**IMPLEMENTACION SUPERCOMPUTADORA**

**GRUPO SUPERCOMPUTADORA**

**LIDER: EDGAR TORRES COY**

**PROFESOR: JOHN EDWARD CASTRO BERNETH**

**FUNDACION UNIVERSITARIA SAN MATEO**

**INGENIERIA DE SISTEMAS**

**COMPUTACION PARALELA Y DISTRIBUIDA**

**BOGOTA**

**2019**

Contenido

[**PROPUESTAS DEL TRABAJO A REALIZAR** 3](#_Toc25509627)

[**Propuesta 1:** 3](#_Toc25509628)

[**Propuesta 2:** 3](#_Toc25509629)

[**Propuesta 3:** 4](#_Toc25509630)

[**PROPUESTA SELECCIONADA** 5](#_Toc25509631)

[**CRONOGRAMA** 6](#_Toc25509632)

[**ROLES** 6](#_Toc25509633)

[**METODOLOGÍA A UTILIZAR (SI SE REQUIERE).** 8](#_Toc25509634)

[**ARQUITECTURA DEL DESARROLLO DEL PROYECTO.** 8](#_Toc25509635)

[**Arquitectura SOA** 8](#_Toc25509636)

[**PARTE 2 DE LA INVESTIGACIÓN DEL PROYECTO REALIZADO.** 9](#_Toc25509637)

[**PVM (Parallel virtual Machine)** 9](#_Toc25509638)

[**MPI (Message Passing Interface)** 10](#_Toc25509639)

[**MOSIX** 10](#_Toc25509640)

[**METODOLOGÍA SCRUM** 11](#_Toc25509641)

[**Introducción** 11](#_Toc25509642)

[**Fundamentación** 12](#_Toc25509643)

[**Manifiesto Ágil** 12](#_Toc25509644)

[**Principios en Scrum** 13](#_Toc25509645)

[**Definición del proceso Scrum** 14](#_Toc25509646)

[**Definición y caracterización de los Eventos.** 17](#_Toc25509647)

[**Definición y caracterización de los Artefactos.** 19](#_Toc25509648)

[**Personas y roles del proyecto.** 20](#_Toc25509649)

[**DISEÑO** 22](#_Toc25509650)

[**CONFIGURACIÓN** 22](#_Toc25509651)

# **PROPUESTAS DEL TRABAJO A REALIZAR**

## **Propuesta 1:**

Se ejecutará el desarrollo de una súper computadora mediante equipos físicos los cuales estarán interconectados mediante un switch y cables de red, permitiendo su comunicación, los equipos contarán con sistema operativo basado en Linux (se sugiere Debian o Pelican).

La metodología que se usará para la construcción de la súper computadora será Scrum ya que es una metodología ágil que nos permite acoplar los procesos de desarrollo que se requieren para la realización del proyecto.

Su arquitectura será la SOA (Arquitectura orientada a servicios), dado que su meta es aportar la flexibilidad mediante la automatización de infraestructuras y herramientas necesarias para ejecutar la creación de una súper computadora.

Como aporte a la materia este trabajo nos permitirá evidenciar físicamente la forma cómo se comporta una supercomputadora, mediante la estructura propuesta, y compararla con un solo clúster, con la finalidad de verificar su desempeño y sus bondades en la automatización de procesos.

## **Propuesta 2:**

Comprobar mediante el uso de máquinas virtuales el desempeño que tiene una supercomputadora, configurando un equipo como master y los demás como nodos, para brindar al master las fuerzas necesarias para ejecutar más eficientemente los procesos, los cuales usaran un sistema operativo basado en Linux.

La arquitectura bajo la cual se desarrollará será SOA (Arquitectura orientada a servicios), la cual nos brinda flexibilidad en la automatización de los procesos que ejecutan cada uno de los nodos al servicio del equipo Master.

La metodología que se pretende usar para el desarrollo de la supercomputadora es Scrum la cual nos brinda la posibilidad de llevar en mejor forma las actividades para el desarrollo del trabajo.

Como aporte a la materia, la demostración de la virtualización de las maquinas nos permitirá ver en forma más clara como se pueden compartir recursos entre varios equipos para mejorar el desempeño de una máquina.

## **Propuesta 3:**

La metodología Ágil que se pretende implementar es KANBAN ya que esta es similar a la metodología SCRUM, no poseemos historia de usuario, nos permite el trabajo en paralelo y la asignación de tareas con sus respectivos estados (To Do, Doing, Don)

Descripción del desarrollo del trabajo

* La duración de los Sprint será equivalente a 7 días
* Las tareas que se identifiquen serán asignadas dependiendo de las habilidades que tenga cada uno de los integrantes
* Los Dailys se realizarán a través del grupo de wathsapp (si se es requerido) cada uno tendrá duración de 15 minutos aprox. En el cual se expondrá las tareas en las que se está trabajando y se presentaran los posibles impedimentos

Arquitectura (backend)

Teniendo en cuenta la necesidad es viable la arquitectura SOA, que nos permite la distribución de los servicios en diferentes servidores para el trabajo en paralelo. Se pretende crear un bus de servicios, esto con el fin de centralizar y gestionar la comunicación entre los diferentes servicios, estos podrían estar alojados en dockers para poder suministrar los recursos necesarios para que cada servicio funcione de manera satisfactoria y en caso de fallar no afecte el funcionamiento de los otros.

Arquitectura (Infraestructura)

Balanceador, para la distribución de peticiones entre las máquinas que se vayan a implementar.

Si es posible, X cantidad de máquinas con las mismas características. En caso contrario, X cantidad de máquinas virtualizadas.

Tecnologías y Herramientas

• Servicios REST con dotNet Core para garantizar la interoperabilidad entre sistemas (Windows y Linux).

• Dockers, la implementación de estos es viable en Windows y Linux.

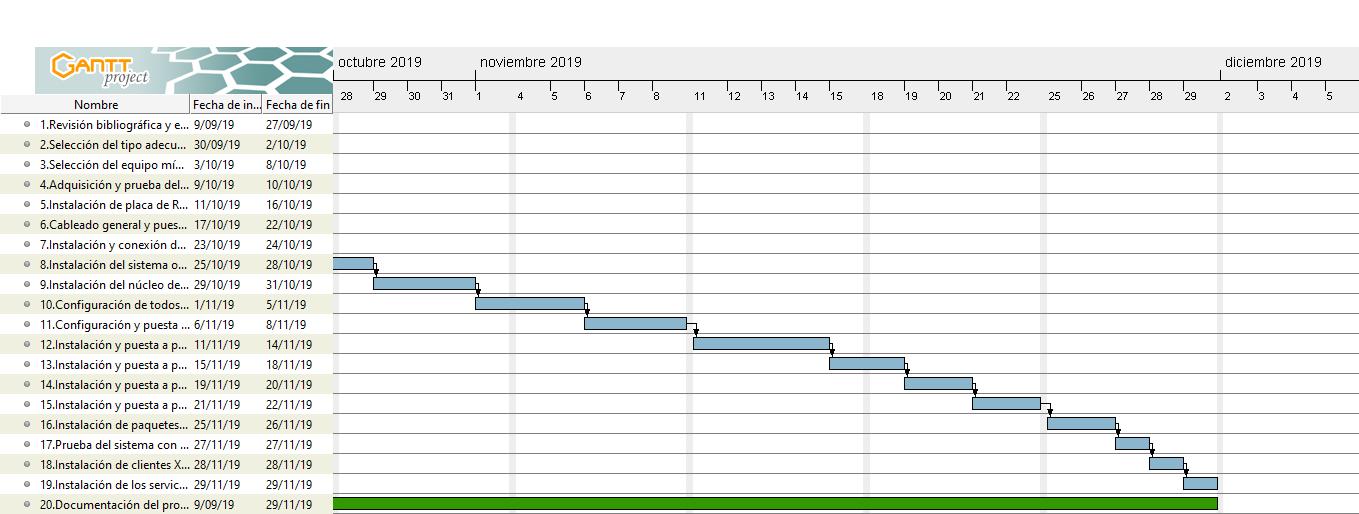
El aporte que pretendemos dar a la clase es el de realizar por medio de una comprobación física o virtual las bondades de la creación de supercomputadoras y como al realizar cualquiera de las tres propuestas estamos realizando una comprobación de los procesos educativos y los temas vistos durante el desarrollo de la materia.

# **PROPUESTA SELECCIONADA**

La propuesta seleccionada para realizar el trabajo es la # 1, teniendo en cuenta lo expuesto en la misma se realizarán las siguientes actividades para poder cumplir con la propuesta:

1. Revisión bibliográfica y estudio de las bases de funcionamiento de los clusters.
2. Selección del tipo adecuado para las necesidades del grupo (MOSIX).
3. Selección del equipo mínimo necesario para implementar el cluster elegido.
4. Adquisición y prueba del hardware seleccionado.
5. Instalación de placa de Red adicional en el nodo maestro.
6. Cableado general y puesta a punto de la conexión del servidor a la red local de la FUS.
7. Instalación y conexión de los nodos mediante un Switch.
8. Instalación del sistema operativo Linux en todas las PCs
9. Instalación del núcleo del sistema MOSIX en ambas PCs
10. Configuración de todos los servicios y protocolos de red necesarios.
11. Configuración y puesta a punto del MOSIX
12. Instalación y puesta a punto del MPI
13. Instalación y puesta a punto del PVM
14. Instalación y puesta a punto del sistema de colas de procesos (PBS).
15. Instalación y puesta a punto del sistema de administración remota (Webmin).
16. Instalación de paquetes, lenguajes y herramientas específicas para cálculo numérico.
17. Prueba del sistema con diferentes aplicaciones (software de redes neuronales, programas de cálculo pesado, etc.)
18. Instalación de clientes X (entorno gráfico) y SSH para acceso desde las terminales Windows.
19. Documentación del proceso (versiones preliminares de este informe)

# **CRONOGRAMA**



# **ROLES**

|  |  |  |
| --- | --- | --- |
| **Rol** | **Responsabilidades:** | **Entregable:** |
| Tester (Jhon Jairo López Sáez) | Encargado de realizar diferentes pruebas desde el primer prototipo - modelo hasta el final en lineamiento con la arquitectura y metodología colaborativa propuesta; asi commo el registro de los cambios y mejoras en la evolución en el tiempo. | Diagramas y resultados de pruebas obtenidas. |
| Testing (Brayam Estiven Duran Cardona) | Encargado de realizar pruebas funcionales y no funcionales con el fin de detectar defectos en el uso de la plataforma | Pruebas documentadas realizadas (test configuration, set de pruebas) |
| Implementador (David Gerardo Urrego Camargo) | Encargado de realizar y verificar los componentes del proyecto de acuerdo con la metodología y la arquitectura establecida al inicio del proyecto, al igual será el responsable de implementar un prototipo y generar un reporte respecto. | Documentación Técnica |
| Desarrollador (Jhonatan Stiv Cuadros Cárdenas) | Encargado de implementar todos los requerimientos relacionados en el documento ya sea con un software o con la implementación y/o configuración de la supercomputadora establecida en el documento, siguiendo los parámetros de la metodología. | Implementación |
| Diseñador (Anderson Rubio) | Delegado de la implementación de la metodología, relaciona el diseño con la implementación incluyendo el nivel de detalle esperado en el diseño antes de que proceda la implementación.  Así como la verificación de los requerimientos para el desarrollo del proyecto.  creación de un concepto de sistema que ayude a cumplir los objetivos del proyecto | Diagramas de estructura  Requerimientos de Vista metodología de proyecto |
| Analista (Tatiana Pinzon) | Encargada de analizar la información para el levantamiento de los requerimientos y las necesidades para la ejecución del proyecto. | Matriz de requerimientos |
| Edwin Alexander Chavez Barreto | Encargado del soporte técnico sobre los equipos de cómputo a intervenir para la implementación de la supercomputadora, instalación de sistemas operativos necesarios y mantenimiento funcional de los aplicativos y conexión y configuración necesaria para la conexión de la red de datos | Informe técnico, usuarios y claves de acceso, informe de configuración de los sistemas operativos en cada equipo de cómputo. |

# **METODOLOGÍA A UTILIZAR (SI SE REQUIERE).**

**Metodología Scrum:** Esta nos permite planificar la realización de la ejecución del proyecto en pequeños springs, que se ejecutaran por semana según el cronograma

El desarrollo de producto tiene un ciclo de vida en la metodología Scrum. Estas son fases en las que se divide un proceso Scrum:

¿Qué y quién? El producto que queremos conseguir una vez terminemos el Sprint, y los roles de equipo con sus tareas asignadas, según cada semana programada que se debe intervenir sobre el proyecto

Se efectuará una reunión para la planificación del Sprint de cada semana. En ella, se divide el tiempo de duración del Sprint, así como el objetivo y entregable del mismo. Además, el equipo de desarrollo deberá saber cómo realizarlo, el cual ya se encuentra constatado en el cronograma establecido, para algunas actividades será un Scrum diario. Para ejecutar actividades para elaborar el plan del día, adicionalmente se verificará si lo objetivos del proyecto se están cumpliendo

Se tendrá también una reunión del sprint del líder del proyecto con el docente encargado para verificar si se encuentra en cumplimiento los requisitos o si se requiere ejecutar algún tipo de modificación, la cual se ejecutará al después de acabar el sprint correspondiente.

# **ARQUITECTURA DEL DESARROLLO DEL PROYECTO.**

## **Arquitectura SOA**

La meta de esta arquitectura es aportar la flexibilidad de activos existentes que permite la automatización de infraestructuras y herramientas necesarias para ejecutar la creación de una súper computadora.

Por su versatilidad es posible que los servicios puedan ser consumidos por los clientes en aplicaciones o procesos de negocios distintos ya que tiene una ventaja que es la interoperabilidad entre ellas

# **PARTE 2 DE LA INVESTIGACIÓN DEL PROYECTO REALIZADO.**

## **PVM (Parallel virtual Machine)**

En su traducción al español Máquina Virtual Paralela, es un sistema o conjunto de bibliotecas que se usa principalmente dentro de la programación paralela, la cual me permite realizar una conexión entre dos o más equipos de cómputo, con el objetivo integrar los recursos de estas computadoras, para finalmente obtener el esquema de una sola máquina.

Este software opera en diferentes plataformas como lo pueden ser (Unix y Windows), aunque para la creación del proyecto se usara o se implementara dentro de la plataforma Linux.

Algunas de las funciones del uso del PVM consiste en la mensajería asincrónica y para el control de procesos entre los diferentes nodos presentes dentro del sistema, además que permite la escalabilidad del mismo.

**Ventajas:**

Entre las diferentes librerías existentes para el envío de mensajes en el paralelismo es uno de los más fáciles de usar.

Control arbitrario de dependencias de estructura, esto permite que la aplicación determine: (Donde y cuando Iniciar o terminar tareas, que maquinas se añaden o se eliminan desde la máquina virtual y finalmente que tareas se puede comunicar y/o sincronizar con otras.)

**Desventajas:**

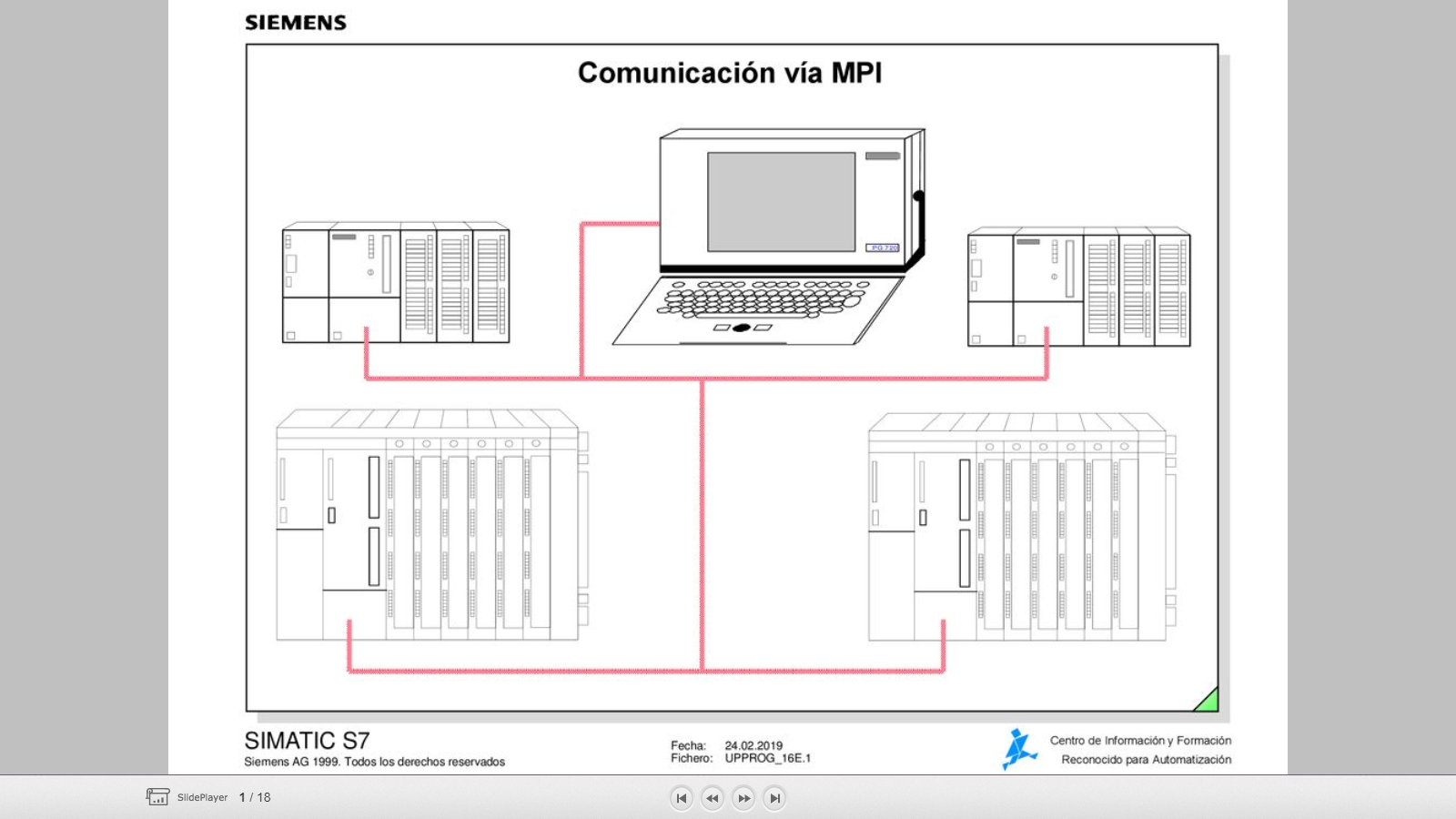
En gran medida el rendimiento del esquema dependerá de la capacidad de procesamiento de los ordenadores o nodos que se encuentran dentro del esquema.

Lo que es una ventaja también se puede convertir en una desventaja, ya que, aunque es muy fácil de usar, esto conlleva a que sea algo deficiente en cuanto al paso de mensaje entre nodos.

## **MPI (Message Passing Interface)**

Es una librería que permite el paso de mensajes, ya sea de forma estándar y/o portable, se utilizan principalmente a la hora de realizar cálculos dentro de un clúster, se usa a través de un programa que permita la migración de procesos entre múltiples procesadores.

El envío de mensaje es una técnica que se usa en programación concurrente que permite la sincronización de diversos procesos que se desean ejecutar entre nodos.



## **MOSIX**

Es un software con capacidades de computación de clústers, específicamente permite una mejora en el Kernel (Intermediario entre software y hardware) de Linux, esta mejora permite que cualquier clúster de estaciones de trabajo y servidores trabajen coordinadamente como parte de un solo sistema, permitirá ejecutar aplicaciones no paralelizados en un clúster, en otras palabras el software permite ejecutar programas convencionales destinados para funcionar en una maquita, en un clúster a través del envío de mensajes o la migración de procesos entre los nodos que se encuentran dentro del sistema.

Una característica muy eficiente que tiene MOSIX dentro del funcionamiento de un clúster, es que realizara un monitoreo constante en los procesos que encuentre o que se están ejecutando y es de esta forma que dividir de forma correcta los procesos entre los nodos y generar un equilibrio, cabe destacar que este proceso antes mencionado lo realizara de forma automática.

Debemos tener en cuenta que solo algunas aplicaciones se beneficiaran bajo este sistema de clúster que permita MOSIX, algunos de estos son:

* Procesos que necesiten para su funcionamiento el uso de una gran cantidad de CPU.
* Procesos que tengan tiempos de ejecución impredecibles.
* Un gran número de usuarios simultáneos (Concurrencia).
* Servidores Web predispuestos a escalabilidad.

# **METODOLOGÍA SCRUM**

## **Introducción**

En primera instancia, los roles involucrados dentro del proyecto serán el: (**Product owner** y el **scrum master**), quien respectivamente se encargarán de realizar el primer contacto con el cliente para la toma de requerimientos para luego crear el product blacklog y de realizar la planeación de la metodología scrum como tal. Esta labor deberá ser realizada durante las primeras dos semanas del proyecto, sin todavía contar con el equipo de desarrollo.

Luego de trascurrido el periodo antes mencionado, se inició el proceso de sprint como tal, que será de **2 semanas** cada uno, aquí es donde con el equipo scrum completo, ya con el development team completo, inician el sprint planning meeting que durará **4 horas** por ser un sprint de 2 semanas, aquí se creará el sprint backlog que a su vez será el sprint goal del sprint.

Luego de llegar a un acuerdo, dar transparencia del product backlog por parte del product owner y especificar de forma clara la metodología a través del scrum master. Se inicia el proceso de desarrollo junto con las reuniones diarias, lideradas por el equipo de desarrollo, esta se hará dentro de las instalaciones por un time box de **15 minutos** y el scrum master se hará cargo de que se realice dicho evento.

El refinamiento se realizará los días lunes, más específicamente la primera semana del sprint con duración no mayor a un 10% del equipo de desarrollo, justo antes del evento de sprint planning del nuevo sprint.

## **Fundamentación**

Gracias a la posibilidad que ofrece el marco scrum de respuesta al cambio, se ha convertido en uno de los marcos agiles más usados para proyectos de creación de software, el uso de iteraciones permitirá un control de las tareas a realizar.

1. Para iniciar se debe tener el **product backlog**, con las tareas de usuario priorizadas dependiendo del criterio del dueño del producto, en el product backlog encontraremos los requerimientos deseados por el cliente, los cuales deberán ser explicados hacia el equipo de desarrollo.
2. Dentro del sprint planning meeting, se analizarán y escogerán las historias de usuarios que se planean realizar o desarrollar dependiendo del timebox seleccionado en el sprint, este tendra como nombre sprint backlog.
3. Luego de esto se inician las iteraciones o en otras palabras los sprint, dentro de este tiempo, se realizarán los eventos estipulados en el marco.
4. Y finalmente se debe entregar un incremento de producto potencialmente funcional.

Durante el proceso el objetivo de usar scrum es crear un ambiente empírico, eliminando impedimentos y de liderazgo, principalmente por el scrum master, con el fin de crear un equipo auto organizado, esencialmente del equipo de desarrollo.

El concepto de timebox con sus respectivos tiempos para cada evento, debe ser aclarado por el scrum master y de ser necesario repetir esta información hasta conseguir claridad entre el equipo scrum.

## **Manifiesto Ágil**

A pesar de que el marco scrum está más enfocado hacia la creación de un producto valor, tampoco podemos descartar el uso de documentación, que permitirá una mayor transparencia principalmente con la idea de negocio (Bussines Case) y el modelo a usarse durante el proyecto (Modelo de trabajo).

La asignación clara de las características y deberes de cada rol debe ser estipuladas y aclaradas, junto con la metodología que se usara, cuando el scrum team este completo, esto beneficiara a que en algún momento no exista indecisión, que conlleve a obtener una meta en común.

La metodología permitirá tener respuesta al cambio, tal como se indica más adelante en el producto backlog, todo proyecto evolucionara por ende las historias de usuarios en muy pocas ocasiones terminaran tal cual fueron estipuladas en la primera reunión entre el cliente y el product owner. Es por este motivo el uso de los sprint dentro del marco scrum, por este medio se tendrá un control de los avances y al igual obtener los posibles cambios que el cliente desee, por este motivo el cliente deberá estar dispuesto cuando se necesite, por lo general el cliente designa una persona quien será la persona que deberá estar dispuesta para estos casos.

## **Principios en Scrum**

Durante el uso de la metodología scrum, se debe llevar un control de proceso que permita conocer el trabajo que se está realizando y que se está cumpliendo con lo estipulado al inicio del proceso, para esto la transparencia es fundamental, y los eventos que conforman el marco scrum nos permiten lograr esta transparencia:

* Daily scrum
* Sprint planning meeting
* Sprint review
* Sprint retrospective

Además, el tener presente los artefactos (son explicados con detalle más adelante) y el objetivo de estos, contribuye a que el concepto de terminado sea igual para todo el equipo scrum. Algunos elementos que facilitan optimizar la transparencia dentro del marco son:

* Burndown chart
* Scrumboard

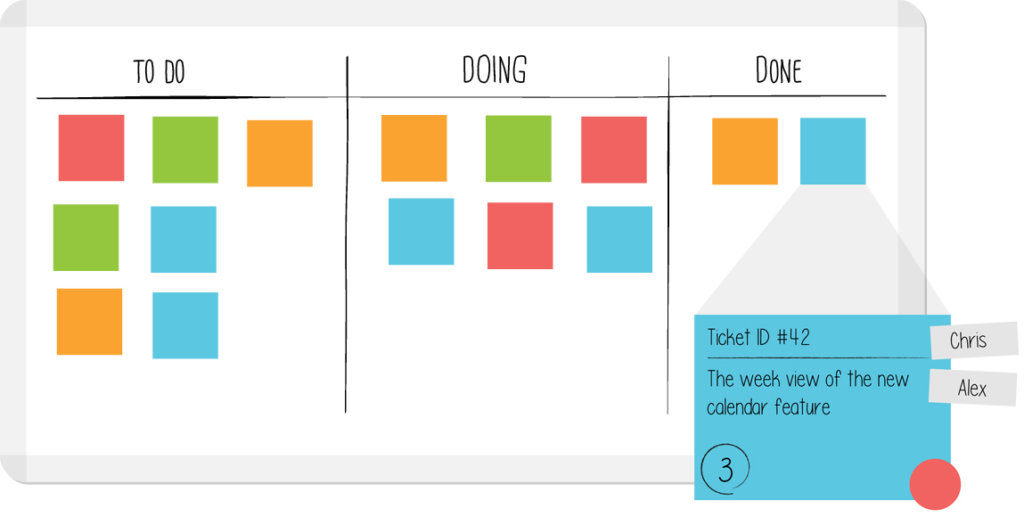


Ilustración 1 Scrum Board

El product owner inicia con el proceso de transparencia, dando prioridad a las historias de usuario y explicando cada una de ellas al equipo de desarrollo, las veces que sea necesarias, al igual el scrum master debe dejar de forma clara la metodología que se usara dentro del marco e incentivar el trabajo empírico y auto organizado por parte del equipo de desarrollo.

## **Definición del proceso Scrum**

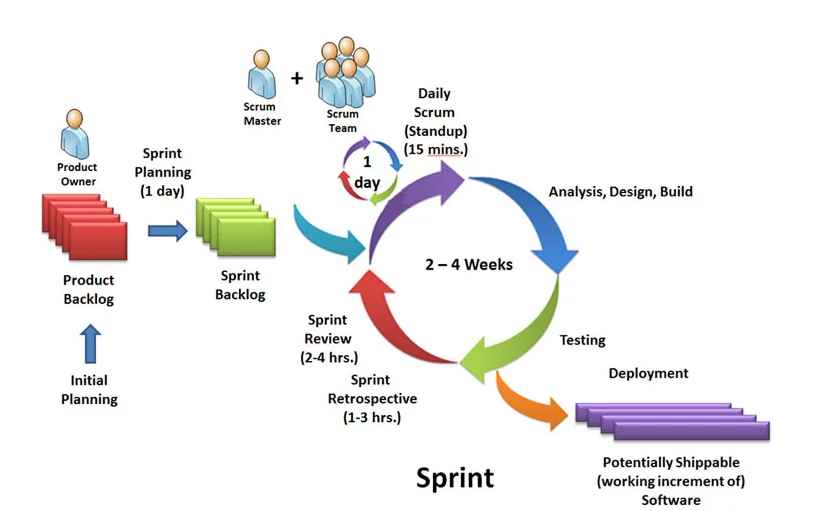


Ilustración 2 Marco Scrum

1. El Primer contacto con el cliente lo debe realizar el product owner, con la idea de crear la lista de requerimientos o historias de usuario, aquí es donde nace el product backlog, que al desarrollar cada una de estas tareas obtendremos el producto final de alto valor. El orden de prioridad del product backlog debe realizarlo el producto owner.
2. El siguiente paso sera el pirmer evento del modelo, llamado sprint planning meeting, para la creacion de la supercomputadora sera de 4 horas cada reunion, debido a los sprint de 2 semanas, aquí el producto owner ya con el product backlog de forma clara y prioriazada, el equipo scrum (Scrum master, Product owner, development team), debe seleccionar o elegir cuales de estas historias de usuario podran ser desarrolladas y entregadas al final del sprint, la lista que fue seleccionada se llamara spirnt backlog.
3. Con el sprint backlog definido se realiza el proceso de tasking, aquí se identificará las tareas que se van a realizar, el tiempo estimado y generalmente se plasman en un taskboard (tablero de tareas).
4. Diariamente durante el time box del sprint, se debe realizara los daily scrum, con el objetivo de que el equipo scrum puede rener una retrospectiva y control del trabajo realizado y por hacer, estos seran de 15 minutos y seran dirigidos por el equipo de desarrollo.
5. Ya al final del sprint, se realiza el sprint review que sera de 2 horas, para el sprint de 2 semanas, el objetivo es evaluar el producto “terminado” o incremento del proyecto que fue estimado durante el sprint planning al inicio del sprint, eso si el product owner decide no cancelar el sprint anteriormente. Los invitados o asistentes a esta reunión son el Scrum master y los invitados autorizados por el product owner generalmente es el cliente como tal, junto con los demás integrantes del equipo scrum.
6. Antes de finalizar el sprint se realiza el sprint retrospective, y todo el equipo scrum debe estar presente, el objetivo de la reunión es analizar lo ocurrido durante el sprint terminado, teniendo en cuenta procesos, herramientas y el incremento realizado, de esta manera determinar posibles mejoras a tener en cuenta en el próximo sprint, consiguiendo un mejor rendimiento; Una forma que podemos usar para tener una guía al momento de determinar los procesos realizados durante el sprint, es responder las siguientes preguntas:

¿Que estuvo bien durante el sprint?

¿Qué se pudo mejorar durante este sprint?

1. Finalmente, para cerrar el sprint se realiza el ultimo evento que es el refinamiento, que es una Reevaluacion del product backlog, dependiento de lo ocurrido durante el sprint que finaliza

Principalmente scrum es utilizado para proyectos complejos, donde debido al protocolo establecido dentro del marco, pueden contribuir a obtener el proyecto de forma exitosa, sin afectar la gran medida la triple restricción (Tiempo, costo y calidad), debido a la facilidad que tiene para adaptarse al cambio.

Sus tiempos en los eventos ya concretos o timebox facilitan el control de los procesos e identificación de posibles impedimentos que tenga el equipo, durante cada sprint.

Se debe tener en cuenta que el “líder” es el scrum master, quien debe elaborar la metodología que se usara y al finalizar explicarlo al equipo, con el objetivo de crear transparencia (punto fundamental dentro del marco scrum), el trabajo empírico debe ser promovido por el scrum master.

El objetivo del uso de los sprint, es que el controlar las tareas a desarrollar y examinar que mejoras se pueden realizar y que no se puede volver a hacer. De esta manera se retroalimenta el equipo, creando una mejor adaptación a las futuras tareas a crear en los siguientes sprint. Al igual se debe tener en cuenta que las historias de usuarios dadas desde la primera reunión o contacto (dueño del producto y cliente) generalmente cambian con el paso del tiempo y de las entregas del incremento, por esta razón el uso de las iteraciones proporciona la manera de modificar estas historias de usuario al final de cada sprint junto después de un incremento “terminado” y aprobado.

Al igual que el scrum master el product owner, debe exponer la información que maneja de manera clara y asegurarse que fue entendida, principalmente información que se maneje del product backlog.

Algunos de los beneficios que podemos encontrar con el uso de scrum son:

* Flexibilidad al cambio
* Mayor productividad
* Predicciones de tiempo
* Reducción de riesgo en la triple restricción
* Mayor claridad de software

## **Definición y caracterización de los Eventos.**

**Sprint:**

Durante este evento se busca la terminación de un incremento de producto funcional que potencialmente puede ser entregado, el tiempo puede variar entre 1 2 3 y máximo 4 semanas, debemos tener claro que durante el evento no se realizaran cambios durante el mismo, para no afectar el objetivo que fue planteado inicialmente, el alcance puede ser renegociado dependiendo del desempeño del equipo, esto se puede realizar con un acuerdo entre el Development team y el product owner.

Para la creación de la supercomputadora se realizarán sprint de 2 semanas, y debido a que se estima a que el proyecto tenga una duración de 2 meses, se puede deducir que durante este periodo el proyecto tenga 4 sprint, cada uno como se debe realizar con un incremento de producto según se planee al inicio de cada sprint.

**Sprint planning:**

Dentro de esta reunión todo el scrum team está presente, con el objetivo de determinar: Que se puede entregar como incremento, y como se conseguirá hacer, durante el periodo de sprint (2 semanas en este caso), para sprint de 1 mes, se debe realizar un sprint planning de 8 horas, para el proyecto será de 4 horas por sprint.

Aquí es donde el development team proyecta que lista de usuario se pueden cumplir y finalmente se forja el sprint goal.

El evento se realizará los días lunes dentro de las instalciones justo luego de entregar el product increment de esta forma daremos inicio al nuevo sprint.

**Daily scrum:**

El objetivo principal de este evento, es responder 3 preguntas por parte del development team, quienes son los encargados de dirigir esta reunión la cual debe tener un time box (Tiempo máximo) de 15 minutos, en este caso no importa que tiempo tenga el sprint, el scrum master dentro de esta reunión solo tiene como objetivo, verificar que se lleve a cabo el evento durante el tiempo antes mencionado.

Las preguntas a contestar son:

¿Qué hice ayer para lograr el objetivo?

¿Qué voy a hacer el día de hoy para conseguir el sprint goal?

¿Qué impedimento he tenido que me retrasen durante la labor?

La reunión se hará a la misma hora y en el mismo lugar (8:00 am, dentro de las instalaciones) durante un time box o periodo de tiempo no mayor a 15 minutos en cada reunión.

**Sprint review:**

El evento debe tener un time box de 4 horas como máximo para sprint de un mes, para el proyecto de supercomputadora será de 2 horas en cada sprint por el periodo de tiempo que se asignó al inicio del proyecto.

Al igual que en el Daily scrum, es el scrum master el encargado de que el evento se lleve a cabo y que los asistentes entiendan el objetivo de dicha reunión.

Se indica que requerimientos se abordaron durante el sprint, cuales faltan por parte del product owner y el development team, se encarga de resolver dudas respecto al incremento.

Esta reunión se realizará en este caso con un desayuno de negocios los días viernes, cerca de las instalaciones, con el fin de presentar el product increment.

**Sprint Retrospective:**

Evento con un time box no superior a 3 horas para sprint de un mes, el proyecto constara de un sprint retrospective de hora y media en cada sprint,

Todos los roles que conforman el equipo deben estar presentes durante esta reunión, aunque es el scrum master el encargado de guiar la reunión y finalmente llegar a una conclusión.

## **Definición y caracterización de los Artefactos.**

**Product Backlog:**

Es la lista de requerimientos o historias de usuario, que crea el product owner con la información que obtuvo del cliente, es almacenada aquí, según como lo considere el dueño del producto dada la prioridad que el asigne, debe estar de forma clara y concisa para un fácil entendimiento del development team.

la lista por lo general sufre ciertas modificaciones principalmente en la cantidad de historia de usuarios durante el transcurso del tiempo y de los sprint.

Esta evolucionara, esto se da debido a que, durante el desarrollo y las entregas generalmente el cliente busca perfeccionarlo, cosa que desde un principio sería una tarea complicada el tener la idea totalmente clara.

La definición de terminado de cada historia de usuario, deberá ser definida y aclarada dentro del scrum team, junto con el cliente, de esta manera crear transparencia y claridad.

**Sprint Backlog:**

Es el conjunto de historias de usuario tomados del product backlog, por el development team en el sprint planning, para hacer desarrollados dentro del sprint con el objetivo de crear el incremento final “terminado”, el cual es el objetivo principal de cada sprint.

En casos especiales el development team puede analizar si alguno de estas tareas no puede cumplirse dentro del time box del sprint, y mediarlo con el product owner, para en caso dado dividir esta historia de usuario en una tarea más pequeña o en su defecto, dejar esta tarea para un próximo sprint, aunque esto es poco recomendable porque puede generar retrasos y afectar de alguna manera la triple restricción.

**Product increment:**

Es el sprint backlog o más específicamente las tareas que fueron seleccionados al inicio del sprint, que fueron terminadas, y en su conjunto forman un incremento o producto “terminado”, listo para poder ser entregado al cliente.

Este deberá ser revisado y aprobado dentro del sprint review, por parte del cliente, una de las condiciones para que esto suceda es que el product increment debe cumplir con la definición de “terminado” acordado al inicio del sprint, además aquí es donde el development team, responderán las preguntas planteadas por el mismo cliente.

Tal como se indica más adelante el responsable de tomar la decisión liberar o no el incremento es el producto owner. La suma de estos incrementos debe generar el producto final.

## **Personas y roles del proyecto.**

**Scrum master:**

Dentro del equipo scrum será el encargado de dar transparencia y claridad a los demás integrantes del equipo scrum, respecto al marco scrum como tal, como lo son sus artefactos, eventos, time box, etc. Además, motiva los cambios con el objetivo de que el equipo de desarrollo cree productos de alto valor, para esto se asegura de guiar al equipo a ser autoorganizado y funcional, teniendo en cuenta los impedimentos que el development team puede tener y eliminarlos, esa es una de las características principales de scrum master, estar en frente del equipo liderando el proyecto y que el marco scrum sea usado de forma correcta. Debe asegurarse de que el equipo trabaje ajustándose a la:

* Teoría
* Practica
* Reglas

Todo dentro de un entorno empírico.

El scrum master también estará encargado de que los eventos que se encuentren dentro del sprint se realicen y con los tiempos indicados dentro del marco.

**Product owner:**

Es el que realice el primer contacto con el cliente, y tomara nota de los requerimientos del proyecto, para crear las historias de usuarios, para luego almacenarlos dentro del product backlog, ya en el Sprint planning, él es el que decide como ordenar el product owner como él lo considere mejor, y a la vez aclararlo ante todo el scrum master para dar claridad y trasparencia.

Entre los diferentes roles, el product owner es el que debe saber más sobre el objetivo de negocio y sobre la lista de producto y asegurarse de que haya sido entendida por el development team.

Durante los diferentes sprint que tenga el proyecto, será el único rol autorizado para cancelar o no el sprint, esto lo podrá determinar con la colaboración del equipo de desarrollo.

**Development team:**

Tal como su nombre lo indica este rol está compuesto por el equipo de desarrolladores presentes en el proyecto, este deberá ser auto organizado, y tendrá el poder de proporcionar las estimaciones del backlog, de esta manera predecir la lista de pendientes e identificar que funcionalidad ira en el siguiente sprint, son los encargados de dirigir el scrum daily anteriormente mencionado, y nadie más puede intervenir durante este tiempo, y durante este evento, se debe analizar principalmente los impedimentos que estos pudieron tener durante su trabajo de desarrollo con el objetivo de cumplir el sprint goal. Estos deberán estar conformado por mínimo por 3 y máximo por 9 desarrolladores, para poder realizar sprint un poco más largos y que el equipo no resulte difícil de manejar.

Para este proyecto se contratarán 7 desarrolladores.

Otra de las características es que no se admiten subgrupos dentro del mismo, todos los integrantes como tal conforman un equipo de trabajo responsables de los incrementos en cada sprint.

# **DISEÑO**

**Hardware:**

Para la creación de la supercomputadora se usará tres equipos, uno de ellos será el master y los dos restantes serán los esclavos los cuales tendrán las siguientes características:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Computadoras** | **Ram** | **Disco Duro** | **IP** | **Sistema operativo** |
| Master | 1 GB | 10 GB | 192.158.0.150 | Ubuntu 14.04 desktop |
| Esclavo 1 | 1 GB | 10 GB | 192.158.0.151 | Ubuntu 14.04 Server |
| Esclavo 2 | 1 GB | 10 GB | 192.158.0.152 | Ubuntu 14.04 Server |

**Software:**

Luego de especificar los programas a usar dentro del proceso se especificará en que nodo se usaran cada uno de estas herramientas:

Para los nodos presentes dentro del proyecto se instalará y configurará la herramienta Mosix cuyo proceso de instalación se especificará más adelante.

# **CONFIGURACIÓN**

**Mosix:**

Dentro de la configuración de Moxis se debe tener en cuenta que primero que todo debemos realizar una configuración del servidor luego de montar el sistema operativo (Ubuntu 14.04) por medio de una máquina virtual, para esto se usó la herramienta (Virtual Box):

1. Para iniciar el proceso de instalación debemos ingresar por medio del usuario Root incorporado dentro Ubuntu, se realizará por medio de este código: **sudo password,** donde le asignaremos una contraseña a al usuario Root.
2. A treves del código **ifconfig** podemos obtener un resumen de la configuración con la que está establecida la máquina.

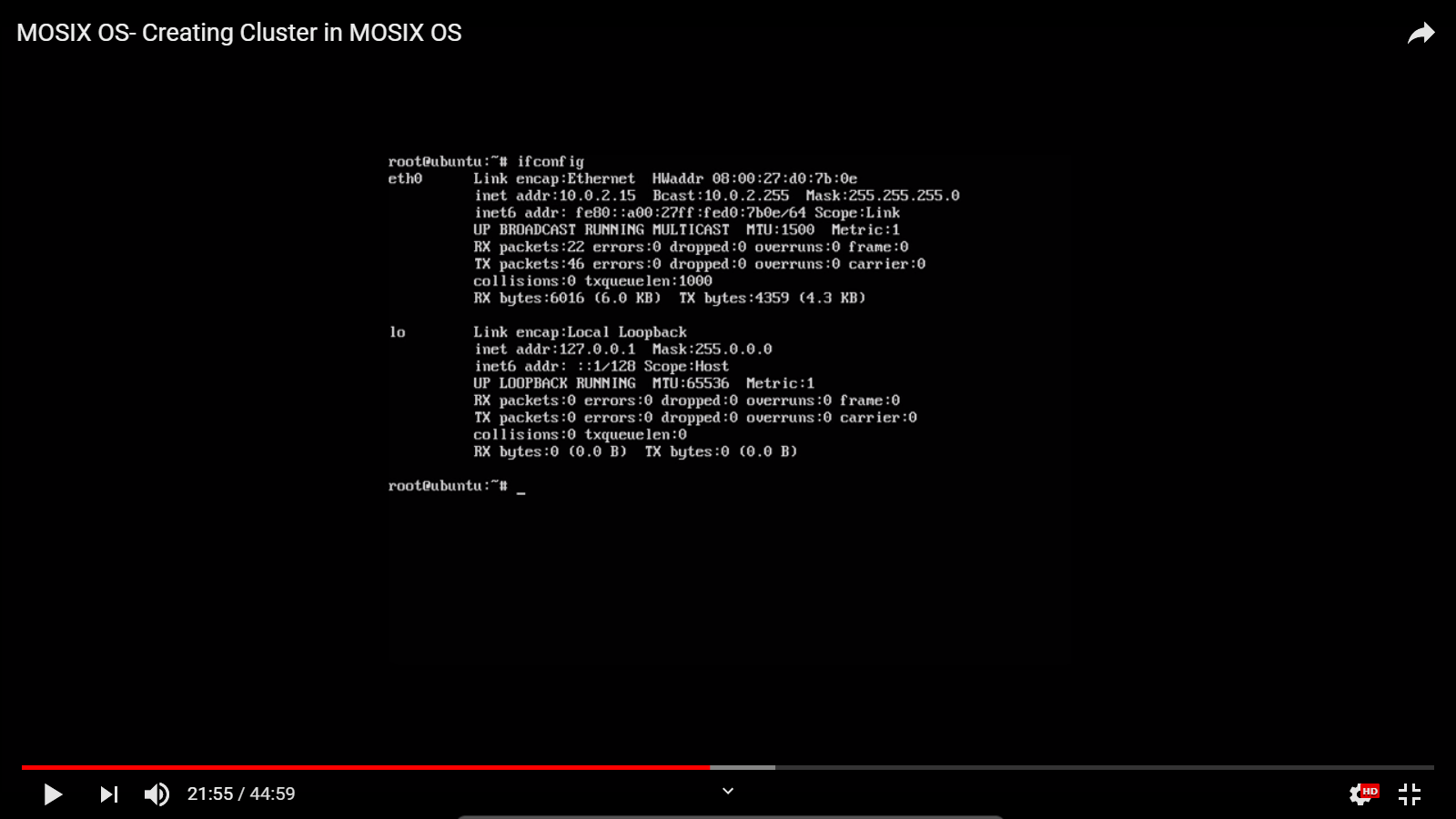


Ilustración 3 Configuracion Maquina

1. Para actualizar los repositorios se usará el código

**apt-get update.**

1. Luego se ejecutará el comando

**apt-get install build-essential**

para instalar el paquete necesario para la instalación de Mosix.

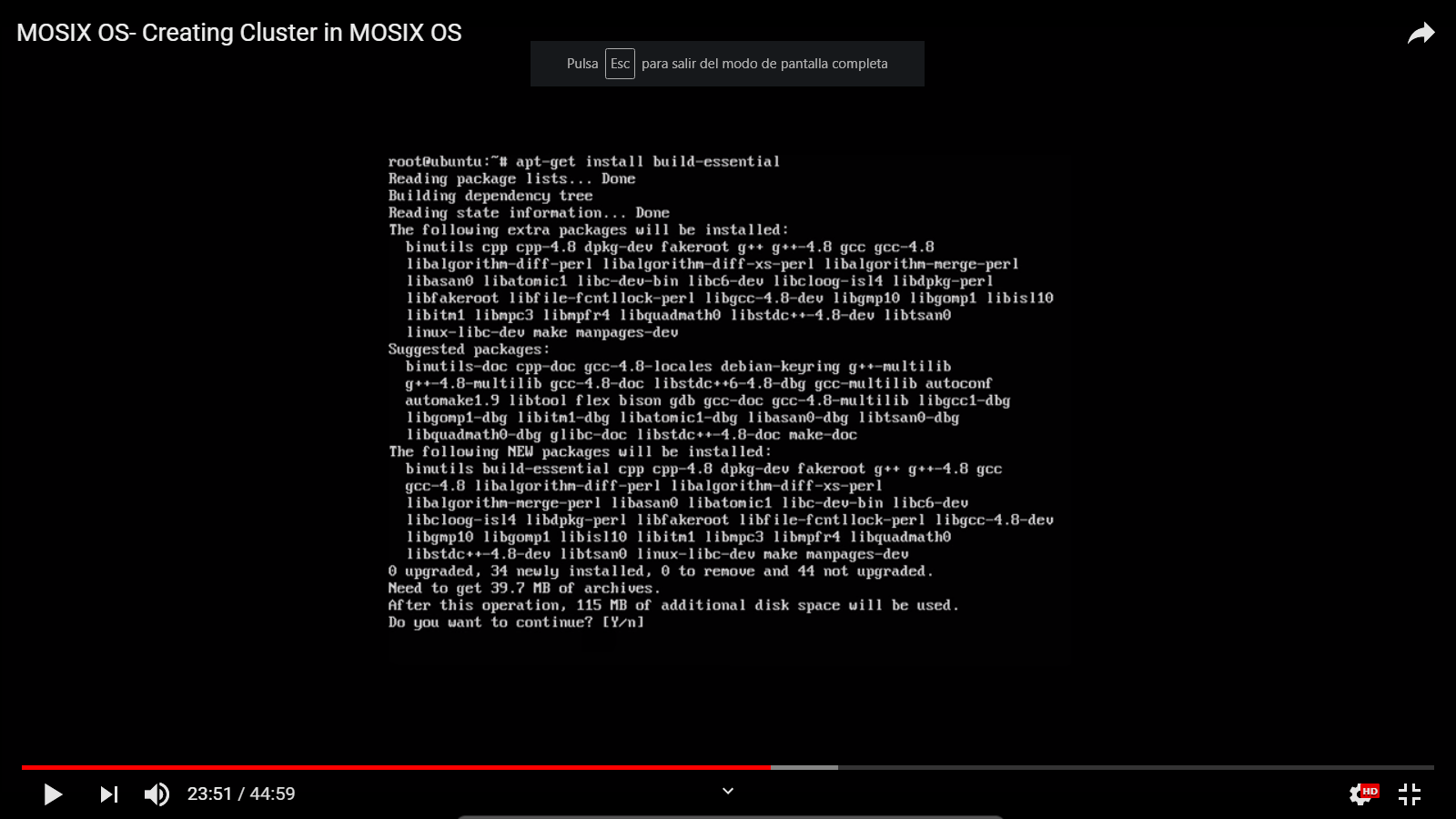


Ilustración 4 Instalación de Paquete

1. Ejecutamos

**apt-get install gcc**

1. Enseguida usaremos

**apt-get install cmake**

1. Se creará el folder por medio del comando

**mkdir /etc/mosix**

he ingresaremos al mismos con ayuda del comando

**cd /ect/mosix**



Ilustración 5 Creacion folder mosix

1. Dentro del folder descargaremos el paquete de Mosix ingresando el siguiente comando:

**www.mosix.cs.huji.ac.il/mos4/mosix-4.4.4.tbz**

1. Ejecutamos

**tar -xjvf MOSIX-4.4.4.tbz**

1. Para establecer la configuración realizada es necesario reiniciar el equipo en este punto del proceso.
2. Realizaremos un cambio de red en nuestra máquina, por defecto esta viene con la configuración:

Conectado a: NAT

La cual cambiaremos por **red interna** tal como se muestra en la siguiente imagen:

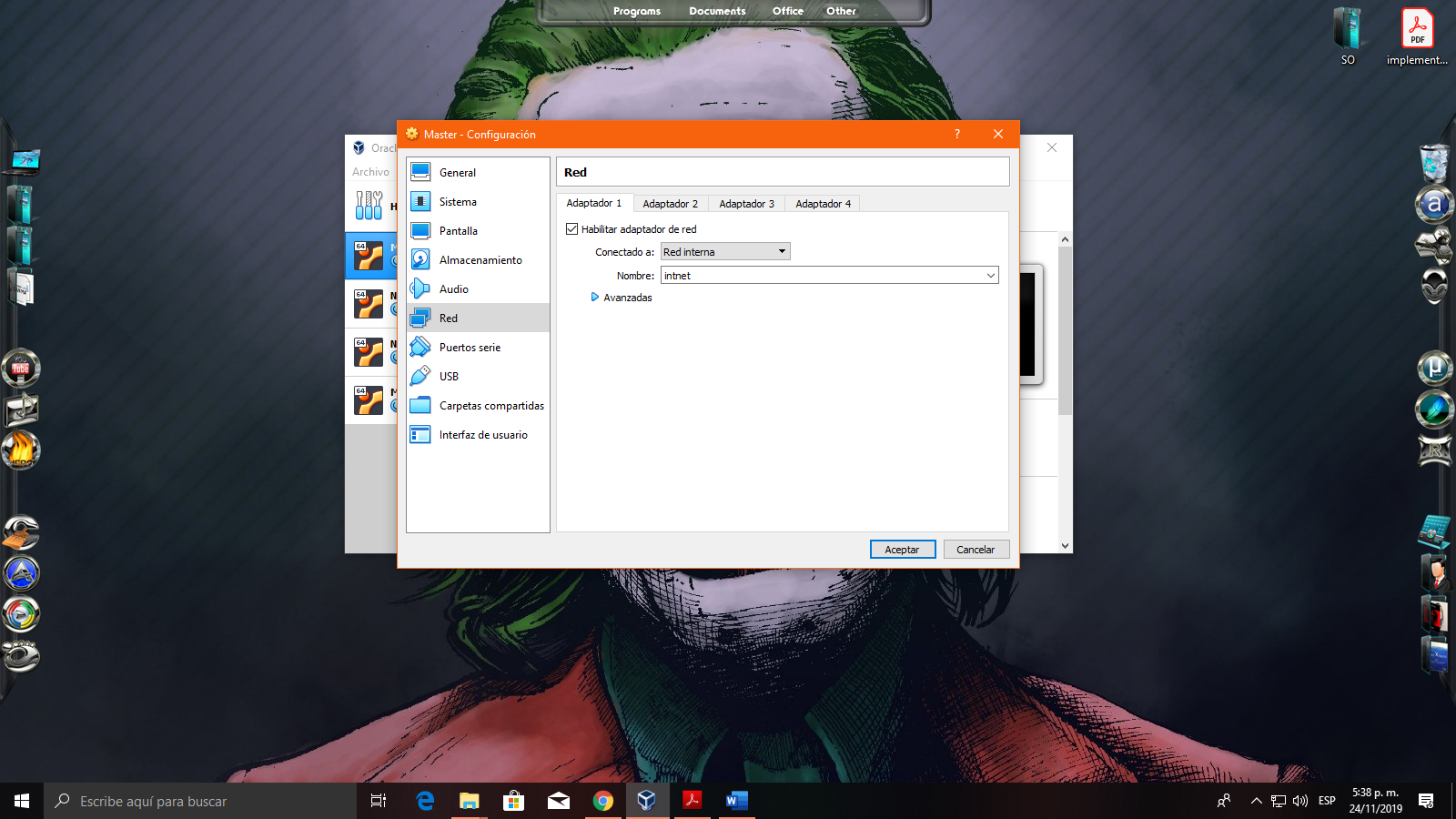


Ilustración 6 Cambio de Red

1. Reiniciamos el equipo y esperamos a que se configure la red seleccionada en el paso anterior.



Ilustración 7 Configuracion de Red

Y luego ingresamos por medio del usuario root

1. Realizamos la configuración de la red en donde principalmente cambiaremos la configuración de la IP cambiándola de dinámica a estática, ingresamos a esta configuración con ayuda del comando

**nano /etc/network/interfacess**

aparecerá lo siguiente:



Ilustración 8 Cambio tipo IP

Dentro de la ventana cambiaremos **dhcp** por la palabra **static** de esta manera realizaremos el cambio anteriormente mencionado.

Dentro de la misma ventana agregaremos la IP que asignaremos, en este caso al nodo Master la cual será **192.168.0.150,** junto con la configuración de mascará que en este caso será de **250.250.250.0,** tal como se puede observar en la siguiente captura de pantalla:



Ilustración 9 Asignacion de IP (Master)

1. Ejecutamos

**/etc/init.d/networking restart**

Y reiniciamos la maquina nuevamente.

1. Ingresamos con el usuario root, y nos dirigimos al folder donde se encuentra nuestro paquete de Mosix por medio del siguiente comando:

**cd /etc/mosix/mosix-4.4.4/**

y verificamos por medio del comando **ls,** que el paquete se encuentra dentro le folder listo para su instalación:

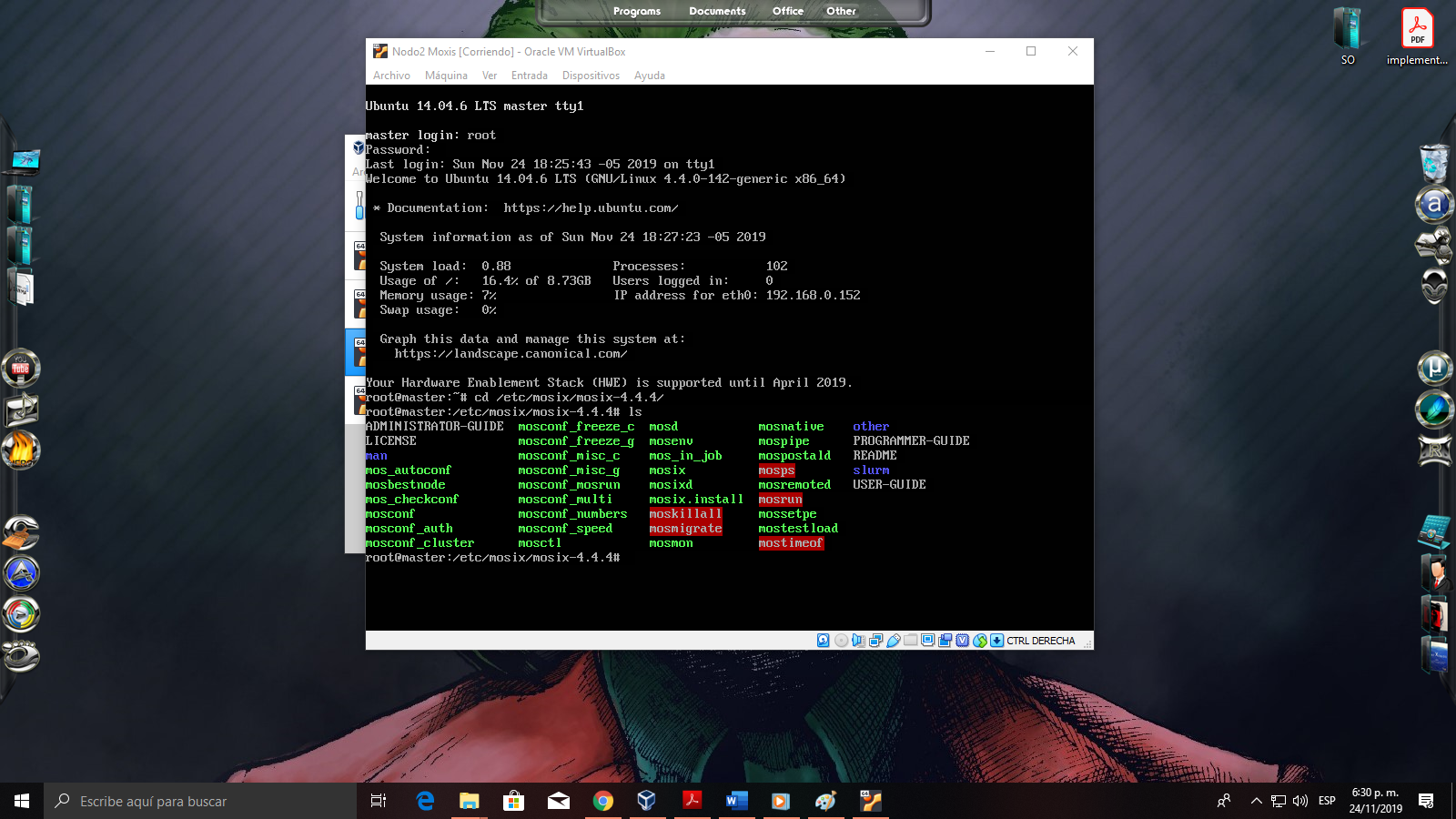


Ilustración 10 Paquete Mosix

1. Finalmente realizaremos la instalación de Mosix por medio del código:

**./mosix.install**

y posteriormente damos click en **ENTER**

1. En el menú que aparece a Econtinuación de la instalación podemos especificar los nodos que usaremos:

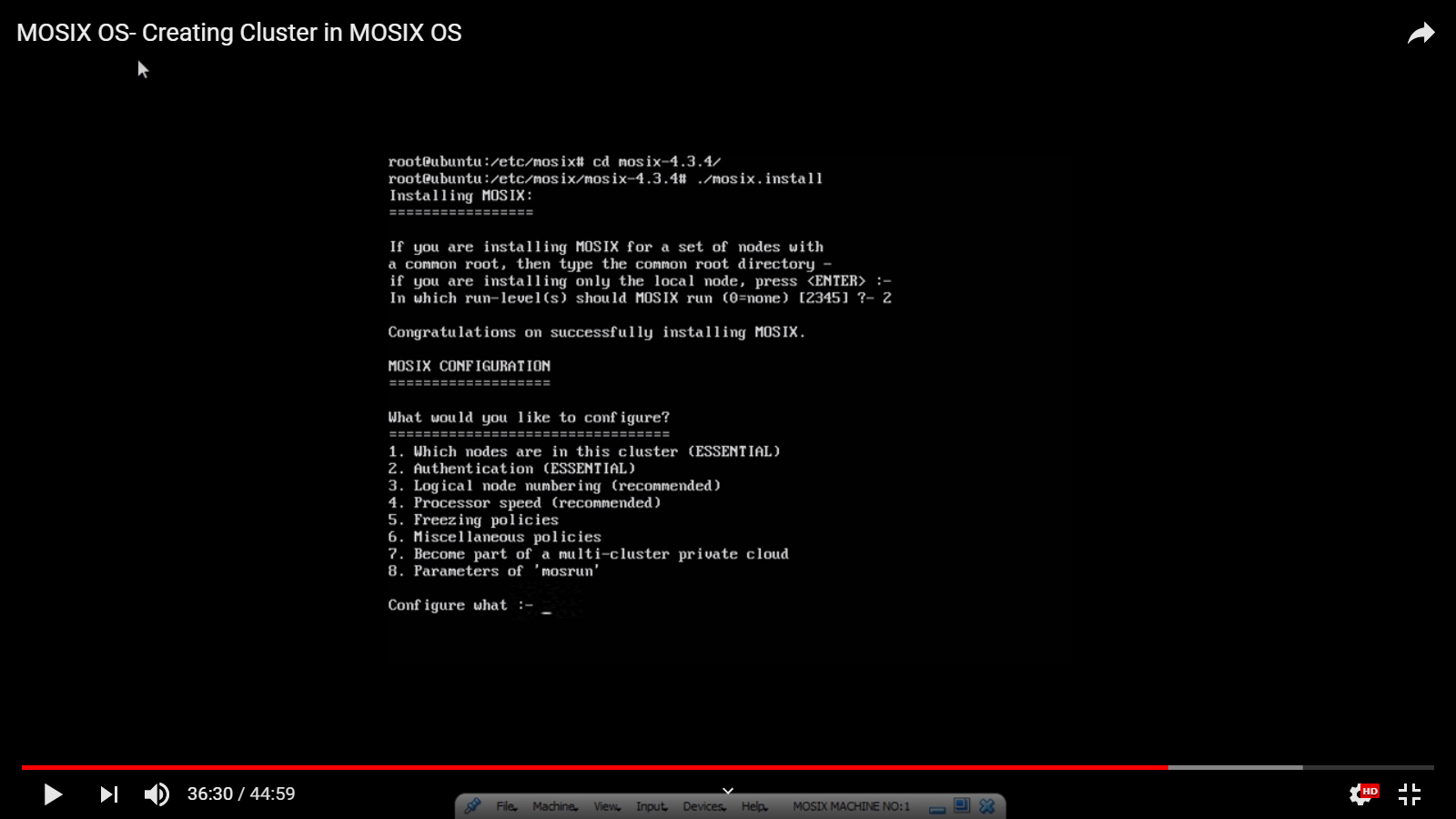


Ilustración 11 Menu Mosix

Para eso ingresamos el número 1, y posteriormente identificamos los 3 nodos (dentro de estos debemos incluir el master) que usaremos para la creación de la supercomputadora, junto con la IP con la cual se identificara nuestro nodo Maestro:

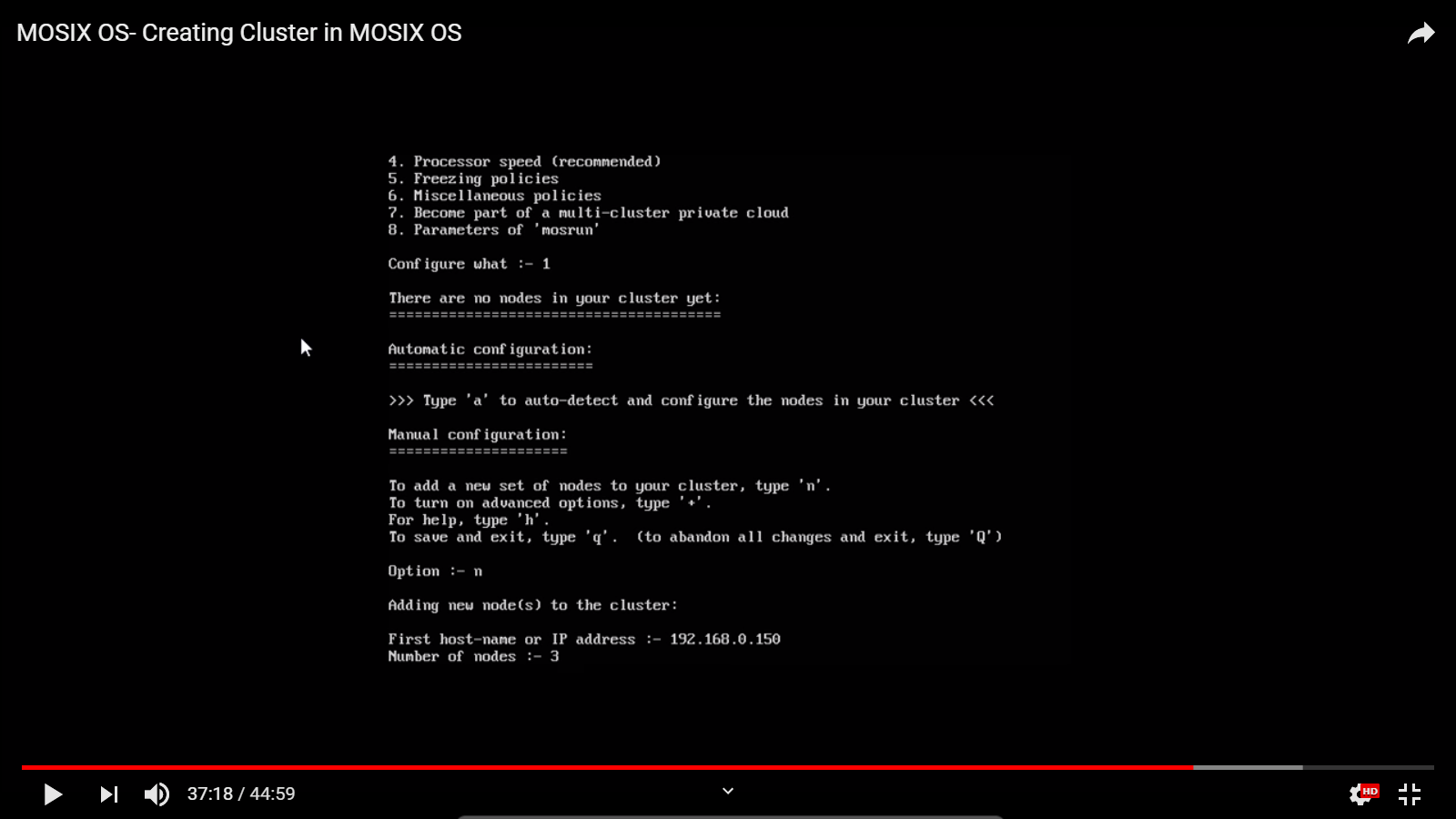


Ilustración 12 Asignación de Nodos

1. Finalmente realizaremos un ultimo reinicio de equipo para dar por termino la configuración de moxis.

Cabe destacar que el proceso especificado anteriormente se debe realizar en cada uno de los nodos ya sea el maestro o los esclavos que estén dentro de la arquitectura, la única diferencia que se realizara entre nodo y nodo es la IP; teniendo en cuenta que el nodo maestro tiene como IP:192.168.0.150, los demás nodos deben seguir la secuencia, por ejemplo

* Nodo 2: 192.168.0.151
* Nodo 3: 192.168.0.152

De esta manera se creará la conexión entre computadoras y así el clúster necesario para la creación de la supercomputadora.