecturas expuestas anteriormente, además cada procesador es capaz de ejecutar su propio programa con diferentes datos, esto significa que los procesadores operan asíncronamente, o en términos simples, pueden estar haciendo diferentes cosas en diferentes datos al mismo tiempo, los sistemas MIMD también poseen una subclasificación: ***los sistemas de memoria compartida*** y ***los sistemas de memoria distribuida***.

- ✓ **Sistemas de memoria compartida**

  En este tipo de sistemas cada procesador tiene acceso a toda la memoria, es decir, hay un espacio de direccionamiento compartido como se muestra en la figura 23, Con esto se tiene tiempo de acceso a memorias uniformes, ya que todos los procesadores se encuentran igualmente comunicados con memoria principal, además el acceso a memoria es por medio de un solo canal. En esta configuración debe asegurarse que los procesadores no tengan acceso simultáneamente a regiones de memoria de una manera en la que pueda ocurrir algún error.

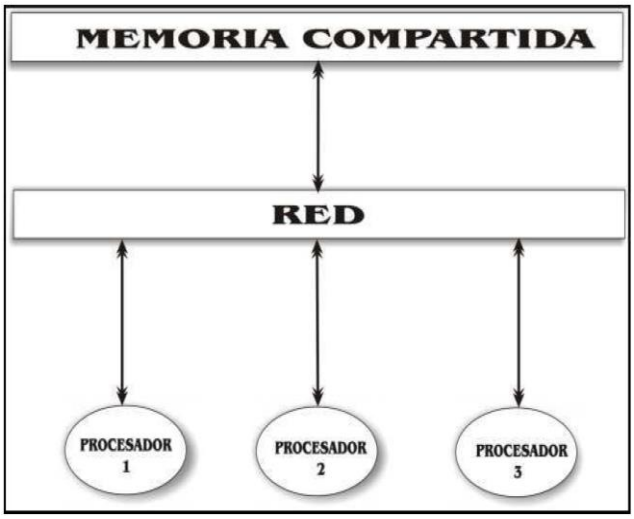


Figura 23 - Sistemas de memoria compartida

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

Este submodelo posee las siguientes ventajas.

- Facilidad de la programación. Es mucho más fácil programar en estos sistemas que en sistemas de memoria distribuida.
- Las computadoras MIMD con memoria compartida son sistemas conocidos como de multiprocesamiento simétrico (SMP), donde múltiples procesadores comparten un mismo sistema operativo y memoria. Otro término con el que se le conoce es máquinas firmemente juntas o de multiprocesadores.

Por otro lado este modelo posee las siguientes desventajas

- Poca escalabilidad de procesadores, debido a que se puede generar un cuello de botella en el número de CPUs.
- En computadoras vectoriales como CRAY, todas las CPU tienen un camino libre a la memoria. No hay diferencia entre las CPU.
- La razón principal de este modelo es el alto precio de la memoria.

Se puede citar el siguiente conjunto de sistemas que poseen el modelo de sistemas de memoria compartida: **SGI/CRAY Power Challenge, SGI/CRAY C90, SGI/Onyx, ENCORE, MULTIMAX, SEQUENT y BALANCE**, entre otras.

✓ **Sistemas de memoria distribuida**

Estos sistemas tienen su propia memoria local. Los procesadores pueden compartir información solamente enviando mensajes, es decir si un procesador requiere los datos contenidos en la memoria de otro procesador, deberá enviar un mensaje solicitándolos, a esta comunicación se le conoce como **paso de mensajes,** se muestra la estructura de los sistemas de memoria distribuida en la figura 24.

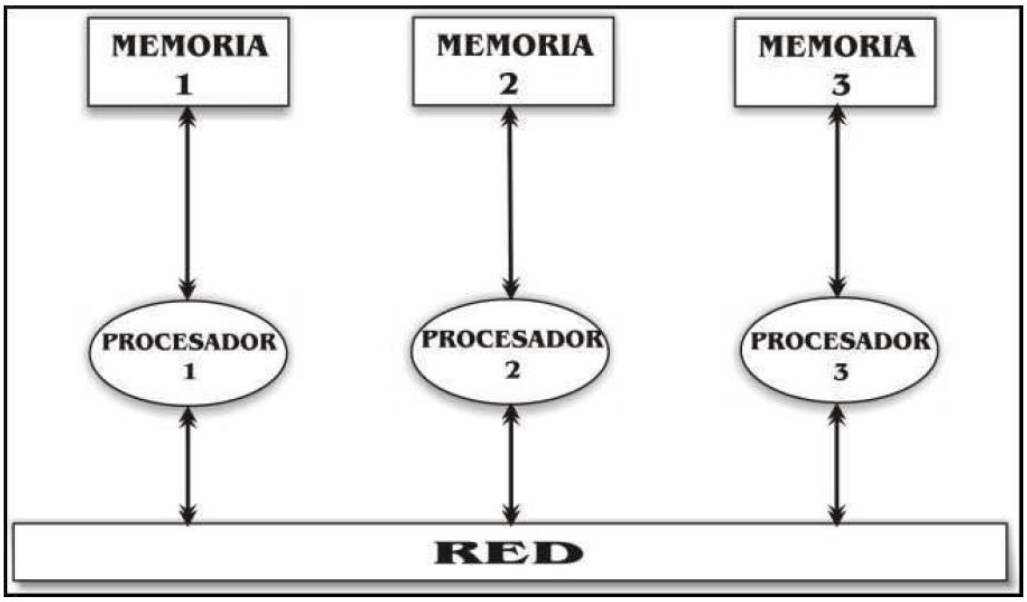


Figura 24. Sistema de memoria distribuida

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

Los sistemas de memoria distribuida poseen notablemente la siguiente ventaja:

- La **escalabilidad**. Las computadoras con sistemas de memoria distribuida son fáciles de escalar, debido a la demanda de los recursos se pueden agregar más memoria y procesador.

Las desventajas de los sistemas de memoria distribuida son las siguientes:

- El acceso remoto a memoria es lento.
- La programación en ocasiones puede ser complicada.

Las computadoras MIMD de memoria distribuida son conocidas como **sistemas de procesamiento en paralelo (MPP),** donde múltiples procesadores trabajan en diferentes partes de un programa, usando su propio sistema y memoria. *También se les llama multicomputadoras, máquinas libremente juntas o* ***clúster.***

2.3.7. Sistemas fuertemente acoplados y débilmente acoplados

"Los sistemas con respecto al hardware pueden ser ***fuertemente acoplados*** *y* ***débilmente acoplados."*** (Deitel, 1987)***.***

En los sistemas fuertemente acoplados se utiliza un almacenamiento único compartido por los diferentes procesadores y un solo sistema operativo que controla todos los procesadores y el hardware del sistema, como se muestra en la figura 25, La comunicación se realiza mediante memoria compartida y el retraso que se experimenta al enviar un mensaje en la computadora a otra es corto y la tasa de transmisión de los datos, es decir el número de bits que se pueden transferir por segundo es grande.

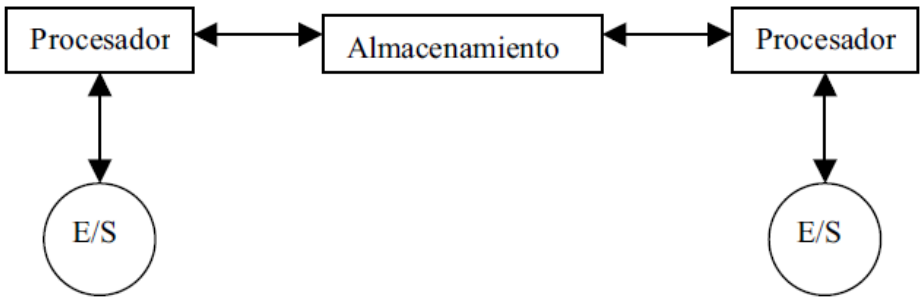


Figura 25. Hardware en un sistema fuertemente acoplado

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

Un sistema débilmente acoplado consiste en conectar dos o más computadoras independientes mediante una red, como se muestra en la figura 26, cada computadora tiene su propio sistema operativo, su propia memoria y almacenamiento, además que pueden funcionar de manera independiente y pueden comunicarse entre sí. También pueden tener acceso a los archivos a otras computadoras por medio de la red. La comunicación entre los procesadores se realiza por medio de ***transferencia de mensajes*** o de llamadas a ***procedimientos remotos.*** El retraso de máquinas es grade y la tasa de transmisión de datos es baja.

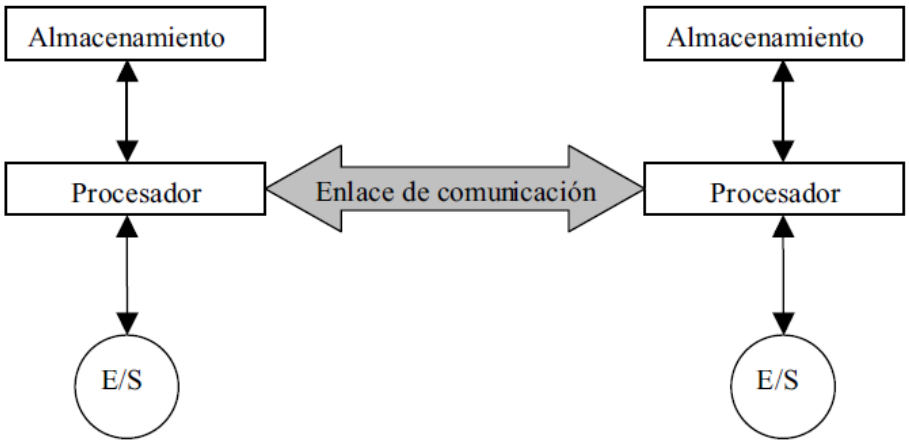


Figura 26. Hardware en un sistema débilmente acoplado

Fuente: Deitel, Introducción a los Sistemas Operativos, 1987

Los sistemas fuertemente acoplados tienden a utilizarse más como sistemas paralelos (para trabajar en un problema en específico) y los débilmente acoplados tienden a utilizarse como sistemas distribuidos (para trabajar con varios problemas relacionados o no relacionados entre sí).

En general los sistemas multiprocesadores tienden a estar más fuertemente acoplados que las multicomputadoras, puesto que comparten la memoria, un reloj global y sus tiempos de acceso a los datos son similares para todos sus procesadores. Por lo tanto, el intercambio de datos se realiza a una velocidad semejante a la velocidad de acceso a memoria, aunque algunas multicomputadoras interconectadas mediante fibra óptica pueden funcionar también con velocidad aproximada a sus memorias, la ejecución de algunas tareas puede ser más tardado, debido a que no se comparte memoria ni reloj y cada computadora cuenta con su memoria local. Los términos ***débilmente acoplados*** y ***fuertemente acoplados***, no son estrictamente apropiados pero si muy útiles área para entender el concepto que se quiere dar a conocer.

2.3.8. Organización simétrica

Aunque el multiprocesamiento simétrico suele ser la organización más compleja al llevarla a la práctica, suele ser muy poderosa. Todos los procesadores son idénticos y el sistema operativo puede utilizar cualquiera de ellos para controlar cualquier dispositivo de entrada o salida o para hacer referencia a cualquier unidad de almacenamiento.

Los sistemas de multiprocesamiento simétrico generalmente son los más confiables ya que si existe alguna falla en un procesador, hace que el sistema operativo expulse el procesador

de su conjunto de procesadores disponibles y tome nota del problema, el sistema puede seguir funcionando aunque a niveles menores.

2.4. Clúster de computadoras

2.4.1. Historia

La historia de los clúster de computadoras comenzó cuando Donald Becker y Thomas Sterling (1994) construyeron un clúster para la NASA, su nombre fue Beowulf con la finalidad de resolver problemas que aparecen en las ciencias de la tierra y del espacio.
Posteriormente en el año 1996, se construye otro clúster llamado *Hyglac* desarrollado por el Instituto Tecnológico de California (Cal Tech) y el Laboratorio de Propulsión Jet (JPL), el otro clúster es, *Loki* construido en el Laboratorio Nacional de los Álamos.

A partir de este proyecto, han surgido numerosas iniciativas en este sentido. Estos clúster se utilizan para cualquier tarea que requiera enormes cantidades de cómputo.
En la actualidad, los clúster han tenido mucho auge en centros de investigación y en las empresas, debido a que ciertos problemas que se deseaban resolver rebasaban la capacidad de cómputo de una computadora personal. La solución más obvia para resolver estos problemas pues sería comprar una supercomputadora, pero existe un problema a esta solución, debido a que una supercomputadora en ocasiones cuesta varios miles de dólares, cantidad que a veces va más allá de los presupuestos de inversión tanto de las empresas como de los centros de investigación.

Pero la necesidad de solucionar los problemas con los recursos que se cuentan, han hecho que personal académico de diversas universidades y centros de investigación se han a la tarea de construir sus propias supercomputadoras conectado computadoras personales y desarrollando software para resolver dichos problemas.

2.4.2. Definición de clúster

Existen varias definiciones de clúster las cuales se asemejan mucho, se tomará como referencia tres definiciones de distintos autores.

Según Branch Bedoya & Mesa Munera (2008)

> Un clúster se puede definir como un grupo de múltiples computadores unidos por una red de alta velocidad de tal forma que el conjunto puede ser visto como una única máquina, pero que por su poder de cómputo resuelve problemas que un solo equipo de escritorio no podría hacer(párr. 8).

Según Sánchez & Heider (2007)

> Un clúster es un conjunto de computadoras interconectadas con dispositivos de alta velocidad que actúan en conjunto usando el poder cómputo de varios CPUs en combinación para resolver ciertos problemas dados.
>
> Se usa un clúster para crear una supercomputadora que puede servir como un servidor en un sistema, reduciéndose el costo de inversión (p. 5)

Según Sterling, Salmon & Becker (1999)

> Un clúster es la interconexión de dos o más computadoras independientes a través de una red, usadas como un recurso unificado de cómputo con el fin de aumentar el rendimiento en la ejecución de tareas.

Analizando las definiciones mencionadas anteriormente un clúster surge a necesidad de resolver problemas complejos pero que sí los puede realizar un equipo de cómputo en un lapso de tiempo muy largo, claro dependiendo de la complejidad del trabajo encomendado.

2.4.3. Estructura de un clúster

Un clúster es la interconexión de dos o más máquinas basadas en sistemas distribuidos que son débilmente acoplados a través de redes de alta velocidad y que en este caso es un sistema débilmente acoplado como se muestra en la siguiente figura 27 (Buyya, 1999).

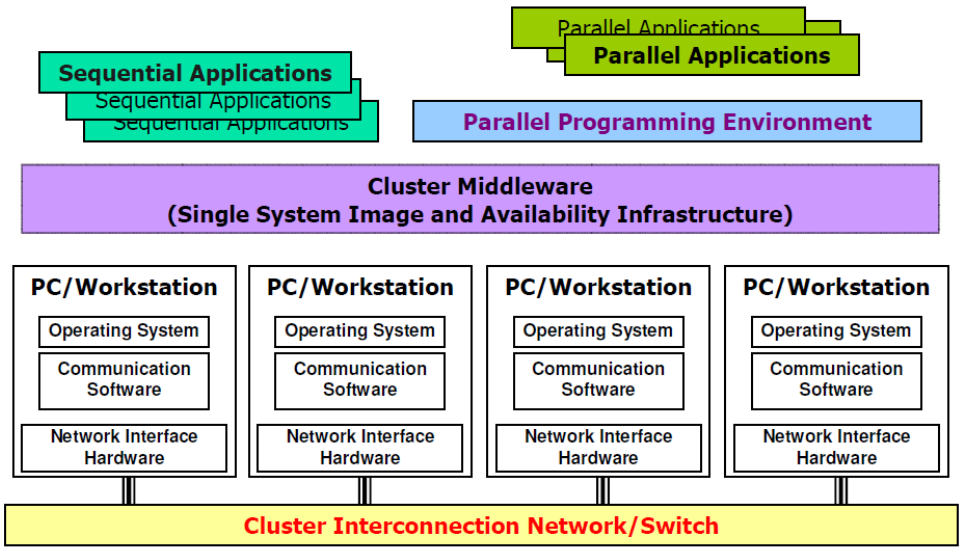


Figura 27. Estructura de un clúster

Fuente: High Performance Clúster Computing: Architectures and Systems - Dept. of Computer Science and Software Enginer , 2009

Según la figura 16 se puede observar que cada nodo del clúster es decir computadoras y/o estaciones de trabajo (***PC/Workstation***) están conectadas a una red de alta velocidad conmutada por un **switch** u otro dispositivo de mayor prestaciones, un dispositivo que permite transmitir información entre equipos de cómputo, cada nodo del clúster posee su propio sistema operativo (***Operating System***) como puede ser Windows o GNU/Linux, su software de comunicación (***communication software***) con los protocolos TCP/IP que viene a ser un conjunto de reglas para asegurar en envío y recepción de información entre equipos y su propia interfaz(***Network interface Hardware***) de red para realizar las transmisión y recepción de datos con adaptadores de red.

Se puede visualizar también que sobre las estaciones de trabajo se halla el ***Middleware*** del clúster esto nos permite trabajar con el clúster como una sola imagen del sistema e infraestructura.

Pfister (1998) afirma que sobre este nivel se halla los entornos de programación paralela sobre las cuales se pueden desarrollar aplicaciones o programas especiales en entorno paralelo.

Un clúster de computadoras puede utilizar diversas ***tecnologías de interconexión y software de comunicación,*** seleccionar una tecnología de interconexión de red depende de diversos factores, la compatibilidad con el hardware del clúster así como el sistema operativo y por ende el rendimiento, hay dos métricas para medir el rendimiento para interconexiones. El ***ancho de banda*** y la ***latencia,*** el ancho de banda está definido como la cantidad de datos que se puede transmitir a través del hardware de red en un determinado tiempo el cual se mide en MBytes/s, mientras que latencia es el tiempo para preparar y trasmitir datos de un nodo origen a un nodo destino que se mide en μs.

- ✓ **Tecnologías de Interconexión**

  En la tabla 3 se muestra un resumen y su respectiva descripción de algunas tecnologías de interconexión.

Tabla 3. Tecnologías de interconexión

Tecnologías de interconexión	Descripción
Gigabit Ethernet	Proporciona razonablemente alto ancho de banda dado su bajo precio, pero sufre relativamente alta latencia, restringiendo así Gigabit Ethernet como una buena opción. Sin embargo, el bajo precio de Gigabit Ethernet es atractivo para los grupos de construcción.
Giganet cLAN	Giganet cLAN se desarrolla con el objetivo de apoyar mediante en hardware y una baja latencia. Sin embargo, sólo proporciona un ancho de banda bajo de menos de 125 MBytes/s, por lo que no es una opción viable para la aplicación de redes de clúster rápidos.
Infiniband	El último estándar de la industria basada en VIA conceptos y puesto en libertad en 2002, Infiniband admite la conexión de diversos componentes del sistema dentro de un sistema tal como inter procesadores, redes, subsistemas de entrada y salida o conmutadores de red multi-protocolo. Esto hace que Infiniband sea independiente de cualquier tecnología en particular.
Mirynet	La corriente que más se utiliza para redes de clúster rápidos. La principal ventaja de Myrinet es que opera en el espacio de usuario, evitando así las interferencias del sistema operativo y retrasos.
QSNet II	La versión de próxima generación de QsNet que se basa en el alto rendimiento PCI-X interfaz, en comparación con la interfaz PCI de QsNet. QsNetII es capaz de lograr 1,064 Mbytes/s, apoyar a 4.096 nodos, y proporcionar la arquitectura de direcciones virtuales de 64 bits.
Scalable Coherent Interface (SCI)	El primer estándar de la tecnología de interconexión especificada para la computación clúster. SCI define un esquema de caché basada en directorios que se pueden mantener los depósitos de conexión procesadores coherentes, por lo tanto capaces de poner en práctica la memoria compartida virtual.

Fuente: High Performance Clúster Computing: Architectures and Systems - Dept. of Computer Science and Software Enginer , 2009

Como anteriormente se mencionó hay dos métricas primordiales para medir el rendimiento de las conexiones de red: el ancho de banda y latencia, adicionalmente en la tabla 4 se examina la comparación de algunos factores de las tecnologías entes mencionadas como son: ancho de banda, latencia, disponibilidad de hardware, soporte para el sistema operativo GNU/Linux, el número máximo de nodos soportados por el clúster, implementación de los protocolos, soporte para la arquitectura de interfaz virtual (VIA) y soporte para interface de paso de mensajes (MPI), se explica a continuación los conceptos de VIA y MPI.

La Arquitectura de Interfaz Virtual

"(VIA) es un estándar de interface de software de comunicación de baja latencia esto fue desarrollado por un consorcio de productores de hardware e instituciones académicas y fue adoptado para uso de muchos clúster" (Cameron & Regnier, 2002)

La interface de paso de mensajes (MPI)

"Es un conjunto de librerías que pueden utilizarse para desarrollar aplicaciones paralelas y distribuidas para sistemas distribuidos, MPI provee una capa de comunicación y portabilidad de código de aplicaciones entre las diversas plataformas de sistemas distribuidos".

(Sun , 2006)

Tabla 4. Comparación entre tecnologías de interconexión

Comparación Criterio	Gigabit Ethernet	Giganet cLAN	Infiniband	Mirynet	QSNet II	Scalable Coherent Interface (SCI)
Ancho de banda (MBytes/s)	Menor a 100	Menor a 125	850	230	1064	Menor a 320
Latencia (µs)	Menor a 100	De 7 a 10	Menor a 7	10	Menor a 3	De 1 a 2
Disponibilidad Hardware	Sí	Sí	Sí	Sí	Sí	Sí
Soporte para Linux	Sí	Sí	Sí	Sí	Sí	Sí
Número máximo de nodos	1000's	1000's	Mayor a 1000's	1000's	4096	1000's
Implementación de protocolos	Hardware	Firmware en adaptador	Hardware	Firmware en adaptador	Firmware en adaptador	Firmware en adaptador
Soporte para Arquitectura de Interfaz Virtual (VIA)	NT/Linux	NT/Linux	Software	Linux	Ninguno	Software
Soporte para interface de paso de mensajes (MPI)	MVICH over M-VIA, TCP	3rd Party	MPI/Pro	3rd Party	Quadrics	3rd Party

Fuente: High Performance Clúster Computing: Architectures and Systems - Dept. of Computer Science and Software Enginer, 2009

La tabla 4 muestra los diversos factores de las tecnologías de interconexión (Baker, Apon, Buyya, & Jin, 2002, pp. 87-125).

Se han realizado varias evaluaciones de desempeño de los sistemas en clúster y de las interconexiones específicas según las siguientes referencias:

Banikazemi, Liu, Panda, & Sadayappan(2001); Lan, & Deshikachar(2003); Vander Steen, (2003); Chen, Wyckoff, & Moor (2000); Hsieh, Leng, Mashayekhi, & Rooholamini,(2000).

Por desgracia, gran parte del trabajo realizado de alguna manera se ha "comparando manzanas con naranjas". Esto se debe a que los análisis de rendimiento más actuales se han visto obligados a comparar los resultados de interconexiones que se utiliza en diferentes sistemas de clúster (ya que cualquier grupo determinado rara vez tiene más de uno o dos interconexiones disponibles).

Un subconjunto de los autores (Pourreza, Eskicioglu y Graham) ha llevado a cabo evaluaciones de desempeño de un número de interconexiones identificado en la Tabla 3, tomando tiempos cuando se ejecutan aplicaciones idénticas en nodos del clúster idénticos. Repetidas pruebas aisladas de una serie de puntos de referencia estándar de computación en clúster (Incluidos los puntos de referencia paralelas NAS y las referencias Pallas), así como algunos del mundo real aplicaciones paralelas en la primera y segunda generación Myrinet, SCI, así como Fast (100 Mbps) y gigabit (1000 Mbps) Ethernet.

Pourreza, Eskicioglu, Graham (2004) hicieron pruebas con las tecnologías de interconexión y los resultados principales se resumen a continuación como se muestran en la figura 28.

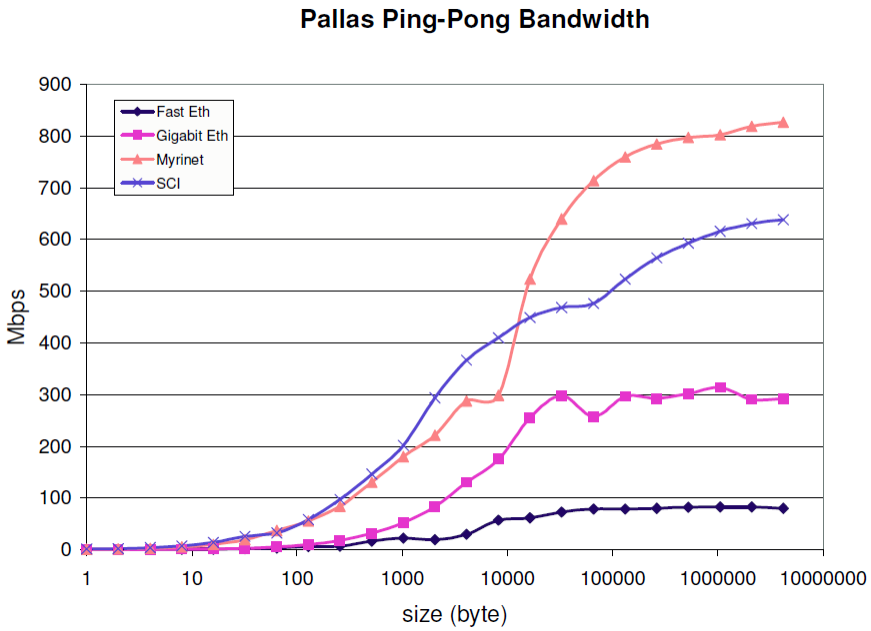


Figura 28. Comparativa entre tecnologías de interconexión en ancho de banda

Fuente: High Performance Clúster Computing: Architectures and Systems - Dept. of Computer Science and Software Enginer, 2009

El rendimiento de las diversas interconexiones (en términos de ancho de banda y latencia) se muestran en las figuras 28 y 29 respectivamente. Las cifras relativas de las cuatro redes son los esperados: con Fast Ethernet siendo claramente inferior a la de los demás interconectas y Gigabit Ethernet siendo notablemente por debajo de SCI y Myrinet a pesar de la publicidad de un ancho de banda de forma similar. Incluso a partir de estos resultados.

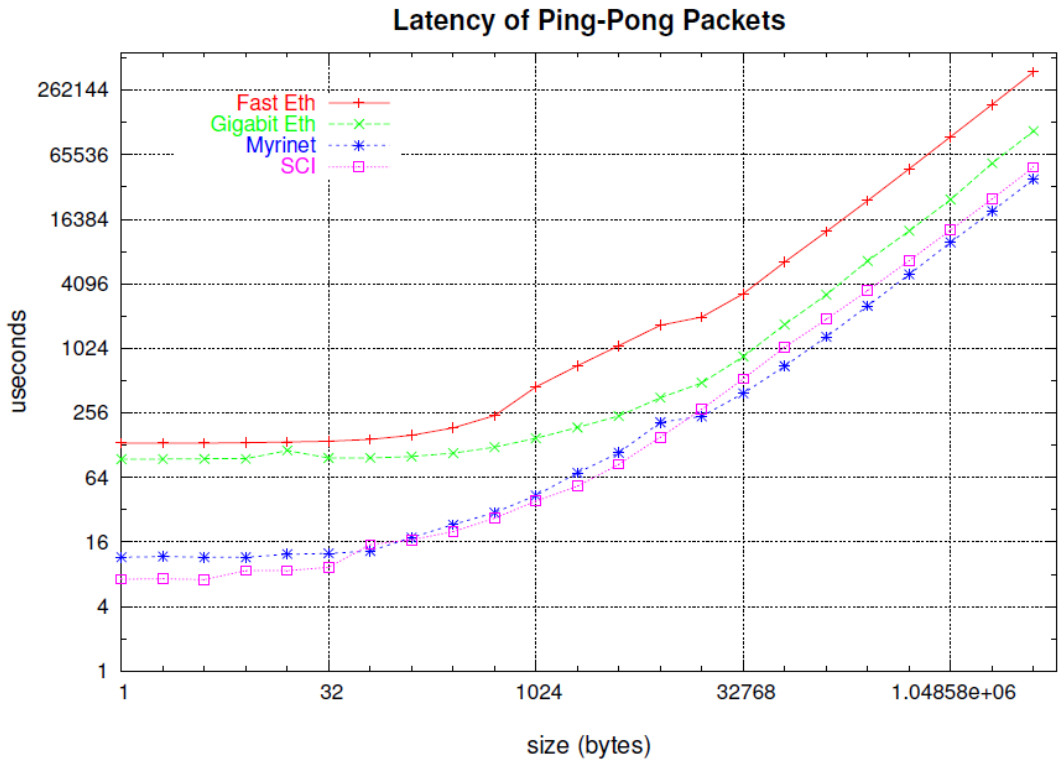


Figura 29. Comparativa entre tecnologías de interconexión en latencia

Fuente: High Performance Clúster Computing: Architectures and Systems - Dept. of Computer Science and Software Enginer, 2009

Para los clúster de cómputo de pequeña escala, la mayoría de los constructores de clúster por primera vez tienden a elegir entre un bajo costo e interconexión de los productos básicos (Fast Ethernet o Gigabit Ethernet) o una interconexión más cara pero más alta en rendimiento (como Myrinet o SCI). En muchos casos, esta elección se hace antes de cualquier investigación de la interconexión, necesidades de las aplicaciones que se ejecutan en el clúster y es a menudo determinado simplemente haciendo la pregunta: "¿Puede nuestro presupuesto permitirse una interconexión rápida?" Este enfoque no es el deseable y puede conducir a la frustración con el rendimiento del clúster resultante debido a dos razones posibles.

- ✓ Una posible razón es que la interconexión de la red seleccionada no sea adecuada para el trabajo a realizar.
- ✓ Otra posible razón es que el dinero gastado en una interconexión podría haber sido mejor gastado en otros componentes útiles, tales como procesadores más rápidos y de memoria más grande.

Los constructores de clúster también podrían fácilmente evitar estos problemas siguiendo algunas pautas simples y teniendo en cuenta el rendimiento y características de las diversas interconexiones según lo determinado por análisis independientes.

En general, el costo de los productos básicos de interconexión es menor que el costo de interconexiones de alto rendimiento. La principal diferencia entre los productos básicos e interconexiones de alto rendimiento es la latencia de envío de mensajes y el ancho de banda disponible para la mensajería. Esto significa, naturalmente, que interconexiones de alto rendimiento son más deseables cuando las aplicaciones que se ejecutan sobre ellos se comunican con frecuencia (sobre todo si se intercambian muchos mensajes pequeños). Las aplicaciones que se comunican con poca frecuencia y que intercambian grandes mensajes suelen realizar muy bien el uso de los productos básicos de interconexiones. El costo de Fast Ethernet (100Mbps) es ahora tan baja que es una estrategia común entre muchos constructores de clúster comenzando por utilizar Fast Ethernet para su interconexión y luego actualizar cuando sea necesario (o cuando tengan una mejor idea de sus características de aplicación y por lo tanto los requisitos). Dado el bajo coste de Fast Ethernet, hay poca preocupación si el equipo se utiliza sólo durante un corto período de tiempo.

Gigabit Ethernet es también tratada de manera similar a pesar de que es más caro. Sin embargo, las ventajas de Gigabit Ethernet son una vida útil más larga como una interconexión activa y la capacidad de soportar una gama mucho más amplia de aplicaciones

2.4.4. Características de un clúster

"Un clúster es un sistema distribuido por lo tanto posee sus características pero adicionalmente proporciona tres beneficios principales: alta disponibilidad, buena escalabilidad y bajo costo". **(Sun , 2006)**

- ✓ **Alta disponibilidad**

  La disponibilidad es la capacidad de que un clúster pueda permanecer funcionando ante diferentes fallas que pudieran existir mientras se encuentra trabajando. Por ejemplo una falla podría ser un procesador o equipo de cómputo estropeado, una falla de disco duro, alguna falla en la conexión de red o falla de energía eléctrica en algún nodo, entonces si un nodo falla los demás nodos captan toda la carga de trabajo.

Para muchas aplicaciones de empresas, el clúster es una buena solución, debido a que el tiempo de recuperación es aceptable. Frecuentemente se programan los tiempos para realizar otras aplicaciones en algún nodo del clúster.

- ✓ **Buena escalabilidad**

  La escalabilidad es la capacidad de un sistema o sistemas de poder crecer de acuerdo con las necesidades de los usuarios. Los clúster son escalables ya que se les puede agregar o sustraer componentes a la computadora, por ejemplo, agregar memoria, discos duros, tarjetas de red, procesadores o agregar otra computadora.

  Las arquitecturas de los clúster pueden ser escalables de dos maneras. La primera, los diseñadores seleccionan las computadoras que formarán los nodos del clúster, para que de esta forma puedan ser escalables, aumentándoles memoria, disco duro, etc. La segunda, los diseñadores aumentan el número de computadoras que forman parte del clúster.

- ✓ **Bajo costo**

  Una arquitectura de clúster es relativamente bajo en costo debido a que un clúster utiliza en el mayor de los casos software libre y gratuito, de esta manera las empresas no tendrían que hacer gasto extra para comprar software, también es importante señalar y especificar que se puede utilizar los mismos equipos de cómputo que posea la organización para construir el clúster sin la necesidad de comprar equipos adicionales.

2.4.5. Ventajas y desventajas de los clúster

Las ventajas que presentan la construcción de un clúster citadas por Torrealba Martinez (2002) son:

- ✓ Cada una de las máquinas en un clúster, puede ser un sistema completo para usarlo en un amplio rango de aplicaciones.
- ✓ El hardware de interconexión de red ha venido experimentando un constante decremento de precio, considerando que además se puedan lograr ahorros adicionales empleando un solo monitor, teclado y ratón.
- ✓ Los clúster de computadoras pueden crecer hasta formar sistemas verdaderamente grandes, es decir, desde dos hasta varios cientos de nodos.
- ✓ Se puede reemplazar fácilmente una computadora del clúster que no esté funcionando correctamente.
- ✓ Si algún componente falla, el o los procesos pueden seguir ejecutándose en los demás nodos.

- ✓ El clúster se puede interconectar a una red de área local permitiendo dar servicio a múltiples usuarios internos y externos a través de internet.
- ✓ Existe software gratuito en la red como el sistema operativo para computadoras de escritorio como GNU/LINUX y software de aplicación para clúster como: MOSIX, CONDOR, LUI, BEOWULF, HOVM, entre otros. (p. 11)

Las desventajas principales en la utilización de un clúster según Torrealba Martinez (2002) son:

- ✓ Si falla el nodo maestro desde el cual se envía una tarea, se pierde la ejecución de la tarea.
- ✓ Puede haber inconsistencia en los datos, provocada por la falla en el software o en el hardware.

Una pregunta puede surgir entonces, si el software es gratis, el hardware barato y los clúster pueden crecer de manera significante. ¿Por qué cualquier empresa o institución no tiene su propio clúster?, pues bien no todo hardware de red está diseñado para realizar procesamiento paralelo, generalmente por que la latencia es alta y el ancho de banda de la red es baja, aún hay muy poco software que soporte a un clúster como un solo sistema(p.12).

2.4.6. Lenguajes de programación para clúster

"Existen varios lenguajes de programación paralela, sobresaliendo de pestos MPI (Message Pasing Interface) – Interface de paso de mensajes y PVM (Parallel Virtual Machine)-máquina virtual paralela, por ser los estándares más aceptados." (Castillo, Olivera Acosta, 2006, p. 21)

- ✓ **MPI**

MPI consiste de una biblioteca para programación paralela en el modelo de intercambio de mensajes. En este estándar se han incluido los aspectos más relevantes de otras bibliotecas de programación paralela.

Entre las ventajas de MPI se encuentra la disponibilidad de varios modos de comunicación, los cuales permiten al programador el uso de buffers para el envío rápido de mensajes cortos, la sincronización de procesos o el traslape de procesos de cómputo con procesos de comunicación. Esto último reduce el tiempo de ejecución de un programa paralelo, pero tiene la desventaja de que el programador debe ser más cuidadoso para evitar la corrupción de mensajes. Dos de las principales distribuciones libres de MPI son: **LAM/MPI y MPICH.**

- ✓ **PVM**

  PVM se comenzó a desarrollar en verano de 1989 por el Oak Ridge National Laboratory, y posteriormente junto con la Universidad Tennessee en los EUA. Es una biblioteca de envío de mensajes, totalmente libre, capaz de trabajar en redes homogéneas y heterogéneas y que hacen uso de los recursos existentes en algún centro de trabajo para poder construir una máquina paralela de bajo costo, obteniendo su mejor rendimiento. Maneja transparentemente el envío y recepción de todos los mensajes, conversión de mensajes y calendarización de tareas a través de una red de arquitecturas incompatibles. Está diseñado para conjuntar recursos de cómputo y proveer a los usuarios de una plataforma paralela para correr sus aplicaciones, independientemente del número de computadoras distintas que utilicen y donde estas se encuentren localizadas. El modelo computacional de PVM es simple y además muy general. El usuario escribe su aplicación como una colección de tareas cooperativas. Las tareas acceden los recursos de PVM a través de una biblioteca de rutinas. Estas rutinas permiten la inicialización y terminación de tareas a través de la red, así como la comunicación y sincronización entre tareas.

2.4.7. Modelos de clúster

En los artículos publicados por Mesa Múnera & Branch Bedoya (2008) existen tres modelos de clúster Mosix / Openmosix y Beowulf.

- ✓ **Beowulf**

  "Beowulf es una clase de computador masivamente paralelo de altas prestaciones principalmente construido a base de clúster de componentes hardware estándar". (Llorens & Peña, 2002)

  "Consta de un conjunto de nodos minimalistas, unidos por un medio de comunicaciones económico. Esto quiere decir que tienen lo mínimo para ejecutar su función, de hecho los nodos por si solos no son capaces de ejecutar ni tan siquiera un sistema operativo" (Arrollo, Nievas, & Pino, 2004)

Se puede ver como un supercomputador paralelo construido con hardware comercial de fácil adquisición, que posee como sistema operativo (GNU/Linux)

"Los clústers Beowulf son extremadamente poderosos, pero no lo son para todas las personas. Su principal desventaja es que requieren de software diseñado para poder aprovechar los recursos del clúster" (Robbins, s.f.)

- ✓ **Mosix**

  Según Pérez (2001) la arquitectura Mosix basada en la misma ideología de la arquitectura Beowulf. Se basa en un conjunto de parches aplicados al kernel de Linux y que se asignan a todo un grupo de nodos un espacio de direcciones y de procesos común, gracias a los cuales los procesos migran de uno a otro con el fin de equilibrar favorablemente la carga del sistema global.

  "Es una herramienta diseñada para realizar balanceo de carga en el clúster de forma totalmente transparente de manera tal que los nodos se comportan como un asola máquina, y así incrementar el aprovechamiento de cada uno de los nodos" (Corrrea, 1999).

La principal ventaja de Mosix frente a Beowulf se basa precisamente en esto. Mosix da mejor respuesta que Beowulf frente a la caída e inserción de nodos. A parte de esto, Mosix permite un cambio constante de aplicación, o incluso, el correr aplicaciones independientes de forma simultánea.

- ✓ **Openmosix**

  Openmosix es una extensión del proyecto Mosix, De (Chirinov, 2003), la idea de este modelo es que la distribución de tareas en el clúster la determina Openmosix de forma dinámica, conforme se van creando tareas. Cuando un nodo está demasiado cargado, las tareas que se están ejecutando pueden migrar a cualquier otro nodo del clúster. Así desde que se ejecuta una tarea hasta que se muere, podrá migrar de un nodo a otro, sin que el proceso sufra mayores cambios.

  De acuerdo a Catalán (2004), la idea de Openmosix es que los procesos colaboren de forma que parezcan que están en un mismo nodo. Como sus algoritmos son dinámicos, tiene una fuerte ventaja sobre los algoritmos estáticos de PVM/MPI, responden a las variaciones en el uso de los recursos entre los nodos migrando procesos de un nodo a otro, de forma transparente para el proceso, para balancear la carga y para evitar fallas de memoria en un nodo.

  Al contrario que PVM/MPI, no necesita una adaptación de la aplicación, ni siquiera que el usuario sepa nada sobre el clúster. Para aprovechar completamente PVM/MPI hay que programar con sus librerías, por lo tanto hay que rehacer todo el código que haya para sacar el mayor provecho del clúster.

Openmosix puede balancear una única aplicación si esta está dividida en procesos lo que ocurre en un gran número de aplicaciones hoy en día, también puede balancear las aplicaciones entre sí, balanceando los procesos en la mínima unidad de balanceo.
Además Openmosix funciona a nivel kernel por lo tanto puede conseguir toda la información que necesite para decidir cómo está de cargado un sistema y que pasos debe seguir para aumentar el rendimiento, además puede realizar más funciones que cualquier aplicación a nivel de usuario.

2.4.8. Aplicaciones de clúster

Entre las aplicaciones más recientes de la computación distribuida se encuentra la computación en la nube o ***cloud computing*** que todos usamos en el acceso a múltiples servicios de internet.

La aplicación de estas técnicas se aplica a dominios muy distintos que tienen en común a necesidad de una gran potencia de cómputo. Algunas aplicaciones tradicionales incluyen la predicción del tiempo, las simulaciones aerodinámicas el estudio de los terremotos y tsunamis o la secuenciación del genoma humano.

Por ejemplo, en los recientes estudios sobre el Bolson de Higgs han tenido un papel central el uso de grandes computadores las técnicas de procesamiento paralelo.
La computación paralela y distribuida también se usa en el estudio en las interacciones entre proteínas y el modelado de moléculas que tienen una gran relación con el diseño de nuevos fármacos.

Otro ejemplo en el cual se utiliza la computación paralela es en la realización de tomografías médica por computador(TAC) que se usa mucho en los diagnósticos médicos y donde el tiempo necesario para realizar el diagnóstico es muy importante, en la figura 30 se muestra una imagen de tomografías médicas realizada por computadores y procesadas con clúster.

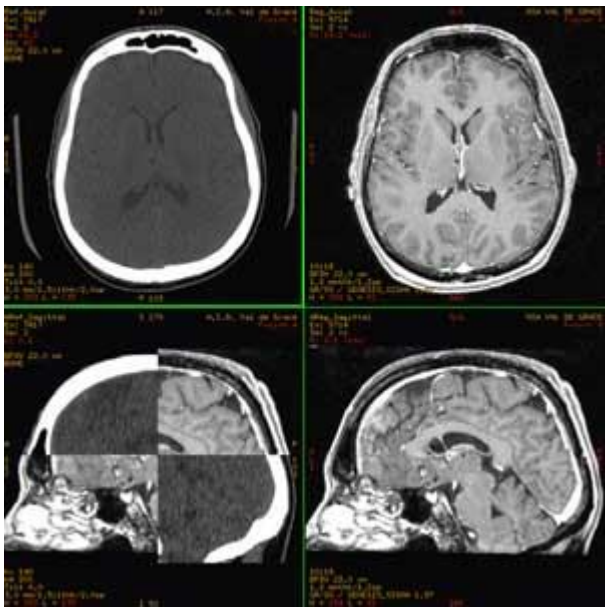


Figura 30. Tomografía médica

http://www.reddediagnosticosmedicos.com.co/2014/03/04/tomografia-axial-computarizada/, 2013

Otra de las categorías de aplicaciones que está siendo adoptado rápidamente en computación clúster según (Barroso, Dean, & Holzle, 2003) son las Aplicaciones de Grand Challenge (GCA). Las aplicaciones Grand Challenge se definen como aplicaciones enfocadas a resolver los problemas fundamentales de la ciencia y la ingeniería, con gran impacto económico y científico cuya solución se puede avanzar mediante la aplicación de Informática y comunicaciones de alto rendimiento. El alto nivel de complejidad en GCA requiere de una gran cantidad de necesidades de recursos, tales como el tiempo de procesamiento, el espacio de memoria y ancho de banda de comunicación. Una característica común de GCA es que implica simulaciones que son computacionalmente intensivos. Ejemplos de GCA se aplican a la dinámica de fluidos, modelización ambiental, la simulación de ecosistemas, imágenes biomédicas, biomecánica, biología molecular, diseño molecular, la cognición y ciencias computacionales.

Aparte de GCA, los clúster de computadoras también se están utilizando en otras aplicaciones que requieren una alta disponibilidad, escalabilidad y rendimiento. Los clúster están siendo utilizados como almacenamiento y servidores de respaldo replicado información que proporcionan la tolerancia a fallos esencial y fiabilidad para aplicaciones críticas. Por ejemplo, la Internet con motores de búsqueda, Google utiliza la computación del clúster para proporcionar servicios confiables y eficientes de búsqueda en Internet. También hay muchos productos de clúster comerciales diseñados para bases de datos distribuidas y servidores web (p. 22 -28).

- ✓ **Google Search Engine:** Los buscadores de internet permiten a los usuarios realizar búsquedas de información en internet mediante la introducción específica de palabras clave. Un motor de búsqueda utilizado, Google utiliza la computación clúster para satisfacer la enorme cantidad de solicitudes de búsqueda en todo el mundo que forman parte de las miles de consultas por segundo. Una sola consulta en Google necesita utilizar al menos decenas de miles de millones de ciclos de procesamiento y acceso a unos pocos cientos de megabytes de datos con el fin de devolver los resultados de búsqueda satisfactorios. Google utiliza la computación en clúster como la solución a la alta demanda de recursos del sistema y las agrupaciones tienen mejores relaciones precio-rendimiento que las plataformas informáticas de alto rendimiento alternativo, y también consumen menos energía eléctrica. Google se centra en 2 factores de diseño importantes: fiabilidad y rendimiento de solicitud. Google es capaz de lograr una fiabilidad a nivel de software de manera que una infraestructura de computación puede ser fiable construida en grupos de 15.000 PCs de los productos básicos distribuidos en todo el mundo. Los servicios de Google son también replicados a través de múltiples máquinas en los grupos para proporcionar la disponibilidad necesaria. Google maximiza solicitudes de rendimiento global mediante la realización de la ejecución en paralelo de las solicitudes de búsqueda individuales. Esto significa que más solicitudes de búsqueda se pueden completar dentro de un intervalo de tiempo específico.

  La figura 31 muestra la forma de operar de Google Search Engine (Barroso, Dean, & Holzle, 2003, p. 22-28)

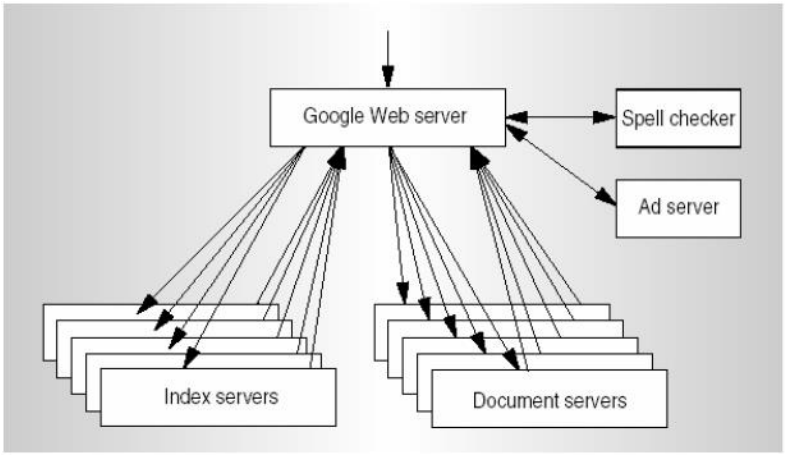


Figura 31. Entorno de Google Search Engine

Fuente: Barroso , Dean, & Holzle, The Google Cluster Architecure, 2003

- ✓ **Render de imágenes:** El Scientific Computing and Imaging y el Instituto de la Universidad de Utah ha explorado la visualización científica basado en clúster utilizando

un clúster de visualización de 32 nodos compuesto por hardware comercial de componentes conectados con una red de alta velocidad. El OpenGL herramienta de visualización científica Simian se ha modificado para crear una versión compatible con clúster Simian que soporta paralelización haciendo uso explícito de los nodos del clúster remotos a través de una interfaz de paso de mensajes (MPI). Simian es capaz de generar Imágenes 3D para simulaciones de incendios que se propagan de esa hipótesis de modelos tales como cuando un misil ubicado dentro de un grupo de combustible de avión se incendia y explota. Con la representación de imágenes para las simulaciones de incendios esta extensión permite a los investigadores tener una mejor visualización de los efectos destructivos.

Normalmente, Simian utiliza un mecanismo de intercambio para gestionar los conjuntos de datos que son demasiado grandes para cargar en la memoria de texturas disponibles, lo que resulta en bajo rendimiento y la interactividad. Para Simian clúster, grandes conjuntos de datos se dividen en sub-volúmenes que pueden ser distribuidos a través de varios nodos del clúster, así lograr el rendimiento interactivo. Esta técnica de "divide y vencerás" se descompone en primer lugar el conjunto de datos en sub-volúmenes antes de la distribución de los sub-volúmenes de varios nodos del clúster remotos. Cada nodo es entonces responsable de la prestación de sus sub-volumen con el hardware de gráficos disponibles localmente. Los resultados se combinaron finalmente utilizando un algoritmo de composición binaria de intercambio para generar la imagen final. Esto permite a Simian clúster visualizar conjuntos de datos a gran escala para mantener las tasas interactivos sin necesitar de la textura de intercambio, en la figura 32 se muestra el resultado de las simulaciones realizadas con el clúster Simian publicadas de los estudios realizados por (Calvin, Hartner, & Hansen, 2003)

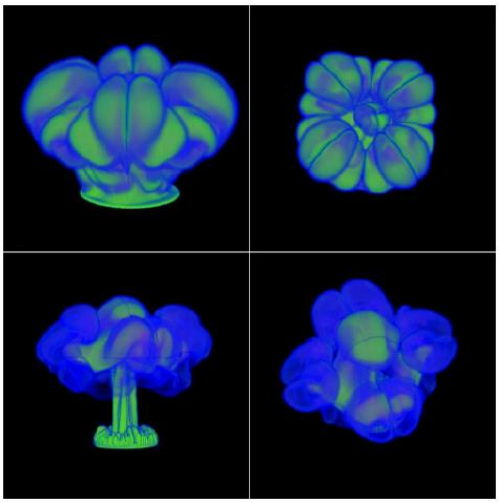


Figura 32. Resultados de simulaciones con Simian

Fuente: Calvin, Hartner, & Hansen, Cluster Based Interactive Volume Rendering With Simian, 2003

2.4.9. Clasificación de clúster

Debido a que los clúster están formados por un grupo de computadoras y estas pueden tener diversas arquitecturas, existe la siguiente clasificación de acuerdo a su homogeneidad.

- ✓ **Clúster homogéneos:**

  En este tipo de clúster todas las computadoras conectadas en red que forman los nodos del clúster tienen la misma arquitectura y utilizan un mismo sistema operativo (GNU/Linux, Unix o Windows), por ejemplo computadoras con arquitecturas Intel, AMD.

- ✓ **Clúster heterogéneos:**

  En estos clúster todas las computadoras conectadas en red que forman los nodos del clúster tienen distintas arquitecturas y utilizan sistemas operativos diferentes, como por ejemplo computadoras Intel, AMD y una multitud de diferentes sistemas operativos para los nodos. Controlar este entorno de un clúster heterogéneo es mucho más difícil que controlar un clúster homogéneo.

- ✓ **Meta clúster**

  Estos son clúster de clúster, es decir los meta clúster son usualmente un grupo de clúster que están geográficamente distribuidos, ya sea en una nación o en el mundo entero, pero pueden ser tratados como un solo recurso de software especial y muy avanzado. Los meta clúster pueden ser también homogéneos o heterogéneos, los meta clúster también suelen ser llamados súper clúster.

2.4.10. Tipos de clúster

Según (Castillo Castillo & Olivera Acosta, 2006)

> Los clúster se pueden clasificar tomando en cuenta diferentes aspectos como la aplicación, disponibilidad, sistema operativo, configuración y el número de nodos.
> Muchos de los trabajos de investigación han clasificado de manera muy similar los clúster, teniendo en cuenta ello para cuestiones prácticas se puede tomar los siguientes tipos de clúster (p.16)

- ✓ **Clúster de tolerancia a fallas(Fail-Over)**

Estos clúster utilizan una conexión de alto desempeño entre las computadoras, ésta conexión es para monitorear cual o cuales de los servicios están en uso, así como la sustitución de una máquina por otra cuando uno de sus servicios ha caído, por ejemplo los mecanismos de trabajo de Google Search Engine mas explicado en la ***1.4.8.***

- ✓ **Clúster de Balanceo de carga (Load-Balancing)**

  Estos clúster son implementados cuando los recursos de los nodos son insuficientes para el procesamiento de datos, el clúster reparte toda la carga de trabajo entre los demás nodos del clúster.

- ✓ **Alto rendimiento (High Performance)**

  Este tipo de clúster se implementa especialmente cuando centros de datos, centros de investigación entre otros necesitan y requieren gran capacidad de procesamiento y de tratamiento de datos así como una potencia de computación extrema.

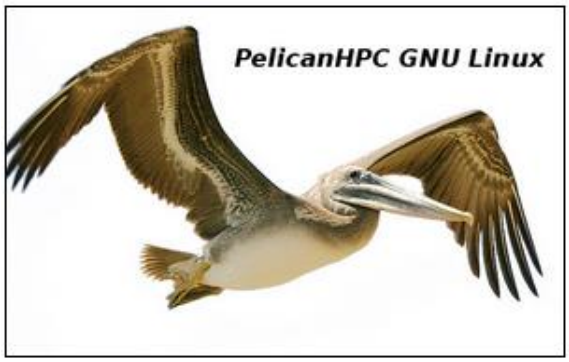
2.5. Exploración tecnológica de las herramientas

2.5.1. Soluciones de clúster de computadoras de alto rendimiento

Durante la exploración tecnológica que se realizó se encontraron diversas soluciones para implementar clúster de computadoras de alto rendimiento que están basados e implementados sobre los clúster pioneros es decir Beowulf, Mosix y Openmosix, estos son: PelicanHPC, clusterknoppix, ABCLinux, Kerrighed, Rock Cluster y Quantian, todas estas soluciones son software libre ya que uno de los objetivos de la presente investigación es construir y realizar la investigación utilizando soluciones libres y todas estas soluciones están puestos en marcha sobre el sistema operativo GNU/Linux, se procede a continuación a realizar la descripción de cada una de estas soluciones así como de sus fundamentales características para posteriormente una comparativa entre ellos y elegir la plataforma para la investigación.

- ✓ **Pelican HPC**

PelicanHPC es una imagen ISO - híbrido (CD o USB) que permite configurar un clúster de computación de alto rendimiento en un par de minutos. Un clúster PelicanHPC permite hacer uso de la computación paralela MPI. Se puede ejecutar PelicanHPC en una sola máquina de núcleo múltiple de y así utilizar usar todos los núcleos para resolver un problema en específico, o utilizar varios equipos de cómputo en red, La instalación del clúster Pelican es muy sencillo ya que se instala el nodo frontal o central y este a través de arranque de red proporciona los mecanismos para instalar los nodos



desde una red local. Todos los nodos del clúster deben tener en su sistema de archivos la misma imagen, por lo que se garantiza que todos los nodos ejecutan el mismo software. Los paquetes instalados y configurados a través de APT. Este sistema operativo está basado en la distribución Debian. Puede utilizarse PelicanHPC orientado a diversas disciplinas, adicionalmente se puede crear una versión personalizada. No es muy difícil. Dos ejemplos de distribuciones especializadas que se suman a la base PelicanHPC son MOLA y birgHPC .

Estas son algunas características de Pelican HPC

- PelicanHPC está creado utilizando herramientas del proyecto Debian Live .
- Contiene la última versión estable de la aplicación OpenMPI de MPI.
- Disponible arquitecturas de 64 bits. también está disponible para 32 CPU bits.
- Contiene programas de ejemplo utilizando GNU Octave . También tiene el punto de referencia Linpack HPL .
- PelicanHPC y todas las pruebas se realiza mediante la versión estable de Squeeze versión de Debian GNU Linux como base.

En la parte inferior se muestra la figura 33 de la solución de pelicanHPC.

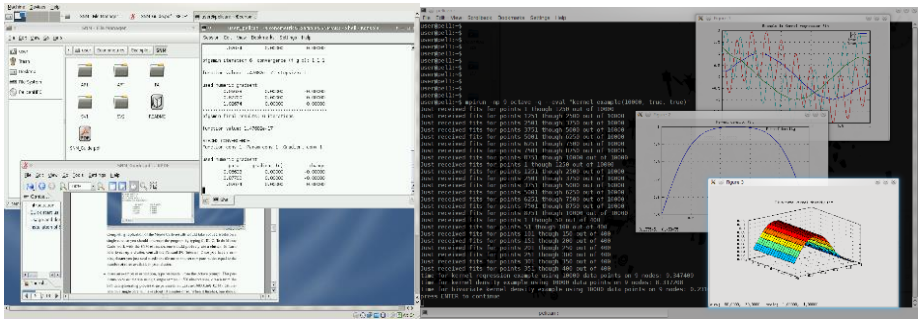


Figura 33. Pelican HPC

Fuente: página oficial de Pelican HPC,http://pareto.uab.es/mcreel/PelicanHPC/pelicanhpc.png, 2013

✓ **Clusterknoppix**

Este sistema operativo basado en Knoppix y montado con Openmosix es un proyecto que surgió de la separación de los dos principales desarrolladores de Openmosix, es un sistema de clúster para el sistema operativo GNU/Linux que consiste en un parche aplicado en el kernel responsable de las migraciones transparentes de procesos, y unas herramientas de área de usuario,

necesarias para calibrar y administrar el clúster. Esto permite que no se tenga que reprogramar nuestras aplicaciones para que aprovechen el clúster, es decir no hay que instalar la aplicación en todos los nodos para ejecutar una tarea.

Los procesos no saben en qué nodo del clúster se ejecutan, y es el propio Openmosix el responsable de "engañarlos", y redirigir las llamadas del sistema al nodo del clúster en el que se lanzó el proceso el cual actúa en ese momento de nodo maestro.

El parche para el kernel funciona en las versiones 2.4 y 2.6, aunque en esta última solo de forma experimental.

Las ventajas de utilizar Openmosix frente a otras arquitecturas de clúster es que no hay que preocuparse por agregar librerías, no es necesario programar las aplicaciones y cuenta con un demonio para descubrir nodos de forma automática denominado omdiscd con Openmosixcollector que se encarga de crear automáticamente una lista con las máquinas existentes en la red, y de estar escuchando en caso de que haya otros demonios de detección de nodos; así como de informar al kernel sobre los nodos operativos para la migración de procesos. Algunas desventajas que ofrece Openmosix en comparación a otros es que, como se mencionó anteriormente tiene un núcleo dependiente; y como no funciona totalmente en el núcleo 2.6 puede dar problemas para el hardware que no soporte la versión de kernel 2.4. En cuanto a la estructura interna de Openmosix utiliza un sistema de archivos propios denominados o MFS (Openmosix File System) que permite que todos los nodos de un clúster tengan acceso al sistema de archivos de todos los otros nodos.

Se muestran en la figura 34 el sistema operativo Clusterknoppix

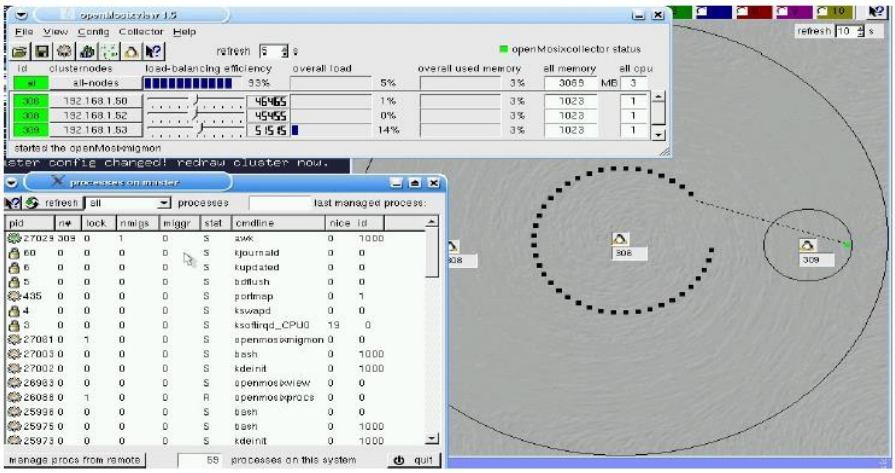


Figura 34. Funcionamiento de Clusterknoppix

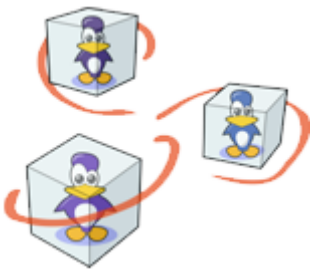
Fuente: http://www.downloadsource.net/17180/clusterKNOPPIX/

✓ **ABCLinux**

La distribución ABC GNU/Linux está basada en Ubuntu y está especializada en la construcción automática de clúster Beowulf de alto rendimiento con tan solo arrancar el sistema en modo "live" en el frontend o siendo instalado en su disco duro. Los nodos arrancan a través de arranque por red. Usa como gestor de ventanas Gnome. Integra el monitor de recursos Ganglia. Se trata de la primera distribución que integra todas estas características. Ha sido publicado un artículo científico sobre este sistema en el IEEE y presentado en el Simposium ICAT2009 celebrado en Sarajevo (Bosnia & Herzegovina). Desarrollado por Iker Castaños Chavarri en el Departamento de Ingeniería de Sistemas y Automática de la EUITI de Bilbao, Universidad del País Vasco.

✓ **Kerrighed**

Kerrighed es una sola imagen del sistema del sistema operativo para clúster Kerrighed ofrece el punto de vista de una única máquina.



Los objetivos de Kerrighed son la facilidad de uso, alto rendimiento de las aplicaciones, la alta disponibilidad del clúster, la gestión eficiente de los recursos, y capacidad de personalización de alta del sistema operativo.

Kerrighed se implementa como una extensión del sistema operativo Linux (un conjunto de módulos y un parche para el kernel).

Principales características específicas son:

- La gestión amplia en grupos de proceso
- Soporte para clúster de memoria compartida
- Sistema de archivos clúster
- Proceso transparente puntos de control
- Alta disponibilidad para aplicaciones de usuario
- Personalizable con respecto a las características del sistema.

✓ **Rock Cluster**

Rocks es una distribución de Linux basada en CentOS y un sistema de gestión de clúster que permite el despliegue rápido de clúster Linux sobre hardware físico o contenedores virtuales Xen. Un clúster de Rocks, es fácil de implementar y ofrece todas las ventajas de la virtualización de los nodos del clúster. Con un mínimo de dos máquinas físicas, Rocks permite un despliegue y una gestión de clúster simple y rápida, liberando al administrador para que se centre en el apoyo a la red informática y a las aplicaciones distribuidas que hacen del clúster una opción atractiva. Rock clúster dispone de rollos para instalar herramientas así como software incluidas en la distribución estándar de Rocks, existen diferentes herramientas de código abierto de computación paralela y distribuida de alto rendimiento, tales como Sun Grid Engine , OpenMPI y Condor. Esta poderosa colección de funciones avanzadas es una de las razones por las que la NASA, la NSA, el Laboratorio de Investigación de IBM en Austin, la Marina de los EE.UU., el MIT, Harvard y la Universidad Johns Hopkins están todos utilizando Rocks por alguna de sus aplicaciones más intensivas.

✓ **Cuantian**

El sistema operativo Quantian es una remasterización de Knoppix / Debian para las ciencias computacionales. El medio ambiente es la a autoconfiguración y directamente de arranque de CD / DVD que convierte cualquier PC o portátil (siempre que se pueda arrancar desde CD-ROM / DVD) en un una estación de trabajos. Quantian también incorpora Clusterknoppix y añade soporte para Openmosix , incluyendo el inicio remoto de clientes ligeros en un contexto de servidor de terminal Openmosix permite una rápida configuración de un SMP clúster ordenador, se muestra en la figura 35 el funcionamiento de Quantian .

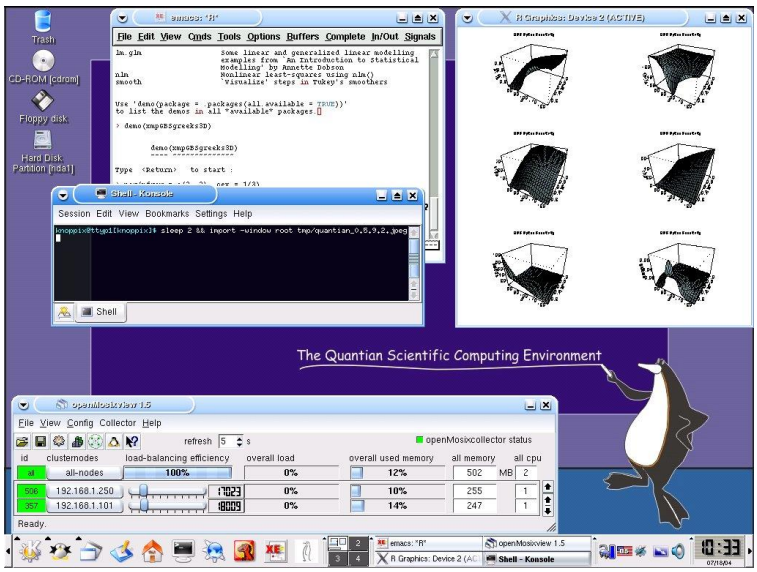


Figura 35. Funcionamiento de Quantian

Fuente:
http://www.google.com.pe/imgres?imgurl=http%3A%2F%2Fdirk.eddelbuettel.com%2Fquantian%2Fquantian_0.5.9.2.jpeg&imgrefurl=http%3A%2F%2FOpenmosix.sourceforge.net%2Finstant_Openmosix_clusters.html&h=768&w=1024&tbnid=hlM177CaJacngM%3A&zoom=1&docid=2LqB-vCc1ZV9SM&hl=es-419&ei=aymRU5fDC4O-sQSA-oCQDw&tbm=isch&ved=0CC8QMygHMAc&iact=rc&uact=3&dur=353&page=1&start=0&ndsp=13, 2013

2.5.2. Tabla de comparación entre soluciones de clúster de alto rendimiento

Como se vio anteriormente los clúster de computadoras en la actualidad se definen en tres tipos estrechamente relacionadas entre sí, por un lado se halla a Beowulf el pionero en soluciones HPC, Mosix su competencia y el que de algún modo u otro está basado en Beowulf y finalmente Openmosix la versión libre de Mosix, la tabla 5 muestra la comparación de las soluciones HPC antes mencionadas.

Tabla 5. Comparativa en soluciones HPC

Herramienta HPC	Sistema	Distribución	Versión inicial	Clase	Idiomas disponibles	Licencia	Última versión
Pelican HPC	GNU/Linux	**Debian**	**-**	**Beowulf**	**Inglés**	**GPL**	**2.8**
ABCLinux	GNU/Linux	**Ubuntu**	**11/11/2009**	**Beowulf**	**Español/Inglés**	**GPL**	**1.0 Beta**
Clusterknoppix	GNU/Linux	**Knoppix**	**10/05/2003**	**Openmosix**	**Español/Inglés**	**GPL**	**3.6**
Quantian	GNU/Linux	**Knoppix**	**1/03/ 2006**	**Openmosix**	**Español/Inglés**	**GPL**	**7.9.2**
Rock Cluster	GNU/Linux	**CentOS**	**4/11/2005**	**Beowulf**	**Español/Inglés**	**GPL**	**6.1**
Kerrighed	GNU/Linux	**Ubuntu**	**-**	**Openmosix**	**Español/inglés**	**GPL**	**2.3.0**

Fuente: elaboración propia

La exploración tecnológica solo incluye aquellas soluciones libres, como se puede apreciar en la tabla 5 todas estas soluciones se ejecutan bajo el sistema operativo GNU/Linux y se puede encontrar en todos ellos características muy en común, es por ello que son todas propicias para utilizarlas en la presente investigación, sin embargo el sistema operativo Clusterknoppix resultó poseer cualidades bastante importantes y que facilitan de cierta forma la realización de la presente investigación, de esta manera no abordar otros temas fuera de lo que se pretenda estudiar en el sentido de que no hay que desarrollar ni programar software para implementar un clúster, para la instalación se realizan unos pocos pasos para la configuración del clúster, además se encontró que hay software para las soluciones HPC con respecto al trazado de rayos , todas estas son soluciones libres para el sistema Clusterknoppix.

El sistema Clusterknoppix posee programas preinstalados y configurados, una de sus características más resaltantes es la sencillez y la transparencia con la cual se puede configurar el clúster.

2.5.3. Programas de Trazado de Rayos

En la actualidad existen gran variedad de software que utilizan la técnica de trazado de rayos, muchos de ellos son software privativo pero en contraparte también se puede encontrar software de versión libre y gratuito, en la tabla 6 se muestra la lista de software para el trazado de rayos.

Tabla 6. Lista de software para el trazado de rayos

Software	Licencia	Plataformas			
		Ventanas	OS X	Linux	Otro
Art of Illusion	GPL	Sí	Sí	Sí	No
Licuadora	GPL	Sí	Sí	Sí	No
BRL-CAD	BSD , LGPL	Sí	Sí	Sí	No
Gelato	Freeware	Sí	No	Sí	No

Software	Licencia	Plataformas			
		Ventanas	OS X	Linux	Otro
Kerkythea	Freeware	Sí	Sí	Sí	No
LuxRender	GPLv3	Sí	Sí	Sí	No
Picogen	GPLv3	Sí	No	Sí	No
Pixie	GPL	Sí	Sí	Sí	No
POV-Ray	AGPLv3	Sí	Sí	Sí	No
Resplandor	BSD	Sí	Sí	Sí	No
Sunflow	MIT	Sí	Sí	Sí	No
Tachyon	GPL	No	Sí	Sí	No
YafaRay	LGPL	Sí	Sí	Sí	No

Fuente: Elaboración propia

La tabla 6 es un filtro de los programas cuyas licencias no son propietarias y además pueden ser ejecutados en entornos GNU/Linux, todos estos programas poseen similares características para la realización de la presente investigación, pero se utilizará la herramienta Povray ya que el programa Povray mostró facilidad de instalación y configuración en entornos GNU/Linux y una compatibilidad total con el sistema Clusterknoppix el cual fue elegido en la exploración tecnológica de las soluciones HPC.

Si bien es cierto la lista de programas antes mencionados son aplicaciones que están programadas y se ejecutan secuencialmente es decir para ordenadores, no para soluciones clúster de ordenadores desafortunadamente no se pudo encontrar aplicaciones para el

trazado de rayos implementados con programación paralela en el resto de programas en mención, es por eso que el criterio por el cual predominó la selección del programa **Povray** fue que existe 1 programa y dos parches que paralelizan el algoritmo del programa el trazado de rayos **Povray**, MPIPOV y PVMPOV estas soluciones son parches aplicados al código fuente original de **Povray** para lograr que trabajen de forma paralela, desafortunadamente se al intentar aplicar los parches y compilar el código fuente de la aplicación en entornos HPC se encontraron muchas incompatibilidades con respecto a la librerías y paquetes adicionales para su compilación y ejecución, se lidió con códigos que utilizaban para su compilación diferentes versiones y en muchas ocasiones estas versiones eran obsoletas no pudiendo compilar, mucho menos instalar ninguno para su funcionamiento, la librería obsoleta que causo muchos problemas fue la librería libpng10-dev, por poseer dependencia de la librería libpng10-0 el cual posee ya una versión reciente libpng12-dev por otro lado la única solución encontrada para el sistema Clusterknoppix fue de 1 programa, Povmosix, Povmosix un programa con interface gráfica que permite dividir una escena en unidades de trabajo más pequeñas, éstos son asignados a los nodos del sistema Clusterknoppix, también nos da la posibilidad de realizar configuraciones como: el tamaño de la imagen resultante en pixeles, el formato de salida de la imagen resultante. También nos muestra las estadísticas por unidades de proceso de las tareas encomendadas a los diversos nodos del clúster, la división de líneas de código por proceso, es estado y el porcentaje de renderizado por unidades de trabajo, todas estas estadísticas e información es muy importante para la investigación y que cubren con el propósito de la misma. En la figura 36 se muestra la interface del programa Povmosix.

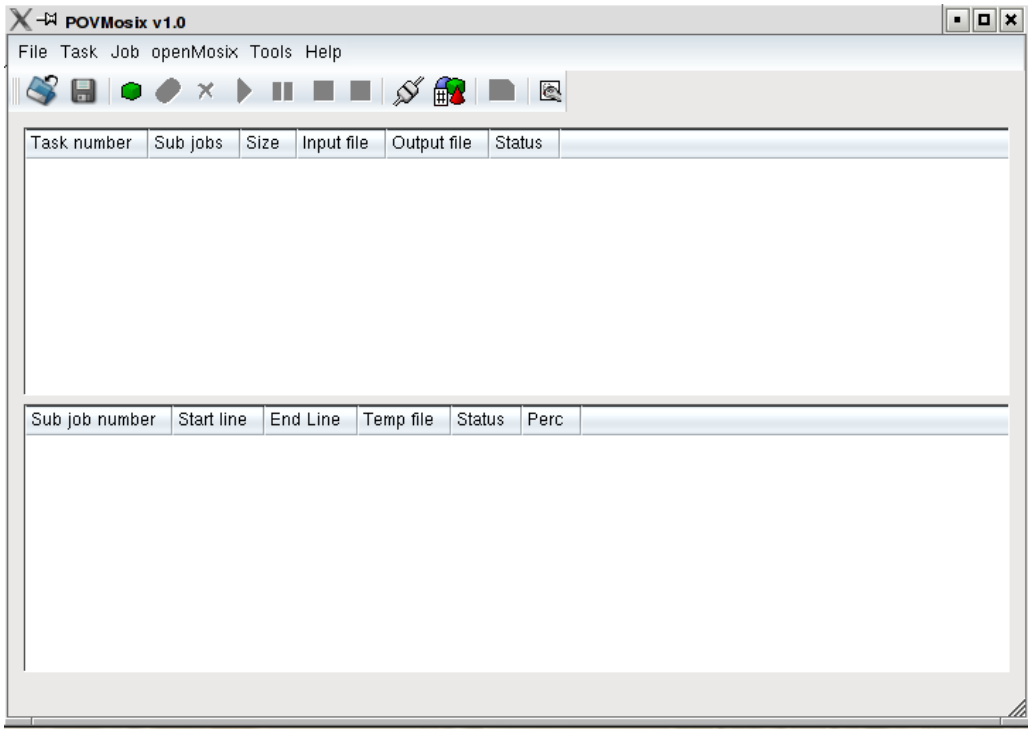


Figura 36. Interface del programa Povmosix

Fuente: Elaboración propia

2.6. Introducción al render y la postproducción

En los últimos 15 años, los gráficos por computador han pasado de formar parte de la comunidad científica a invadir el mercado cinematográfico, los videojuegos, la visualización médica y los sistemas de diseño asistido por computador. Una de las líneas de trabajo que más repercusión han tenido en todas estas áreas es la de representar imágenes de objetos tridimensionales, cuya descripción matemática está almacenada en la memoria de un computador, con el mayor realismo posible de forma que un observador humano no pueda diferenciar si se trata de una fotografía o de una imagen generada mediante una computadora. Este campo de trabajo se ha denominado Síntesis de Imagen Realista (Realistic Image Synthesis (RIS)), el nivel de realismo alcanzado está directamente relacionado con los componentes de la interacción de la luz con las superficies que se simule. En esta sesión se estudian algunos de los principales métodos de generación de imagen realista, describiendo sus ventajas, inconvenientes y limitaciones.

2.6.1. Perspectiva histórica

Desde la época de las primeras pinturas rupestres en el paleolítico, el hombre ha tenido la necesidad de construir imágenes bidimensionales que representen un mundo real o imaginario tridimensional. No fue hasta el renacimiento en el siglo XVI cuando Albrecht Dürer alcanzó un importante nivel de fotorealismo gracias a sus revolucionarios estudios de proyección (Figura 24) y perspectiva (Figura 25), La Figura 38 muestra el método de estudio de la proyección en perspectiva sobre una imagen 2D empleando una cuerda. La figura 37 por su parte muestra el uso de una rejilla de hilos empleada para dividir la escena con la misma proporción que la cuadrícula sobre la que dibujaba en la mesa de trabajo.

Figura 37. Plantilla de dibujo en perspectiva

Fuente: Carlos Gonzales Morcillo,
http://www.esi.uclm.es/www/cglez/fundamentos3D/05.01.Introduccion.html, 2013

Estas ideas desarrolladas por el famoso pintor alemán fueron mejoradas por otros artistas posteriores como Vermeer y Rembrandt y adaptadas a la informática por primera vez por Arthur Appel en 1968 y mejoradas por Boutknight en 1970 con el método de render

Scanline. En estos métodos la rejilla que empleaba Dürer se reemplaza por un plano de imagen en el que cada pixel se corresponde con una cuadrícula de la rejilla.

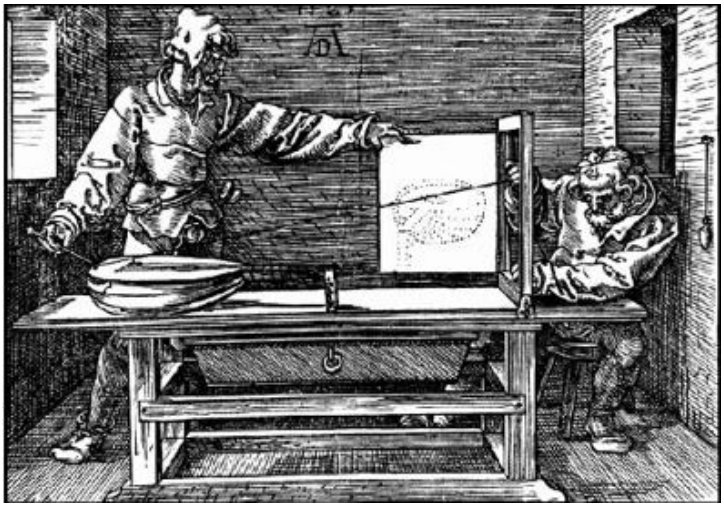


Figura 38. Estudio de proyección mediante una cuerda

Fuente: Carlos Gonzales Morcillo,
http://www.esi.uclm.es/www/cglez/fundamentos3D/05.01.Introduccion.html, 2013

El estudio de la perspectiva mediante "trazado de cuerdas" fue estudiado en 1637 por Descartes en los primeros tratados de óptica geométrica, donde se explicaba la forma del arco iris. Esta primera aproximación al trazado de rayos fue, en esencia, la misma en la que se basó Whitted en 1980 para formular el método recursivo de trazado de rayos. En este método se basan la mayoría de los algoritmos de renderizado realistas que se emplean en la actualidad.

2.6.2. Modelos tridimensionales fotorealistas

El fotorealismo es la cualidad de una imagen generada por computadora que trata de imitar las imágenes generadas por cámaras fotográficas mediante complejos cálculos y algoritmos matemáticos que simulan los efectos/defectos que la luz (halos, destellos) las sombras (coloreado de sombras, difusión), las texturas (aspereza, brillo, reflejos, refracción) y la radiosidad (coloreado de la luz ambiente) producen en las imágenes resultantes.

La generación de imágenes fotorealistas se obtiene a través de la síntesis de imágenes.

2.6.3. Síntesis de imágenes

La Síntesis de Imágenes es la disciplina que se dedica al estudio y desarrollo de procesos que permitan sintetizar imágenes a partir de modelos tridimensionales. El proceso de síntesis tridimensional abarca diversas etapas como se puede observar en la figura 39 (Fernandez, 2003)

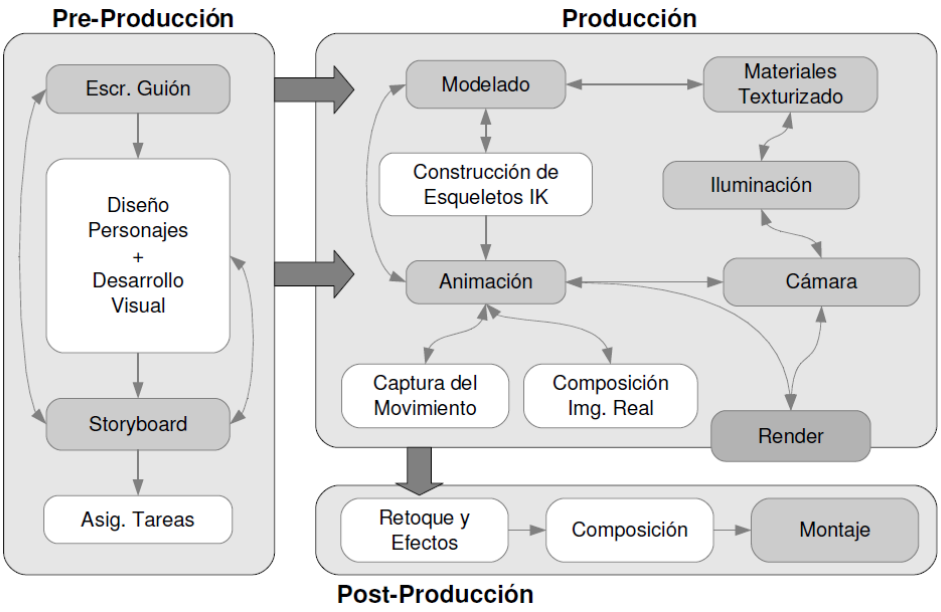


Figura 39. Etapas de producción de modelos tridimensionales

Fuente: José Antonio Fernandez, Sistema GRID para el Render de Escenas 3D, 2003

Las etapas del subproceso de producción son las más relacionadas con la informática gráfica y en tal sentido el estudio estará centrado es este aspecto, en especial en el proceso de *render.*

Los principales pasos para obtener una imagen sintética partiendo de unos bocetos de la escena son:

Modelado: Consistente en la construcción de la geometría de los objetos de la escena. La técnica utilizada para representar estos objetos depende del fin para el que están destinados.

Hay principalmente dos esquemas de representación:

- ✓ **Modelado de Fronteras (o B-Rep)** que define la superficie que limita al objeto. En función de los datos que se almacenen del objeto, se habla de modelado poligonal (que almacena información sobre vértices, aristas y caras) o modelado de superficies curvas (en el que se guarda información sobre puntos de control, puntos de tangente, etc.).
- ✓ **Modelado por Geometría Sólida Constructiva (o CSG):** que define el sólido como un conjunto de expresiones lógicas booleanas. Caracteriza implícitamente el interior del objeto y su superficie. El más utilizado en los motores de render (y a veces el único) es el modelado poligonal.

  Proporciona una representación consistente y eficiente (el hardware trabaja bien convértices y aristas). El inconveniente es que es costoso en almacenamiento.

- ✓ **Texturizado:** Es la fase en la cual se asignan propiedades a los objetos de la escena. Se definen las texturas que se aplicarán, las propiedades de reflexión y refracción de losmateriales, etc. Pueden usarse tanto imágenes como texturas.
- ✓ **Iluminación:** Se definen las fuentes de luz de la escena. Es un paso íntimamente relacionado con el proceso de render debido a que define el modelo de iluminación a utilizar.

Las fuentes de luz se definen por una serie de componentes:

- *Posición y Orientación*
- *Color*
- *Intensidad*.
- *Factor de Caída*

- ✓ **Animación:** Fase que consiste en definir cómo cambian unas determinadas propiedades, por lo general posición, orientación y escala de un objeto a lo largo de una línea de tiempo. No aparece en los proyectos en los que la salida es una sola imagen estática.
- ✓ **Render:** Consiste en la generación de una imagen bidimensional a partir de la descripción de la geometría de la escena junto con las definiciones de las luces, de la cámara y de los materiales como se muestra en la figura 40.

Figura 40. Imagen fotorealista

Fuente: Salón de la fama, página oficial de Povray, http://www.hof. povray.org, 2013

Existen multitud de métodos de render que aplican distintas técnicas y que serán revisadas en las siguientes secciones.

2.6.4. La síntesis de imágenes fotorealistas

Para poder comprender la síntesis de las imágenes fotorealistas se debe conceptualizar antes la generación de gráficos por computadora.

"La generación de gráficos por computadora es un área de investigación cuyo objetivo es la creación de algoritmos y métodos que permitan generar una imagen sintetizada artificialmente en un computador" (Cohen & Wallace, 1993)

El realismo es la característica fundamental y el cual se sientan las bases de la síntesis de las imágenes fotorealistas en tal sentido.

Según Pharr & Humphreys *(2004)*

> El realismo en graficación por computadora es definido como la producción de una imagen indistinguible de una fotografía del correspondiente lugar. Para lograrlo, es necesario simular la interacción de la luz con las superficies de acuerdo con las leyes de la Física.

Es así que el fenómeno de la interacción de la luz en las superficies es simulado mediante un modelo matemático. Entre mayor sea la correspondencia del modelo con la teoría física, mayor será el nivel de realismo exhibido por la imagen generada en el computador.

La base de la generación de imágenes se encuentra en la adecuada simulación de la luz, El término utilizado en la síntesis de imágenes fotorealistas es el de ***iluminación global*** que consiste en simular toda la luz dispersa en un modelo tridimensional respetando las leyes de la física.

Las primeras investigaciones pioneras en modelos de iluminación global iniciaron con los trabajos de Arthur Appel, quien ideó una técnica de rayos para determinar sombras y zonas ocultas de los objetos. También Golsteiny Nagel usaron una técnica especial derivada de la simulación de los proyectiles balísticos y partículas atómicas (Foley, 1997)
Posteriormente se denominó algoritmo clásico del raytracer a la propuesta de Ted Whitted (1980) que consistía en usar una técnica recursiva para trazar el recorrido de los rayos de la luz y simular efectos como sobras, reflexión y refracción.

2.6.5. Los elementos de la iluminación

Con el objetivo de comprender las diferentes técnicas de iluminación, sus ventajas, sus inconvenientes y los efectos visuales que logran, es necesario realizar un breve repaso de los elementos principales que forman la iluminación en los gráficos por ordenador.

Según (Raya Gonzales, 2009)

Se entiende por ***modelo de iluminación*** el cálculo de la intensidad de cada punto de la escena. En dicho cálculo intervienen los siguientes factores:

- El tipo e intensidad de las fuentes de luz.
- El material del objeto.
- La orientación del objeto con respecto a la luz.

Cuando la luz incide sobre un objeto, este reacciona en una o más de las formas siguientes, dependiendo si el objeto es: transparente, translucido, opaco, suave, rugoso, liso.

La luz podría ser:

- Total o parcialmente trasmitida.
- Total o parcialmente reflejada.
- Total o parcialmente absorbida

A continuación se detallan cada uno de los tres efectos que puede tener la luz incidente.

✓ **Transmisión**

La transmisión tiene lugar cuando la luz pasa a través de un objeto sin cambiar esencialmente; en este caso, se dice que el objeto es transparente. De acuerdo al índice de refracción del material, se produce una alteración en la forma en que la luz es transmitida. El **índice de refacción** (IR) es el cociente entre la velocidad de la luz en el vacío con respecto a la velocidad de la luz en el material.

El punto en donde se encuentran dos superficies de distinto IR se denomina **superficie borde.** En este punto, un rayo de luz trasmitida (el rayo incidente) cambia de dirección de acuerdo a la diferencia entre el índice de refracción y el ángulo de incidencia en el objeto. A este efecto se le denomina **refracción.** La luz que incide normalmente en la superficie del objeto pasa sin refractarse, aunque si puede reflejarse dependiendo del material; la luz que incide con otro ángulo será tanto parcialmente refractada como reflejada, dependiendo también en este caso, de las características del material.

✓ **Reflexión**

Cuando la luz incide sobre un objeto opaco(es decir, un objeto que no transmite luz), la superficie del objeto juega un papel importante en la determinación de la luz es completamente reflejada, completamente difundida o algo de ambos fenómenos.

El **coeficiente de reflexión** describe la amplitud (o la intensidad) de una onda reflejada respecto a la onda incidente. Este coeficiente muestra la relación entre el rayo incidente y la radiación reflejada por éste en una superficie.

Una superficie suave o brillante es aquella que está hecha con partículas se igual o semejante índice de refracción (IR). Estas superficies reflejan la luz con una intensidad y ángulo igual al del rayo incidente como se puede observar en la figura 41.

- ✓ **Reflexión especular**

Dentro de los modelos de iluminación, se dice que un material es especular si la fuente que lo ilumina procede de una dirección concreta y se refleja en una única dirección, se muestra en la figura 42 la reflexión de la luz perfectamente especular. Esto hace que se produzcan brillos intensos como, por ejemplo, los producidos en objetos metálicos o barnizados como se muestra en la imagen 29, en algunos materiales, los brillos solo se producen cuando la luz incide con un ángulo cercano a los 90ºcon respecto a la normal de la superficie (por ejemplo el cristal).

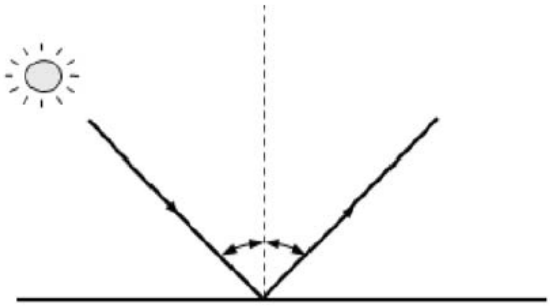


Figura 41. Reflexión de la luz perfectamente especular

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

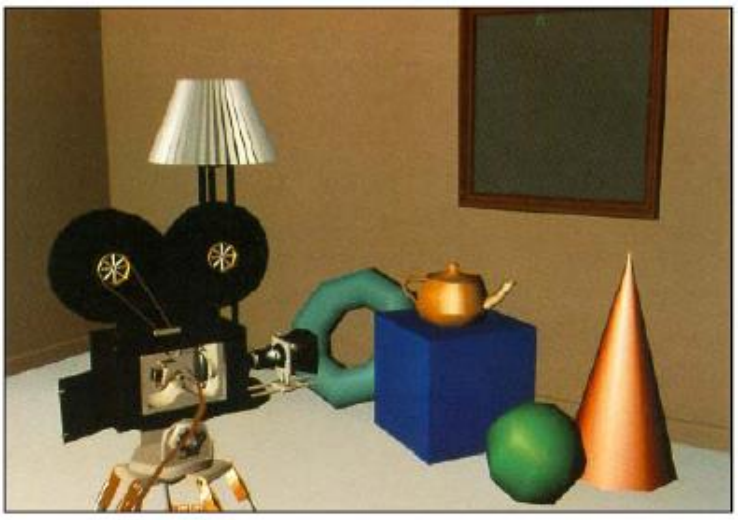


Figura 42. Escena con luz especular

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

✓ **Reflexión difusa**

La difusión o la dispersión es otro aspecto de la reflexión. Cuando una sustancia contiene partículas de diferente índice refractivo, un rayo de luz que incide en dicha sustancia será dispersado. Es decir, la luz proviene en una dirección pero se refleja en todas direcciones. La cantidad de luz dispersada depende de la diferencia entre los índices de refracción (*cociente entre la velocidad de la luz en el vacío con respecto a la luz de la velocidad en el material*) y también el tamaño de las partículas. Este tipo de luz corresponde a una luz completamente difusa como se muestra en la figura 43, se puede observar en la figura 44 una escena con luz difusa.

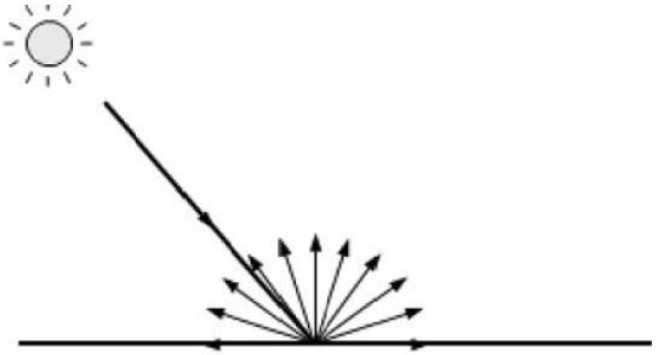


Figura 43. Reflexión de la luz completamente difusa

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

Figura 44. Escena con luz difusa

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

Sin embargo, en muy pocas ocasiones los objetos reflejan de manera perfectamente difusa o perfectamente especular, creando reflexiones mixtas como se muestra en la figura 45.

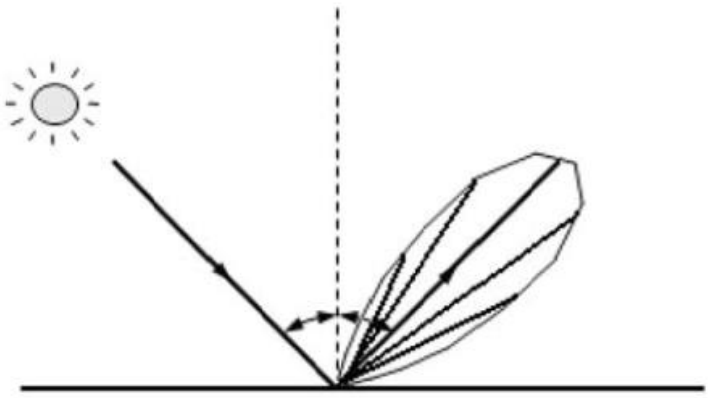


Figura 45. Reflexión de la luz mixta

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

Las técnicas de iluminación global persiguen calcular el valor de la intensidad de la luz en cada uno de los puntos de la escena, el valor dependerá de la interacción de las distintas fuentes con los elementos de la escena.

Por último cabe mencionar la **luz ambiental,** que proviene de todas direcciones e ilumina todas las caras del objeto por igual. Es más básica y, si solo se emplea este tipo de luz a la hora de sintetizar una imagen, se obtendrá una apariencia poco realista, como puede apreciarse en la figura 46.

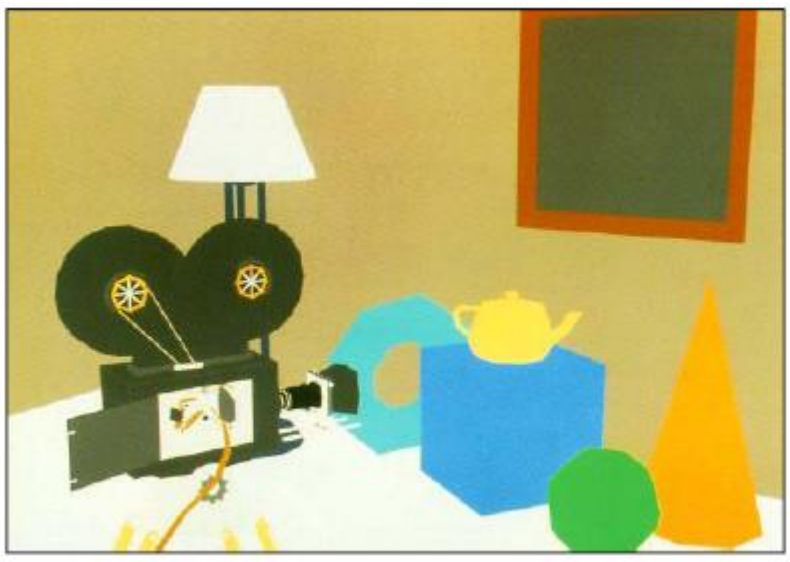


Figura 46 – Escena con luz ambiental

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

El modelo de iluminación de Phong es un modelo empírico simplificado para iluminar puntos de una escena, pero sirve para dar una visión sencilla de cómo se produce la iluminación en los gráficos por ordenador. En él, se indica que la iluminación en un punto es una combinación lineal de tres tipos de iluminación, iluminación ambiental, iluminación difusa e iluminación especular como se muestra en la figura 47.

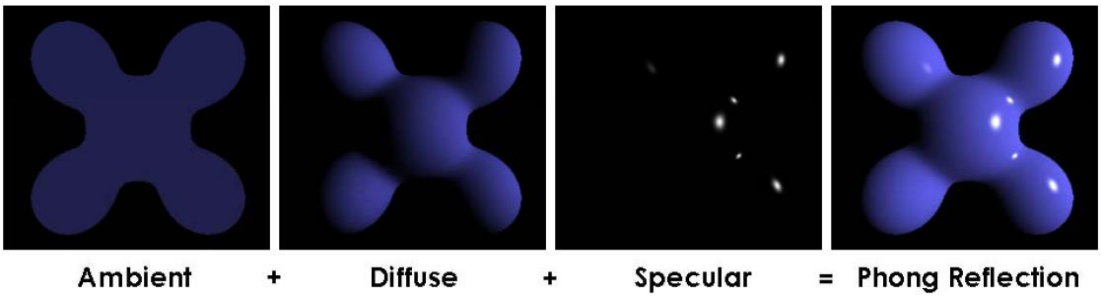


Figura 47. Modelo de iluminación de Phong

Fuente: Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009.

✓ **Absorción**

Toda luz o parte de ella puede ser absorbida dependiendo de la pigmentación del objeto. Los pigmentos son colorantes naturales que absorben algunas o todas las longitudes de onda de la luz, Lo que se ve como color, son las longitudes de onda de la luz que no es absorbida.

2.6.6. Modelos de iluminación de sombras

✓ **Iluminación local**

La **Iluminación local** considera, únicamente la luz que llega desde las fuentes emisoras de luz hasta los objetos. Se calcula el color del rayo de luz en función, únicamente de:

- El color del material del punto del que proviene el rayo.
- Las fuentes de luz presentes en la escena.

Si bien es cierto que este modelo de iluminación cumple con uno de los objetivos principales de la misma (como iluminar la escena), la apariencia resultante del renderizado se encuentra muy lejos de ser realista, es un modelo abreviado de la iluminación global, pero demasiado simple para convencer al espectador. Para mejorar su apariencia existen múltiples algoritmos que añaden efectos visuales, como pueden ser sombras suaves, en el diseño por ordenador existen dos tipos de sobras las suaves y duras, las sobras resultan de gran importancia para dotar de realismo las escenas virtuales.

- ✓ **Iluminación global**

  Se considera **iluminación global** (IG) a aquellas técnicas de iluminación que tienen en cuanta las interacciones entre los distintos objetos de la escena, es decir, considera la luz reflejada por un punto teniendo en cuenta toda la luz que llega y no solo la energía que proviene de las fuentes de luz, como en caso de la iluminación local.

  Los efectos que se consiguen con este tipo de técnicas son sombras suaves, reflexiones entre objetos, iluminación indirecta y trasparente, importantes si se busca una apariencia fotorealista o imágenes físicamente correctas. El conjunto de estos efectos hace que la apariencia visual de los modelos sea mucho más realista que utilizando técnicas de carácter local. Sin embargo, su coste computacional las restringe a aplicaciones con un número limitado de objetos.
  Hoy en día, un gran porcentaje de aplicaciones gráficas que no requieren tiempos de respuesta bajos o interactividad utilizan iluminación global.

- ✓ **Tipos de sombras**

  Las sombras son claves para obtener una correcta relación espacial de los objetos. Dependiendo del tipo de fuente de luz, existen dos tipos de sobras: Las **sobras duras** (Hard Shadows) y las **sombras suaves** (Soft Sadows).

  Las sombras duras provienen del tipo de fuente de luz puntual (la luz no tiene un área) y puesto que en la naturaleza no hay fuentes de luz puntuales, tampoco existen los tipos de sobras duras.

  Por otro lado, las sombras suaves se originan con fuentes de luz que ocupan un área determinada, lo que provoca zonas de umbra y zonas de penumbra en la sombra del objeto como se muestra en la figura 48.

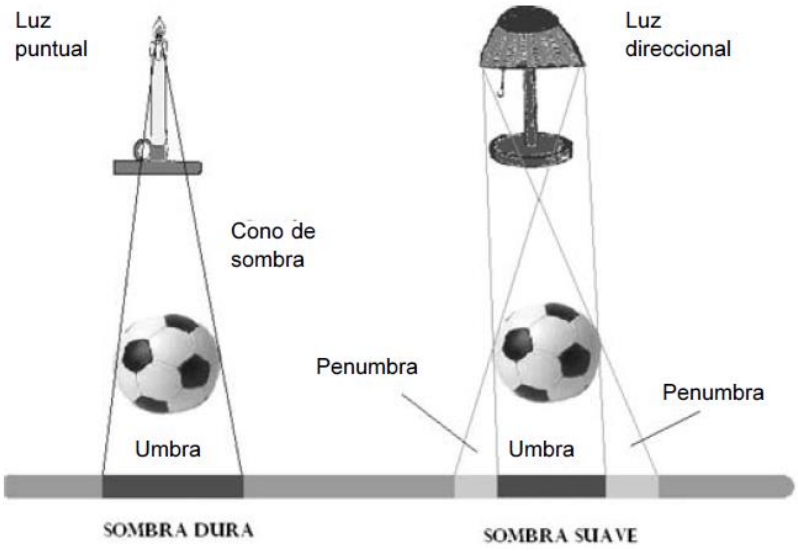


Figura 48. Diferencias entre sombras suaves y sombras duras.

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

2.6.7. Técnicas de iluminación global

Las técnicas de iluminación global persiguen calcular el valor de la intensidad de la luz en cada uno de los puntos de la escena, el valor dependerá de la interacción de las distintas fuentes con los elementos de la escena.

La iluminación de escenarios 3D es un tema indispensable si se quiere dotar de realismo a los *renders.* Por desgracia, la luz es demasiado compleja para que sea fácil de modelar con las coordenadas actuales por lo que existen diferentes técnicas que aproximan los principios de la luz que permiten obtener resultados que engaña al ojo.

Dentro del mundo de la informática gráfica, numerosos investigadores han dedicado sus trabajos a analizar, diseñar e implementar diferentes técnicas de iluminación global que den lugar a un renderizado más fotorealista y un buen rendimiento.

Se exponen a continuación algunas de las técnicas con mayor relevancia.

- ✓ **Traza de rayos(ray tracing)**

  En 1980, el ray tracing, probablemente el algoritmo de render más popular, fue presentado por *Turner Whitted*. Esta técnica consiste en lanzar rayos que intersecciones con los distintos objetos de la escena. En este algoritmo se determinan las superficies visibles en la escena que se quiere sintetizar, trazando rayos desde el observador

(cámara) hasta la escena a través del plano de imagen. Se calculan las intersecciones del rayo con los diferentes objetos de la escena y aquella interapartado que esté más cerca del observador determinará cual será el objeto visible. Realiza además, un proceso de sombreado (cálculo de la intensidad del pixel) que tiene en cuenta efectos globales de iluminación como pueden ser reflexiones, refracciones y sombras arrojadas.

Para simular los efectos de reflexión y refracción se trazan rayos sucesivamente desde el punto de interapartado que se está sombreando, dependiendo de las características del material del objeto intersecado como se muestra en la figura 49.

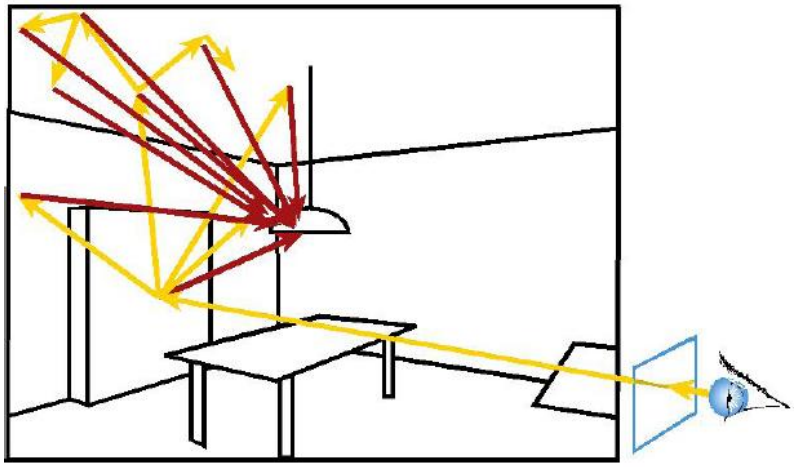


Figura 49. Técnica de trazado de rayos

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

Existen varios tipos de traza de rayos, dependiendo de la trayectoria que sigan los rayos:

- **Directa.** Los rayos comienzan desde las fuentes de luz y rebotan en los sucesivos objetos hasta llegar a la posición de la cámara. Aunque es el algoritmo de traza de rayos original, muchos de los rayos enviados no contribuirán a la imagen

- **Inversa.** Para evitar el problema anterior, una alternativa es hacer que los rayos comiencen en la cámara y recorran la escena rebotando por los diferentes objetos hasta que lleguen a las fuentes de luz. El inconveniente de este tipo de trayectoria es que no se conoce a priori cuántos rebotes se han de calcular hasta llegar a las luces.

- **Bidireccional.** Debido a que el número de rayos lanzados influye notablemente en la calidad final de la escena, una de las opciones más utilizadas en la elección de la trayectoria de los rayos es la bidireccional. Se lanzan rayos que comienzan desde la cámara y desde las fuentes de luz, esperando que gran porcentaje de ellos coincida en

algún punto de la escena, se puede observar la aplicación de este método en la figura 50.

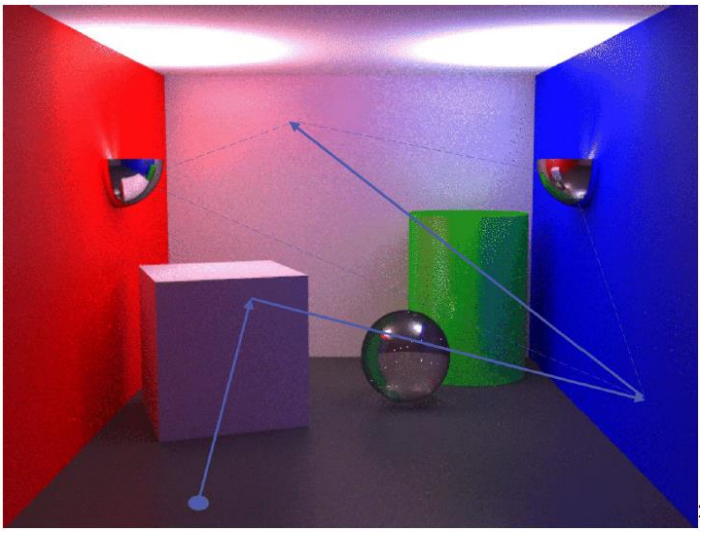


Figura 50. Traza de rayos bidireccional.

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

La técnica de la traza de rayos requiere un ***elevado tiempo de cómputo.*** Ello es debido a que se basa en muestrear un subconjunto finito, pero representativo, del conjunto infinito de caminos posibles; a más muestras, más tiempo y/o memoria es necesaria y, como contrapartida, se logrará más exactitud.

El muestreo estadístico que suele utilizarse, como *Monte Carlo,* produce resultados aproximados a la solución exacta, aunque converge en ella.

La traza de rayos proporciona resultados más realistas en superficies especulares, brillos, reflejos pero no consigue sombras suaves ni efectos de iluminación difusa. Además, este tipo de efectos de iluminación so dependientes del punto de vista, lo que hace que no se puedan generar texturas como *lightmaps* (mapas de luces) para simular brillos o reflejos en entornos dinámicos, se puede visualizar en la figura 51 una imagen generada con traza de rayos.

Figura 51. Imagen generada con traza de rayos.

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

- ✓ **Radiosidad**

  Este modelo se basa en el concepto de **Intercambio de luz entre superficies** y fue creado por investigadores de la Universidad de Cornell.

  Se muestra en la figura 52 en modelo de las Cajas de Cornell, resultado de la investigación de los creadores de este método.

Figura 52. Cajas de Cornell basado en radiosidad

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

Para calcular este cambio de energía, se subdivide el modelo en pequeñas unidades denominadas *parches*, a partir de las cuales se obtiene la distribución de luz de la escena.

En el modelo de radiosidad, cada superficie tiene asociados dos valores: la intensidad luminosa que recibe y la cantidad de energía que emite.

Una de las características que define a las técnicas de radiosidad es que proporcionan una solución de iluminación independiente del punto de vista a diferencia del modelo de trazado de rayos. Una vez calculada, se puede utilizar la iluminación resultante para renderizar desde diferentes ángulos, no obstante ésta solución es costosa tanto en tiempo como espacio de almacenamiento, se puede observar en la figura 53 la síntesis de una imagen fotorealista generado con radiosidad.

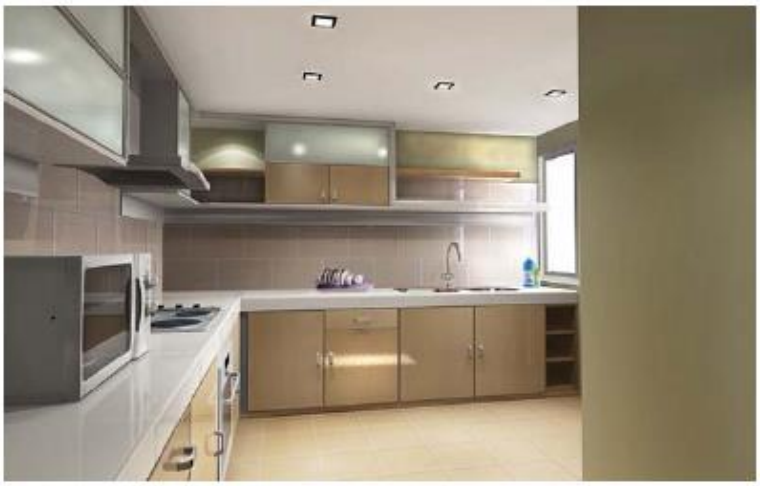


Figura 53. Imagen generada con radiosidad

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

- ✓ **Path tracing**

En 1986, *Kajiya* publicó la técnica del **path tracing.** Es uno de los algoritmos que mejor reproduce el comportamiento de la luz en todas sus facetas. Consiste en trazar rayos desde cada punto en direcciones más afines a recibir luz y usando una distribución estocástica para estimar la ecuación de la luz. Es indispensable trazar gran cantidad de rayos para conseguir resultados interesantes, por lo que, aunque se consigan resultados muy realistas, su coste computacional es muy elevado al igual que el de traza de rayos, la figura 54 muestra una imagen generada con path tracing.

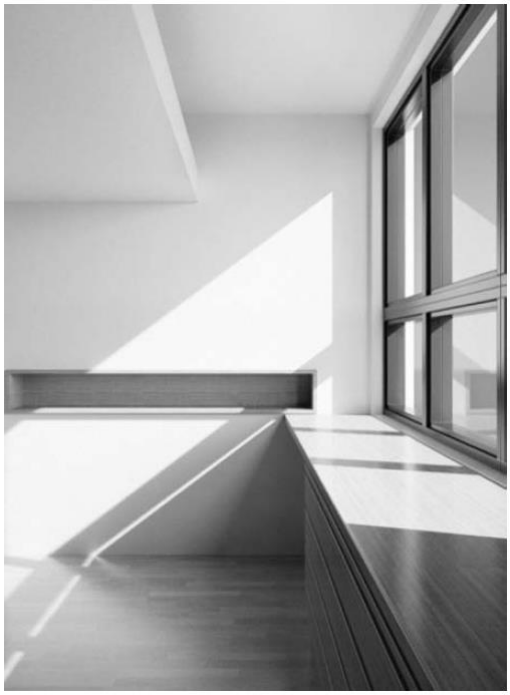


Figura 54. Imagen generada con path tracing

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

- ✓ **Caches de irradiancia**

  En 1998, *Ward* publicó un artículo, ampliado en 1992 en (con Heckbert), en el que proponía un método de recolección e interpolación que pasó a denominarse de radiancia o ***caché de irradiancia.***

  Está técnica consigue iluminación global en superficies difusas y difuso-especulares. Evita el cálculo de numerosos caminos que requiere el trazado de rayos, utilizando muestreo por importancia, por lo que mejora la eficiencia de la técnica en general.
  La reflexión sobre superficies especulares se calcula de forma semejante a la traza de rayos, al igual que la iluminación directa sobre superficies difusas.

  Por otro lado, la iluminación indirecta sobre superficies difusas se calcula en algunas intersecciones y se aproxima por interpolación a partir de otras almacenadas en la caché de irradiancia. De esta manera, se consiguen efectos de iluminación global para superficies especulares y difusas. Sin embargo, un inconveniente de esta técnica que es la interpolación puede reproducir errores donde las superficies no son planas. Por otro lado, el almacenamiento y consulta de las irradiancias pre calculadas tiene coste en tiempo y memoria, En la figura 55 se muestra un ejemplo del uso de chaches de irradiancia.

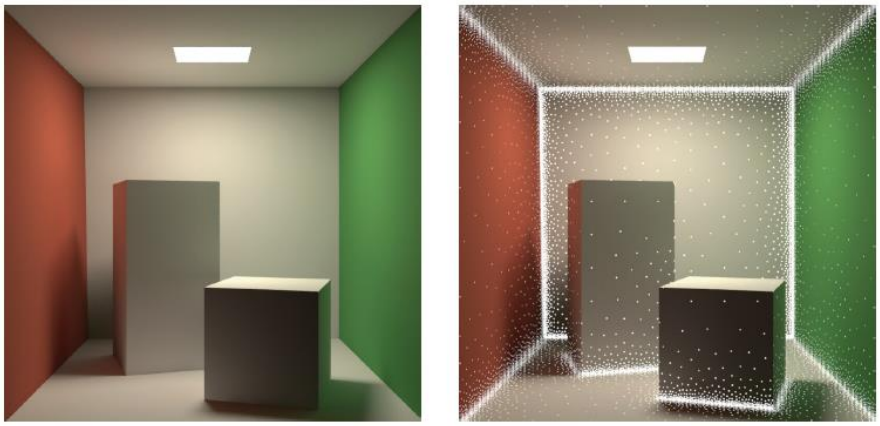


Figura 55. Imagen generada con caches de irradiancia

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

- ✓ **Mapeado de fotones(Photon mapping)**

  En 1995, *Jensen* dio a conocer otro método, el **photo mapping**. Recrea el comportamiento físico de la luz, lanzando fotones desde los focos de luz que pueden rebotar o adherirse a los objetos.

  Se trata de un algoritmo versátil, capaz de simular iluminación global, incluyendo causticas, difundir interreflexiones, medios participantes en escenas complejas.
  El algoritmo de iluminación global basado en mapas de fotones es un método de dos pasadas:

  - En el primer paso se constituye la estructura conocida como mapa de fotones emitidos desde las fuentes de lux hasta la escena. Esta estructura suele formarse por un *k-d tree* y en ella solo se almacenan los fotones que golpean objetos no especulares.
  - El segundo paso, la fase de rendering, utiliza técnicas estadísticas en la estructura de mapas de fotones para extraer información sobre el flujo de entrada y radiancia reflejada en cualquier punto de la escena.

  En la figura 56 se puede visualizar una imagen generada con *photon mapping.*

Figura 56. Imágenes generadas con photon mapping

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

El mapa de fotones es disociado de la representación geométrica de la escena. Esta es una característica clave del algoritmo, por lo que es capaz de simular iluminación global de escenas complejas que contiene millones de triángulos, a pesar de su geometría y de sus procedimientos complejos. Comparado con la técnica de radiosidad con elementos finitos, los métodos *photo mapping* tienen la ventaja de que no se requiere de mallado. Si bien es cierto que la radiosidad es el algoritmo más rápido para escenas simples difusas, a medida que la complejidad de la escena aumenta, los mapas de fotones tienden a escalar mejor y ser más rápido. Además los métodos basados en *photon mapping* pueden manipular superficies no difusas y cáusticas como se muestra en la figura 57, características que no se consigue con los métodos de radiosidad tradicionales.

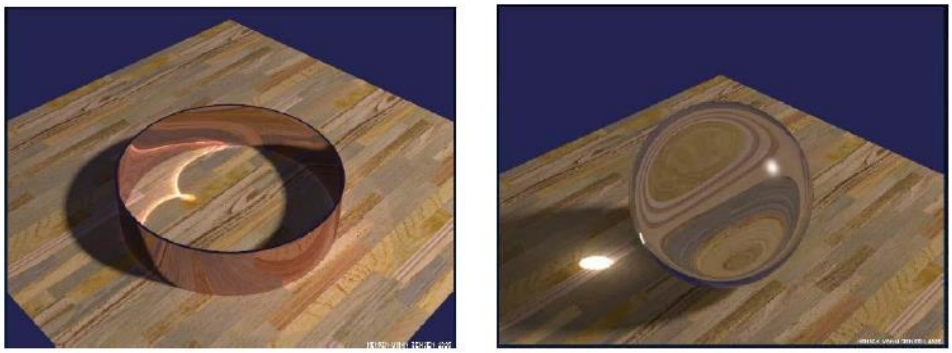


Figura 57. Causticas generadas con photon mapping

Fuente: Raya Gonzales, técnicas de iluminación para gráficos por ordenador, 2009

2.6.8. Factores a tomar en cuenta en los tiempos de renderización.

En la actualidad los programas de trazado de rayos brindan abundantes datos estadísticos con respecto a los procesos de renderización de escenas, muchos otros programas que utilizan otros algoritmos de renderizado también poseen las mismas cualidades, las estadísticas de

estos programas incluyen información referidas al procesamiento y a la renderización de la escena.

En la figura 58 se muestra las estadísticas que genera del programa Povray.

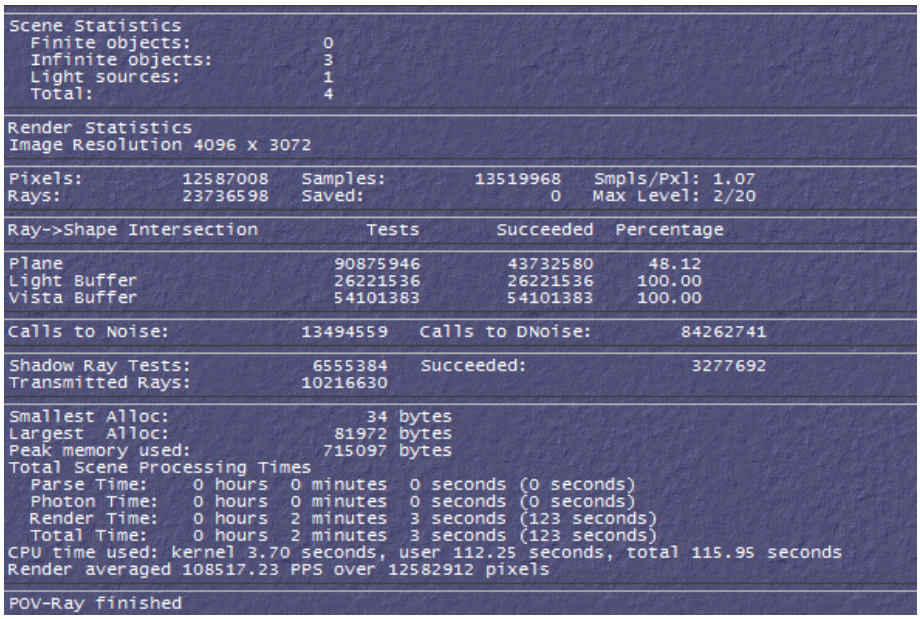


Figura 58. Estadísticas generadas por el programa Povray

Fuente: Elaboración propia

Las estadísticas y factores a tomar en cuenta por los programas de trazado de rayos se clasifican en 2 bloques, por un lado se hallan las estadísticas de la escena y por otra las estadísticas del render, en las estadísticas de la escena se pueden encontrar datos y estadísticas como son : la cantidad de primitivas tridimensionales que posee la escena, cantidad de luces propagadas en la escena, cantidad objetos entre otros dependiendo a la complejidad de elementos que posean las escenas, con respecto a las estadísticas de render se puede hallar también datos importantes como son: la resolución de las escenas, la cantidad de pixeles, cantidad de rayos, muestras, el porcentaje de las luces, las vistas propagadas en el entorno, las sombras y rayos transmitidos, el total de memoria utilizada para el renderizado y por último y no menos importante los tiempos de interpretación, el tiempo de mapeo de fotones, el tiempo de renderización y el tiempo total de renderizado, a todo ello se le suma el tiempo usado de por el CPU así como el núcleo del sistema y del usuario, esta amplia cantidad de datos y estadías también se dan en el resto de los programas que utilizan esta técnica.

La realización de la presente investigación se enfoca solo a los tiempos de renderización en sí, es por lo cual se utilizan solo los datos de los tiempos totales del proceso de renderización, así de esta manera poder determinar la incidencia de mejora con respecto a los tiempos.

CAPITULO III
APLICACIÓN DE LA METODOLOGÍA

3.1. IMPLEMENTACIÓN DEL SISTEMA CLUSTERKNOPPIX.

Como bien se definió en el marco teórico y parte de la exploración tecnológica, el sistema a implantar será el sistema Clusterknoppix, actualmente el laboratorio 4 de la escuela profesional de ingeniería de sistemas cuenta con equipos de cómputo de última generación, en la tabla 7 se especifican las características de las mismas.

Tabla 7. Características de las computadoras del laboratorio 4 de la EPIS

CARACTERISTICAS	
PROCESADOR	Intel CORE i7-3770 CPU 3.40 GHz
MODELO	HP Compaq Pro 6300 MT
DISCO DURO	1 TB
MEMORIA RAM	8 GB
PROCESADORES LOGICOS	4
NÚCLEOS	2
CPUS	8

Fuente: Elaboración propia

Es importante mencionar que todos los 40 equipos disponibles poseen las mimas características y por ende los mismos recursos computacionales, para la implementación del sistema Clusterknoppix se procedió a instalar el sistema operativo como se describe a continuación.

3.1.1. Instalación Del Nodo Maestro

La instalación del nodo maestro se realizó utilizando el disco de instalación del sistema Clusterknoppix mencionado anteriormente, la instalación del sistema requiere la creación de dos unidades o particiones del disco duro, una partición primaria la cual está destinada para la instalación del sistema operativo base y otra partición lógica que servirá para el área de intercambio, es importante mencionar que los sistemas GNU/Linux requieren esta partición de área de intercambio para la virtualización de memoria, por otro lado en el anexo 6 se especifican detalladamente los pasos que se realizaron para la instalación del nodo maestro, sin embargo no todo concluye con la instalación del nodo maestro, y a que es necesario realizar configuraciones adicionales.

3.1.2. Configuración Del Nodo Maestro

La configuración del nodo maestro consistió en realizar las siguientes configuraciones:

- Configuración de la interface de red del nodo maestro.
- Configuración del sistema de archivos MFS del nodo maestro.
- Configuración de SSH para la comunicación entre procesos.

Con respecto al primer punto se utilizó un espacio de direcciones IP de la clase C, en la tabla 8 se puede apreciar las direcciones IP asignadas a los nodos del clúster, el nodo maestro es el nodo 1.

Tabla 8. Direcciones IP asignadas a los nodos del clúster

Nodo	Dirección IP asignada
1	192 . 168 . 1 . 100 (nodo maestro)
2	192 . 168 . 1 . 101
3	192 . 168 . 1 . 102
4	192 . 168 . 1 . 103
5	192 . 168 . 1 . 104
6	192 . 168 . 1 . 105
7	192 . 168 . 1 . 106
8	192 . 168 . 1 . 107
9	192 . 168 . 1 . 108
10	192 . 168 . 1 . 109
11	192 . 168 . 1 . 110
12	192. 168. 1 . 111
13	192 . 168 . 1 . 112
14	192 . 168 . 1 . 113
15	192 . 168 . 1 . 114
16	192 . 168 . 1 . 115
17	192 . 168 . 1 . 116
18	192 . 168 . 1 . 117
19	192 . 168 . 1 . 118
20	192 . 168 . 1 . 119
21	192 . 168 . 1 . 120
22	192 . 168 . 1 . 121
23	192 . 168 . 1 . 122
24	192 . 168 . 1 . 123

Fuente: Elaboración propia

En el anexo 7 se detalla los pasos a seguir para realizar la configuración de la interface de red del nodo maestro y análogamente del resto de los nodos del clúster.

Seguidamente se realizó la configuración del sistema de archivos MFS un sistema de archivos distribuido propiamente de Openmosix el cual permite utilizar un único sistema de archivos, dicho proceso consistió en crear un directorio mfs, realizar las configuraciones necesarias editando los archivos de configuración del sistema y reiniciando los demonios necesarios, en el anexo 8 se explica detalladamente los pasos a seguir para realizar las configuraciones antes mencionadas.

También se realizaron las configuraciones de SSH para la comunicación entre procesos, las comunicaciones SSH permiten que un nodo determinado ejecute tareas y tome control de otros equipos a través de la red de área local, muy importante para ejecutar tareas en los nodos remotamente, el servicio SSH se configuró para que las comunicaciones entre los nodos se realicen de manera transparente, es decir que no se necesitará autenticación de usuario, para ello se generó una clave pública y otra privada con las cuales se pueden comunicar los procesos de manera trasparente. En el anexo 9 se detalla cómo se realizaron las configuraciones de SSH para la comunicación entre procesos.

3.1.3. Instalación del software Povray y Povmosix

La instalación del programa Povray solo se realizó en el nodo maestro ya que por medio este nodo se iniciarán los trabajos para que los nodos esclavos ayuden en la ejecución de la misma, el sistema Clusterknoppix ejecuta los trabajos desde cualquier nodo en específico no siendo necesario que el programa este instado en todos los nodos sino solamente en el nodo que iniciará la tarea, para instalar el programa Povray se descargó las fuentes del programa desde la página oficial de Povray, la versión que se instaló fue la versión 3.6, en el anexo 10 se detallan los pasos a seguir para instalar el software de Povray, en la apartado siguiente se explica la configuración del programa Povmosix.

3.1.4. Configuración de Povmosix

Antes de comenzar a utilizar el software de Povmosix se debe realizar las configuraciones iniciales, para ello se debe hacer clic en el botón  para acceder a la ventana de configuración, se debe ubicar el archivo binario del programa Povray el cual se encuentra en la dirección siguiente: /usr/local/bin/Povray, se explica cada uno

de los parámetros configurables en el programa, se muestra en la figura 59 la ventana de configuración de Povmosix.

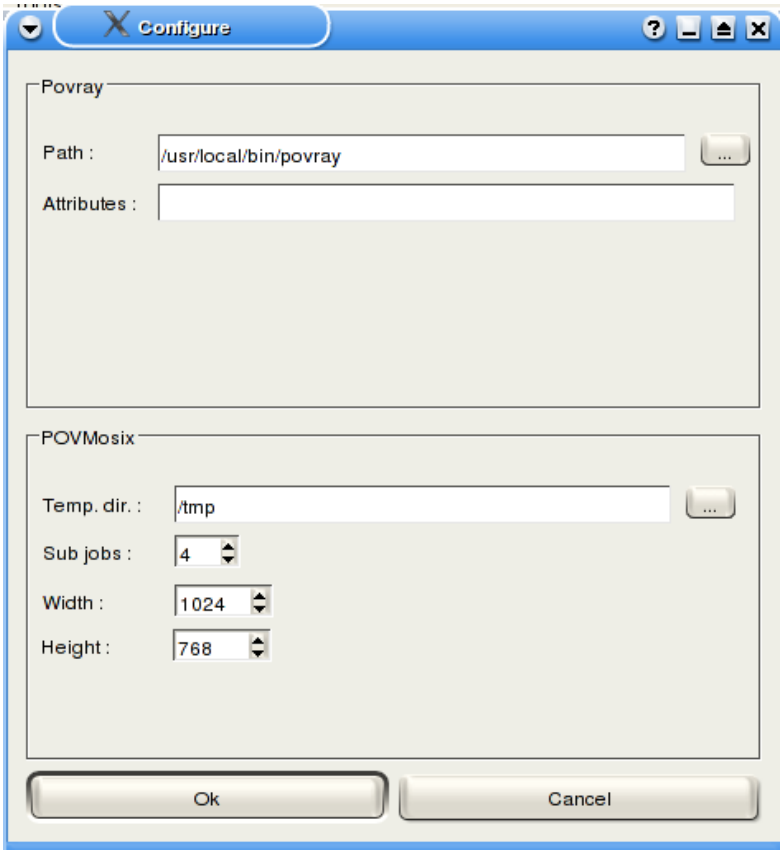


Figura 59. Ventana de configuración de Povmosix

Fuente: Elaboración propia

- **Temp. Dir (Directorio temporal)**

  Es el directorio donde se almacena temporalmente todos los archivos generados para subdividir y unir los trabajos.

- **Sub Jobs (Sub trabajos)**

  Aquí se establece la cantidad de trabajos o división de unidades de trabajo, para la presente investigación se utilizaron 96 sub trabajos para renderizar todas las escenas.

- **Width (Ancho):**

  Ancho en pixeles de la imagen resultante de la renderización, para nuestro caso es de 1024 pixeles.

- **Height (Alto):**

  Alto en pixeles de la imagen resultante de la renderización, para el presente caso es de 768 pixeles.

  Finalmente para guardar los cambios se debe hacer clic en el botón "Ok"

3.1.5. Instalación De Los Nodos Esclavos

La instalación de los nodos esclavos se hizo de la misma forma como se realizó la instalación del nodo maestro especificado en el anexo 6, la diferencia radica en que se le asignará una única dirección IP según según la tabla 8 expuesta anteriormente no siendo necesario la instalación del programa Povray ni Povmosix en los nodos esclavos.

3.2. FASE DE PRUEBAS INICIALES

3.2.1. PRUEBAS CON EL GRUPO CONTROL

Inicialmente se renderizó las escenas tridimensionales del modelo benchmark correspondientes al grupo experimental 6 (grupo control), dicha renderización se realizó sin intervención del clúster de computadoras, es decir con un solo nodo, los resultados obtenidos se muestran en la figura 60.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES
26	17	39	1059	4		6	
39	24	25	1465	4		6	
45	19	30	1170	4		6	
47	12	52	772	4		6	
48	12	50	770	4		6	
56	27	42	1662	4		6	
60	10	34	634	4		6	
78	10	35	635	4		6	
81	25	40	1540	4		6	
83	12	18	738	4		6	

Figura 60. Tiempos de renderización del grupo experimental 6

Fuente: Elaboración propia

Como se puede observar en la figura 60 la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 4, se considerará la cantidad de sub trabajos mayores cuando se utilice más de dos nodos. En la carpeta "anexo 11" del DVD "anexos" se muestra los datos de la figura 60 de manera completa , en esta hoja electrónica contiene los tiempos de renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 61 se muestran las escenas obtenidas del grupo experimental 6, adicionalmente en la carpeta "anexo 12" del DVD "anexos" se puede encontrar las escenas obtenidas del plan de pruebas iniciales de la investigación.

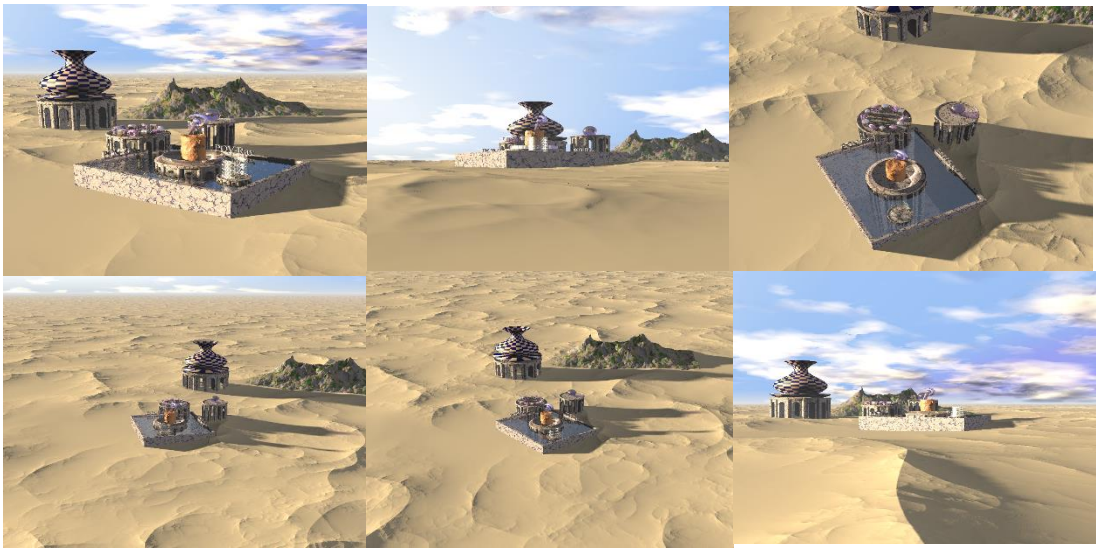


Figura 61. Escenas obtenidas del grupo control

Fuente: Elaboración propia

3.3. FASE DE PRUEBAS FINALES

Para la realización de las pruebas finales se procedió distribuir la cantidad de nodos a los grupos experimentales sujetos a observación, se puede visualizar dicha distribución en la tabla 9.

Tabla 9. Cantidad de nodos asignados a los grupos experimentales

Grupo Experimental	Cantidad de CPUS	Cantidad de Sub Procesos	Cantidad de Nodos
1	96	96	24
2	64	96	16
3	32	96	8
4	16	96	4
5	8	96	2
6	4	-	1

Fuente: Elaboración propia

3.3.1. Pruebas con el grupo experimental 5

Como se puede observar en la figura 62 la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 8, los sub trabajos son 96. En la carpeta "anexo 13" del DVD "anexos" se muestra los datos de la figura 62, se puede hallar los tiempos de renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 63 se muestran las escenas generadas y obtenidas del grupo experimental 5,

adicionalmente en la carpeta “anexo 14” del DVD anexos se muestra las escenas obtenidas del grupo experimental 5.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES
2	3	59	239	8	96	5	
9	6	20	380	8	96	5	
13	4	27	267	8	96	5	
15	7	9	429	8	96	5	
21	6	33	393	8	96	5	
55	7	45	465	8	96	5	
66	7	33	453	8	96	5	
67	6	51	411	8	96	5	
84	5	52	352	8	96	5	
86	6	50	410	8	96	5	

Figura 62. Tiempos de renderización del grupo experimental 5

Fuente: Elaboración propia

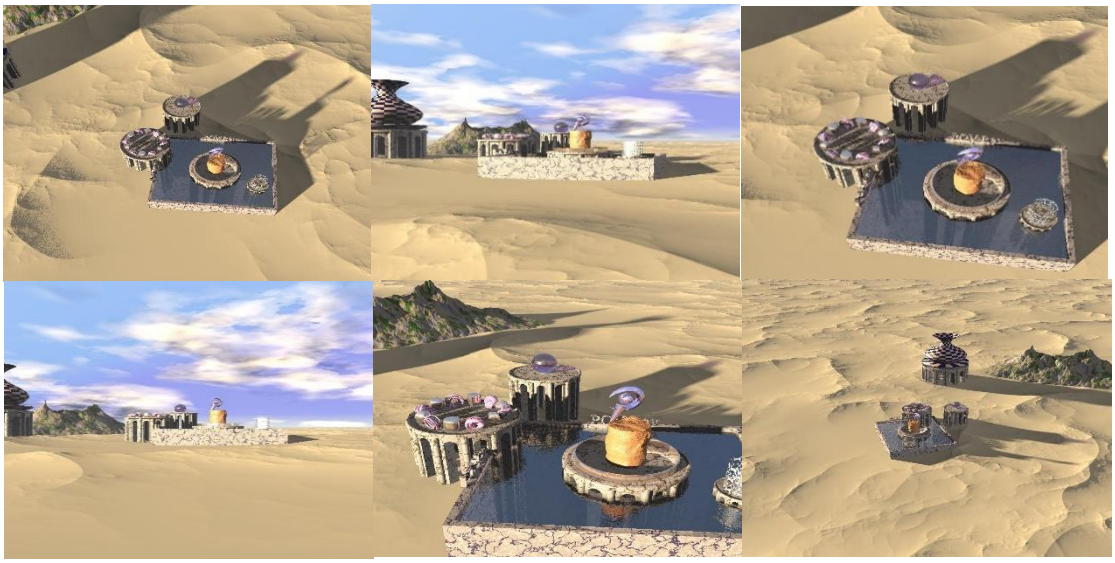


Figura 63. Escenas obtenidas del grupo experimental 5

Fuente: Elaboración propia

3.3.2. Pruebas con el grupo experimental 4

Como se puede observar en la figura 64 la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 16, los sub trabajos son 96. En la carpeta “anexo 15” del DVD “anexos” se muestra los datos de la figura 64, se pueden hallar los tiempos de

renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 65 se muestran las escenas obtenidas del grupo experimental 4, adicionalmente en la carpeta "anexo 16" del DVD "anexos" se muestra las escenas obtenidas del grupo experimental 4.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resultantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES
7	4	38	278	16	96	4	
8	3	59	239	16	96	4	
19	4	60	300	16	96	4	
24	4	9	249	16	96	4	
32	4	46	286	16	96	4	
33	4	16	256	16	96	4	
34	3	40	220	16	96	4	
35	4	30	270	16	96	4	
36	3	37	217	16	96	4	
37	3	45	225	16	96	4	

Figura 64. Tiempos de renderización del grupo experimental 4

Fuente: Elaboración propia

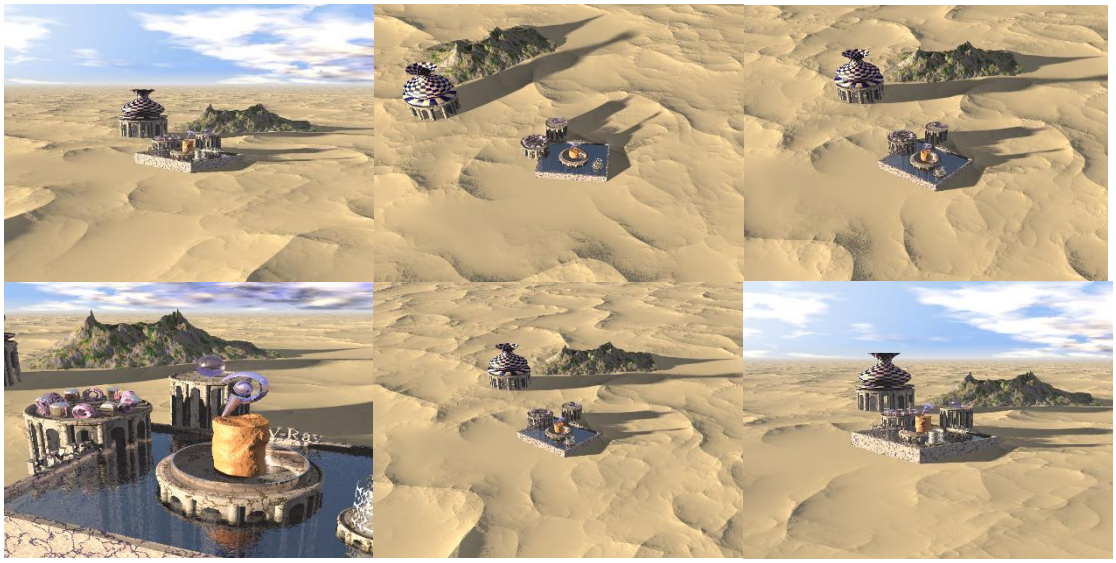


Figura 65. Escenas obtenidas del grupo experimental 4

Fuente: Elaboración propia

3.3.3. Pruebas con el grupo experimental 3

Como se puede observar en la figura 66 la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 32, los sub trabajos son 96. En la carpeta “anexo 17” del DVD “anexos” se muestra los datos de la figura 66, se puede hallar los tiempos de renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 67 se muestran las escenas obtenidas del grupo experimental 3, adicionalmente en la carpeta “anexo 18” del DVD “anexos” se muestra las escenas obtenidas del grupo experimental 3.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES	
3	1	46	106	32	96	3		
4	1	48	108	32	96	3		
12	2	1	121	32	96	3		
14	1	57	117	32	96	3		
17	3	2	182	32	96	3		
22	1	51	111	32	96	3		
23	1	58	118	32	96	3		
27	3	2	182	32	96	3		
28	1	52	112	32	96	3		
30	1	54	114	32	96	3		

Figura 66. Tiempos de renderización del grupo experimental 3

Fuente: Elaboración propia

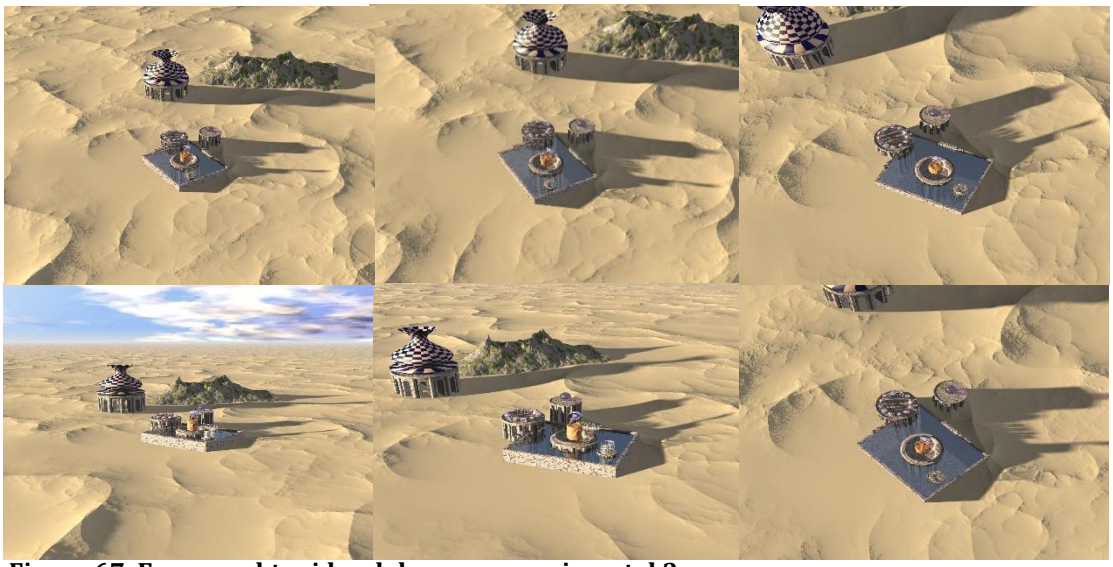


Figura 67. Escenas obtenidas del grupo experimental 3

Fuente: Elaboración propia

3.3.4. Pruebas con el grupo experimental 2

Como se puede observar en la figura 68 la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 64, los sub trabajos son 96. En la carpeta “anexo 19” del DVD “anexos” se muestra los datos de la figura 68, se pueden hallar los tiempos de renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 69 se muestran las escenas generadas y obtenidas del grupo experimental 2, adicionalmente en la carpeta “anexo 20” del DVD “anexos” se muestra las escenas obtenidas del grupo experimental 2.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES	
5	2	6	126	64	96	2		
6	2	29	149	64	96	2		
11	2	6	126	64	96	2		
16	2	0	120	64	96	2		
18	2	0	120	64	96	2		
29	1	55	115	64	96	2		
38	2	2	122	64	96	2		
53	2	2	122	64	96	2		
54	1	56	116	64	96	2		
59	2	7	127	64	96	2		

Figura 68. Tiempos de renderización del grupo experimental 2

Fuente: Elaboración propia

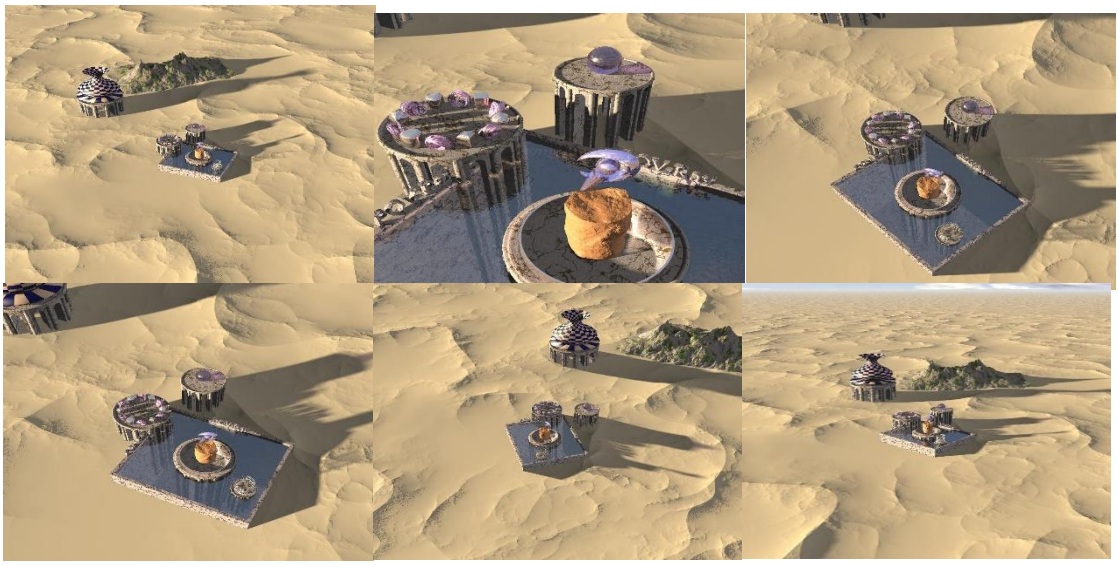


Figura 69. Escenas obtenidas del grupo experimental 2

Fuente: Elaboración propia

3.3.5. Pruebas con el grupo experimental 1

Como se puede observar en la figura 70, la cantidad de CPUs asignados a cada una de las posiciones de las cámaras es 96, los sub trabajos son 96. En la carpeta “anexo 21” del DVD “anexos” se muestra los datos de la figura 70, se pueden hallar los tiempos de renderización en minutos y segundos respectivamente así como la numeración de las escenas. En la figura 72 se muestran las escenas generadas y obtenidas del grupo experimental 1, adicionalmente en la carpeta “anexo 22” del DVD “anexos” se muestra las escenas obtenidas del grupo experimental 1.

RESULTADOS DE LAS RENDERIZACION DE POSICIÓNES DE LAS CÁMARAS

- Las posiciones de las cámaras estan limitadas en el eje "Y" mayor o igual a 5 ya que no se pretende renderizar el modelo por debajo del nivel del suelo por carecer de validez para nuestro estudio.
- Los números aleatorios están comprendidas desde el 0 hasta el 20 ya que si tomamos números mayores, el área de renderización será el menor posible, por lo cual carecería de validez al no poder ser apreciado durante el proceso de la observación.
- El tamaño de las imágenes resutantes será 1024x738 píxeles

POSICIONES DE CÁMARAS DEL MODELO BENCHMARK

Nº	Minutos	Segundos	Tiempo total en segundos	CPUS	SUB TRABAJ	GRUPO	GRUPOS EXPERIMENTALES	
1	2	36	156	96	96	1		
10	2	32	152	96	96	1		
20	2	36	156	96	96	1		
25	2	41	161	96	96	1		
40	2	5	125	96	96	1		
41	2	30	150	96	96	1		
46	2	21	141	96	96	1		
49	2	37	157	96	96	1		
50	2	18	138	96	96	1		
51	2	34	154	96	96	1		

Figura 70. Tiempos de renderización del grupo experimental 1

Fuente: Elaboración propia

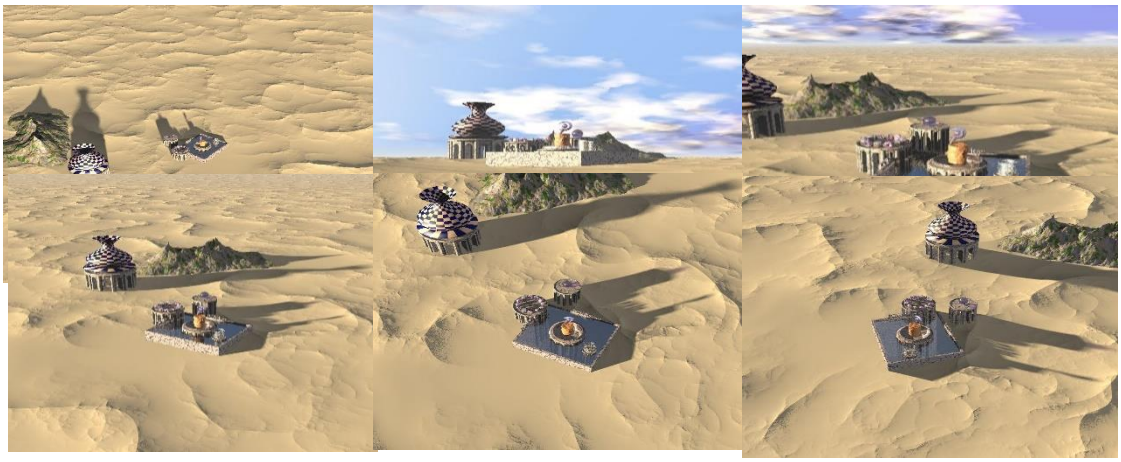


Figura 71. Imágenes obtenidas del grupo experimental 1

Fuente: Elaboración propia

3.4. Hipótesis

3.4.1. Hipótesis general

La implementación de un clúster de computadoras de alto rendimiento disminuirá notablemente el tiempo de renderización de modelos tridimensionales fotorealistas usando razado de rayos.

3.4.2. Hipótesis específicas

- ✓ La implementación del clúster de computadoras de alto rendimiento será una solución creada con la aplicación de soluciones libres, es decir software libre en su totalidad.

- ✓ La implementación del clúster de computadoras de alto rendimiento contendrá información de los procesos, porcentajes así como los tiempos por unidad de trabajo los cuales servirán para determinar los factores y el tiempo de renderización de modelos tridimensionales fotorealistas usando trazado de rayos.

- ✓ La implementación de clúster de computadoras de alto rendimiento permitirá interconectar los equipos de cómputo del laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas y trabajar en un entorno distribuido.

- ✓ La implementación de clúster de computadoras de alto rendimiento nos permitirán evaluar la reducción de los tiempos de renderización de modelos tridimensionales fotorealistas con trazado de rayos en un 60%.

CAPITULO IV

RESULTADOS

4.1. PRUEBAS ESTADÍSTICAS

4.1.1. Pruebas de Hipótesis

a) Hipótesis general del investigador

La implementación de un clúster de computadoras de alto rendimiento disminuirá notablemente el tiempo de renderización de escenas tridimensionales fotorealistas usando trazado de rayos.

H_1=Existe una diferencia significativa entre las medias correspondiente a los tiempos de renderización entre los diversos grupos de comparación.

H_0=No existen diferencias significativas entre las medias correspondientes a los tiempos de renderización entre los diversos grupos de comparación.

Determinar el valor de α (porcentaje de error)

El porcentaje de error será del 5%.

Alfa= 5%=0.05

Las pruebas estadísticas a utilizar para el presente estudio dependerán de la prueba de normalidad que se realiza a los diversos casos, dependiendo si estas siguen la distribución normal se utilizaran las pruebas paramétricas y en caso de no cumplir los casos de estudio con una distribución de normalidad se procederá a seleccionar una prueba no paramétrica.

4.1.2. Pruebas estadísticas

a) Prueba de Normalidad

Se debe corroborar que la variable aleatoria en los grupos de la presente investigación se distribuye normalmente. Para ello se utiliza la prueba de Kolmogorov-Smirnov K-S una muestra cuando las muestras son grandes (tamaño de muestra mayor a 30), o la prueba de Chapiro Wilk cuando el tamaño de la muestra es menor a 30, las hipótesis quedarán de la siguiente forma para determinar si la variable aleatoria se distribuye normalmente.

Las hipótesis de la prueba de normalidad serán las siguientes.

H_0: Los datos provienen de una distribución normal.

H_1: los datos no provienen de una distribución normal.

- ✓ Si el P- valor es mayor o igual a α, entonces se acepta H_0.
- ✓ Si el P- valor es menor a α, entonces se acepta H_1.

Se utilizó el software estadístico SPSS para obtener los estadísticos descriptivos así como la prueba de normalidad, la tabla 10 muestra el resumen de procesamiento de los casos y la tabla 11 muestra los estadísticos descriptivos de los grupos de comparación, también la tabla 12 se muestra los resultados de la prueba de normalidad exigida.

Tabla 10. Resumen de procesamiento de casos

	Casos					
	Incluidos		Excluidos		Total	
	N	Porcentaje	N	Porcentaje	N	Porcentaje
tiempo * clúster	384	100,0%	0	0,0%	384	100,0%

Fuente: Elaboración propia

Tabla 11. Descriptivos del tiempo de renderización en segundos del clúster

Tiempo

clúster	N	Media	Desv. típ.	Mínimo	Máximo
Clúster con 96 CPUs	64	153,55	7,950	125	171
Clúster con 64 CPUs	64	171,45	6,175	156	180
Clúster con 32 CPUs	64	181,59	15,214	133	218
Clúster con 16 CPUS	64	322,31	30,761	262	366
Clúster con 8 CPUs	64	674,94	117,602	541	833
No clúster	64	891,64	304,350	541	1767
Total	384	399,25	314,007	125	1767

Fuente: Elaboración propia

Ya que nuestra muestra es mayor a 30, se utilizará el Sig. De Kolmogorov-Smirnov

Tabla 12. Pruebas de normalidad para la muestra

		tiempo
N		384
Parámetros normales[a,b]	Media	399,25
	Desviación típica	314,007
Diferencias más extremas	Absoluta	,224
	Positiva	,224
	Negativa	-,198
Z de Kolmogorov-Smirnov		4,391
Sig. asintót. (bilateral)		,000

a. La distribución de contraste es la Normal.

b. Se han calculado a partir de los datos.

Fuente: Elaboración propia

En la Tabla 12, se puede visualizar que el valor de Z de Kolmogorov-Smirnov es mayor a 0.05, por lo tanto la distribución de contraste es la normal.

Entonces ya que nuestra variable tiempo se comporta normalmente y nuestra variable dependiente es numérica, se utilizará el método de ANOVA de un factor para determinar si hay diferencias significativas, es importante mencionar que al realizar el ANOVA se tendrá que verificar si el ANOVA es significativo para la comparación inter-grupos, concluido este análisis, si nuestro resultado se obtiene que: "no es significativo esta comparación", entonces no existen diferencias significativas entre los grupos, es decir se aceptaría la hipótesis nula H_0, pero si es que la comparación inter-grupos es significativo el paso siguiente será analizar si las varianzas son iguales, si el supuesto de igualdad de varianzas nos resulta positivo, es decir hay igualdad de varianzas, entonces se podrá utilizar una de las siguientes pruebas POST-HOC: DMS, Bonferroni, Sidak, Scheffé, R-E-G-WF, R-E-G-QW, S-N-K, Tukey, Tukey-b, Duncan, GT2 de Hochberg, Gabriel, Waller-Duncan, Dunnett. En caso de que nuestro resultado a la igualdad de varianzas haya sido negativa se procederá a utilizar una de las siguientes pruebas POST-HOC: T2 de Tamhane, T3 de Dunnett, Games-Howell, C de Dunnet.

Finalmente al utilizar y aplicar una de las pruebas POST-HOC independientemente de los resultados obtenidos en la fase anterior, se evaluará entonces si el valor de Sig (Significancia), si este valor es menor o igual a 0.05, se aceptará la hipótesis alterna H1, y en caso contrario se aceptaría la hipótesis nula. Finalmente se determinará el porcentaje de mejora en los tiempos de renderización entre los diversos grupos comparando los valores promedios acerca de los tiempos.

En la figura 72, se puede apreciar el esquema de pasos a seguir detalladamente para calcular el ANOVA de un factor en SPSS19 **(Bakiera, Gonzales Such, & Jornet, 2010)**.

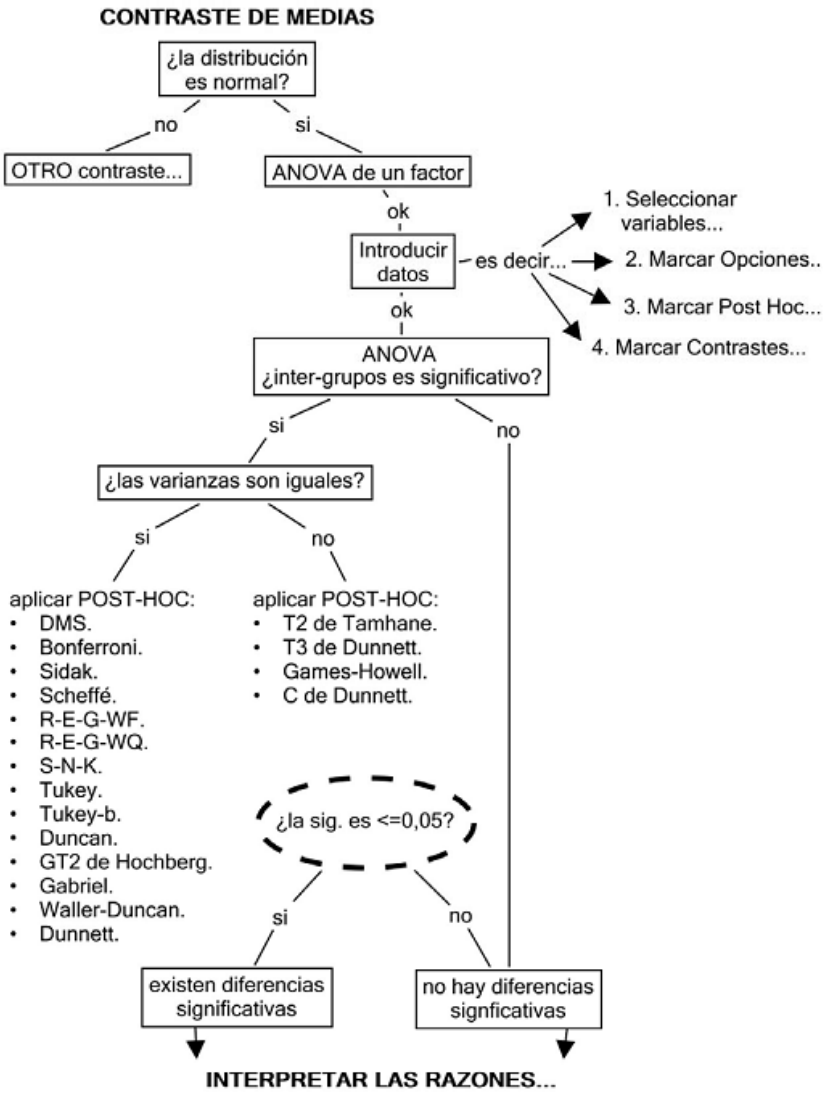


Figura 72. Pasos para aplicar ANOVA de un factor

Fuente: Bakiera, Gonzales Such, & Jornet, SPSS: Anova de un Factor, 2010.

b) ANOVA de un factor

De similar manera que los pasos anteriores, se procederá a formular nuestras hipótesis para ver si las diferencia entre grupos es o no significativa.

H_0: Los grupos de estudios son iguales.

H_1: Los grupos de estudios son diferentes.

- ✓ Si el Sig es mayor o igual a α, entonces se acepta H_0.
- ✓ Si el Sig. es menor a α, entonces se acepta H_1.

Una vez realizado el análisis de varianza para los tiempos de renderización (segundos), en el software de SPSS, se obtuvieron los siguientes resultados que se muestran detalladamente en la tabla 13.

Tabla 13. Resultados de ANOVA de un factor.

	Suma de cuadrados	gl	Media cuadrática	F	Sig.
Inter-grupos	30976460,107	5	6195292,021	345,020	,000
Intra-grupos	6787499,391	378	17956,348		
Total	37763959,497	383			

Fuente: Elaboración propia

Como se puede observar en la tabla 13, el p-valor de comparación de inter-grupos es menor a 0.05, entonces se comprueba que los tres grupos de estudio son distintos por lo cual se acepta H_1.

c) Igualdad de varianzas

Utilizaremos la prueba de Levene, para corroborar la igualdad de varianza entre los grupos, de similar manera se plantea las hipótesis.

H_0: Las varianzas son iguales.

H_1: Las varianzas son diferentes.

- ✓ Si P-valor es mayor o igual a α, entonces se acepta H_0.
- ✓ SI P-valor es menor a α, entonces se acepta H_1.

En la tabla 14 se puede visualizar la prueba de homogeneidad de varianzas, utilizando el estadístico de Levene en SPSS.

Tabla 14. Prueba de homogeneidad de varianzas

Estadístico de Levene	gl1	gl2	Sig.
91,041	5	378	,000

Fuente: Elaboración propia

Como se puede observar en la tabla 14 el valor de Sig es menor a 0.05, por lo tanto se concluye que las varianzas no son iguales, es decir se acepta H_1.

d) Pruebas POST-HOC

Para poder visualizar las diferencias significativas entre los grupos de estudio, se procede a utilizar la prueba T2 de Tamhale.

En la tabla 15 se puede visualizar las comparaciones múltiples entre los diversos grupos.

Tabla 15. Comparaciones múltiples entre los diversos grupos

Tamhane

(I) clúster	(J) clúster	Diferencia de medias (I-J)	Error típico	Sig.	Intervalo de confianza al 95%	
					Límite inferior	Límite superior
Clúster con 96 CPUs	Clúster con 64 CPUs	-17,906*	1,258	,000	-21,67*	-14,15
	Clúster con 32 CPUs	-28,047*	2,146	,000	-34,49*	-21,60
	Clúster con 16 CPUS	-168,766*	3,971	,000	-180,80*	-156,74
	Clúster con 8 CPUs	-521,391*	14,734	,000	-566,21*	-476,57
	No clúster	-738,094*	38,057	,000	-853,89*	-622,29
Clúster con 64 CPUs	Clúster con 96 CPUs	17,906*	1,258	,000	14,15*	21,67
	Clúster con 32 CPUs	-10,141*	2,052	,000	-16,33*	-3,95
	Clúster con 16 CPUS	-150,859*	3,922	,000	-162,76*	-138,96
	Clúster con 8 CPUs	-503,484*	14,721	,000	-548,27*	-458,70
	No clúster	-720,188*	38,052	,000	-835,98*	-604,40

	Clúster con 96 CPUs	28,047*	2,146	,000	21,60*	34,49
	Clúster con 64 CPUs	10,141*	2,052	,000	3,95*	16,33
Clúster con 32 CPUs	Clúster con 16 CPUS	-140,719*	4,290	,000	-153,61*	-127,82
	Clúster con 8 CPUs	-493,344*	14,823	,000	-538,39*	-448,30
	No clúster	-710,047*	38,091	,000	-825,94*	-594,16
	Clúster con 96 CPUs	168,766*	3,971	,000	156,74*	180,80
	Clúster con 64 CPUs	150,859*	3,922	,000	138,96*	162,76
Clúster con 16 CPUS	Clúster con 32 CPUs	140,719*	4,290	,000	127,82*	153,61
	Clúster con 8 CPUs	-352,625*	15,195	,000	-398,65*	-306,60
	No clúster	-569,328*	38,238	,000	-685,59*	-453,06
	Clúster con 96 CPUs	521,391*	14,734	,000	476,57*	566,21
	Clúster con 64 CPUs	503,484*	14,721	,000	458,70*	548,27
Clúster con 8 CPUs	Clúster con 32 CPUs	493,344*	14,823	,000	448,30*	538,39
	Clúster con 16 CPUS	352,625*	15,195	,000	306,60*	398,65
	No clúster	-216,703*	40,785	,000	-339,72*	-93,69
	Clúster con 96 CPUs	738,094*	38,057	,000	622,29*	853,89
	Clúster con 64 CPUs	720,188*	38,052	,000	604,40*	835,98
No clúster	Clúster con 32 CPUs	710,047*	38,091	,000	594,16*	825,94
	Clúster con 16 CPUS	569,328*	38,238	,000	453,06*	685,59
	Clúster con 8 CPUs	216,703*	40,785	,000	93,69*	339,72

***. La diferencia de medias es significativa al nivel 0.05.**

Fuente: Elaboración propia

Se puede observar en la tabla 15 que el valor de Sig. (Significancia) para las diversas comparaciones entre grupos en orden directo e inverso es menor a 0.05, este orden está determinado por dos componentes de comparación: "i" que representa individualmente a cada uno de los grupos experimentales y "j" que representa a los otros grupos experimentales, así de esta manera se realiza la comparativa entre un grupo experimental en concreto "i" con todo los diversos grupos experimentales "j", los valores de sig. muestran que la diferencia en los tiempos de renderización es significativo, es decir hay diferencias significativas entre todos los grupos

experimentales, por otro lado los límites superior e inferior muestran el aumento o reducción de los tiempos en segundos entre los diversos grupos de comparación, seguidamente, seguidamente se determina los porcentajes de mejora en el apartado siguiente.

e) Porcentajes de mejora en los tiempos de renderización.

Para determinar los porcentajes de mejora con respecto a los tiempos de renderización se toma los valores medios de los tiempos de renderización de los grupos calculados en la tabla 11, en la tabla 16 se muestra el porcentaje de mejora en la renderización de las escenas tridimensionales entre todas las posibles combinaciones entre los grupos experimentales.

Tabla 16. Porcentajes de mejora en los tiempos de renderización.

Grupos Experimentales	Grupos de comparación	Media de tiempo de renderización (Segundos)	Porcentaje mejora de tiempos de renderización
Grupo Control 6 (4 CPUs)	Grupo 5	674,94	- 24,30%
	Grupo 4	322,31	- 63,85%
	Grupo 3	181,59	- 79,63%
	Grupo 2	171,45	-80,77%
	Grupo 1	153,55	-82,77%
Grupo Experimental 5 (8 CPUs)	Grupo 6	891,64	32,10%
	Grupo 4	322,31	-52,25%
	Grupo 3	181,59	-73,09%
	Grupo 2	171,45	-74,59%
	Grupo 1	153,55	-62,43%
Grupo Experimental 4 (16 CPUs)	Grupo 6	891,64	176,64%
	Grupo 5	674,94	109,40%
	Grupo 3	181,59	-43,66%
	Grupo 2	171,45	-46.80%
	Grupo 1	153,55	-52,35%
Grupo Experimental 3 (32 CPUs)	Grupo 6	891,64	391,01%
	Grupo 5	674,94	271,68%
	Grupo 4	322,31	77,49%
	Grupo 2	171,45	-5,58%
	Grupo 1	153,55	-15,44%
Grupo Experimental 2 (64 CPUs)	Grupo 6	891,64	420,05%
	Grupo 5	674,94	293,66%
	Grupo 4	322,31	87,99%
	Grupo 3	181,59	5,91%
	Grupo 1	153,55	-10,44%

Grupo Experimental 1 (96 CPUs)	Grupo 6	891,64	480,68%
	Grupo 5	674,94	339,55%
	Grupo 4	322,31	109,90%
	Grupo 3	181,59	18.26%
	Grupo 2	171,45	11,65%

Fuente: Elaboración propia

En la tabla 16 se puede observar que los porcentajes de los tiempos de renderización en algunos casos aumentan y en otros disminuyen, cuando los porcentajes tienen signos positivos se halla un incremento o aumento en los tiempos de renderización, y en caso de que muestren signo negativo es señal de que se ha reducido en dicho porcentaje.

4.6.1. Discusión de Resultados

Antes de desarrollar la discusión de resultados se realiza la contrastación del objetivo principal de la investigación, si éste fue alcanzado mediante el cumplimiento de todos los objetivos específicos a lo largo del desarrollo de la investigación, también se realiza la contrastación de la hipótesis general como las hipótesis específicas para aceptar o rechazar lo expuesto en el planteamiento metodológico.

a) Contrastación de objetivos

Objetivo General

- Evaluar la influencia de un clúster de computadoras de alto rendimiento en el tiempo de renderización de modelos 3D fotorealistas en la Universidad Nacional José María Arguedas.

Objetivos específicos

- ✓ Realizar una exploración de las herramientas tecnológicas para implementar un clúster de computadoras de alto rendimiento así como las herramientas para la renderización de modelos tridimensionales fotorealistas basadas para el trazado de rayos y seleccionar la más adecuada para la investigación.

- ✓ Realizar un estudio sobre las herramientas y factores para determinar los tiempos de procesamiento en la renderización de modelos tridimensionales fotorealistas utilizando trazado de rayos.

- ✓ Implementar un clúster de computadoras de alto rendimiento en el laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas para así determinar y evaluar

los recursos computacionales del clúster como memorias y procesadores asignados.

- ✓ Evaluar los tiempos de renderización de modelos tridimensionales fotorealistas con trazado de rayos.

Se corroborara el primer objetivo específico, en la presente investigación se realizó la exploración tecnológica en el apartado 2.5 de todas las herramientas para implantar un clúster de computadoras de alto rendimiento así como las aplicaciones para renderizar modelos y escenas tridimensionales por lo que **el primer objetivo específico queda cumplido.**

Con respecto al segundo objetivo específico se utilizó como herramienta el software de Povray como resultado de la exploración tecnológica para determinar los factores a tomar en cuenta para la renderización, sin embargo no se consideraron todos estos factores por no necesitar de ellos para la evaluación de los tiempos de renderización, por lo que **el segundo objetivo específico queda cumplido**.

El tercer objetivo específico queda comprobado ya que se implantó un clúster de computadoras de alto rendimiento que bajo la exploración tecnológica nos permitió evaluar los recursos computacionales del clúster como memoria y procesadores, por lo **cual el tercer objetivo específico queda cumplido**.

Finalmente el cuarto objetivo específico queda comprobado de la misma manera, a razón de que se logró evaluar y analizar estadísticamente los resultados en el apartado 4.1, **el cuarto objetivo específico queda cumplido.**

En su defecto ya que se cumplió con los objetivos específicos planteados se da por **cumplido nuestro objetivo general.**

b) Contrastación de hipótesis

Hipótesis general

La implementación de un clúster de computadoras de alto rendimiento disminuirá notablemente el tiempo de renderización de modelos tridimensionales fotorealistas usando razado de rayos.

Hipótesis Específicas

- ✓ La implementación del clúster de computadoras de alto rendimiento será una solución creada con la aplicación de soluciones libres, es decir software libre en su totalidad.

- ✓ La implementación del clúster de computadoras de alto rendimiento contendrá información de los procesos, porcentajes así como los tiempos por unidad de trabajo los cuales servirán para determinar los factores y el tiempo de renderización de modelos tridimensionales fotorealistas usando trazado de rayos.
- ✓ La implementación de clúster de computadoras de alto rendimiento permitirá interconectar los equipos de cómputo del laboratorio 4 de la Escuela Profesional de Ingeniería de Sistemas y trabajar en un entorno distribuido.

- ✓ La implementación de clúster de computadoras de alto rendimiento nos permitirán evaluar la reducción de los tiempos de renderización de modelos tridimensionales fotorealistas con trazado de rayos en un 60%.

Para corroborar la aceptación o rechazo de la primera hipótesis se incide en la exploración tecnológica desarrollada en la apartado 2.5, en ella se puede encontrar que existe una amplia gama de aplicaciones de software libre para implementar clúster de computadores de alto rendimiento así como software para trazado de rayos, todas y cada una de ellas se sometieron a comparaciones entre sus principales características, ya que la solución final presentada en el capítulo III, está constituida en su totalidad por soluciones de software libre. Por lo que **se acepta la primera hipótesis específica.**

Con respecto a la segunda hipótesis específica en el apartado 1.8.1 se puede visualizar el programa Openmosixview, éste programa permite monitorear el clúster Openmosix y brinda la información necesaria acerca del clúster, entre las informaciones brindadas se hallan: los identificadores de cada nodo, las direcciones IP de cada nodo, los procesos de cada nodo, el balanceo de carga de cada nodo, la cantidad de memoria de cada nodo, la cantidad de CPUs de cada nodo, adicionalmente también se puede hallar datos importante como son la capacidad total del clúster con respecto a memoria RAM y CPUs, gracias a todo ello **se acepta la segunda hipótesis especifica**.

Con respecto a la tercera hipótesis específica, solo se interconectó los equipos de cómputo del laboratorio 4, siendo no necesario utilizar todos los equipos ya que la cantidad prevista en el apartado 3.3 fue de 24 equipos de cómputo el con el cual se renderizó el grupo experimental 1, según la tercera hipótesis se debió interconectar

todos los equipos de cómputo del laboratorio 4, esto no fue impedimento ya que se pudo realizar las pruebas del grupo experimental 6 sin ningún problema con una menor cantidad de nodos, sin embargo **se rechaza la tercera hipótesis específica** ya que no se utilizó todos los equipos de cómputo del laboratorio 4.

La cuarta hipótesis específica requiere un poco más de análisis, la aceptabilidad o rechazo de esta hipótesis se puede analizar con lo desarrollado en la tabla 16 de la sección 4.1.2. ya que con respecto a las pruebas la cantidad de nodos asignados inicialmente a la renderización no fue lo suficiente para reducir drásticamente los tiempos de renderización, pero a medida que incrementó la cantidad de CPUs, se pudo observar que el porcentaje de reducción llega hasta un 52% y como es de esperar si se continua agregando más CPUs al clúster, la reducción en el tiempo de renderización es cada vez menor con respecto al porcentaje anterior hasta llegar al 5%, con ello el porcentaje de mejora total para aceptar o rechazar la cuarta hipótesis específica se basa en la comparación de tiempos entre los grupos experimentales 1 y 6 constituidos por 96 y 4 CPUs respectivamente.

La reducción total con respecto al grupo control con 4 CPUs y el grupo Experimental con 96 CPUs es de 82,77%.

Por lo que se rechaza la cuarta hipótesis específica, ya que el porcentaje de reducción es de un 82% un valor mucho mayor al que se esperaba.

La hipótesis general de la investigación se rechaza ya que el porcentaje de mejora es mayor de lo esperado, con ello se comprueba que los tiempos de renderización utilizando clúster de computadores con 96 CPUs reducen drásticamente los tiempos de renderización hasta un 82.77%, sin embargo cabe aclarar que inicialmente al ir agregando 1 nodo más al clúster la mejora es notable llegando al 24.30%, seguidamente se agrega 2 nodos más al clúster y los tiempos de renderización se reducen muy notablemente llegando a un 63.85%, a partir de este punto la mejora en los tiempos de renderización obviamente son significantes pero sin mucho incremento con respecto a el porcentaje obtenido anteriormente con el otro grupo experimental ya que con 4 nodos más, la mejora es de 79.63%, con 8 nodos más, el 80.77% y finalmente con 8 nodos más, el 82.77% , éstos dos últimos incrementos son solo del 2 o 3%, una cifra no muy notable.

CONCLUSIONES

- Al finalizar la exploración tecnológica de la investigación se llegó a la conclusión de que existe una amplia y muy variada colección de programas y paquetes de software libre disponibles que pueden utilizarse para implantar soluciones de clúster de computadoras de alto rendimiento, se concluyó la exploración tecnológica con la comparativa sobre estas diversas soluciones y la elección del sistema Clusterknoppix. Por otro lado también se realizó la exploración tecnológica sobre las herramientas para renderizar escenas tridimensionales con trazado de rayos, de la misma manera se realizó una comparativa entre las diversas soluciones encontradas y se escogió las herramientas Povray y Povmosix para realizar el trazado de rayos.

- Al concluir el estudio realizado sobre los factores a tomar en cuenta para determinar los tiempos de procesamiento en la renderización de modelos tridimensionales fotorealistas utilizando trazado de rayos se concluyó que los factores se clasifican en dos grandes grupos, por un lado se halla las estadísticas con respecto a información del modelo, el cual contiene la cantidad de objetos, luces, rayos entre otros, por otro lado se hallan los tiempos de renderización propios, se tomó solo el tiempo de renderización total por ser el más importante y suficiente para determinar la incidencia de mejora.

- Se logró implantar un clúster de computadoras de alto rendimiento en el laboratorio 4 de la escuela profesional de ingeniería de sistemas de la Universidad Nacional José María Arguedas, el clúster utiliza el sistema Clusterknoppix junto con su monitor de recursos Openmosixview el cual permite evaluar y obtener los recursos computacionales del clúster.

- Se evaluó los tiempos de renderización con la renderización de 384 escenas tridimensionales tomadas al azar del modelo benchmark con la utilización del clúster de computadoras de alto rendimiento y el software Povmosix, asimismo se determinó los porcentajes de incidencia de mejora éntrelos diversos grupos experimentales de la investigación.

- Los clúster de computadoras de alto rendimiento reducen drásticamente los tiempos de renderización de modelos y escenas tridimensionales fotorealistas utilizando la técnica de trazado de rayos, la influencia de mejora se puede visualizar claramente cuando se utiliza un gran número de ordenadores, más por el contrario si se utiliza unos pocos ordenadores la incidencia de mejora es muy poca, pero a medida que se incrementa una buena cantidad de nodos al clúster se puede ver muy notablemente esta diferencia en los tiempos de renderización llegando a ser considerada como drástica con un 82.77%, pero llega también el punto en el que no se puede reducir más el tiempo de renderización al ir incrementado

muchos más nodos esto se debe a que la coordinación de los trabajos se vuelve más complicada , por ende los tiempos de renderización disminuyen en muy poca proporción llegando la reducción mínima de nuestro caso de estudio en 2.00% entre el grupo experimental 1 y 2 una cifra no tan notable.

- A lo largo de todo este tiempo el hombre ha ido construyendo súper ordenadores que le ayuden a resolver problemas cuya complejidad computacional suele ser elevada, los clúster de computadoras de alto rendimiento son prueba de ello, la solución en clúster aún se mantiene vigente y es utilizada en el mundo de la computación gráfica así como en otros sectores para acelerar los cálculos como se muestra en el anexo 5(Top 500 supercomputers).

RECOMENDACIONES

- La estadística es una gran herramienta para procesar los datos de una investigación y llegar a conclusiones con el apoyo de software estadístico como SPSS, las investigaciones experimentales exigen que deban cumplirse muchas pruebas y requisitos como son la homogeneidad de la muestra, que los datos cumplan con una distribución normal, que los datos cumplan con la homogeneidad de varianzas, todo esto debe ser cumplido estrictamente antes de poder realizar cualquier análisis, sin embargo no siempre se pueden cumplir estos requisitos, para ello se recomienda utilizar pruebas no paramétricas para acercarse a los resultados. Existe también el caso de que si nuestros datos no cumplen con la distribución normal se puedan transformar dichos datos con una función logarítmica para que sigan una distribución normal, se recomienda considerar estas soluciones.

- El sistema Clusterknoppix es un sistema en donde se pueden realizar muchas configuraciones adicionales, no se abordó estos temas por no estar dirigida nuestra investigación en ese sentido pero para estudios enfocados hacia otras ramas puede ser necesario abordar dichas configuraciones adicionales, se recomienda revisar la documentación respectiva expuesta en la bibliografía sobre el sistema Clusterknoppix.

- En algunos casos puede no existir software que trabaje en entornos distribuidos, para realizar estudios, como se vio en el marco teórico los clúster de computadoras manejan un lenguaje de programación denominado MPI, en tal sentido se recomienda utilizar este lenguaje para desarrollar aplicaciones a medida y según se necesite para entornos distribuidos ya que las soluciones en clúster en su totalidad soportan este lenguaje.

BIBLIOGRAFÍA

A. Apon, R. Buya, & H. Jin. (Enero de 2002). Cluster Computing and Applications. *Encyclopedia of Science and Tecnology*, 87-125.

Arrollo, R., Nievas, F., & Pino, O. (2004). *Los clusters como Plataforma de Procesamiento Paralelo*. Obtenido de http://usuarios.lycos.es/lacaraoculta/descargas/Clusters_definitivo.pdf

Bakiera, M., Gonzales Such, J., & Jornet, J. (20 de Enero de 2010). SPSS: Anova de un Factor. Valencia, Valencia, España.

Banikazemi, M., Liu, J., & Sadayappan, P. (Setiembre de 2001). Implementing Tread Marks over Virtual Interface Architecture on Myrinet and Gigabit Ethernet: Challenges, Design Experience and Performance Evaluation. *International Conference on Parallel Processing(ICPP)*, 167-174.

Barroso, L. A., Dean, J., & Holzle, U. (2003). Web Search for a Planet. *The Google Cluster Architecure*, 22-28.

Bilbao Eguia, J. (2004). Medidas de rendimiento. En *Arquitectura de Computadoras* (págs. 28-36). Lima.

Branch Bedoya, J. W., & Mesa Munera, A. (2008). Implementación de un cluster homogéneo para la solución de problemas de alta complejidad computacional. *Avances en Sistemas e Informática, 5*(3), 1.

Buyya, R. (1999). *High Performance Cluster Computing: Architectures and Systems.* Prentice Hall.

Calvin, X., Hartner, M., & Hansen, C. (2003). *Cluster Based Interactive Volume Rendering With Simian*, 3-17.

Cameron, D., & Regnier, G. (2002). *Virtual Interface Architecture.* Intel Press.

Castillo Castillo, J., & Olivera Acosta, R. B. (2006). *Implementación de un Cluster Openmosix para Cómputo Científico.* México.

Catalán, M. (2004). *El manual para clustering con Openmosix*. Obtenido de http://www.alumnes.eps.udl.es/b4767512/07.Openmosix/oM_como.html

Chen, H., Wyckoff, P., & Moor, K. (Agosto de 2000). Cost/Performance Evaluation of Gigabit Ethernet and Myrinet as Cluster Interconnects. *Conference on Network and Application Performance (OPNETWORK)*, 50-62.

Chirinov, R. (4 de Marzo de 2003). *Proyecto Cluster Openmosix(Linux)*. Obtenido de http://www.noticias3d.com/articulo.asp?idarticulo=248pag=4

Cohen, M., & Wallace, J. (1993). *Radiosity and realistic image synthesis.* Morgan Kauffman.

Corrrea, M. (2 de Enero de 1999). *Instalación y Uso del Cluster Beowulf en CecalCULA*. Obtenido de http://www.cecalc.ula.ve/documentacion/tutoriales/beowulf/node1.html

Deitel, H. M. (1987). *Introducción a los Sistemas Operativos.* Mexico: Addison-Wesley Iberoamericana.

Dormido, S., Hernández, R., Ros, S., & Sánchez, J. (2003). *Procesamiento Paralelo Teoría y Programación.* Madrid.

Eguia, B. (2006). Medidas de rendimiento. En *Arquitectura de computadoras* (págs. 22-29). Lima.

Fernandez, J. (13 de Junio de 2003). Sistema GRID para el Render de Escenas 3D. La mancha, Castilla, España.

Foley, J. (1997). *Computer Graphics, Principles and Practice in C.* Addison-Wesley.

Forum, Message Passing Interface. (12 de Enero de 2006). *MPI Forum*. Obtenido de http://www.mpi-forum.org

Gorlatch, S., Fragopoulou, P., & Priol, T. (2008). *Grid Computing: Archiveents and Prospects.* Canada: Springer.

Hsieh, J., Leng, T., Mashayekhi, V., & Rooholamini, R. (Noviembre de 2000). Architectural and Performance Evaluation of GigaNet and Interconnects on Clusters os Small-Scale SMP Servers. *ACM/IEEE Conference on Supercomputing(SC2000)*.

Lan, Z., & Deshikachar, P. (Diciembre de China). Performance Analysis of a Large-Scale Cosmology Application on Three Cluster Systems. *IEEE International Conference on Cluster Computing*, 56-63.

Llorens, E., & Peña, M. (2002). *Computación Cluster Beowulf*. Obtenido de http://personals.ac.upc.edu/enric/PFC/Beowulf/beowulf.html

Patterson, D. A., & Hennessy, J. L. (2000). *Estructura y Diseño de Computadoras.* Reverté.

Pérez, M. (2001). *Arquitecturas paralelas*. Obtenido de http://www.dragones.org/Biblioteca/Articulos/ArquitecturaParalela2.pdf

Pharr, M., & Humphreys, G. (2004). *Physically Based Rendering From Theory to implementation.* Morgan Kau_man.

Porreza, H., Eskicioglu, R., & Graham, P. C. (Mayo de 2004). Preliminary Performance Assessment of Four Cluster Interconnects on Identical Hardware. *18th International Symposium on High Performance Computing System and Applications(HPCS2004)*.

Raya Gonzales, L. (2009). *Técnicas de Iluminación Para Gráficos por Ordenador.* Mexico.

Robbins, D. (s.f.). *Openmosix*. Obtenido de http://www.intel.com/cd/ids/developer/asmo%ADna/eng/20449.htm

Sánchez, E., & Heider, Y. (2007). *Cluster and Grid Computing.* Mexico: Prentice Hall.

Silberschatz, A., Galvin, P. B., & Gagne, G. (2006). *Fundamentos de sistemas operativos.* España: Mc Graw Hill.

Sterling, T., Salmon, J., & Becker, D. (1999). *How to Build a Beoful: a guide to the implementation and aplication of PC cluster.* Cambridge: The MIT Press.

Sun . (22 de Enero de 2006). Obtenido de Sun Cluster Architecture: http://www.sun.com./software/grid/SunCLusterArchitecture.pdf

Tanembaum, A. S. (1996). *Sistemas Operativos Distribuidos.* Mexico: Prentice Hall.

Top 500 Supercomputers. (17 de Junio de 2013). *List Of Top 500 Supercomputers*. Obtenido de http://s.top500.org/static/lists/2013/06/TOP500_201306_Poster.png

Torrealba Martinez, L. M. (Diciembre de 2002). Construción de un Cluster Mosix con Pruebas con Simulación de Halo. Oxaca, Huajuapan, Mexico.

Vander Stenn, A. J. (Octubre de 2003). An Evaluation of Some Beowulf Clusters. *Cluster Computing*, 287-297.

Whitted, T. (1980). An Inproved Ilumination Model For Shaded Display. *Comunications of the ACM 23*, 343-349.

Wikipedia. (8 de Marzo de 2013). *Fotorealismo*. Obtenido de Wikipedia: http://es.wikipedia.org/wiki/Fotorrealismo

Wikipedia. (7 de 02 de 2014). Obtenido de http://en.wikipedia.org/wiki/List_of_ray_tracing_software

ANEXOS

ANEXO 1

POSICIONES DE LAS ESCENAS ALEATORIAS

ANEXO 2

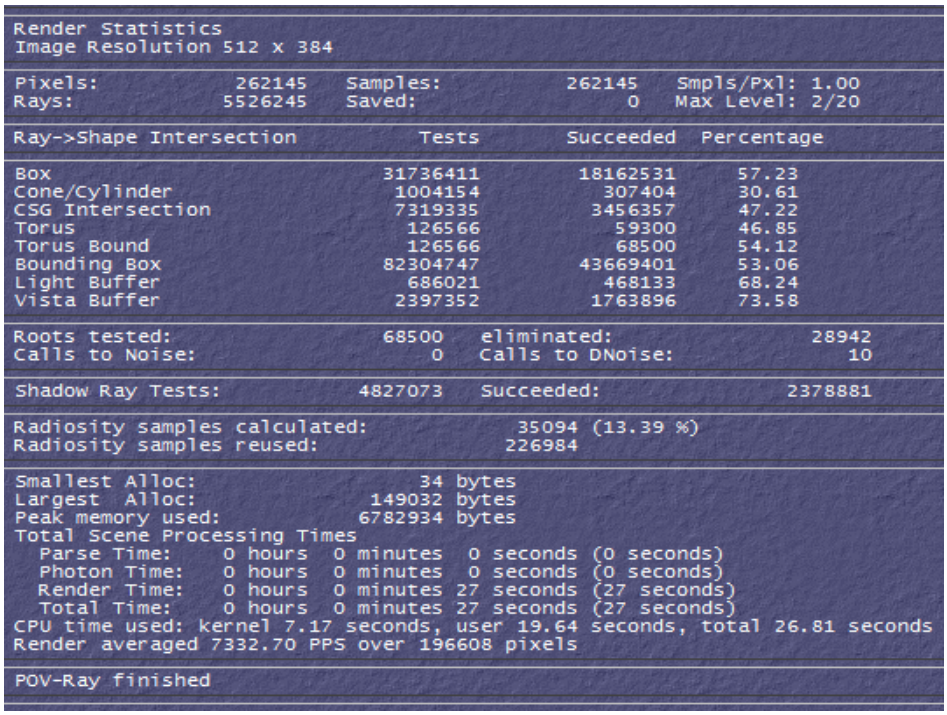
PLANTILLAS UTILIZADAS PARA LA RECOLECCIÓN DE DATOS

ANEXO 3

PLANTILLA UTILIZADA PARA MEDIR LA CONFIABILIDAD

ANEXO 4

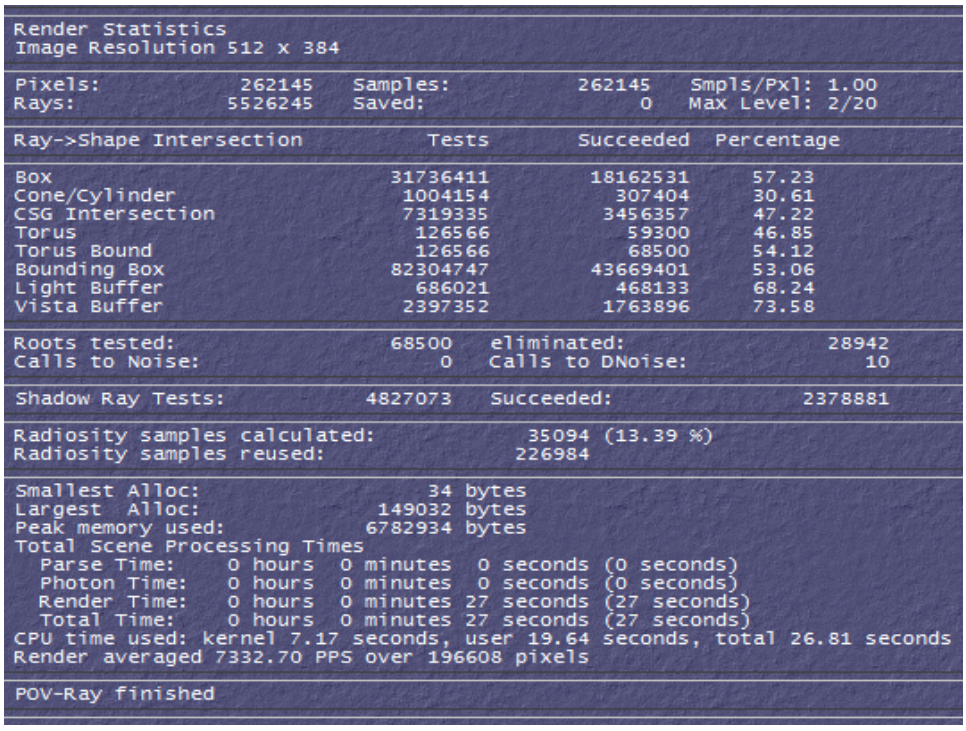
RESULTADOS DE LAS PRUEBAS DE CONFIBILIDAD



```
Render Statistics
Image Resolution 512 x 384

Pixels:          262145   Samples:          262145   Smpls/Pxl: 1.00
Rays:           5526245   Saved:                 0   Max Level: 2/20

Ray->Shape Intersection        Tests       Succeeded  Percentage

Box                         31736411        18162531     57.23
Cone/Cylinder                1004154          307404     30.61
CSG Intersection             7319335         3456357     47.22
Torus                         126566           59300     46.85
Torus Bound                   126566           68500     54.12
Bounding Box                82304747        43669401     53.06
Light Buffer                  686021          468133     68.24
Vista Buffer                 2397352         1763896     73.58

Roots tested:                  68500   eliminated:              28942
Calls to Noise:                    0   Calls to DNoise:            10

Shadow Ray Tests:            4827073   Succeeded:             2378881

Radiosity samples calculated:          35094 (13.39 %)
Radiosity samples reused:             226984

Smallest Alloc:                   34 bytes
Largest  Alloc:               149032 bytes
Peak memory used:            6782934 bytes
Total Scene Processing Times
  Parse Time:     0 hours  0 minutes  0 seconds (0 seconds)
  Photon Time:    0 hours  0 minutes  0 seconds (0 seconds)
  Render Time:    0 hours  0 minutes 27 seconds (27 seconds)
  Total Time:     0 hours  0 minutes 27 seconds (27 seconds)
CPU time used: kernel 7.17 seconds, user 19.64 seconds, total 26.81 seconds
Render averaged 7332.70 PPS over 196608 pixels

POV-Ray finished
```

Figura 73. Resultados de la prueba 1

Fuente: Elaboración propia



```
Render Statistics
Image Resolution 512 x 384

Pixels:          262145   Samples:          262145   Smpls/Pxl: 1.00
Rays:           5526245   Saved:                 0   Max Level: 2/20

Ray->Shape Intersection        Tests       Succeeded  Percentage

Box                         31736411        18162531     57.23
Cone/Cylinder                1004154          307404     30.61
CSG Intersection             7319335         3456357     47.22
Torus                         126566           59300     46.85
Torus Bound                   126566           68500     54.12
Bounding Box                82304747        43669401     53.06
Light Buffer                  686021          468133     68.24
Vista Buffer                 2397352         1763896     73.58

Roots tested:                  68500   eliminated:              28942
Calls to Noise:                    0   Calls to DNoise:            10

Shadow Ray Tests:            4827073   Succeeded:             2378881

Radiosity samples calculated:          35094 (13.39 %)
Radiosity samples reused:             226984

Smallest Alloc:                   34 bytes
Largest  Alloc:               149032 bytes
Peak memory used:            6782934 bytes
Total Scene Processing Times
  Parse Time:     0 hours  0 minutes  0 seconds (0 seconds)
  Photon Time:    0 hours  0 minutes  0 seconds (0 seconds)
  Render Time:    0 hours  0 minutes 27 seconds (27 seconds)
  Total Time:     0 hours  0 minutes 27 seconds (27 seconds)
CPU time used: kernel 7.17 seconds, user 19.64 seconds, total 26.81 seconds
Render averaged 7332.70 PPS over 196608 pixels

POV-Ray finished
```

Figura 74. Resultados de la prueba 2

Fuente: Elaboración propia

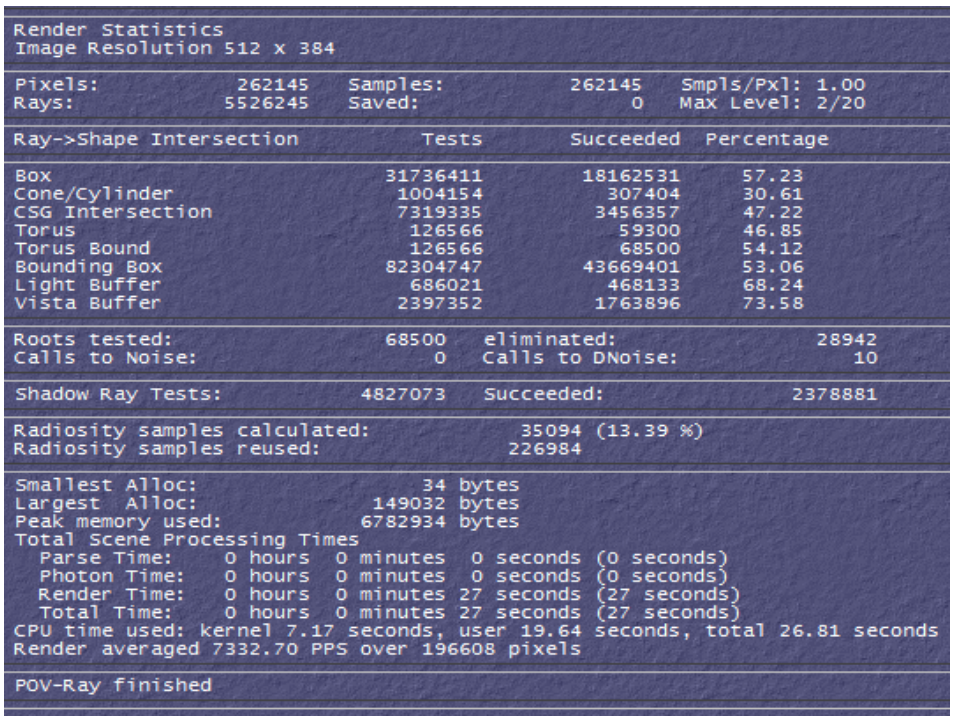


Figura 75. Resultados de la prueba 3

Fuente: Elaboración propia

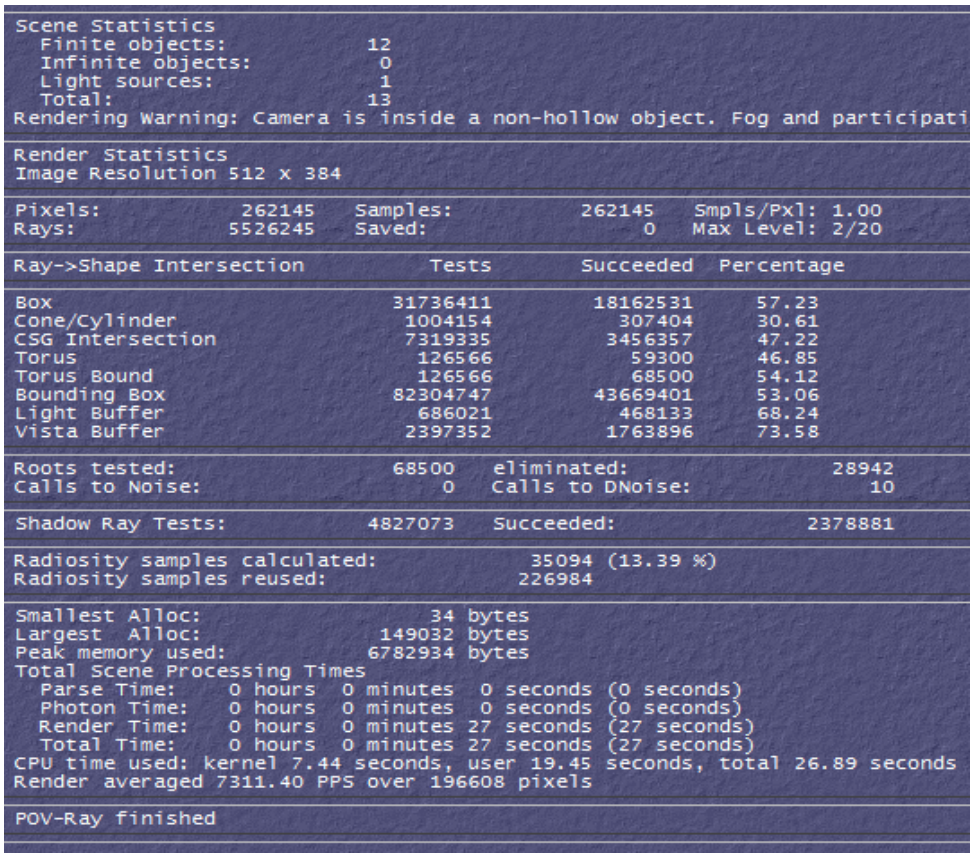


Figura 76. Resultados de la prueba 4

Fuente: Elaboración propia

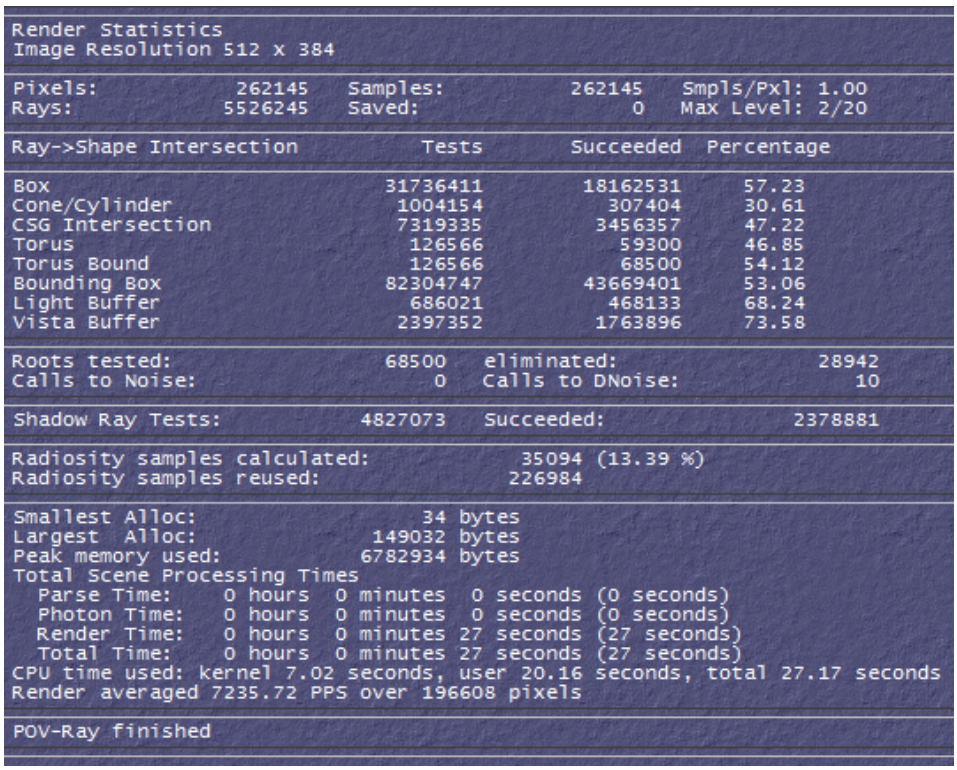


Figura 77. Resultados de la prueba 5

Fuente: Elaboración propia

ANEXO 5

ANEXO 6

INSTALACIÓN DEL SISTEMA CLUSTERKNOPPIX

Para la instalación del nodo maestro se procedió a configurar y establecer como dispositivo principal de arranque la unidad de DVD, una vez realizado ello aparecerá una ventana en la cual se escribirá la palabra "**expert**" para realizar la instalación experta del sistema Operativo como se muestra en la figura 78, el sistema Clusterknoppix posee otros modos de instalación entre los cuales figuran versiones del sistema operativo y procesadores así como arquitecturas distintas.

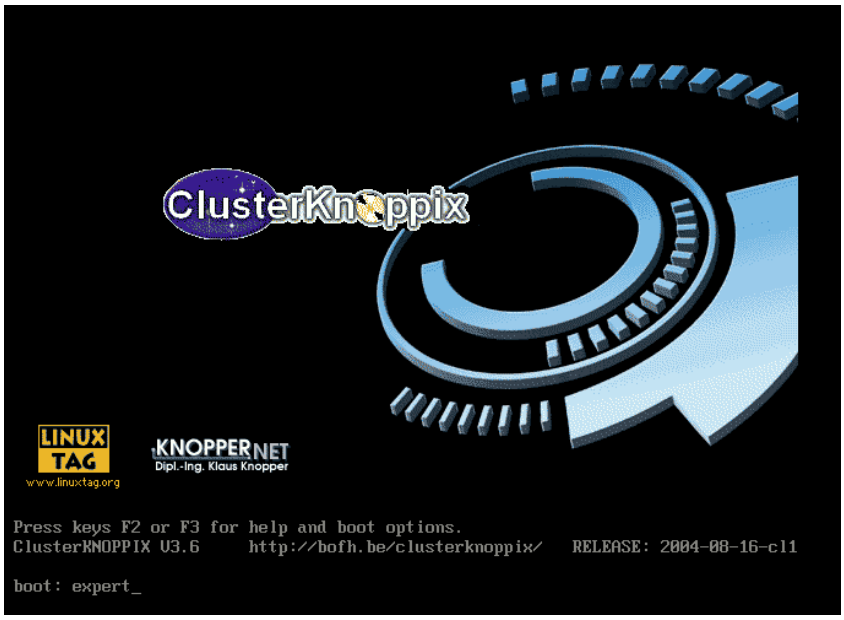


Figura 78. Arranque de la instalación de Clusterknoppix

Fuente: Elaboración propia

La instalación preguntará si se desea cargar los módulos SCSI como se muestra en la figura 79, pulsar la tecla **enter** para cargar dichos módulos.

Figura 79. Cargando SCSI

Fuente: Elaboración propia

Seguidamente también el proceso preguntará si se desea cargar los controladores de diskette, ya que actualmente ya no se utilizan, se debe negar que se carguen los controladores escribiendo la letra **"n"** y pulsando la tecla **enter** como se muestra en la figura 80.

Figura 80. Cargar controladores de diskette

Fuente: Elaboración propia

Seguidamente se realizará la configuración del mouse, la tarjeta de sonido y la tarjeta de video presionar **"n"** para para utilizar la auto detección de los mismos, posteriormente presionar la tecla **"enter"** como se muestra en la figura 81.

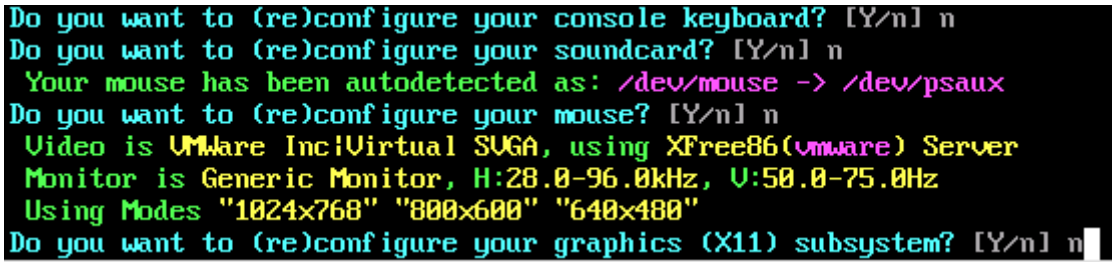


Figura 81. Configurar dispositivos de teclado, mouse, tarjeta de sonido y video

Fuente: Elaboración propia

Una vez realizado estos pasos se mostrará el entorno gráfico del sistema Clusterknoppix como se muestra en la figura 82.

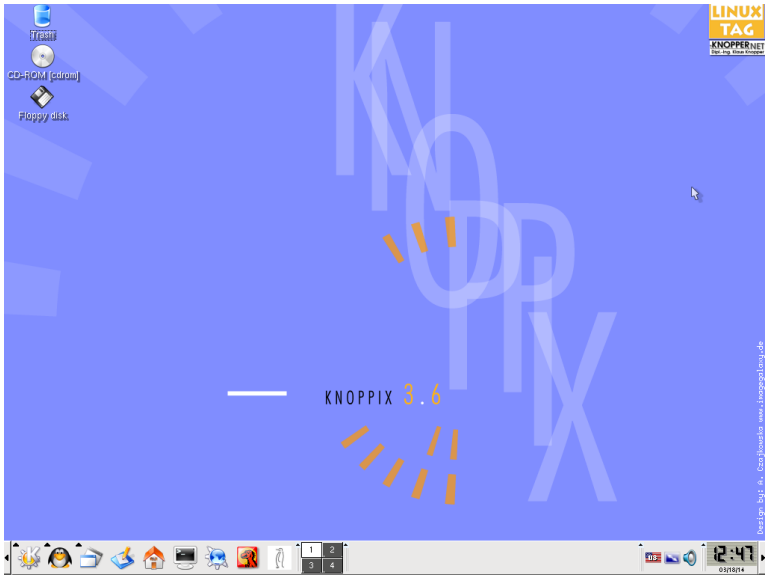


Figura 82. Entorno gráfico de Clusterknoppix

Fuente: Elaboración propia

El sistema inicia en modo live es decir solo se carga en la memoria RAM por el momento, seguidamente se reala la configuración del idioma del teclado en la parte inferior derecha de la pantalla como se muestra en la figura 83, ya que por defecto está en inglés.

Figura 83. Área de configuración del idioma

Fuente: Elaboración propia

Hacer clic derecho en la bandera y seleccionar la opción "**Configure**", eliminar los idiomas que no se utilizarán y pasar al apartado de idiomas activos el idioma español, para confirmar hacer clic en "**ok**" como se muestra en la figura 84.

Figura 84. Configuración de idioma

Fuente: Elaboración propia

Una vez realizado estos pasos, se procederá a instalar el sistema en nuestro disco duro, para lo cual será necesario abrir el terminal  y acceder a los comandos como súper usuario para ello se debe escribir el comando "su" como se muestra en la figura 85.

Figura 85. Logueo como súper usuario

Fuente: Elaboración propia

Para comenzar el proceso de instalación se debe escribir el comando **"knoppix-installer"** y presionar la tecla enter, aparecerá una ventana que se muestra la información del instalador, hacer clic en "ok" para confirmar como se muestra en la figura 86.

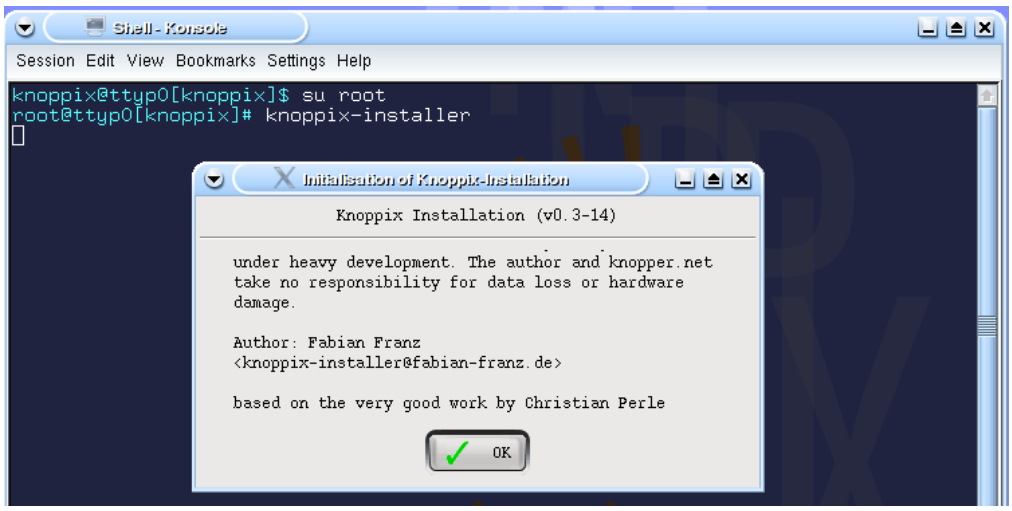


Figura 86. Bienvenida a la instalación de Clusterknoppix

Fuente: Elaboración propia

Posteriormente aparecerá una ventana en la cual se especifican los requisitos mínimos para la instalación del sistema, hacer clic en "**ok**" como se muestra en la figura 87.

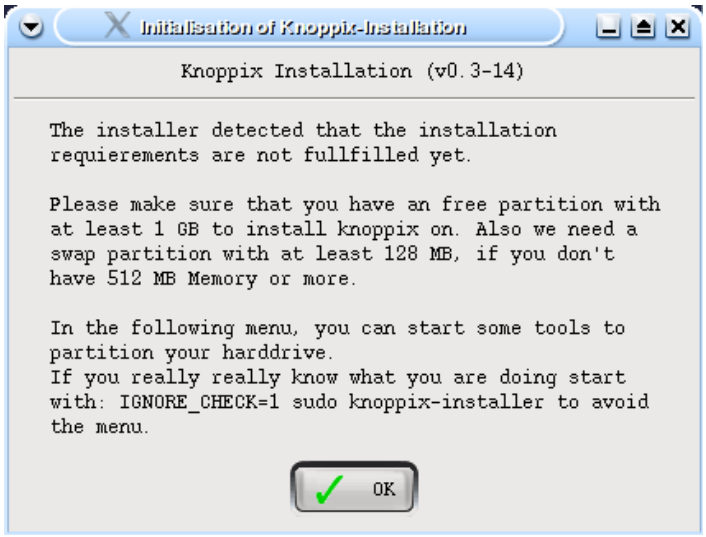


Figura 87. Requisitos mínimos para la instalación de Clusterknoppix

Fuente: Elaboración propia

Para continuar se deberá particionar la unidad de disco en dos particiones, la primera alojara el sistema operativo y la segunda partición será el de área de intercambio para el proceso de virtualización de memoria, para ello hacer clic en la primera opción y presionar la tecla enter como se muestra en la figura 88.

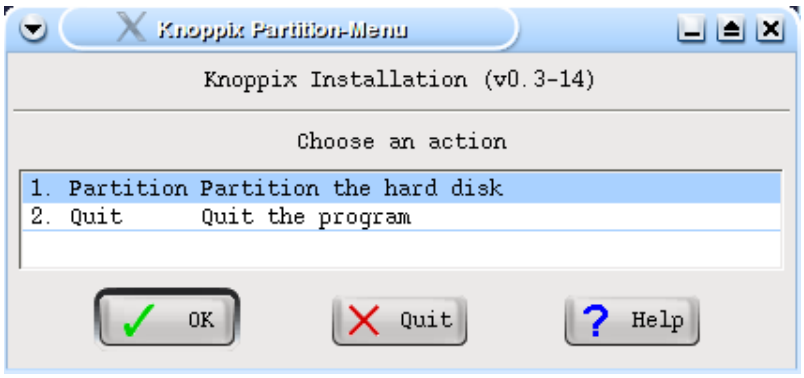


Figura 88. Partición de disco

Fuente: Elaboración propia

A continuación se abrirá la siguiente ventana en la cual se debe seleccionar la unidad de disco **"01 partition Table"**, hacer clic derecho sobre ella y seleccionar la opción "Make a New Patition Table" para preparar la nueva parición como se muestra en la figura 89.

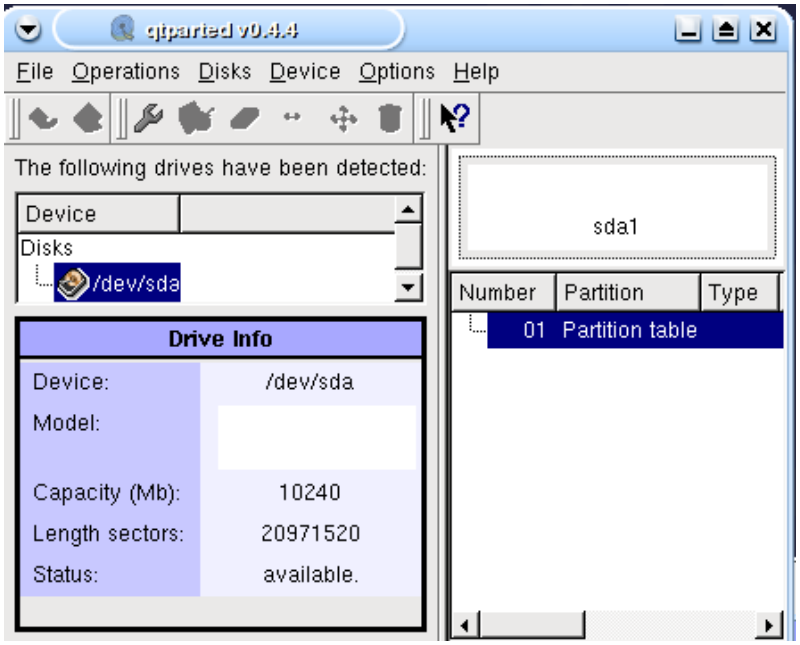


Figura 89. Preparando la nueva tabla de partición

Fuente: Elaboración propia

Al realizar este paso aparecerá una nueva ventana de notificación advirtiendo de que los datos se perderán al realizar esta operación, confirmar haciendo clic en **"yes"** como se muestra en la figura 90.

Figura 90. Confirmación de borrado

Fuente: Elaboración propia

Una vez creada la tabla de particiones se debe proceder a la creación de la partición del sistema operativo, se debe seleccionar la unidad de disco **"01 partition Table"**, hacer clic derecho sobre ella y seleccionar la opción "Create" como se muestra en la figura 91.

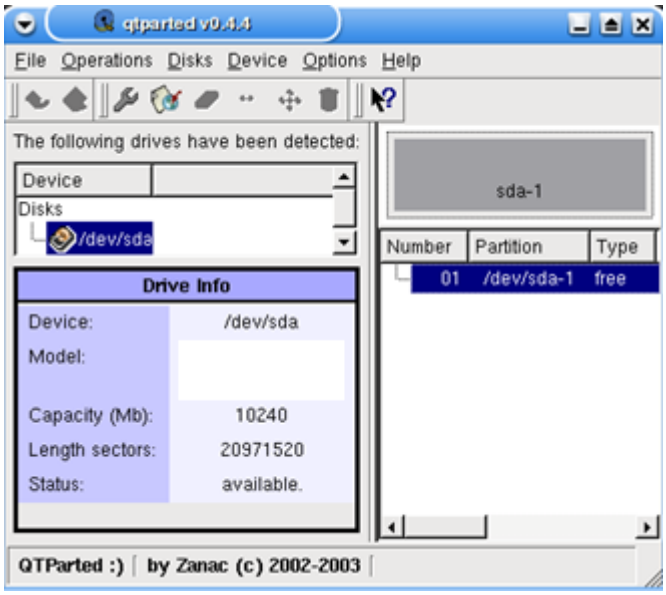


Figura 91. Creando la tabla de partición

Fuente: Elaboración propia

En seguida se debe especificar el tamaño de la parición en GB así como el sistema de archivos a utilizar, para confirmar se debe hacer clic en "**Ok**" como se muestra en la figura 92.

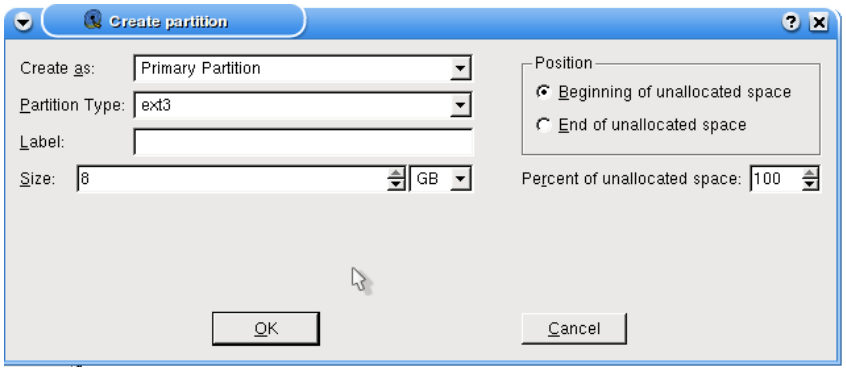


Figura 92. Creación de la partición para el sistema operativo

Fuente: Elaboración propia

Se visualiza en seguida la creación de la partición, se debe seleccionar el área libre y hacer clic derecho sobre él, seleccionar la opción "create" para crear el área de intercambio respectivo como se muestra en la figura 93.

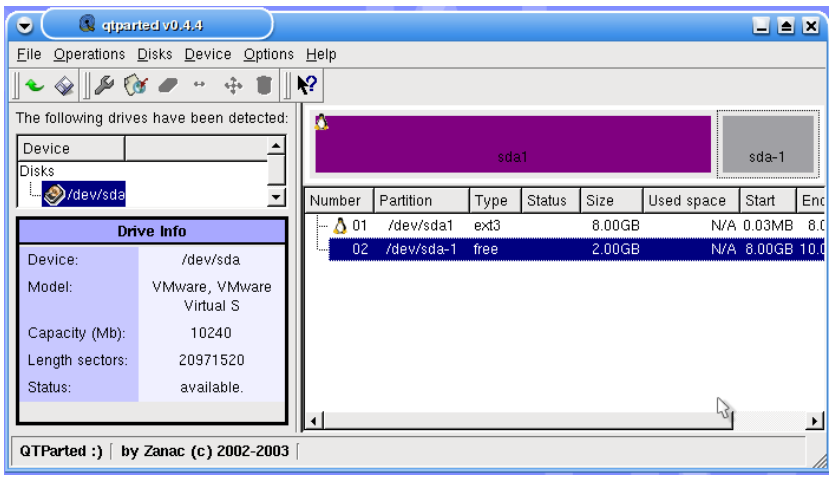


Figura 93. Preparando la partición de área de intercambio

Fuente: Elaboración propia

Seguidamente se debe establecer el tamaño en GB, seleccionar además el tipo de partición "**linux-swap**" y hacer clic en "**ok**" como se muestra en la figura 94.

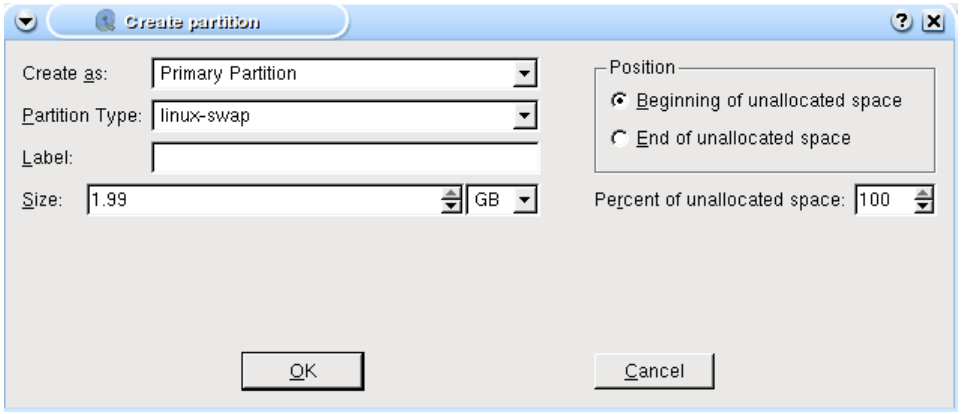


Figura 94. Creación del área de intercambio

Fuente: Elaboración propia

Para guardar los cambios hacer clic en el botón commit  como se muestra en la figura 95.

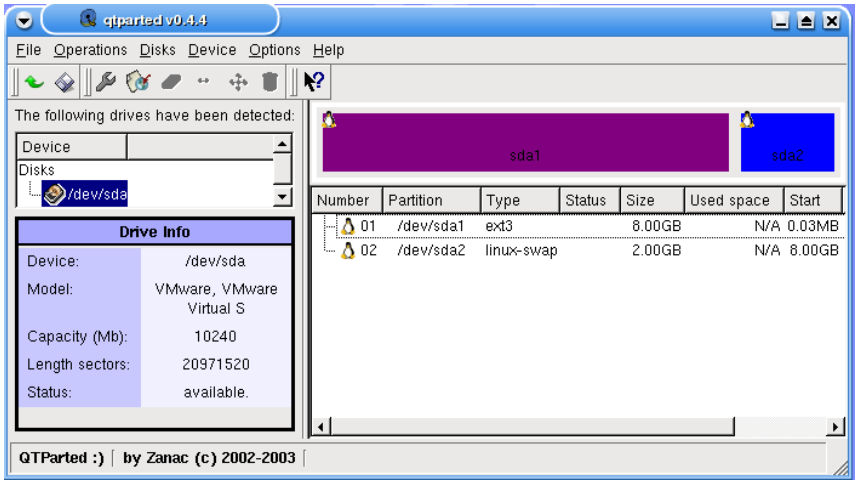


Figura 95. Guardado cambio en las particiones

Fuente: Elaboración propia

Aparecerá una ventana de advertencia en el cual se informa que los datos se perderán al necesitar borrar las pariciones, confirmar haciendo clic en "**yes**" como se muestra en la figura 96.

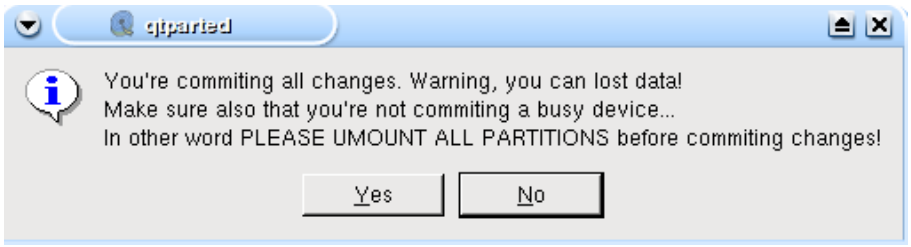


Figura 96. Confirmación de borrado

Fuente: Elaboración propia

Aparecerá la ventana de progreso del proceso de particionado, se debe esperar a que concluya, para finalizar particionado hacer clic en "ok" como se muestra e la figura 97.

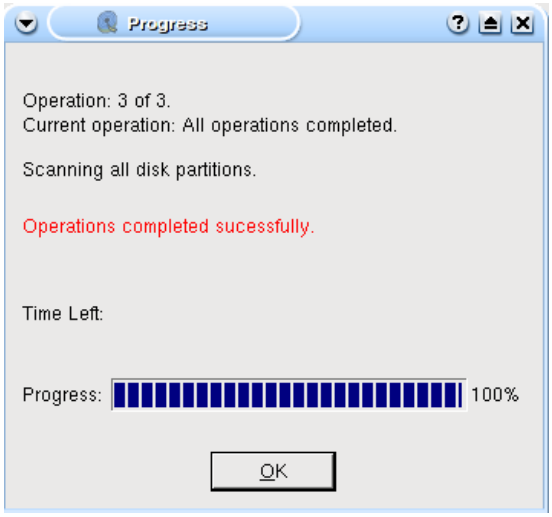


Figura 97. Progreso de la partición de unidades

Fuente: Elaboración propia

En seguida se muestra la ventana donde se debe realizar las pariciones y aparecerá la siguiente ventana en la cual se debe hacer clic en ok para continuar con el siguiente paso que es la configuración del sistema como se muestra en la figura 98.

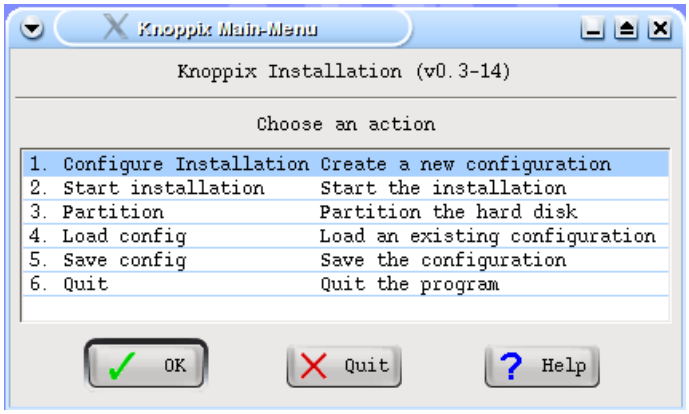


Figura 98. Iniciando la configuración del sistema

Fuente: Elaboración propia

En la siguiente ventana se pedirá que tipo de sistema a utilizar, seleccionar el tipo debían "**debían"** y hacer clic en el botón "next" como se muestra en la figura 99.

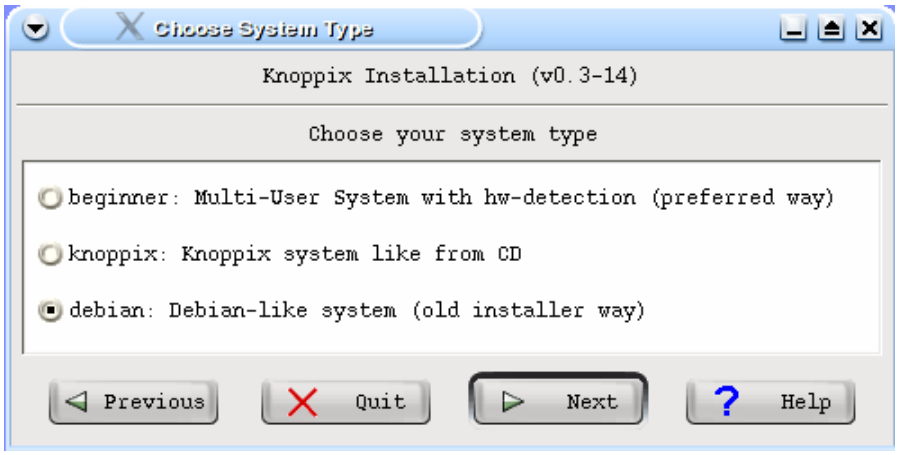


Figura 99. Configuración del sistema a utilizar

Fuente: Elaboración propia

En seguida se pedirá escoger la partición en la cual se instalará el sistema, por defecto se utilizará la parición creada "**sda1"** anteriormente y hacer clic en el botón "next" como muestra la figura 100.

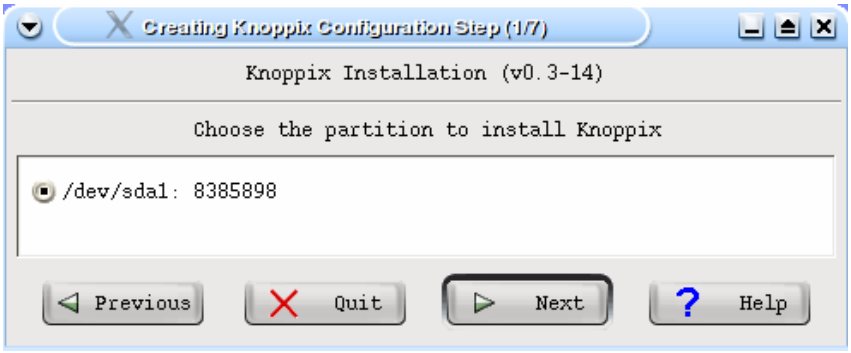


Figura 100. Selección de la partición a usar para el sistema operativo

Fuente: Elaboración propia

También se pedirá el tipo de sistema de archivos utilizar, seleccionar "ext3" por ser el más utilizado y hacer clic en el botón "**next**" como muestra la figura 101.

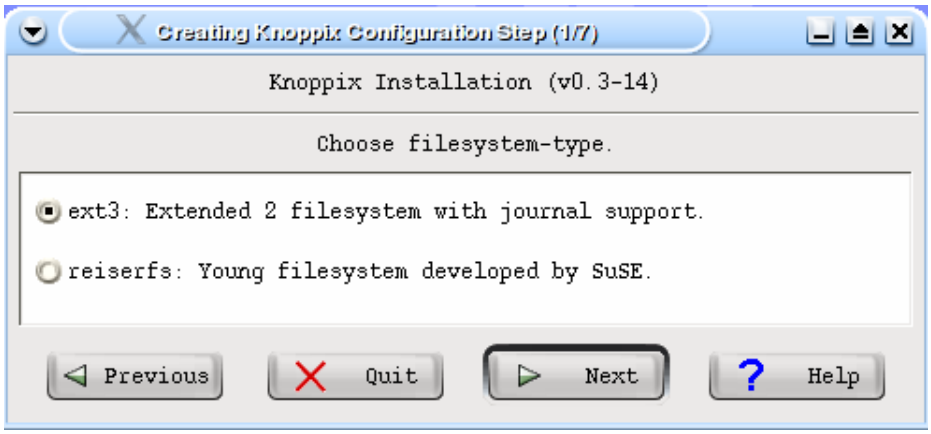


Figura 101. Seleccionando el tipo de sistema de archivos a utilizar

Fuente: Elaboración propia

En seguida se pedirá el nombre del nodo, en este caso se especificó "unajma" y se hará clic en el botón "**next**" como muestra la figura 102.

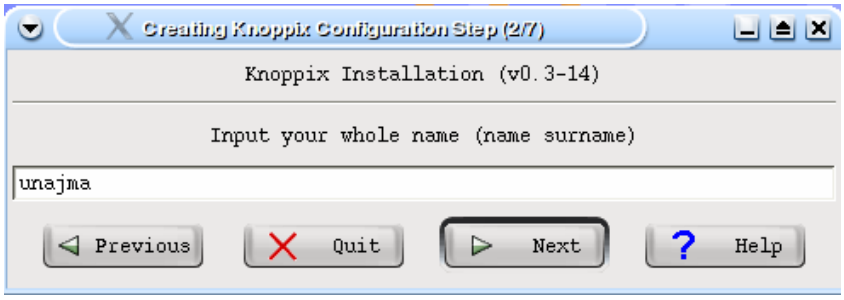


Figura 102. Estableciendo nombre del nodo

Fuente: Elaboración propia

A continuación también se pedirá un nombre para la cuenta de usuario, de la misma forma se escoge un nombre, para este caso se especificó el nombre "**yinmel**", hacer clic en el botón "**next**" como muestra la figura 103.

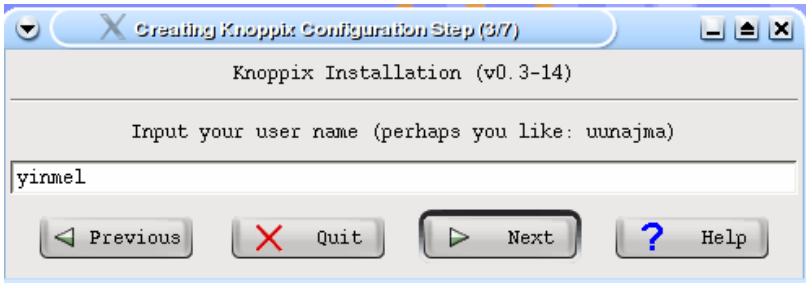


Figura 103. Estableciendo el nombre de usuario del sistema

Fuente: Elaboración propia

Posteriormente se introducirá la contraseña del usuario creado en el paso anterior y hacer clic en el botón "**next**" como muestra la figura 104.

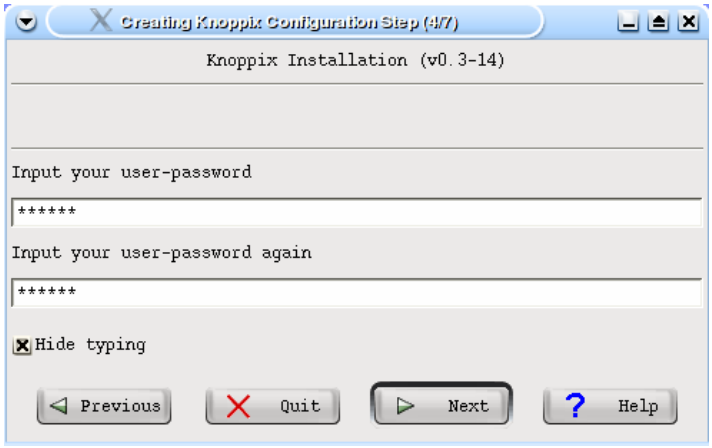


Figura 104. Estableciendo la contraseña del nuevo usuario

Fuente: Elaboración propia

En la siguiente ventana se debe especificar la contraseña del administrador y hacer clic en el botón "**next**" como se muestra en la figura 105.

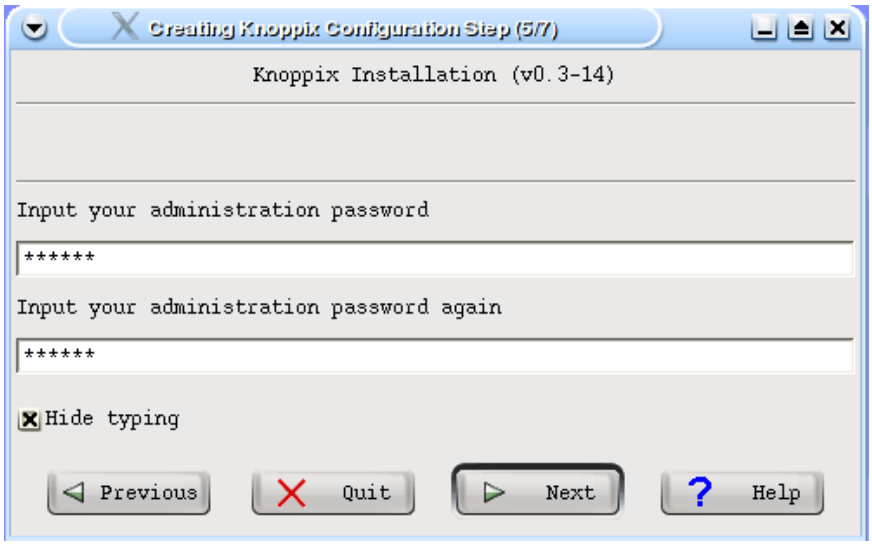


Figura 105. Establecido la contraseña del administrador

Fuente: Elaboración propia

Se debe especificar también el nombre preferido para los host entrantes, esta especificación es opcional, dejar por defecto en el valor mostrado y hacer clic en el botón "**next**" como muestra la figura 106.

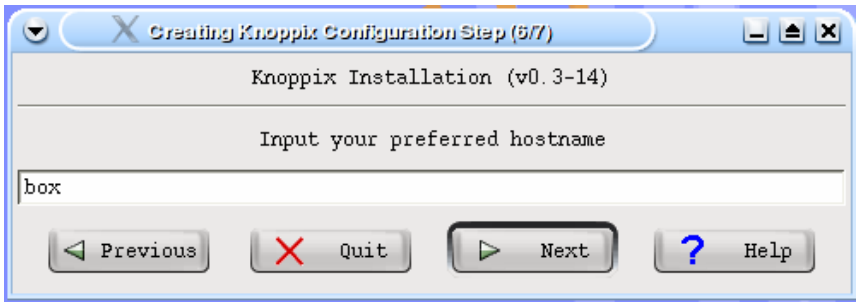


Figura 106. Estableciendo el nombre preferido del sistema

Fuente: Elaboración propia

En el paso final se pedirá escoger el sistema de administración para el arranque del sistema, se debe dejar por defecto en MBR y hacer clic en el botón "**next**" como muestra la figura 107.

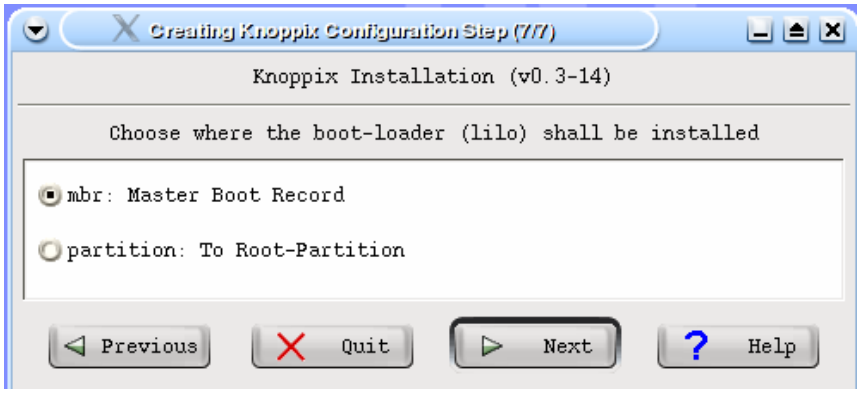


Figura 107. Seleccionado la administración del arranque

Fuente: Elaboración propia

Para comenzar la instalación seleccionar "**Start Instalation**" hacer clic en el botón "**ok**" como muestra la figura 108.

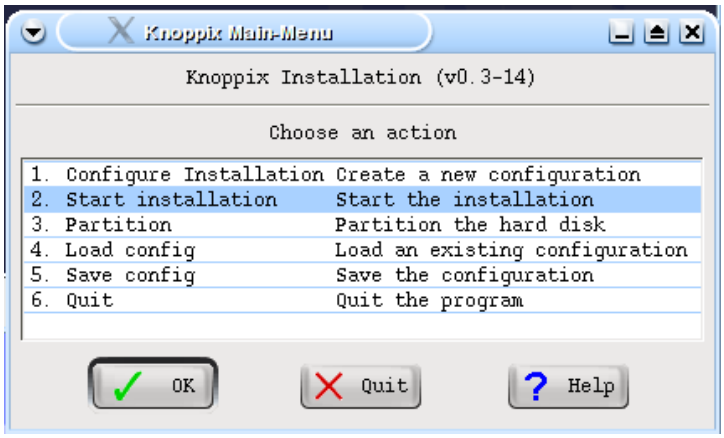


Figura 108. Comenzando la instalación

Fuente: Elaboración propia

En seguida se mostrará un resumen todas ala configuraciones realizadas, para confirmar se debe hacer clic en el botón "**next**" como se muestra en la figura 109.

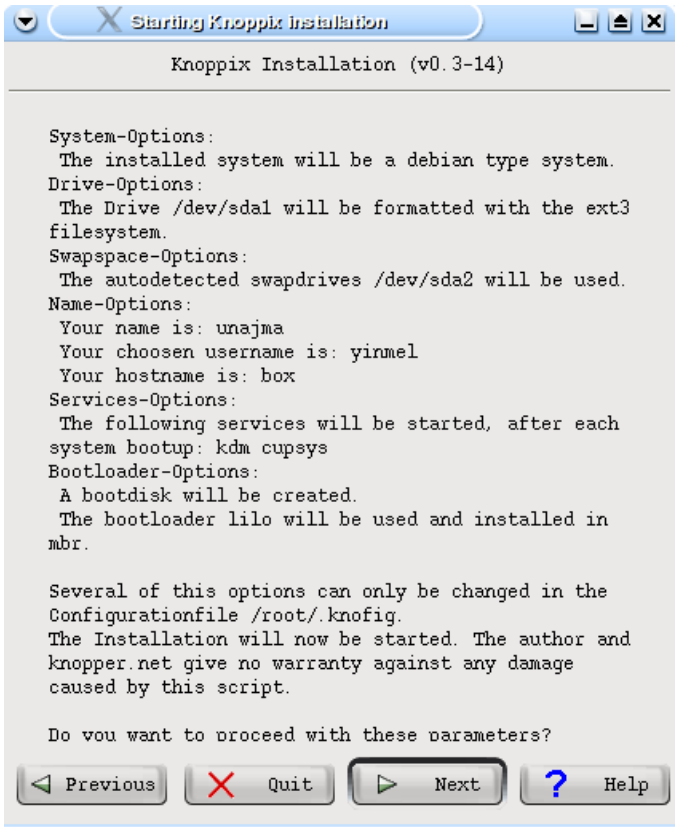


Figura 109. Resumen de las configuraciones realizadas

Fuente: Elaboración propia

Comenzará entonces el proceso de instalación del sistema como se muestra en la figura 110, se debe esperar a que concluya el proceso.

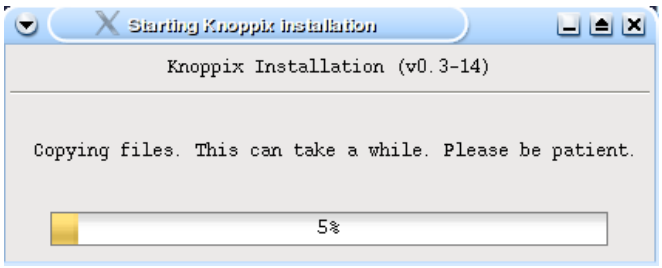


Figura 110. Progreso de la instalación del sistema

Fuente: Elaboración propia

Al finalizarla instalación se pedirá crear un disco de recuperación del sistema, no será necesario, por lo cual se hará clic en el botón "**no**" como se muestra en la figura 111.

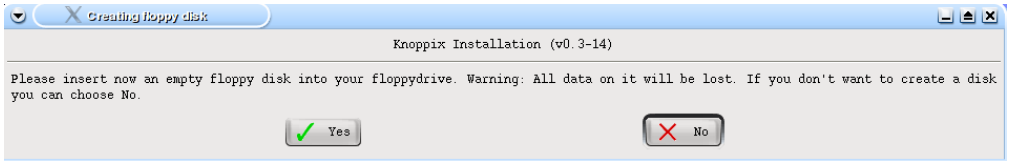


Figura 111. Creación de disco de recuperación

Fuente: Elaboración propia

Se mostrará el aviso de que se ha instalado el sistema en el disco duro satisfactoriamente como se muestra en la figura 112, hacemos clic en el botón "ok" y se procederá a reiniciar el sistema.

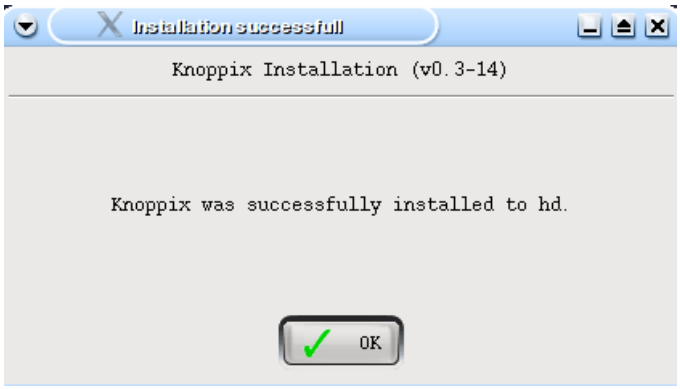


Figura 112. Aviso de instalación finalizada

Fuente: Elaboración propia

Entonces se debe iniciar el sistema con el usuario por defecto creado en los pasos anteriores como se muestra en la figura 113.

Figura 113. Inicio de sesión con la cuenta creada

Fuente: Elaboración propia

ANEXO 7 CONFIGURACIÓN DE INTERFACE DE RED DE UN NODO

Se procede a explicar cómo es que se realizó las configuraciones posteriores a la instalación del nodo maestro, los siguientes pasos se realizan también para el resto de los nodos.

Para empezar se debe configurar la interface de red para las comunicaciones, para ello ingresar al botón de acceso  , al menú **knoppix, Network/Internet, Network Card Configuration.**

Se debe la contraseña del administrador para poder realizar dicha configuración y hacer clic en el botón **"Ok"** como se muestra en la figura 114.

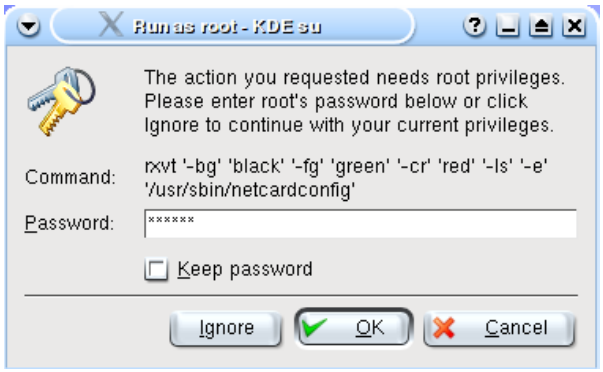


Figura 114. Autenticación para configuraciones

Fuente: Elaboración propia

El proceso preguntará si se desea utilizar el servicio DHCP para la configuración de la red, no se utiliza DHCP para ello se debe hacer clic en "**no**", ya que se configurará todos los parámetros manualmente. A continuación se debe escribir en la ventana la dirección IP, mascara de subred, dirección broadcast y la puerta de enlace predeterminada de nuestra interface de red y confirmar dichas configuraciones haciendo clic en el botón "**ok**" como se muestra en la figura 115.

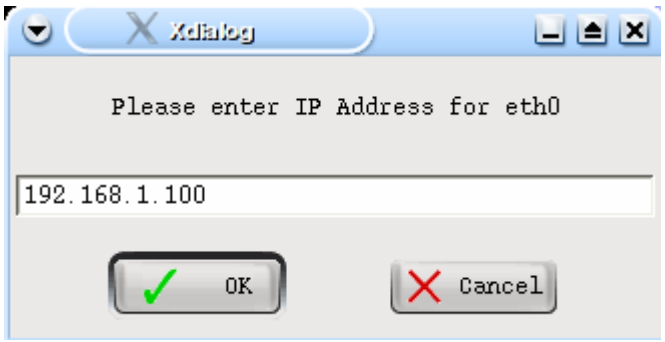


Figura 115. Ingreso de la dirección IP

Fuente: Elaboración propia

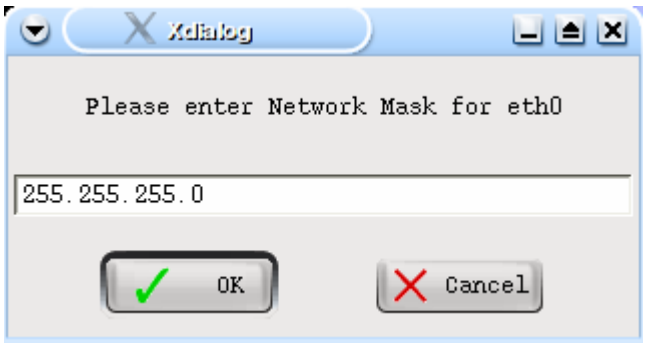


Figura 116. Ingreso de la dirección de máscara de subred

Fuente: Elaboración propia

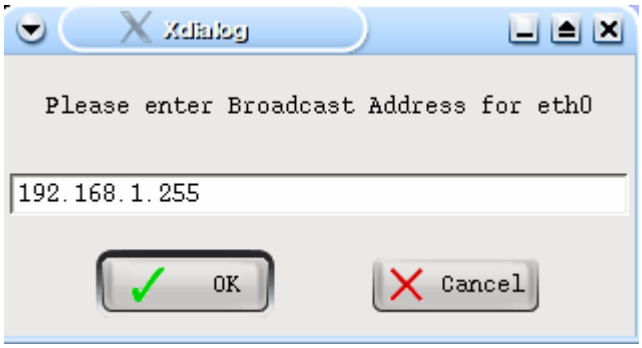


Figura 117. Ingreso de la dirección de broadcast de la red

Fuente: Elaboración propia

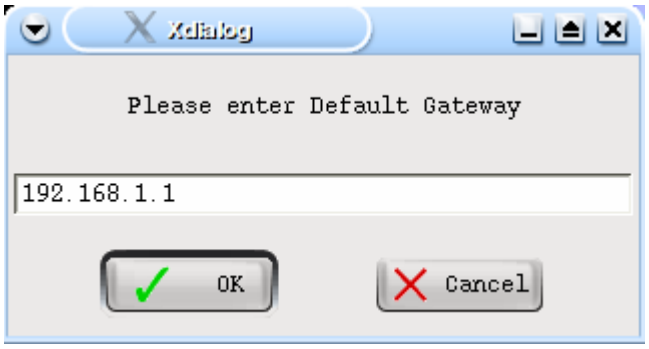


Figura 118. Ingreso de la dirección de puerta de enlace por defecto

Fuente: Elaboración propia

Se debe agregar también una dirección DNS a utilizar para la interface de red en la siguiente venta de como muestra la figura 119.

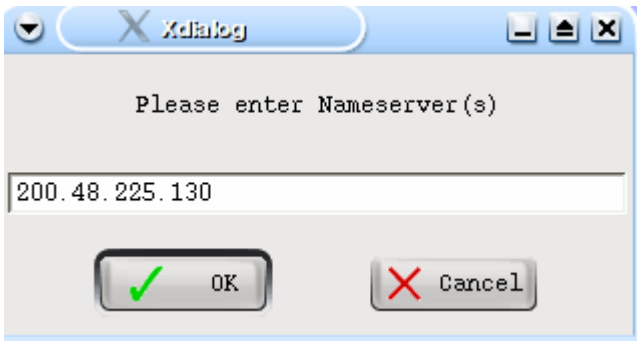


Figura 119. Ingreso de la dirección DNS

Fuente: Elaboración propia

Para verificar que la configuración sea correcta se debe escribir en una consola el comando “ifconfig”. Y deberá mostrase los parámetros de configuración especificados en los pasos anteriores como se muestra en la figura 120.

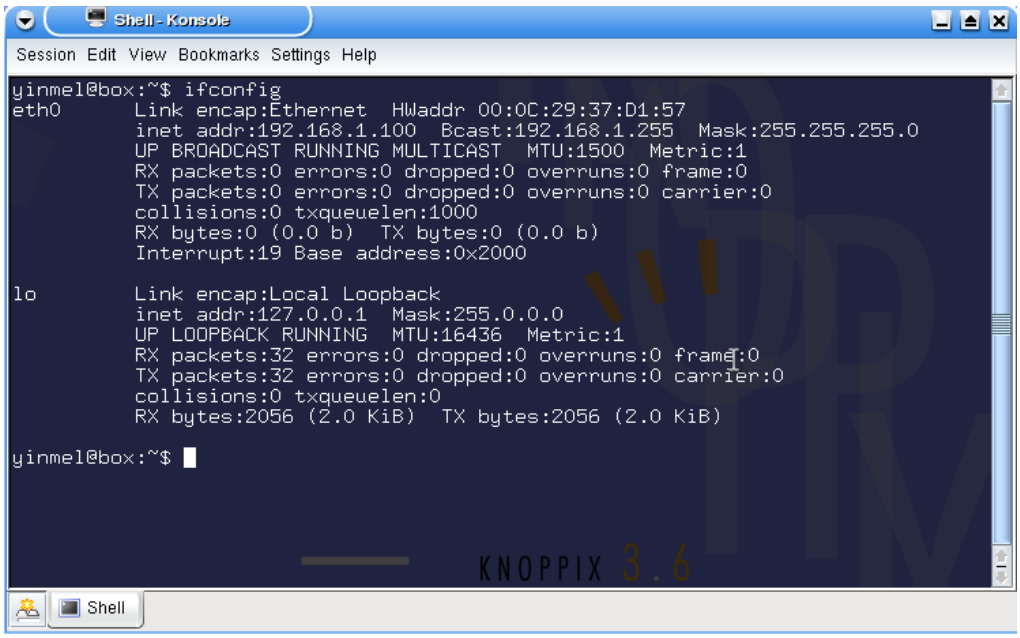


Figura 120. Verificación de la configuración de la interface de red

Fuente: Elaboración propia

ANEXO 8

MONTAR EL SISTEMA DE ARCHIVOS MFS

Openmosix funciona con un sistema de fichero llamado MFS, este permite que se puedan repartir los distintos procesos entre los nodos del clúster para aumentar el rendimiento. Sin este sistema de ficheros el clúster no funcionaría por lo que es algo imprescindible. Pero por defecto no está montado, por lo que hay que realizar dos tareas: primeramente se debe editar el fichero /etc/fstab para que monte el mfs de forma automática y posteriormente se debe montar el mfs en un directorio que deberá crearse con el nombre de "mfs".

Todas las configuraciones se deben hacer como súper usuario, para ello se debe escribir "su" ingresar la contraseña del administrador y editar el archivo /etc/fstab escribiendo en consola el siguiente los siguientes comandos como muestra la figura 121.

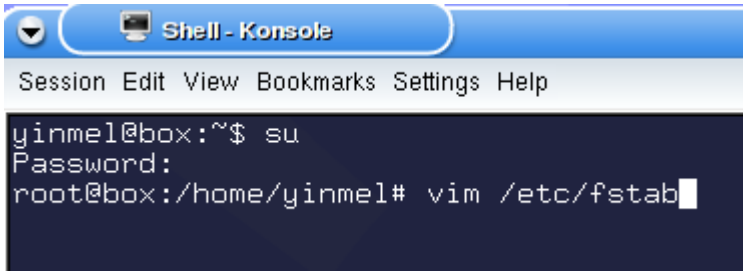


Figura 121. Editando el archivo fstab

Fuente: Elaboración propia

Se abrirá el siguiente el archivo en el cual se debe agregar al final la línea, para especificar el directorio.

mfs /mfs mfs dfsa=1 0 0

Es importante mencionar que para el uso del editor VIM pueden usarse estos pasos sencillos:

- Se debe pulsar la letra "i" para insertar cambios.
- Una vez hecho los cambios pertinentes se debe pulsar la tecla "Esc" para salir del modo de edición.
- Para guardar los cambios se debe pulsar ": x".

Los cambios realizados se muestran en la figura 122, después de realizar los cambios pertinentes se debe guardar el archivo:

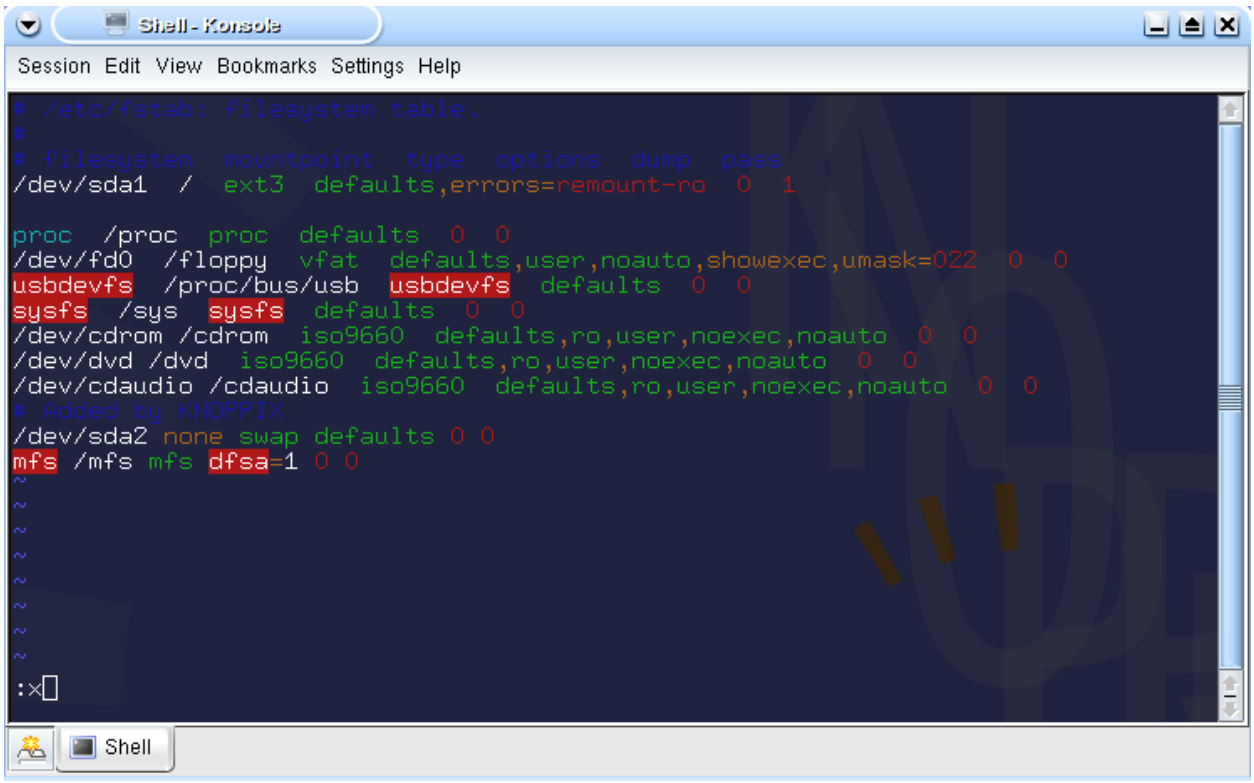


Figura 122. Configuración final del archivo fstab

Fuente: Elaboración propia

Para montar MFS es necesario tener un directorio, para crear dicho directorio se debe escribir el comando como se muestra en la figura 123.

Figura 123. Creando el directorio para mfs

Fuente: Elaboración propia

Se debe montar el sistema de archivos en la carpeta mfs utilizando el comando mount como se muestra en la figura 124.

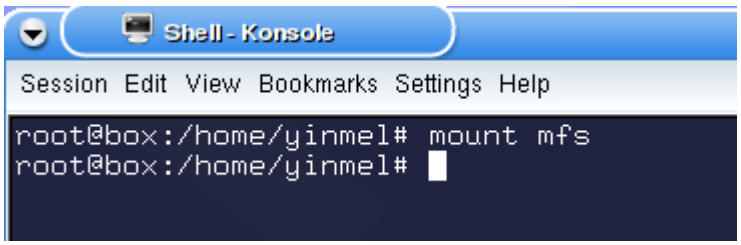


Figura 124. Montando el sistema de archivos mfs

Fuente: Elaboración propia

Se debe verificar el correcto funcionamiento del sistema MFS, para ello reinicie demonio Openmosix con el siguiente comando en consola como se muestra en la figura 125.

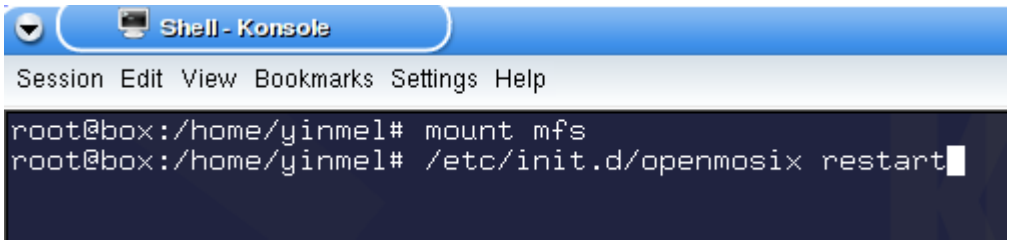


Figura 125. Reiniciando el demonio Openmosix

Fuente: Elaboración propia

En la última línea se debe mostrar el mensaje "MFS Access already enabled", esto significa que el acceso al sistema MFS está habilitado como se muestra en la figura 126.

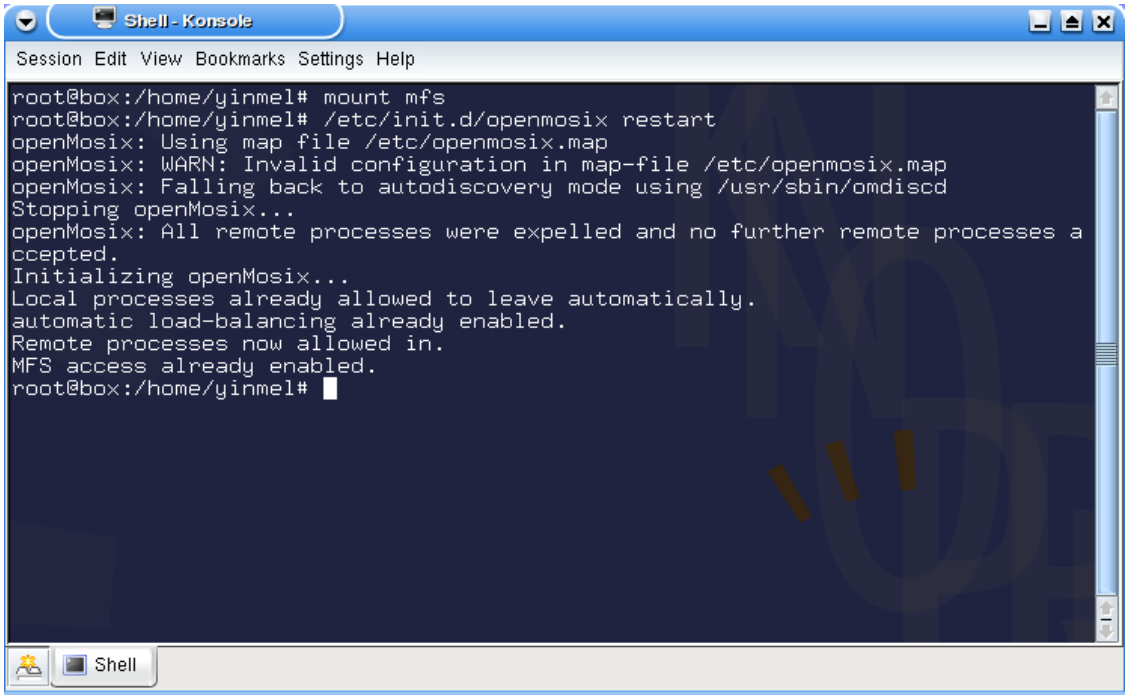


Figura 126. Verificación del funcionamiento de mfs

Fuente: Elaboración propia

ANEXO 9

CONFIGURACION DE SSH PARA LA COMUNICACIÓN ENRE PROCESOS

Openmosix comunica todos y cada uno de sus nodos mediante conexiones de SSH. A través de este canal pasa los mensajes de los procesos a los distintos nodos y estos a su vez lo devuelven al nodo que actúe en ese momento de maestro (el nodo que ejecutó el proceso). Esto mediante el método convencional de conexiones SSH no sería del todo factible. La razón es que cada vez que se iniciara un proceso, saldría peticiones para introducir la clave del usuario del nodo al que se va a conectar por SSH y un nuevo mensaje cuando dicho nodo quiera conectarse al master. Si se quiere automatizar este proceso la única forma es utilizar claves públicas y privadas de SSH para que la comunicación sea transparente para el usuario o administrador del clúster.

Eso es precisamente lo se va configurar. Se debe crear las claves y especificar al clúster que toda comunicación se haga a través de las claves públicas que estarán añadidas en el fichero authorized_keys. Para ello se procede a realizar los siguientes pasos.

Para crear las claves se utilizará el comando que se muestra en la figura 127.

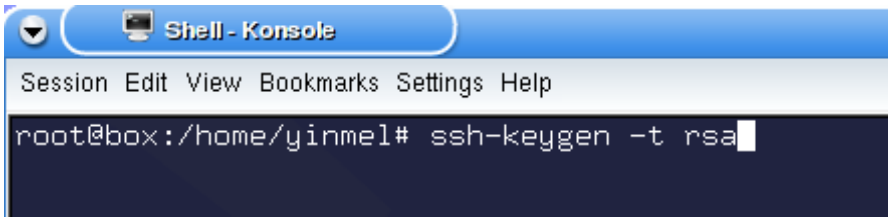


Figura 127. Generando clave pública y privada

Fuente: Elaboración propia

El proceso pedirá guardar las claves (se especificará para ello el fichero /root/.SSH/id_rsa), también se pedirá una frase pero no es necesario escribirla presionando solamente la tecla enter, con esto se debe tener creadas la clave pública y privada del equipo y su ubicación, en la figura 102 se muestra el proceso:

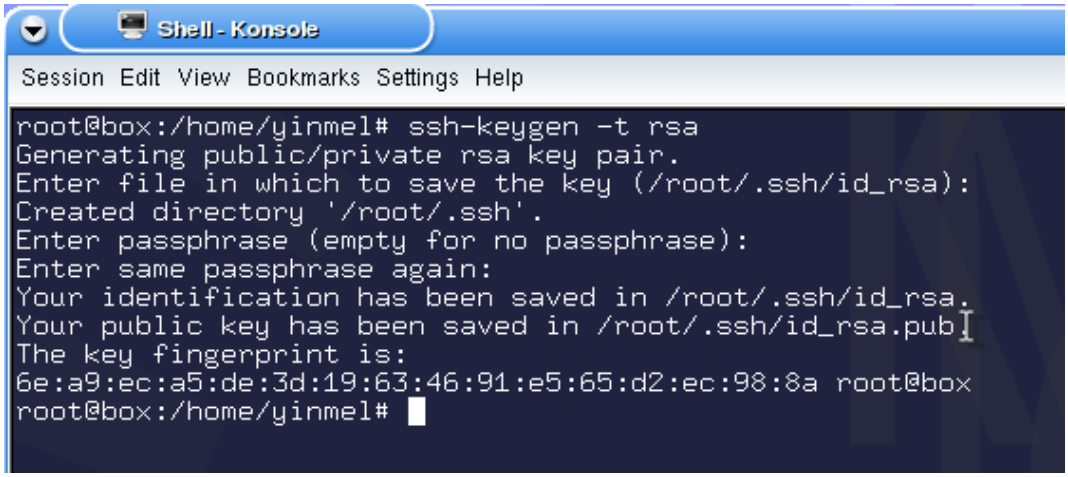


Figura 128. Generación de la clave pública y privada

Fuente: Elaboración propia

Se debe verificar que los archivos se han creado de forma satisfactoria, la comprobación de las claves se realiza con el comando mostrado en la figura 129.

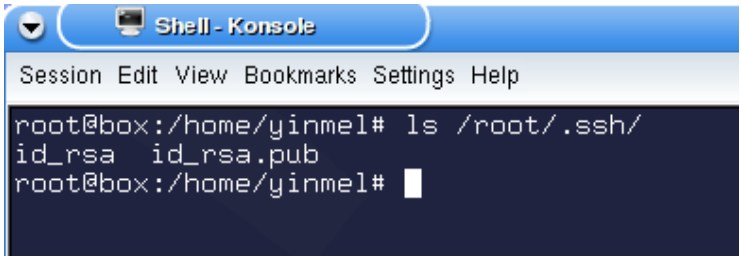


Figura 129. Verificación de la creación de las claves

Fuente: Elaboración propia

Para que la conexión transparente tenga éxito se debe crear el fichero authorized_keys para autorizar la conexión SSH de ese nodo y así igualmente con las claves de los nodos a los cuales se quiere que se conecten. Para ello se usa la siguiente salida del comando cat para redirigir el contenido al fichero antes indicado como se muestra en la figura 130.

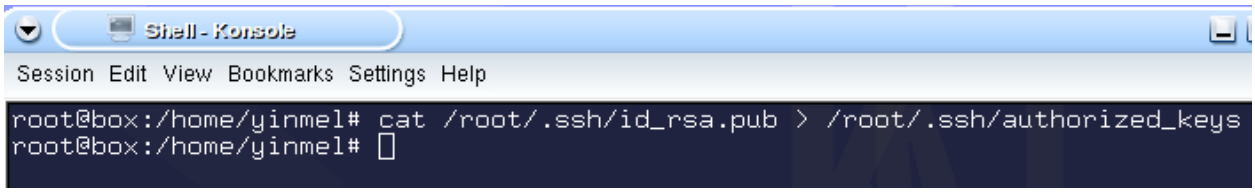


Figura 130. Redirigiendo contenido entre ficheros de configuración

Fuente: Elaboración propia

Una buena práctica es publicar en el servidor web que incorpora el clúster-knoppix, la clave pública para que desde los otros nodos se pueda añadirla al fichero de autorización de los demás con el fin de que todos se puedan conectar con todos de forma automática. Para ello se debe copiar el fichero id_rsa.pub al directorio del servidor web como se muestra en la figura 131:

Figura 131. Publicando la clave pública

Fuente: Elaboración propia

Para que los cambios surtan efecto se debe reiniciar el servidor web apache con el comando que se muestra en la figura 132.

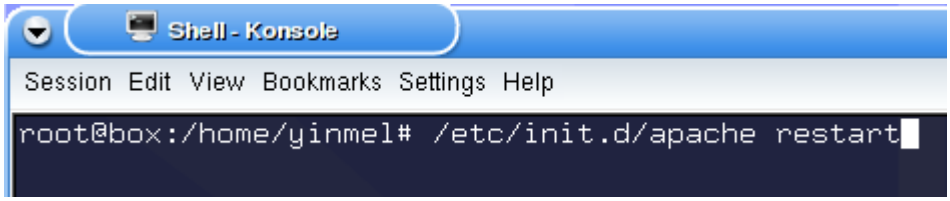


Figura 132. Reiniciado el servidor web

Fuente: Elaboración propia

Se debe verificar desde un navegador web que la calve está publicada digitando la siguiente dirección en la barra de direcciones del navegador como se muestra en la figura 133.

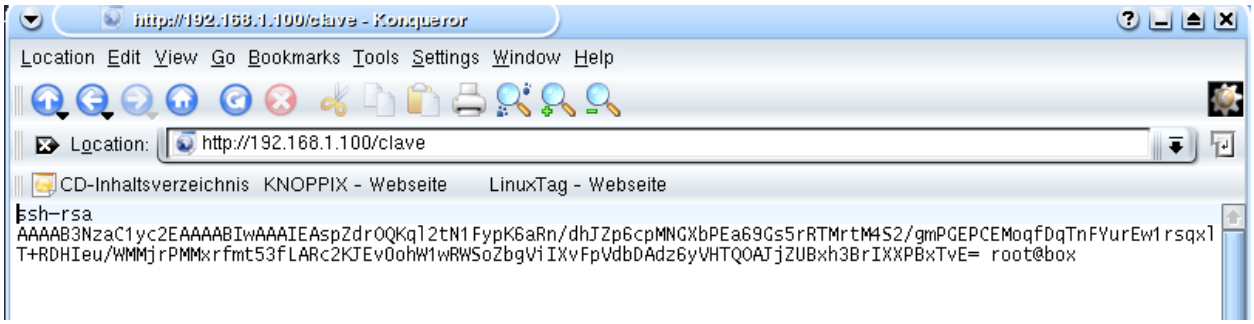


Figura 133. Verificación de la publicación de las claves

Fuente: Elaboración propia

Seguidamente se debe configurar el fichero SSHD para que utilice la autenticación por claves al usar conexiones SSH con el siguiente comando mostrado en la figura 134.

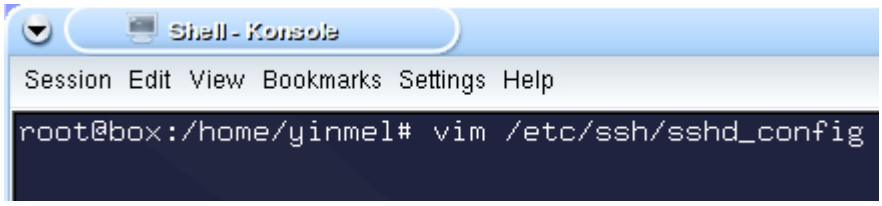


Figura 134. Edición del archivo de configuración del servicio SSH

Fuente: Elaboración propia

En el fichero se deben agregar las siguientes líneas:

• RSAAuthentication no: no realiza autenticaciones de llaves RSA.

• PasswordAuthentication yes: requiere autenticación de claves.

• PubkeyAuthentication yes: realiza autenticación de claves públicas.

• PermitEmptyPasswords yes: permite el ingreso de claves vacías (sin caracteres).

• AutorizedKeysFile /root/.SSH/authorized_keys: Ruta del archivo con las claves públicas.

El archivo de configuración debe quedar como se muestra en la figura 135.

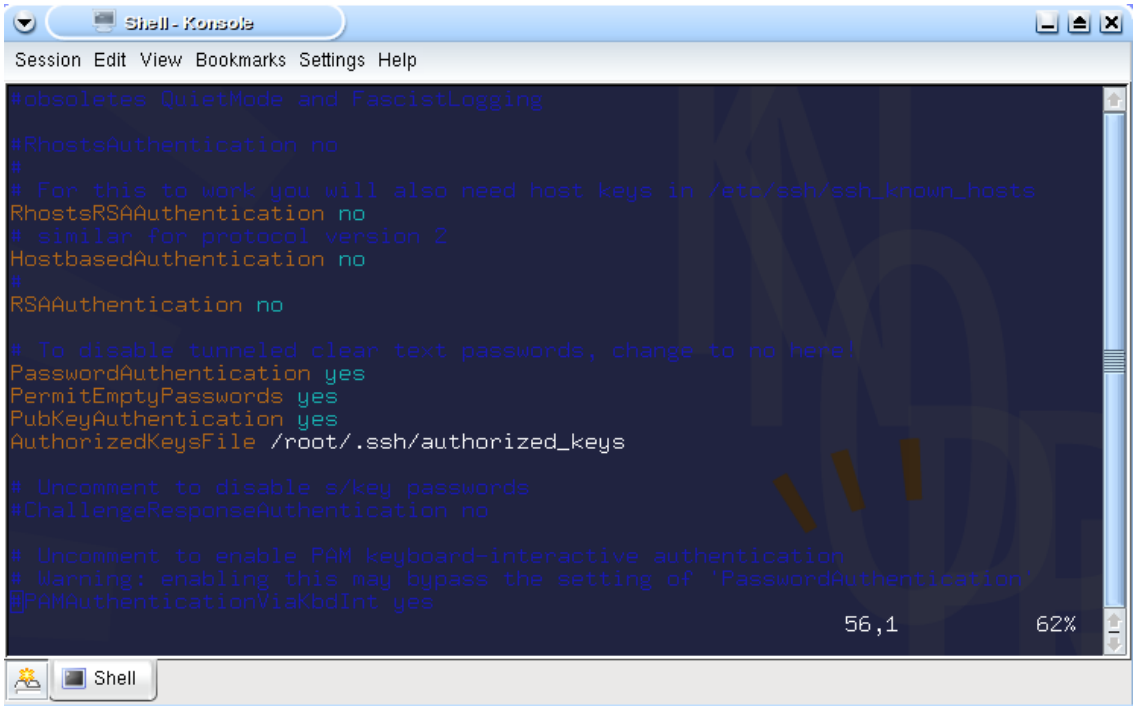


Figura 135. Archivo de configuración final de SSH

Fuente: Elaboración propia

Presionar la tecla "**Esc**" guardar los cambios con ":x" , seguidamente se procede a probar que no se necesite autenticación al realizar conexiones vía SSH con la dirección IP del ordenador como se muestra en la figura 136.

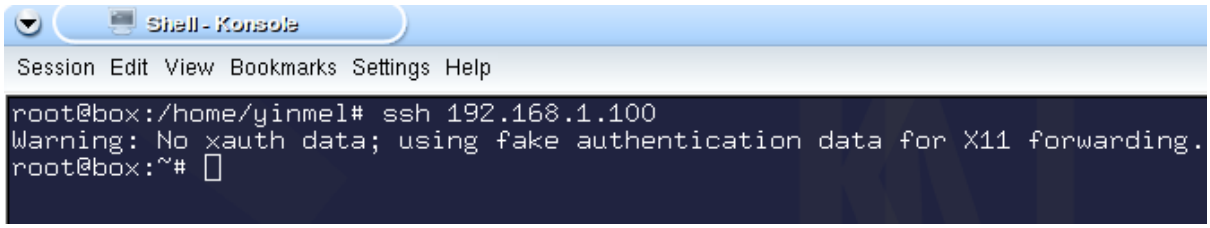


Figura 136. Verificación de la conexión SSH

Fuente: Elaboración propia

Esta configuración de SSH permite que las conexiones del clúster siempre se hagan de forma cifrada. Evitando que intrusos se cuelen en ella, así como que nodos no autorizados accedan a los procesos y estos puedan llegar a ser alterados.

Una vez finalizado todo el proceso de configuración del sistema se debe proceder a **reiniciar el equipo.**

Seguidamente se puede observar la aplicación que sirve para monitorear el clúster, Openmosixview es un programa que contiene información correspondientes al clúster como: Identificado de nodo, balanceo de carga, dirección IP del nodo, total de memoria utilizada por el nodo, cantidad de CPUs del clúster, cantidad de memoria RAM del clúster, identificador de procesos por nodo.

Se puede apreciar en la siguiente figura el programa de Openmosixview.

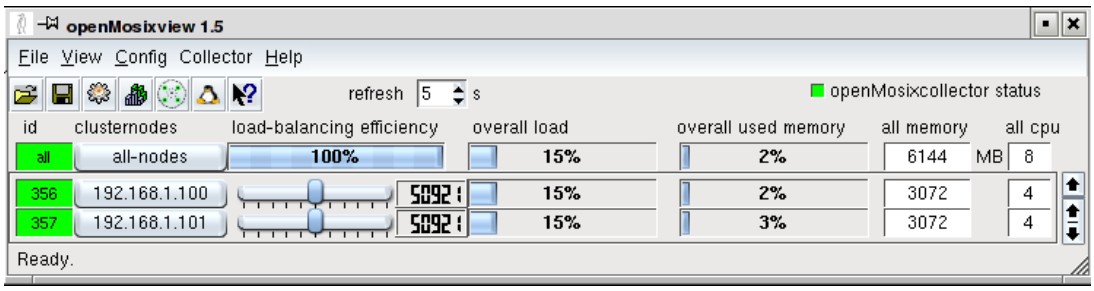


Figura 137. Interface de Openmosixview

Fuente: Elaboración propia

Como se puede observar en la figura, se muestran todas las características del programa Openmosixview.

ANEXO 10

INSTALACIÓN DE POV RAY EN EL NODO MAESTRO

Para instalar la aplicación de Povray se debe ingresar al entorno grafico como súper usuario desde la pantalla de logueo especificando la clave de la misma como muestra la figura 138.

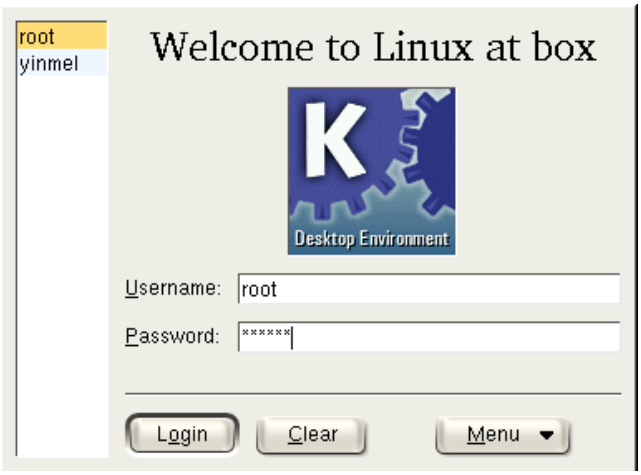


Figura 138. Ingreso como súper usuario al sistema

Fuente: Elaboración propia

Se descargaron los archivos Povray-3.6.1.tar.gz y Povmosix.-1.0-b2.tar.bz2 a una memoria flash usb, por defecto el sistema Clusterknoppix no detecta unidades usb, este proceso se debe realizar manualmente, para ello se debe ingresar a una terminal y escribir el comando "fdisk -l" para poder visualizar los dispositivos conectados como muestra la figura 139.

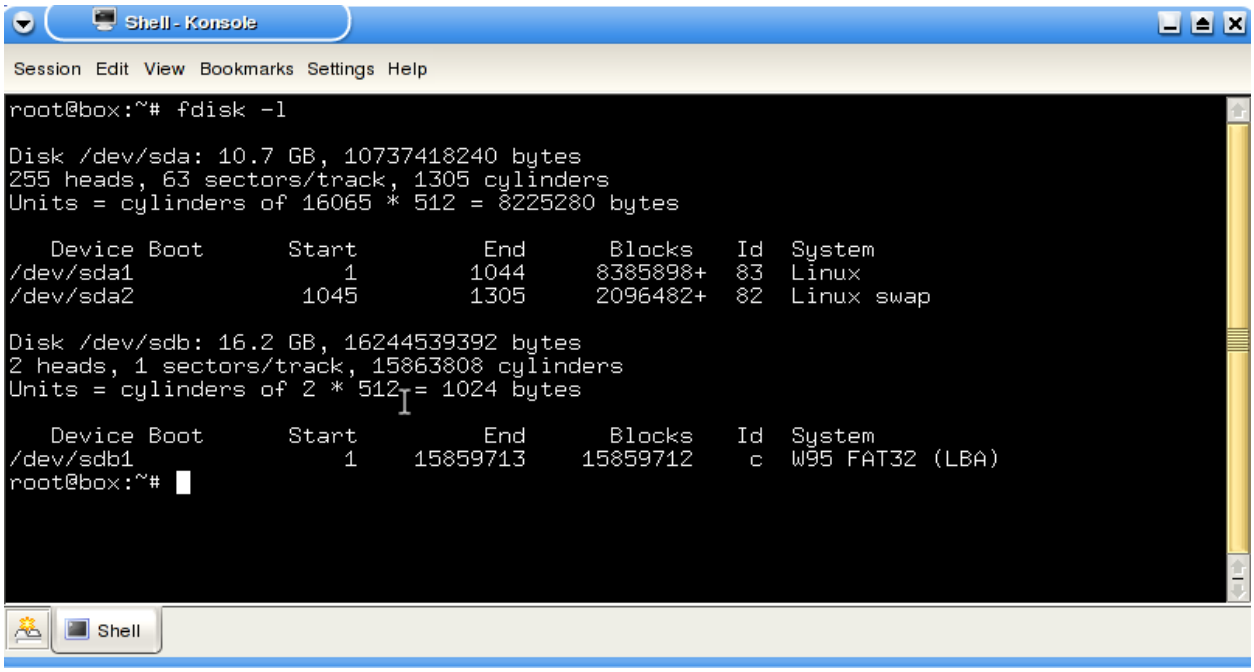


Figura 139. Dispositivos conectados

Fuente: Elaboración propia

Como se puede ver el nombre de la unidad usb es sdb1, por lo cual se debe proceder a crear un directorio con nombre “usb” con el comando “mkdir usb” como muestra la figura 140.

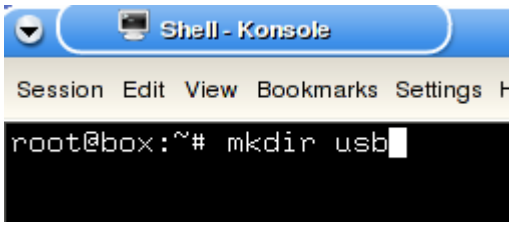


Figura 140. Creación del directorio para montar memorias flash

Fuente: Elaboración propia

Se procederá a montar la memoria usb con el comando mount y los parámetros especificados en la figura 141.

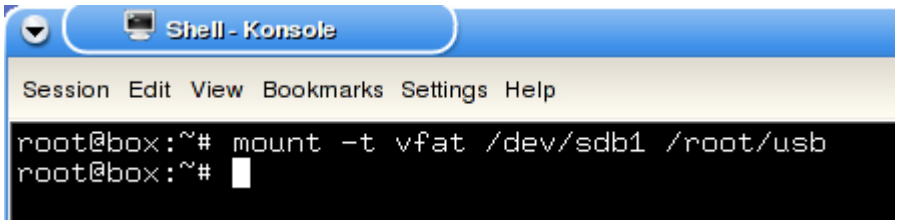


Figura 141. Montando una unidad USB

Fuente: Elaboración propia

Ahora se debe compilar los dos archivos de instalación a la carpeta personal de root como se muestra en la figura 142.

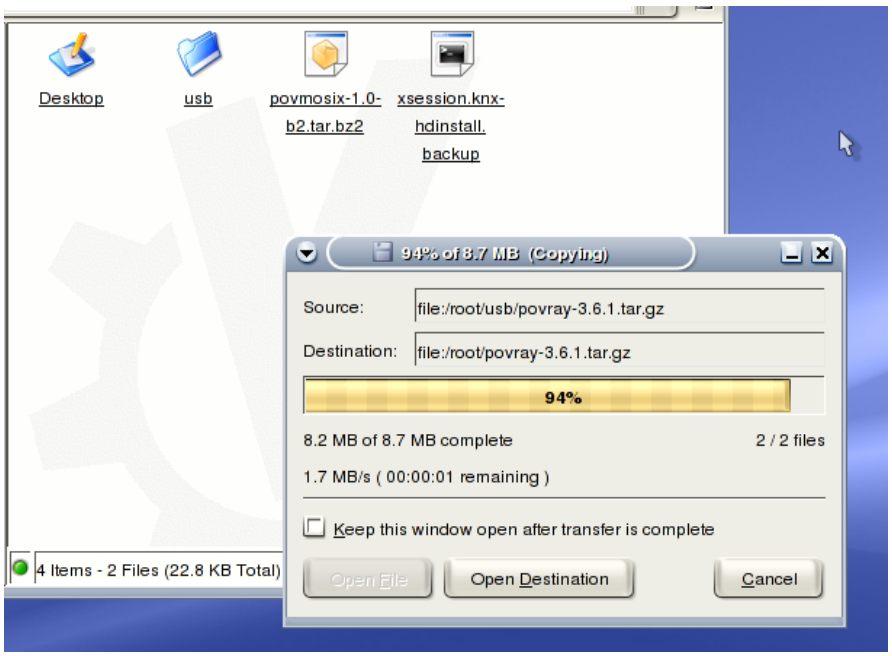


Figura 142. Copiando la carpeta de instalación en la carpeta root

Fuente: Elaboración propia

Se debe descomprimir los dos archivos simplemente haciendo clic derecho sobre ellos y escogiendo la opción “Actions” y haciendo clic en “Extrac here”, se generarán dos carpetas con nombres povmosix y Povray-3.6.1 como se muestra en la figura 143.

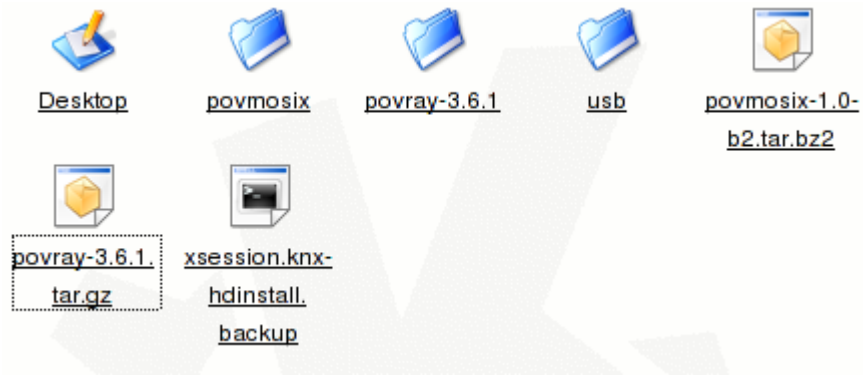


Figura 143. Extracción de los archivos de instalación

Fuente: Elaboración propia

Seguidamente se debe ingresar a un terminar y direccionar la carpeta Povray-3.6.1 con el siguiente comando como muestra la figura 144.

Figura 144. Accediendo al directorio de Povray

Fuente: Elaboración propia

Escribir el siguiente comando para iniciar el proceso de compilación ingresando como parámetro el nombre de la persona que compilará el código fuente como muestra la figura 145.

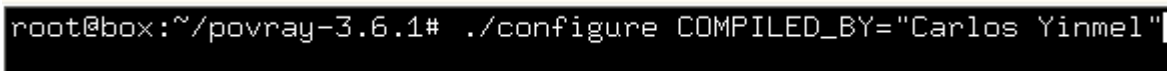


Figura 145. Compilando las fuentes del programa Povray

Fuente: Elaboración propia

Se debe esperar a que culmine el proceso de compilación como muestra la figura 146.

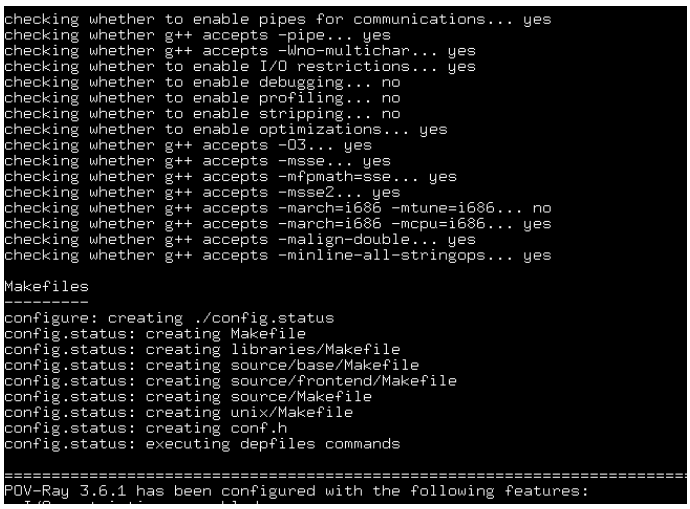


Figura 146. Compilación terminada de Povray

Fuente: Elaboración propia

A continuación se debe escribir el comando “make“ y esperar la preparación de los archivos como se muestra en la figura 147.

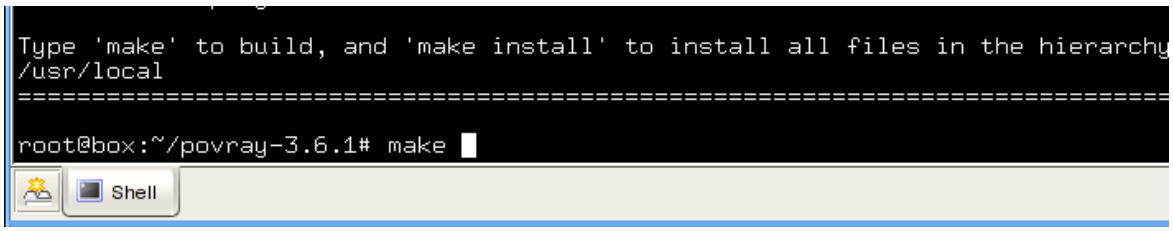


Figura 147. Preparando la instalación

Fuente: Elaboración propia

Seguidamente se debe preparar la instalación con el comando “make install” como se muestra en la figura 148.

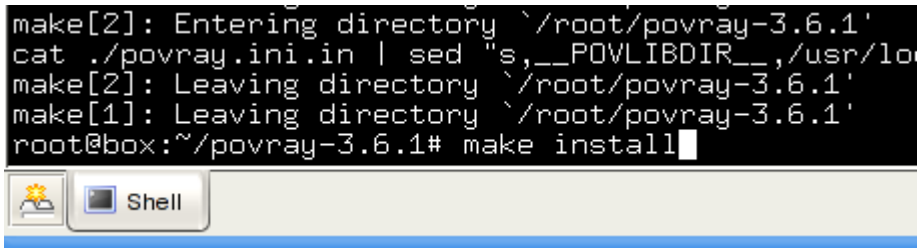


Figura 148. Instalando Povray

Fuente: Elaboración propia

Una vez finalizado el proceso se debe tener instalado el programa de Povray, se debe retroceder un directorio e ingresar a la carpeta Povmosix con el comando cd como muestra la figura 149.

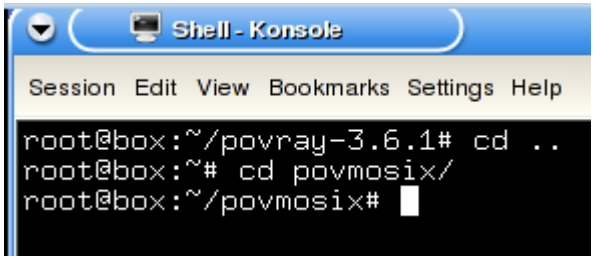


Figura 149. Ingresando al directorio de Povmosix

Fuente: Elaboración propia

Se realizará la compilación de las fuentes de Povmosix, para ello se debe escribir el comando "qmake" y seguido el comando "make" como muestra la figura 150.

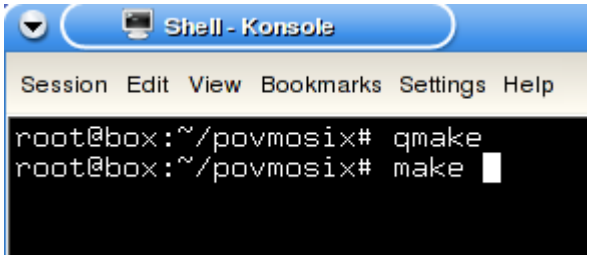


Figura 150. Compilar fuentes

Fuente: Elaboración propia

Se compilará el código fuente del programa Povmosix, se debe esperar a que finalice la instalación como se muestra en la figura 151.

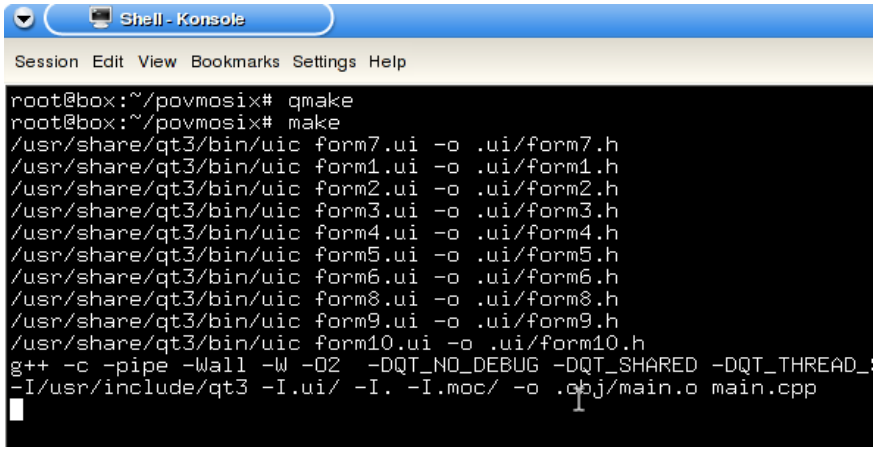


Figura 151. Finalización de la compilación de Povmosix

Fuente: Elaboración propia

Esto generará un ejecutable con nombre Povmosix en el directorio /root/povmosix, para ejecutarlo simplemente se debe abrir el archivo mencionado como se muestra en la figura 152.

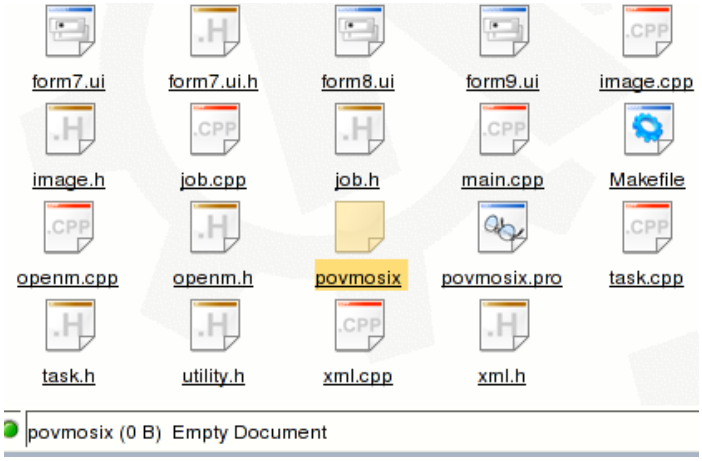


Figura 152. Archivos generados tras la compilación de Povmosix

Fuente: Elaboración propia

Se tendrá entonces instalado el programa Povmosix, se muestra en la figura153 la interface del programa Povmosix**.**

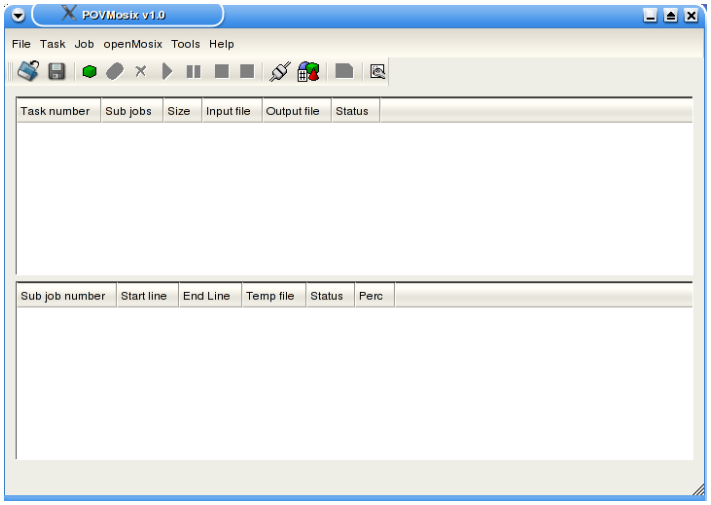


Figura 153. Interface del programa Povmosix

Fuente: Elaboración propia

GLOSARIO DE TÉRMINOS

AGPL: La licencia pública general de Affero (en inglés, Affero General Public License, también Affero GPL o AGPL) es una licencia copyleft derivada de la Licencia Pública General de GNU diseñada específicamente para asegurar la cooperación con la comunidad en el caso de software que corra en servidores de red.

BSD: Berkeley Software Distribution o BSD (en español, "distribución de software Berkeley") es un sistema operativo derivado del sistema Unix nacido a partir de los aportes realizados a ese sistema por la Universidad de California en Berkeley.

CACHE: Es un búfer especial de memoria que poseen los ordenadores. Funciona de una manera similar a como lo hace la memoria principal (RAM), pero es de menor tamaño y de acceso más rápido. Es usado por la unidad central de procesamiento para reducir el tiempo de acceso a datos ubicados en la memoria principal que se utilizan con más frecuencia.

CAUSTICA: Una cáustica es la envolvente de los rayos de luz reflejados o refractados por una superficie curva u objeto, o la proyección de esa envolvente de rayos en otra superficie. El término cáustica puede también referirse a la curva en la cual los rayos de luz son tangentes, determinando una frontera de la envolvente de rayos como una curva de luz concentrada.

CLAN (Clúster LAN): Es una tecnología de interconexión de ordenadores para soluciones en clúster.

CLÚSTER: El término clúster (del inglés cluster, "grupo" o "racimo") se aplica a los conjuntos o conglomerados de computadoras construidos mediante la utilización de hardwares comunes y que se comportan como si fuesen una única computadora.

FAIL OVER: Tolerancia a fallos, hace referencia a la capacidad de un sistema de acceder a la información, aun en caso de producirse algún fallo o anomalía en el sistema.

FAST ETHERNET: Fast Ethernet o Ethernet de alta velocidad es el nombre de una serie de estándares de IEEE de redes Ethernet de 100 Mbps (megabits por segundo). El nombre Ethernet viene del concepto físico de ether. En su momento el prefijo fast se le agregó para diferenciarla de la versión original Ethernet de 10 Mbps.

FLOPS: Floating point operations per second, operaciones de coma flotante por segundo, son una medida del rendimiento de una computadora, especialmente en cálculos científicos que requieren un gran uso de operaciones de coma flotante.

GCA: La Color Graphics Adapter (Adaptador de Gráficos de Color) o CGA, comercializada en 1981, fue la primera tarjeta gráfica en color de IBM (originalmente llamada Color/Graphics Adapter or IBM Color/Graphics Monitor Adapter,1), y el primer estándar gráfico en color para el IBM PC.

GIGABIT ETHERNET: Gigabit Ethernet, también conocida como GigaE, es una ampliación del estándar Ethernet (concretamente la versión 802.3ab y 802.3z del IEEE) que consigue una capacidad de transmisión de 1 gigabit por segundo, correspondientes a unos 1000 megabits por segundo de rendimiento contra unos 100 de Fast Ethernet (También llamado 100BASE-TX).

HPC: Clúster de alto rendimiento es un conjunto de ordenadores que está diseñado para dar altas prestaciones en cuanto a capacidad de cálculo.

LATENCIA: En redes informáticas de datos se denomina latencia a la suma de retardos temporales dentro de una red. Un retardo es producido por la demora en la propagación y transmisión de paquetes dentro de la red.

LGPL: La Licencia Pública General Reducida de GNU, o más conocida por su nombre en inglés GNU Lesser General Public License (antes GNU Library General Public License o Licencia Pública General para Bibliotecas de GNU), o simplemente por su acrónimo del inglés GNU LGPL, es una licencia de software creada por la Free Software Foundation que pretende garantizar la libertad de compartir y modificar el software cubierto por ella, asegurando que el software es libre para todos sus usuarios.

LOAD BALANCING: El balance o balanceo de carga es un concepto usado en informática que se refiere a la técnica usada para compartir el trabajo a realizar entre varios procesos, ordenadores, discos u otros recursos. Está íntimamente ligado a los sistemas de multiprocesamiento, o que hacen uso de más de una unidad de procesamiento para realizar labores útiles.

MIDDLEWARE: Conjunto de servicios o facilidades a la que es posible acudir en el ámbito de una arquitectura.

MIMD: En la arquitectura de computadores hace referencia a los modelos de ordenadores que procesan múltiples instrucciones con múltiples datos.

MISD: En la arquitectura de computadores hace referencia a los modelos de ordenadores que procesan múltiples instrucciones con un solo dato.

MOSIX: Es una arquitectura para soluciones en clúster que se basa en un conjunto de parches aplicados al kernel de Linux y que se asignan a todo un grupo de nodos un espacio de direcciones y de procesos común.

MPI: Message Passing Interface, Interfaz de Paso de Mensajes, es un estándar que define la sintaxis y la semántica de las funciones contenidas en una biblioteca de paso de mensajes diseñada para ser usada en programas que exploten la existencia de múltiples procesadores.

MPICH: Es un programa de biblioteca de desarrollo de libre disposición implementación portable de MPI, una norma estándar de paso de mensaje para aplicaciones de memoria distribuida que utilizan computación paralela. MPICH es software gratuito y disponible para la mayoría de Unix (incluyendo Linux y Mac OS X) y Microsoft Windows.

MIRYNET: Es una red de interconexión de clúster de altas prestaciones. Sus productos han sido desarrollados por Myricom desde 1995. Myrinet físicamente consiste en dos cables de fibra óptica, upstream y downstream, conectados con un único conector. La interconexión se suele realizar mediante conmutadores y encaminadores. Estos dispositivos suelen tener capacidades de tolerancia a fallos, con control de flujo, control de errores y monitorización de la red. Desde su creación se ha incrementado su rendimiento hasta alcanzar latencias de 3 microsegundos y anchos de banda de 10 Gbit/s:

OPENMOSIX: Es una extensión del proyecto Mosix, la idea de este modelo es que la distribución de tareas en el clúster la determina Openmosix de forma dinámica, conforme se van creando tareas.

OPENMOSIXCOLLECTOR: Demonio de Openmosix encargado de recolectar información acerca de los nuevos nodos agregados al clúster, nodos dados de baja al clúster, este proceso de identificación se realiza utilizando tecnologías de interconexión de redes de área local.

OPENMOSIXVIEW: Programa de Openmosix que permite monitorizar el clúster, entre los datos más importante que brinda este programa podemos hallar: los identificadores de nodos, números de procesos, balanceo de carga, migración de procesos, memoria utilizada, cantidad de nodos, cantidad de CPUs, cantidad de Memoria RAM entre otros.

POVMOSIX: Programa que utiliza Openmosix de base para renderizar escenas tridimensionales basadas en trazado de rayos con la utilización de software de Povray.

PVM: La Máquina Virtual Paralela (conocida como PVM por sus siglas en inglés de Parallel Virtual Machine) es una biblioteca para el cómputo paralelo en un sistema distribuido de computadoras. Está

diseñado para permitir que una red de computadoras heterogénea comparta sus recursos de cómputo (como el procesador y la memoria RAM) con el fin de aprovechar esto para disminuir el tiempo de ejecución de un programa al distribuir la carga de trabajo en varias computadoras.

POVRAY: Programa que interpreta lenguaje de computación gráfica basada en trazado de rayos para el proceso de renderización de escenas tridimensionales.

RENDERIZAR: Render en inglés, es un término usado en jerga informática para referirse al proceso de generar una imagen o vídeo mediante el cálculo de iluminación GI partiendo de un modelo en 3D. Este término técnico es utilizado por los animadores o productores audiovisuales (CG) y en programas de diseño en 3D como por ejemplo Studio 3DMax, Maya, Blender, Solid works, etc

SIMD: En la arquitectura de computadores hace referencia a los modelos de ordenadores que procesan solamente una instrucción con múltiples datos.

SISD: la arquitectura de computadores hace referencia a los modelos de ordenadores que procesan solamente una instrucción con un solo dato.

SSH: Secure Shell, en español: intérprete de órdenes segura, es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red. Permite manejar por completo la computadora mediante un intérprete de comandos, y también puede redirigir el tráfico de X para poder ejecutar programas gráficos si tenemos un Servidor X en sistemas Unix y Windows corriendo.