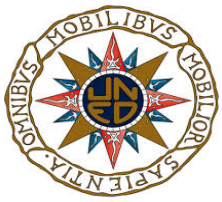


Ciemat
Centro de Investigaciones Energéticas, Medioambientales y Tecnológicas

Proyecto Fin de Máster en Ingeniería de Sistemas y Control.

SIMULACIÓN E IMPLEMENTACIÓN DE UN ENTORNO COLABORATIVO DE COMPUTACIÓN PARA APLICACIONES Y MÉTODOS DE APRENDIZAJE AUTOMÁTICOS EN FUSIÓN

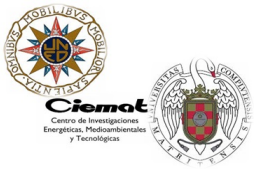
SANTIAGO GONZÁLEZ GONZÁLEZ

INGENIERO TÉCNICO EN INFORMÁTICA GESTIÓN

DEPARTAMENTO DE INFORMÁTICA Y AUTOMÁTICA DE LA ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

U.N.E.D. - U.C.M.


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas

SEPTIEMBRE 2014

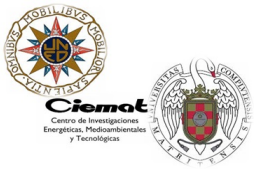
DEPARTAMENTO DE INFORMÁTICA Y
AUTOMÁTICA DE LA ESCUELA TÉCNICA
SUPERIOR DE INGENIERÍA INFORMÁTICA

SANTIAGO GONZÁLEZ GONZÁLEZ
INGENIERO TÉCNICO EN INFORMÁTICA GESTIÓN

Tutores/Directores:

Dr. D. Jesús Antonio Vega Sánchez (C.I.E.M.A.T.)
Dr. Rodrigo Castro Rojo (C.I.E.M.A.T.)
Dr. D. Sebastián Dormido Canto (U.N.E.D.)


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas

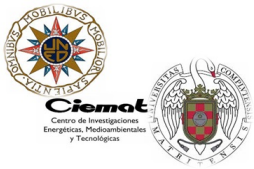

Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas

Autorización:

Autorizo a la Universidad Complutense y a la U.N.E.D. a difundir y utilizar con fines académicos, no comerciales y mencionando expresamente al autor de este trabajo, tanto la memoria de este Trabajo Fin de Máster, como el código, la documentación y/o el prototipo desarrollado.

Firmado: Santiago González González


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas

AGRADECIMIENTOS:

Esta andadura en el campo académico comenzó hace muchos años, tras una pausa que por necesidades familiares tuve que adoptar en mi época adolescente. Comenzó con una propuesta inocente por parte de mi pareja, Aurora, la cual vio en mí las capacidades para afrontar un reto de formación invitándome a obtener el título de ingeniero informático, cosa que en un principio me arrancó una carcajada de incredulidad.

Tras muchos años estudiando, con grandísimos esfuerzos y satisfacciones y como no puede ser de otra manera, muchas decepciones, es el momento de realizar estos agradecimientos a toda esta gente de la que he aprendido muchísimo y me han aguantado otro tanto.

Quiero dar las gracias a mi pareja **Aurora** por ese pistoletazo inicial que cambió mi vida, por ayudarme a creer más en mí, por haberme aguantado tantas horas de ausencia, por mi dedicación a los libros sin menospreciar los cambios de humor producidos por esas decepciones estudiantiles. Obteniendo de ella siempre ayuda, apoyo y comprensión, sin la que de ninguna de las maneras estaría escribiendo esto.

Quiero agradecer a la **U.N.E.D.** por haberme dado la oportunidad de la realización de mis estudios como ingeniero y máster en ingeniería dentro de ella. Agradeciendo su filosofía como institución para poder ofertarnos a la gente que como yo, por suerte, tenemos trabajo, la posibilidad de trabajar y estudiar, algo bastante complejo de compatibilizar.

Quiero agradecer al **Dr. Sebastián Dormido Canto** la confianza depositada en mí, y su ayuda permanente desde que nos conocimos en el Centro asociado de la U.N.E.D en las Rozas y me impartía sus clases de Tecnología de computadores en Ingeniería de Sistemas, hasta hoy como co-director del Máster.

Quiero, de igual manera agradecer al **Dr. Jesús A. Vega Sánchez y al Dr. Rodrigo Castro Rojo,** sus lecciones como directores del proyecto y toda la ayuda prestada a lo largo de la gestación del mismo, por todo lo que he aprendido de ellos, la dedicación y empeño transmitido gracias !!!.

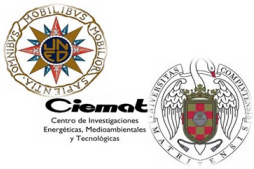
Quiero agradecer a **Pilar**, su ayuda incondicional para todo lo que he necesitado, dentro y fuera de la Universidad.

Quiero agradecer su aporte a toda la gente que ha estado a mi lado durante toda esta etapa luchando codo con codo, aportándome ideas, cariño, compañerismo, profesionalidad y amistad, que sin ellos se hubiera hecho mucho más cuesta arriba este desarrollo.

A su vez, quiero agradecer a todas las personas que se han visto salpicadas por esta andanza su compresión en mis momentos de máxima flaqueza, disculpándome si en algún momento pude llegar a incomodar.

Quiero agradecer el haber conocido a “gente del otro lado del charco” tan maravillosa con la que he vivido ese estrés por la preparación de tesis doctoral, tomando algún café que otro en la cafetería de la universidad. Claudia muchas gracias.

Quiero agradecer a mi madre allí donde esté, todo, ya que sin ella nada hubiera sido posible, te quiero madre !!!


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas


Ciemat
Centro de Investigaciones
Energéticas, Medioambientales
y Tecnológicas

Índice de Figuras.

1. INTRODUCCIÓN

Según transcurren los siglos desde que el ser humano puebla el planeta tierra, sus requerimientos energéticos aumentan progresivamente y directamente proporcionales a las necesidades, entre otras causas, a los avances tecnológicos que se producen. Estas necesidades se inician con el descubrimiento del fuego por parte de algún antiguo ejemplar de Homo Erectus, hace 1,4 millones de años, y se prolongan en el tiempo hasta nuestros días, en los que las investigaciones existentes para el descubrimiento y explotación de nuevas fuentes energéticas son fruto directo de la demanda impuesta para cubrir las necesidades de producción industrial, transporte, iluminación, servicios, climatización etc.

En países desarrollados se están realizando planes y criterios de eficiencia energética y ahorro de la misma por lo que se prevé que el crecimiento de la demanda eléctrica en el próximo siglo sea bastante menor que en el pasado.

Sin embargo, el consumo per cápita de los países en vías de desarrollo es diez veces inferior que el de los países industrialmente avanzados. A medida que estos países mejoren su nivel de vida su demanda energética también aumentará con rapidez. Según qué escenarios se consideren para la evolución de la demanda energética, el consumo podría triplicarse a mediados del siglo XXI.

Estos datos vienen avalados por la previsión emitida por la Organización de las Naciones Unidas que vaticina una población mundial para mediados del presente siglo en torno a 9500 millones de habitantes.

Actualmente los combustibles fósiles (petróleo, gas, carbón) son las principales fuentes de energía por su reducido coste. Estos recursos han sido masivamente explotados y las reservas existentes, principalmente de petróleo y gas, no garantizan su disponibilidad a largo plazo. Por otro lado, los residuos lanzados a la atmósfera durante su combustión tienen un impacto negativo en el medio ambiente, en especial los gases de efecto invernadero; Dióxido de Carbono (CO_2), Metano (CH_4), Óxido de Nitrógeno (N_2O) y Gases Fluorados. Esta concienciación lleva al planteamiento de búsqueda de otras fuentes energéticas. En el “Año Internacional de la Energía Sostenible para Todos“, año

2012, la ONU anunciaba que el 12,9 % de la demanda energética mundial era procedente de energías renovables, presentado en mayo pasado, establece que para cubrir el 85,1% de la demanda mundial de energía se emplean fuentes fósiles como el petróleo y sus derivados (34,6%), el carbón (28,4%) y el gas natural (22,1%).

En este informe se determina que este tipo de energía es responsable del 60% de las emisiones de gases de efecto invernadero que afectan a la capa de ozono y aceleran el calentamiento global.

En cuanto a las soluciones viables a día de hoy, se destacan dos: las energías renovables y la energía nuclear. Las energías renovables son en general bien aceptadas por la sociedad al no plantear aparentemente riesgos para los trabajadores y ser consideradas amigables con el medioambiente. A pesar de ello, la base para la construcción de paneles solares, por ejemplo, para la producción de energía fotovoltaica es el silicio (basado en el procesado del sílice), cuya extracción es altamente costosa.

Los molinos para la producción de energía eólica son poco rentables y su instalación, poco agradable a la vista. El enfoque nuclear, por otro lado, se presenta como una solución poderosa ya que es capaz de producir grandes cantidades de energía.

Esta puede ser generada mediante procesos de fisión *[1]* nuclear, en las que se dividen núcleos de alto número másico, o de fusión *[2]* nuclear, donde se unen núcleos atómicos ligeros. La energía producida mediante fisión, descubierta en 1939 por los investigadores alemanes Hahn y Strassmann, se genera como una reacción en cadena.

Este tipo de reacciones, hoy en día, pueden ser controladas con seguridad. Sin embargo, la extracción, enriquecimiento y utilización de materiales de alto peso atómico (como el uranio) para los procesos de fisión nuclear generan residuos radioactivos que tardan miles de años en perder su nocividad para el medioambiente. Debido a estas circunstancias, existen ciertos reparos en parte de la opinión pública para la producción de energía mediante la fisión.

La fusión termonuclear *[3]* en cambio, y en particular la obtenida mediante confinamiento magnético, requiere un preciso y continuo control del combustible para la operación del dispositivo y la pérdida de este control no conlleva peligros medioambientales.

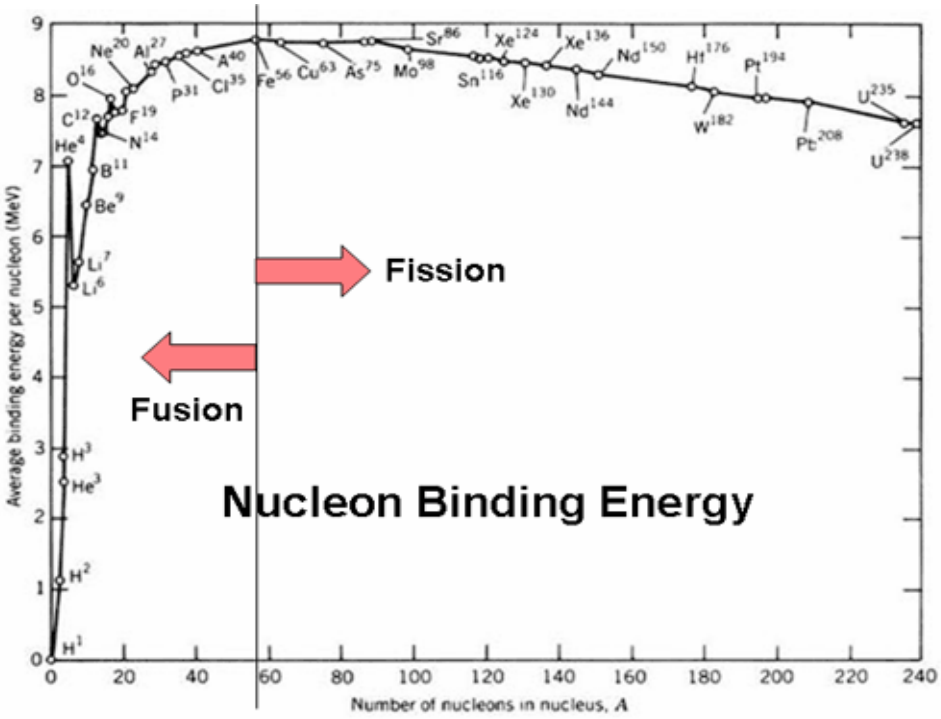


Figura. 1: *Muestra de energía necesaria tanto para los procesos de fusión como de fisión nuclear*.

La energía obtenida mediante la unión de dos átomos ligeros es del orden de 4 veces mayor a la generada mediante la fisión. Allí se representa la energía de enlace por nucleón frente al número másico “A”. Esta se obtiene dividiendo la energía de enlace del núcleo por sus A nucleones. Puede observarse en los núcleos livianos un aumento abrupto de la energía de enlace por nucleón frente al número másico.

Una ventaja añadida de la fusión es que no necesita una ardua tarea de extracción minera, ya que el Deuterio *[4]* y el Tritio *[5]* son abundantes en la naturaleza. Además, su obtención es significativamente más sencilla que la de elementos de alto número másico.

1.1 La física del plasma.

Como se ha mencionado con anterioridad, desde los principios de la humanidad el hombre admiró la "inagotable" producción de energía generada por el sol, venerándolo en algunas civilizaciones como a un Dios. El primer paso científico hacia el desarrollo de una teoría que explicara esta poderosa fuente de radiación estelar se dio en 1905 cuando Albert Einstein introdujo su famosa ecuación sobre la equivalencia entre masa y energía:

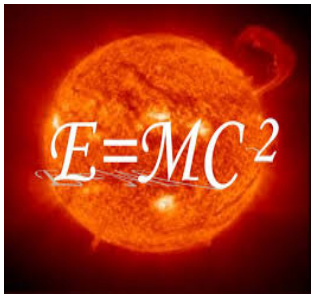


Figura. 2: *Fórmula en la que se muestra la equivalencia entre la masa y la energía dada por la expresión de la teoría de la relatividad de Albert Einstein.*

Estableciendo que pequeñas variaciones en la masa pueden transformarse en una gran cantidad de energía. Francis Aston

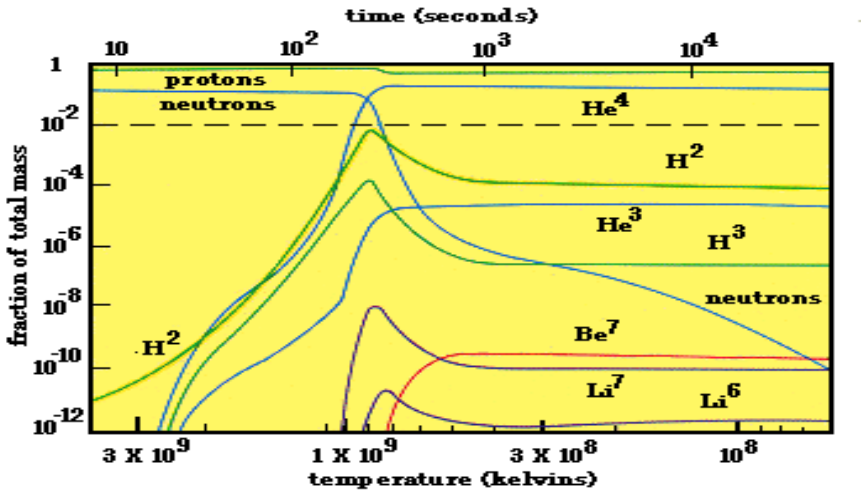


Figura. 3: *Temperaturas obtenidas en los procesos de fusión nuclear.*

15 años después, logró otro avance fundamental al poder demostrarlo con resultados experimentales mediante la invención del espectrómetro de masas. *[6].*

Finalmente, en 1939 Hans Bethe (premio nobel de física en 1967) publicó un artículo revolucionario explicando la generación de energía por fusión en las estrellas. El principio de la fusión es simple ya que involucra dos de los más abundantes elementos en el universo, el hidrógeno y el helio. La gran fuerza gravitacional que aparece en el interior de las estrellas (como en nuestro sol) comprime enormemente el hidrógeno presente y logra que se produzca la fusión de los núcleos. La energía generada es el resultado de la conversión de dos núcleos atómicos que se unen para formar uno de mayor masa atómica. Aplicando la fórmula de Einstein, esta diferencia de masa es transformada en energía.

Sin embargo, las presiones y las temperaturas en el núcleo de las estrellas distan de las condiciones normales en la Tierra. En la década de 1950 se formalizaron los primeros intentos para la producción de energía mediante fusión.

Dentro de las posibles reacciones de fusión, las más atractivas son las denominadas como ciclo D-T (deuterio-tritio) por tener la mayor sección eficaz a menores temperaturas.

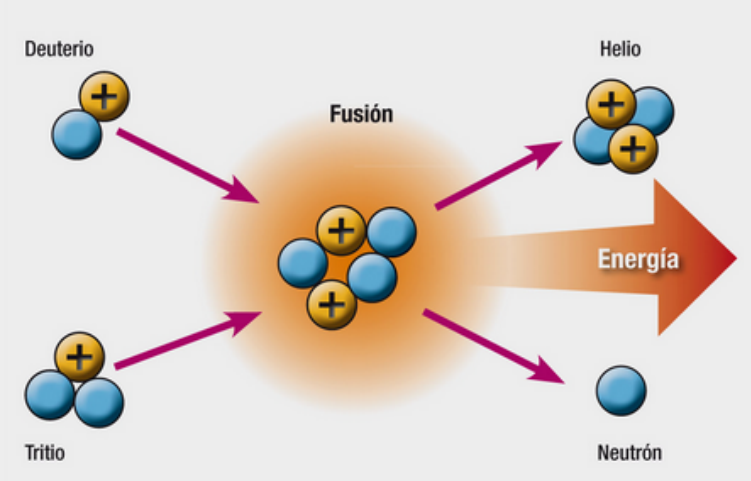


Figura. 4: *Figura proceso representativo ciclo de fusión nuclear denominados Deuterio-Tritio(D-T).*

El Deuterio es un elemento estable y abundante (30 gramos de cada metro cúbico de agua, es deuterio). El tritio, sin embargo, no se encuentra en la naturaleza y debe producirse artificialmente.

Puede generarse mediante la activación del hidrógeno contenido en el agua o por bombardeo de litio (elemento abundante en la corteza terrestre) con neutrones.

Los átomos de hidrógeno se encuentran totalmente ionizados a las temperaturas requeridas para que se produzcan las reacciones de fusión. A este estado de la materia se le denomina “plasma” *[7]* El plasma contiene un número significativo de partículas cargadas (iones y electrones) cuya dinámica presenta efectos colectivos.

Para que un reactor de fusión sea energéticamente rentable, las reacciones deberán generar una energía significativamente superior a la aplicada para la creación y el mantenimiento del plasma. Según el criterio de Lawson *[8]* la energía de las reacciones (en particular de las partículas α) resultan suficientes para calentar el plasma sin necesidad de un aporte energético externo. Lawson definió inicialmente un umbral mínimo, mediante el triple producto de la densidad electrónica (t_E), la temperatura electrónica (n_e) y el tiempo de confinamiento de la energía del plasma.

Este triple producto, considerando un rendimiento del 33%, debe ser igual o superior a 10^{21} keVs / m^3 para que el plasma alcance la ignición *[9].*

La investigación actual para el desarrollo de un futuro reactor de fusión se centra principalmente en dos vías, la fusión por confinamiento magnético y la fusión por confinamiento inercial. En los dispositivos de confinamiento magnético se utilizan campos para crear barreras que aíslen el plasma de la cámara de vacío en la cual está contenido.

El plasma, con una densidad de hidrógeno superior a 10^{20} m^{-3}, se ha de calentar a temperaturas de decenas de KeV (siendo 1eV ~ 11000 K) de tal manera que el producto de la densidad por el tiempo de confinamiento sea 10^{23} m^{-3} s. La estimación de densidad para el confinamiento inercial debe ser del orden de 10^{23}l, se genera mediante el calentamiento y compresión de esferas de hidrógeno (las densidades alcanzadas son del orden de y las temperaturas de decenas de KeV) durante períodos muy cortos (nanosegundos). Para la compresión y el calentamiento de las esferas se requieren láseres de alta

potencia, cuyo desarrollo y posible aplicación en estado continuo plantean problemas de difícil solución *[10].*

1.2 Fusión por confinamiento magnético.

1.2.1 Confinamiento en dispositivos toroidales.

Las primeras aproximaciones para conseguir un dispositivo de confinamiento magnético se basaron en solenoides que confinaban partículas radialmente permitiéndoles fluir axialmente. Para reducir las pérdidas, la solución propuesta consistía en intensificar el campo magnético en los extremos (concepto conocido como *espejo magnético)*.A pesar de dicho esfuerzo, el confinamiento resultante en este tipo de dispositivos ha demostrado ser demasiado pobre por lo que han desaparecido del panorama de fusión.

Una solución alternativa para la reducción de pérdidas de partículas en los extremos es cerrar las líneas de campo magnético sobre sí mismas mediante la creación de campos magnéticos toroidales. En un sistema de campo magnético puramente toroidal, la curvatura y el gradiente del módulo de campo magnético provoca una deriva vertical en dirección opuesta para electrones e iones. Ésta crea un campo eléctrico que induce a su vez una deriva hacia el exterior en todo el plasma, por lo que una configuración puramente toroidal es intrínsecamente inestable. Para evitar esta separación de cargas es necesario retorcer las líneas de campo mediante una componente magnética adicional (poloidal). Una sola línea de campo describe así una superficie de flujo. El transporte perpendicular al campo magnético B está restringido por la fuerza de Lorentz, y por tanto, en una misma superficie de flujo (dirección paralela a las líneas de campo) los parámetros del plasma se pueden considerar prácticamente constantes, mientras que las principales variaciones se producen en la dirección perpendicular a las líneas de campo magnético.

Se destacan dos configuraciones en las que se pueden obtener reacciones de fusión por confinamiento magnético mediante estructuras de campo magnético con componentes toroidales y poloidales: las “Stellarator” y las Tokamaks.

El concepto de “Stellarator” fue ideado por el astrofísico Spitzer en 1951 *[11].*

La estructura de campo magnético en esta configuración se genera mediante conductores externos bobinados al toro. La corriente que circula por las bobinas puede ser controlada desde el exterior y permite un modo de operación continuo.

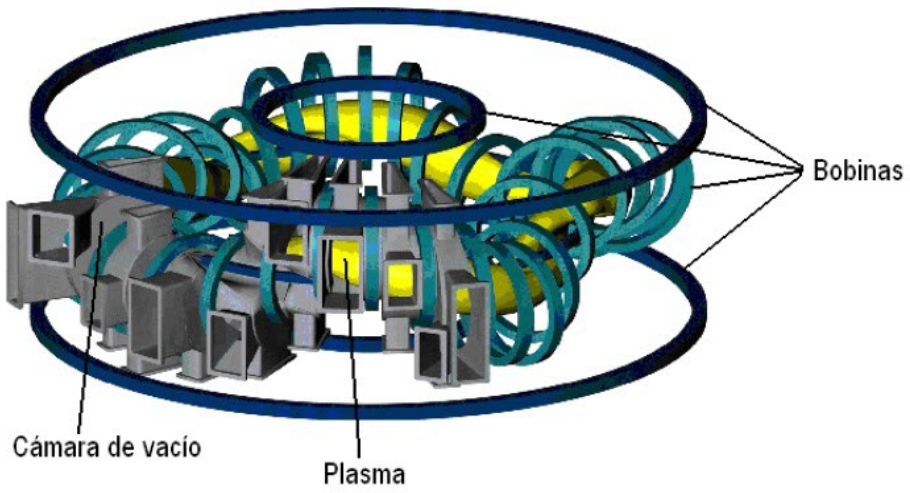


Figura. 5: *Vista esquemática "Stellarator" (TJ II) reactor experimental para la fusión por confinamiento magnético. Se representan el plasma (en amarillo), las bobinas (azul y celeste) y la cuarta parte de la cámara de vacio (gris)*

La intrincada geometría de las bobinas retuerce las líneas de campo magnético alrededor del toro. Sin embargo, la precisión necesaria para el ensamblaje del dispositivo (en especial de las bobinas) es un verdadero reto para la construcción de este tipo de máquinas.

La configuración Tokamak fue propuesta por dos científicos rusos, Tamm y Sakharov. El nombre se deriva de las palabras rusas cuyo significado puede traducirse como "cámara toroidal con campo magnético". La estructura de campo magnético se genera mediante bobinas toroidales y la componente poloidal la produce una corriente que se hace circular por el plasma. Esta corriente se induce mediante un transformador central. Hoy en día la configuración Tokamak es la más difundida y la que se utilizará para el futuro dispositivo experimental ITER ("International Thermonuclear Experimental Reactor"). Desde el punto de vista de ingeniería, presenta menos complicaciones para su construcción que un "Stellarator". Sin embargo, una de

sus principales desventajas se debe a pérdidas abruptas e inevitables de la energía del plasma (que ocurren sólo en este tipo de configuraciones), llamadas disrupciones.

1.2.2 Diagnóstico.

El plasma se encuentra a temperaturas elevadísimas, siendo ésta 10.000 veces superior a la temperatura alcanzada en la superficie solar, confinándolo en una cámara de vacío, cuyo acceso es complicadísimo. Para que el plasma pueda ser observado y seguida su evolución se requieren sistemas especiales de medida, llamados diagnósticos en la nomenclatura habitual de fusión. Para Los dispositivos desarrollados hasta el momento tienen como principales objetivos el estudio de los fenómenos físicos que ocurren en el plasma así como acercarse a las condiciones de ignición.

Cada dispositivo para el estudio de la fusión dispone de un conjunto de diagnósticos que posibiliten determinar las características principales de los plasmas producidos.

1.3 Puesta en escena, "STELLARATOR" TJ-II.

En Madrid se encuentra en funcionamiento el "Stellarator" el más grande de toda Europa el TJ-II. Es de tipo heliac flexible de tamaño medio. Sus objetivos principales son el estudio del confinamiento y del transporte tanto de las partículas como de la propia energía.

El propio diseño de las bobinas de este modelo, permite un amplio diseño de posibilidades en las confinaciones magnéticas, así como y un amplio rango de valores de la transformada rotacional. Esta puede oscilar, en el centro del TJ-II, entre 0.9 y 2.

Esta característica lo distingue de otros Stellarators al ser capaz de generar una gran variedad de geometrías magnéticas del plasma. El eje magnético que se proporciona, al ser helicoidal, le otorga una extremada tridimensionalidad. El TJ-II tiene una periodicidad M=4 con un campo magnético central de ~1 T y un radio mayor medio de 1.5 m.

Los estudios iniciales sobre la configuración magnética del TJ-II se realizaron conjuntamente entre el laboratorio ORNL (Oak Ridge, EE.UU.) y el CIEMAT.

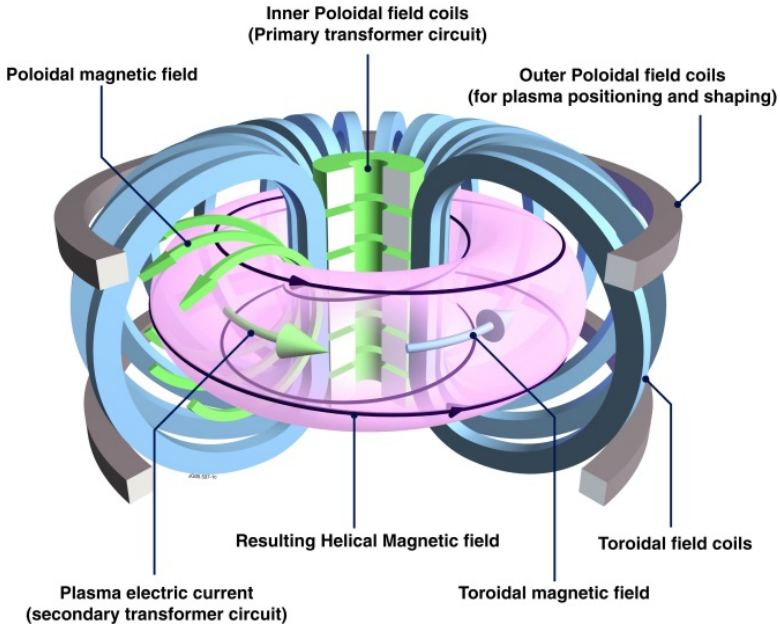


Figura. 6: *Imagen esquemática del campo helicoidal del campo magnético con la bobina.*

La máquina, incluyendo la cámara de vacío, las bobinas y la estructura que las soporta, tiene un diámetro de cinco metros, una altura de dos metros sobre la base de la plataforma experimental y un peso de sesenta toneladas. Las primeras descargas con plasma se consiguieron a finales de 1997 *[12].*

Calentamiento.

El sistema más utilizado para el calentamiento de plasmas en este tipo de reactores se basa en la Emisión Ciclotrónica Electrónica Resonante, ECRH ("Electron Cyclotron Resonance Heating").

Esta emplea ondas electromagnéticas sintonizadas al segundo armónico del giro ciclotrónico de los electrones del plasma alrededor de las líneas de campo. Las ondas son inyectadas perpendicularmente a las superficies magnéticas de la configuración. La frecuencia a la que son absorbidas las ondas depende de la intensidad del campo magnético, existiendo un valor máximo de la densidad electrónica para la cual no se propaga ECRH (fenómeno llamado de "reflexión total"). Cuando se alcanza este nivel de densidad las ondas son reflejadas y no calientan el plasma.

El TJ-II tiene instalados dos girotrones (generadores de microondas de alta potencia), sintonizados al segundo armónico de la resonancia ciclotrónica electrónica. Cada girotrón es capaz de inyectar en el plasma 300 kW a la frecuencia de 53.2 GHz durante un pulso de duración máxima de 1 segundo.

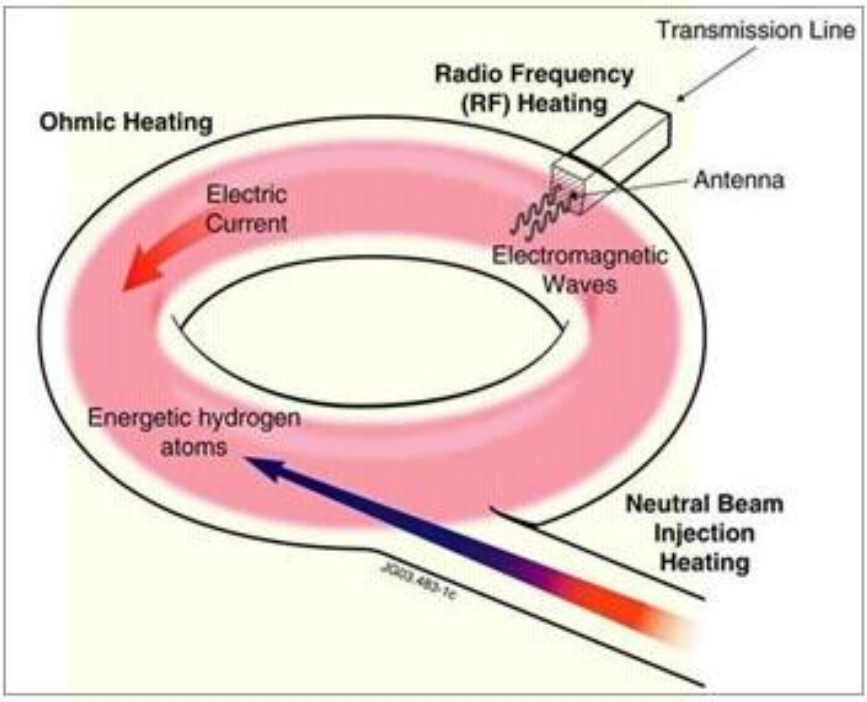


Figura. 7: *Imagen esquemática de la inyección de átomos neutros en el plasma ya precalentado.*

El segundo sistema de calentamiento está compuesto de dos inyectores de haces neutros (NBI, del inglés “Neutral Beam Injection”). Este sistema consiste en la inyección de átomos neutros en el plasma pre-calentado.

El haz de átomos posee una alta energía cinética que al colisionar con las partículas del plasma (entre ellas electrones) se cargan eléctricamente y consecuentemente quedan atrapadas por el campo magnético del dispositivo. Estos nuevos iones poseen una velocidad mucho mayor que la media de las partículas del plasma y consecuentemente provocan una serie de colisiones: ión-ión, ión-electrón y electrón-electrón. Consecuentemente, la velocidad media de los átomos confinados se incrementa.

1.3.1 El confinamiento.

El potente campo magnético del TJ-II se crea a través de un compendio de bobinas externas. El número de vueltas que posee cada bobina, son variables siendo a su vez, la corriente por vuelta dependiente del tipo de bobina y su configuración magnética. Todas las bobinas se refrigeran con agua que circula a través de orificios longitudinales en los conductores de cobre estando agarradas por una estructura mecánica para evitar deformaciones.

La cámara de vacío posee con varios sistemas de observación, sistemas de calentamiento e inyección de gas.

JET ("*Joint European Torus*") es el dispositivo de fusión por confinamiento magnético más grande del mundo. **JET**, sigla en inglés de *Joint European Torus*, es un dispositivo de fusión del tipo tokamak.

Se encuentra situado en una vieja base de la RAF cerca de Culham, en las afueras de Oxford, en el Reino Unido. A la espera de la construcción de ITER, JET es el único gran dispositivo de fusión con capacidad de usar como combustible, el Deuterio-Tritio.

2. IDENTIFICACIÓN DE LA HIPÓTESIS DE PARTIDA.

Tras la introducción plasmada con anterioridad, se puede llegar a deducir que los datos arrojados por todos los estudios y parametrizaciones obtenidos en el campo de la energía termonuclear, son inmanejables fuera del ámbito de la supercomputación. A su vez y siempre directamente proporcional a los recursos de cálculo necesarios, se encuentra el coste, alcanzando precios elevadísimos, no muy asumibles en los días actuales por la situación económica mundial en la que estamos inmersos.

Como alternativa a estas estructuras, surge la iniciativa de la migración de arquitecturas completas hacia y con soportes de software de código abierto. A su vez e intentando obtener una resolución extra a esta problemática, se debe tener en cuenta la arquitectura hardware que debe sustentar estos "supercomputadores de cálculo" *[13]* la cual debe ser específica y concreta para el fin que tiene encomendado, aunque no siempre es así.

En la actualidad se diseñan y adquieren infraestructuras de cálculo de carácter muy específico, las cuales en mucha ocasiones pasan una gran parte de su tiempo inactivas, por lo que resultan infrautilizadas.

De ahí surge el planteamiento de este proyecto, intentar aprovechar al máximo los recursos de distintos centros de cálculo, unidades de supercómputo e incluso sistemas clusterizados de propósito general con el mínimo coste, utilizando software de código abierto, y poniendo como garantía, la seguridad, confidencialidad y disponibilidad de los datos utilizados para el /los fines

requeridos de una manera completamente desatendida y permitiendo a una comunidad determinada, el acceso, gestión y control absoluto de sus trabajos.

3. OBJETIVOS.

El objetivo principal del desarrollo de este proyecto, es la implementación de un entorno, altamente colaborativo de recursos de computación para aplicaciones en fusión, poniendo a su disposición y de una manera lo más eficiente posible, todos los recursos asequibles para la organización así como en paralelo a ello un control exhaustivo de los recursos de cálculo y de los trabajos que se pretenden realizar.

3.1 PBS, como gestor.

En la actualidad las posibilidades que se ofertan en el entorno tecnológico de gestión de trabajos, así como transferencia de ficheros entre distintos dispositivos de hardware(servidores, cloud, redes, ubicaciones de almacenamiento, etc.) todo ello recreado mediante la correspondiente simbiosis hardware-software, requiere de una intervención abusiva,en su inmensa mayoría, del/los usuarios que demandan o proveen dichos trabajos ó resultados.

Esta tarea para casos en los que los números de procesos requeridos así como el tamaño de los datos a tratar sean de una entidad cuantitativa no demasiado elevada, se pueden llevar a cabo utilizando los métodos semi manuales que nos ofrece la tecnología "*común*" al alcance de prácticamente todos los usuarios.

Debido al volumen de datos a tratar en el CIEMAT proporcionados por el dispositivo de fusión termonuclear JET (bases de datos obrantes de más de 80 Terabytes con capacidad de duplicidad de almacenaje bianualmente, así como más de 10.000 señales obtenidas de monitorizaciones) y los requerimientos solicitados para una comunidad de usuarios considerable, se hace completamente necesario la existencia de un control exhaustivo en todos los procesos que compongan las tareas requeridas para la correcta y completa realización de los trabajos solicitados, siendo a su vez lo más trasparente posible para los demandantes de las mismas, logrando así una inocuidad prácticamente

absoluta en los datos requeridos/obtenidos. No pudiéndonos olvidar del gran potencial computacional requerido para la realización de los mismos.

Por eso se ha optado por la tecnología PBS (Portable Batch System) es un sistema que engloba un planificador de tareas (encolador) así como un balanceador de carga entre las diferentes unidades de cálculo (de ahora en adelante MOM).

Fue desarrollado en sus orígenes para la administración de los recursos computacionales de la NASA (*National Aeronautics and Space Administration)*.

PBS es el líder en la administración de recursos y considerado el estándar de facto para los sistemas de planificación bajo sistemas Linux, sin desmerecer en ningún momento al denominado Cloud Computing, aunque de naturaleza ampliamente diferenciada de lo aquí expuesto y requerido. *[14].*

El gestor PBS Torque (Tera-scale Open-source Resource and Queue manager) es un administrador de recursos que proporciona funcionalidades para controlar trabajos sobre distintos MOM's. Permite gestionar el estado de ellos y su utilización.

Basado en OpenPBS, es el sistema de colas batch de código abierto líder en el mercado. Soporta el procesamiento en paralelo de múltiples trabajos y colas interactivas de gestión de los mismos.

En la figura 8, se representa la arquitectura implementada para conseguir los objetivos propuestos.

3.1.1 Representación a alto nivel.

A continuación se describe a un nivel alto de abstracción, una arquitectura típica basada en tecnología torque.

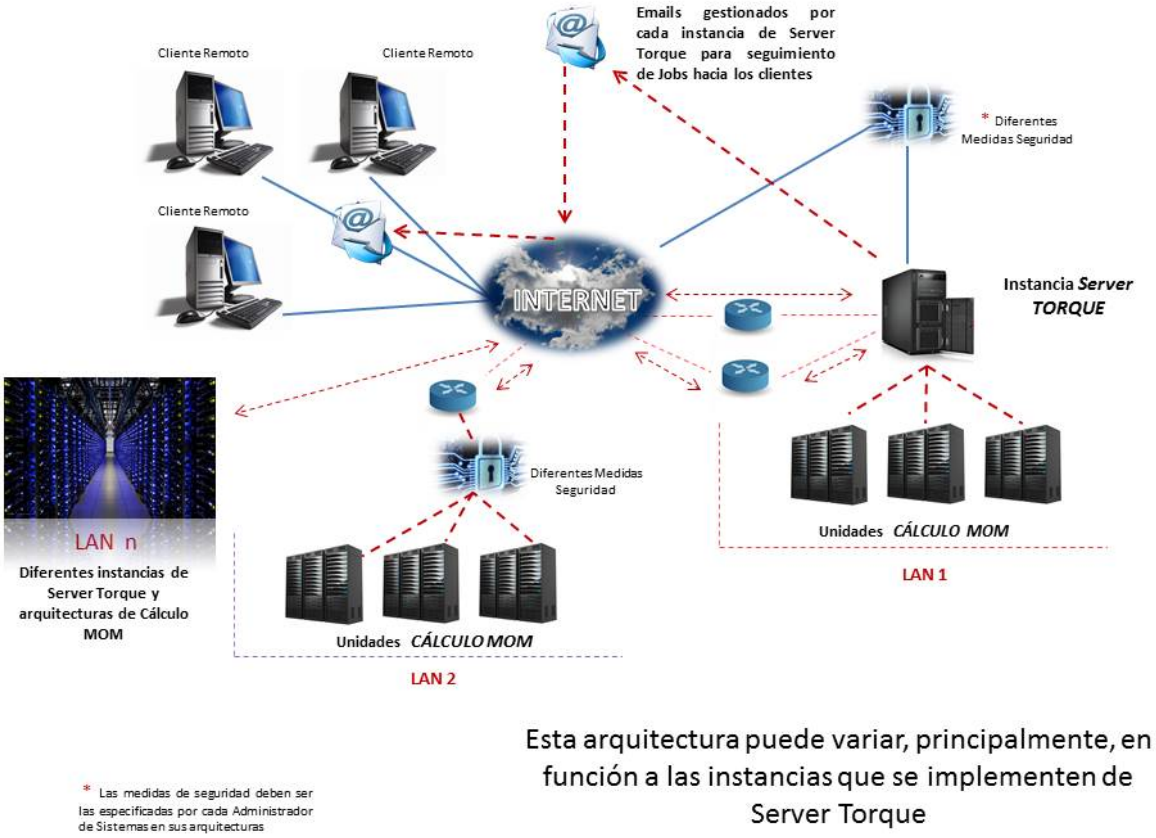


Figura. 8: *Arquitectura típica en el despliegue de TORQUE.*

- El ***Cliente remoto del sistema***, será el que podrá conectarse y utilizar la arquitectura desarrollada.
- Instancias del ***Servidor Torque***, encargado de administrar el conjunto de unidades de cálculo MOM. Es importante destacar que este software permite a su vez y bajo una misma máquina física, crear al unísono varias instancias de clientes MOM bajo la administración de una instancia Server *[15].*
- ***MOM´s (nodos de cómputo)*** que serán los encargados de prestar sus recursos de cómputo al servidor TORQUE. Este en su instalación básica implementa un encolador básico tipo FIFO, el cual gestionará los nodos de cálculo en función de cómo estén nombrados en el fichero en el que se definen los nodos de cómputo

(pbs_nodes). Esto como se verá más adelante cuando se detallen las arquitecturas de los MOM, no es del todo cierto, puesto que a la hora de definir unas colas de ejecución en ellas se implementan unos recursos de hardware específicos que si no son aportados por un solo elemento de cálculo, se requerirán otros adicionales.

Este tipo de despliegue (pbs_server, con sus pbs_mom definidos en una arquitectura específica) puede complementarse con dos encoladores de gestión de trabajos más avanzados Maui y Moab, el primero de libre distribución y ambos dos completamente integrables con la arquitectura definida con anterioridad.

- ***Moab Workload Manager*** es un sistema de programación y gestión altamente avanzado diseñado para clusters y sistemas de computación de utilidad bajo demanda. A un alto nivel, Moab aplica las políticas diseñadas al efecto permitiendo optimizar recursos en la administración de los trabajos, siendo a su vez un optimizador de, aparte de la carga de cálculo de los servicios a través de la combinación ideal de red, potencia de cálculo y los recursos propios de almacenamiento.

Moab permite una verdadera computación adaptable permitiendo modelarse en tiempo de ejecución según las necesidades cambiantes de los recursos. Moab aumenta la disponibilidad de los recursos del sistema, ofrece diagnósticos extensos de clusterización, ofrece potentes características QoS / SLA, y proporciona una rica visualización de rendimiento del clúster a través gráficos, estadísticas e informes.

- A continuación se aportan los diez puntos más significativos (según Adaptive Computing) que diferencian Maui de Moab, intentando aportar un parámetro diferenciador entre ambos.

Muchos usuarios, una vez probado el gestor Maui migran a Moab para conseguir que sus sistemas HPC (High Performance Computing sistemas de alto rendimiento de cómputo) sean más eficientes y se encuentren más optimizados.

1) *Escalabilidad de recursos*: Moab se adapta con facilidad a las necesidades del momento. Es capaz de gestionar más de 15.000 nodos, cientos de miles de trabajos en cola y más de 500 usuarios. Organizaciones con más de 2500 nodos de cálculo o más de 15.000 puestos de trabajo con tan sólo 100 usuarios han encontrado problemas con el rendimiento de Maui. El cambio a Moab HPC Suite

puede eliminar estos problemas y permitir sin errores, más puestos de trabajo con una mayor carga de trabajo que se ejecutan en el clúster por más usuarios.

2) *Mayor flexibilidad a la hora de la adaptación sobre el dinamismo de cambios, para optimizar los recursos.*
3) *Prestaciones de carácter profesional para el mantenimiento del funcionamiento del sistema a máximo rendimiento***.** Servicio que se ofrece para los clientes proporciónenles soporte técnico 24 horas, 7 días a la semana.
4) *Gestión inteligente de la automatización de los procesos para reducir la complejidad de su desarrollo, controlando a su vez el balanceo tiempo/recursos y mejorando la eficiencia del sistema con la integración del gestor de recursos.*
5) *Automatización Uptime para maximizar la fiabilidad del sistema y la productividad Moab.*
6) *Herramientas de administración potentes bajo entorno gráfico para una gestión más eficiente.*
7) *El envío de trabajos de HPC simplificados y la gestión de los usuarios para aumentar la productividad y disminuir la carga de los administradores.*
8) *Mejoras en lo relativo a la gestión de la carga de los trabajos y así optimización de los recursos del sistema.*
9) *Grid y capacidades de gestión de carga de trabajo multi-cluster.*
10) *Incorpora una parte contable, como módulo, con el que se pueden sacar rendimientos y optimizaciones de los recursos empresariales.*

Para el desarrollo de este proyecto, y como punto de partida del mismo, se ha optado por la utilización del encolador que se incorpora en la distribución de Torque, puesto que no es el objetivo principal de este proyecto, dotar de un encolador más eficiente sino del aporte cuantitativo en recursos computacionales a la arquitectura que se diseñe.

4. JUSTIFICACIÓN CIENTÍFICO SOCIAL

Una de las principales razones que han motivado la realización del presente proyecto, ha sido la optimización de los recursos disponibles sin necesidad de crear una red de excesiva complejidad para la unificación de recursos de cálculo para un fin común.

Efectivamente existen en el mercado y en el mundo de la computación varias opciones que ya se encuentran implementadas y todas ellas dedicadas a estos fines de cálculo. Como es sabido la "Computación Grid" es una tecnología innovadora que permite utilizar de forma coordinada todo tipo de recursos (entre ellos cómputo, almacenamiento y aplicaciones específicas) que no están sujetos a un control centralizado. En este sentido es una nueva forma de computación distribuida, en la cual los recursos pueden ser heterogéneos (diferentes arquitecturas, supercomputadores, clusters..) y se encuentran conectados mediante redes de área extensa (por ejemplo Internet).

Este entorno surge en entornos científicos a principios de los años 1990, su entrada al mercado comercial se ha producido siguiendo la llamada Utility computing *[16]* supone una importante revolución.

El término *grid* se refiere a una infraestructura que permite la integración y el uso colectivo de ordenadores de alto rendimiento, redes y bases de datos que son propiedad y están administrados por diferentes instituciones. Puesto que la colaboración entre instituciones envuelve un intercambio de datos, o de tiempo de computación, el propósito del *grid* es facilitar la integración de recursos computacionales. Universidades, laboratorios de investigación o empresas se asocian para formar *grid* utilizando algún tipo de software que implemente este concepto.

El gestor PBS TORQUE (*Tera-scale Open-source Resource and Queue manager*) es un administrador de recursos que nos provee de mecanismos para controlar trabajos sobre distintos nodos de un clúster o de nodos situados en distintas redes, siendo éste uno de sus principales puntos fuertes. Permite gestionar el estado de los nodos, su uso y a la vez permite añadir o quitar nodos de forma interactiva.

Está basado en OpenPBS, es el sistema de colas batch de código abierto líder en el mercado. Soporta el procesamiento en paralelo de múltiples colas batchs y colas de trabajo interactivas.

5. METODOLOGÍA DESARROLLADA (ALCANCE DE OBJETIVOS).

Para poder llevar a cabo toda esta implementación, desde un principio se pensó desarrollarlo con el apoyo de la virtualización puesto que iba a facilitar la implantación del desarrollo y así a posteriori poderlo llevar de manera

relativamente sencilla, a otras arquitecturas de hardware en las que se pudiera implantar.

Las ventajas de la virtualización son muchas, las más evidentes son el ahorro en hardware, normalmente en una relación de 10 a 1 servidores en el caso más optimista, y de 6 a 1 en el caso más pesimista (VMWare Inc., 2010). Esto conlleva reducciones en costes de mantenimiento y administración, además de reducciones en el consumo eléctrico. Se podría llegar a ahorrar hasta 7.000 KWh/año por servidor (lo que supone 1.500 € anuales) (VMWare Inc., 2008a). El ahorro energético es un aspecto a tener en cuenta ya que existen informes (Fundación Vida Sostenible, 2010) que confirman que la tendencia del precio de la energía va a ser creciente. Además, nos ofrece la posibilidad de alinearnos con políticas medioambientales, ya que si más del 50% de la energía eléctrica en España procede de combustibles fósiles (REE 2010), al reducir el consumo eléctrico se reducirán las emisiones de dióxido de carbono a la atmósfera. Se podrían dejar de emitir 4 toneladas de dióxido de carbono al año por servidor virtualizado, el equivalente a quitar 1,5 millones de vehículos de la circulación (VMware Inc., 2010).Todo en consonancia con la situación socio económica por la que el planeta está pasando en los días que vivimos.

La virtualización también ofrece ventajas en la seguridad y movilidad de los sistemas. La realización de copias de seguridad es un servicio prestado por cualquier software de virtualización que, además, al permitir copias de la máquina virtual completa, facilita su traslado a nuevas ubicaciones y su puesta a punto es inmediata.

5.1 ¿Qué es la virtualización?.

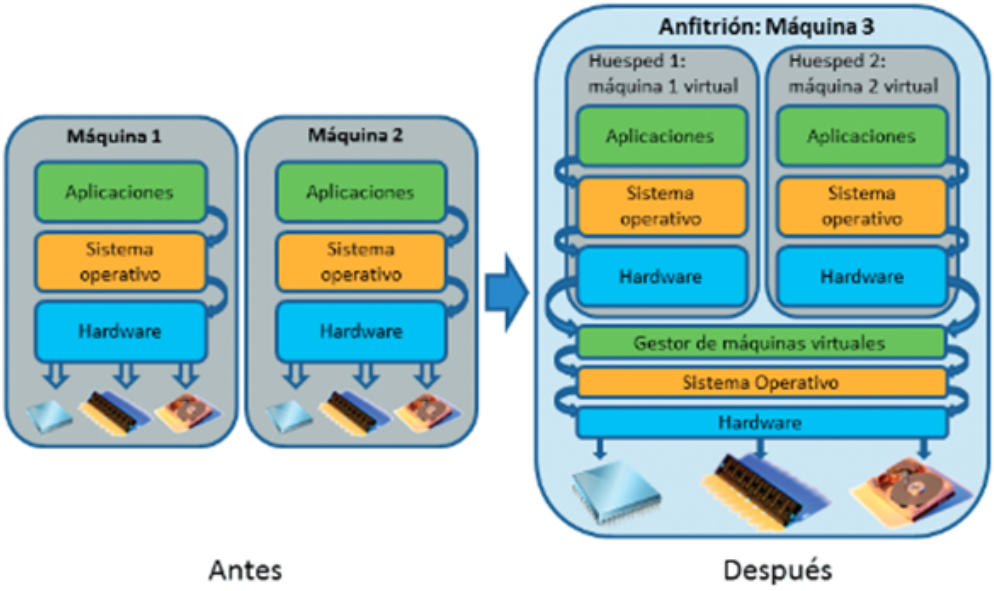


Figura. 9: *Representación de la abstracción planteada para un entorno virtualizado (máquinas anfitrionas y huéspedes).*

En la Figura 9 se observa la abstracción de los recursos de una computadora (Turban et al., 2008) y su puesta en funcionamiento como máquina virtual en otra máquina física. El término “máquina anfitriona” se refiere a la máquina física donde se lleva a cabo la virtualización mientras que el término “máquina huésped” se refiere a la máquina virtual (también llamada hypervisor o virtual machine monitor VMM).

Un sistema de virtualización debe ser capaz de ofrecer una interfaz en la máquina anfitriona para poder interactuar con el sistema operativo de la máquina virtual. Además, la máquina anfitriona debe ofrecer una interfaz de sus recursos a la máquina virtual para que pueda utilizarlos. De estas interfaces de comunicación se encarga un software que se instala en la máquina anfitriona para poder ejecutar las máquinas virtuales. Existen muchos tipos de software de virtualización en el mercado, tanto software propietario como software libre y para ejecutarse sobre máquinas anfitrionas que funcionen como servidores, o bien sobre cualquier computadora personal. Entre los más utilizados están VirtualBox y Virtual PC para computadoras personales, Xen y KVM para servidor, y VMware para ambos.

5.2 Ventajas de la virtualización:

- *Reducción de costes de hardware*.
- *Reducción del consumo eléctrico y el dióxido de carbono emitido a la atmósfera al reducir el número de máquinas.*

Esto en los tiempos que corren y bajo el paradigma de la fusión termonuclear como alternativa de fuente de energía viene al hilo de manera fundamental.

- *Mejora de TCO y ROI*.

Al reducir el número de máquinas se está disminuyendo el coste total de la propiedad (total cost of ownership, TCO) y por lo tanto se está consiguiendo un retorno de la inversión (return of investment, ROI) mejor. Si se añade está ahorrando en la factura eléctrica, el ROI será mejor aun.

- *Reducción de los costes de espacio.*

Al tener menos máquinas se necesitará menos espacio físico para poder desplegar el centro de procesamiento de datos. Administración global centralizada y simplificada: son menos máquinas físicas las que hay que administrar, además de que el software de virtualización ayuda a la gestión remota de las máquinas virtuales.

- *Rápida incorporación de nuevos recursos para los servidores virtualizados.*

Se trata de una tecnología escalable donde es fácil la incorporación de nuevas máquinas virtuales en una misma máquina anfitriona.

- *Mejora en los procesos de clonación y copia de seguridad del sistema.*

Mayor facilidad para la creación de entornos de prueba que permiten utilizar nuevas aplicaciones sin afectar a la producción en un entorno controlado agilizando el proceso.

Este aspecto ha sido uno de los más importantes a la hora de llevar a cabo este proyecto, puesto que el clúster, en el que a la finalización del presente estudio, quedará implantado el gestor TORQUE es un recurso que está siendo utilizado por diferentes usuarios, demostrando así uno de los pilares que sustentan el desarrollo de aplicaciones que conduzcan a la optimización de recursos, como se hizo ver en la exposición de los hechos que apoyan este proyecto (optimización de recursos en sus estados de inactividad).

- *Aislamiento.*

Un fallo general de sistema de una máquina virtual no afecta al resto de máquinas virtuales.

Según un estudio realizado por Network Instruments (2009) la virtualización también ofrece algunas desventajas así como soluciones a problemas acaecidos durante el desarrollo de este proyecto de no fácil resolución. *[15].*

- *Balanceo dinámico.*

Este balanceo dinámico de las máquinas virtuales entre los servidores físicos que componen el pool de recursos, garantiza que cada máquina virtual se ejecute en el servidor físico más adecuado, proporcionando así un consumo de recursos homogéneo y óptimo en el conjunto de toda la infraestructura.

- *Migración en caliente*.

Esta migración en caliente de máquinas virtuales (sin pérdida de servicio) de un servicio físico a otro elimina la necesidad de paradas planificadas por mantenimiento de los servidores físicos.

5.3 ORACLE Virtual box como entorno de virtualización.

Es un software para poder llevar a cabo la virtualización habiendo sido creado en sus orígenes por la empresa alemana Innotek GmbH.

Actualmente está siendo desarrollado por Oracle Corporation como parte de su familia de productos de virtualización.

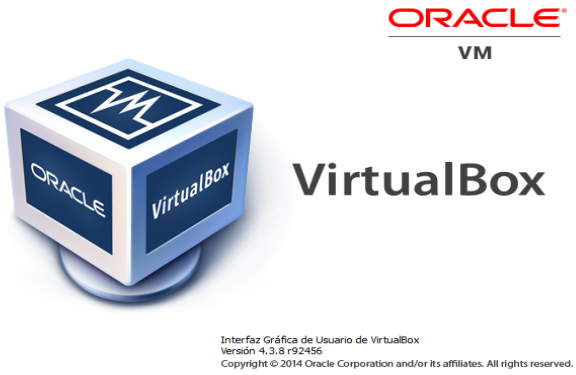


Figura. 10: *Logo del entorno de virtualización utilizado para la recreación de la arquitectura x86(ORACLE VIRTUAL BOX de la empresa Oracle Corporations).*

Entre los sistemas operativos soportados (en modo anfitrión) se encuentran GNU/Linux, Mac OS X, OS/2 Warp, Windows, y Solaris/OpenSolaris, y dentro de éstos es posible virtualizar los sistemas operativos FreeBSD, GNU/Linux, OpenBSD, OS/2 Warp, Windows, Solaris, MS-DOS y muchos otros.

En comparación con otras aplicaciones de virtualización de pago (VMware Workstation o Microsoft Virtual PC), VirtualBox carece de algunas funcionalidades, pero provee de otras como la ejecución de maquinas virtuales de forma remota, por medio del Remote Desktop Protocol (RDP), soporte iSCSI.

Soporta el conjunto de instrucciones de virtualización en procesadores Intel VTx y AMD-V, esto mejora el rendimiento situándolo con muy buena puntuación en recientes comparativas de virtualización.

Mantiene una edición de código abierto **Open Source Edition (OSE)**, aunque con unas pocas menos funcionalidades.

Se puede instalar en una amplia variedad de sistemas operativos: Debian, Fedora, Mandriva, Ubuntu, RedHat, Open SOlaris, Mac OS X, Xandros, openSUSE, PCLinux OS.

Dispone de una amplísima documentación técnica y de usuario *[16]*

5.3.1 Instalación de entorno de Virtualización.

Figura. 11: *Imagen de la pantalla de bienvenida al comienzo de la instalación del entorno de virtualización.*

A continuación se pasa a detallar los pasos dados para la creación del entorno virtualizado del sistema PBS TORQUE.

La versión con la que se ha trabajado, es la referenciada en la figura 11.

Oracle Virtual Box, de manera regular oferta la posibilidad de actualizar la versión del software así como de las versiones de las "extensions packs", que son el conjunto de extras que permiten, entre otras, las conexiones de dispositivos usb entre anfitrión—virtualizado, virtualizado—virtualizado, así como compartición de carpetas en las redes creadas en el entorno.

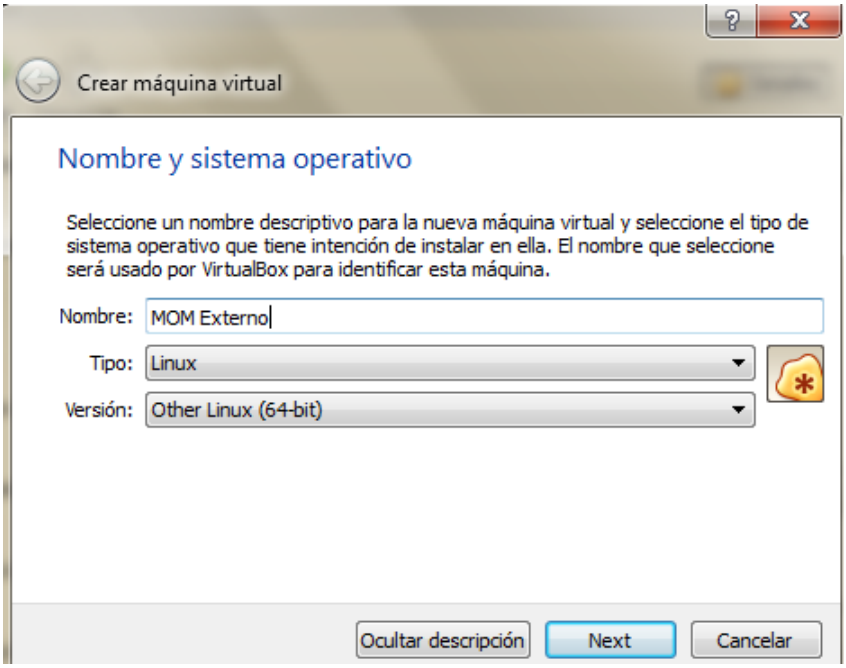


Figura. 12: *Imagen representativa del entorno de instalación de una máquina virtual bajo VirtualBox(Pantalla de configuración).*

Una vez instalado el entorno, se procede a desplegar las máquinas virtualizadas que harán de unidades de cálculo (MOM) denominada "MOM externo" el cual estará soportado bajo una arquitectura de 64 bits y un S.O. Linux, y otra que soportará la infraestructura de Server TORQUE.

El sistema operativo a instalar en esta arquitectura será CentOS 64 en su versión 6, tanto para el MOM como para el Server.

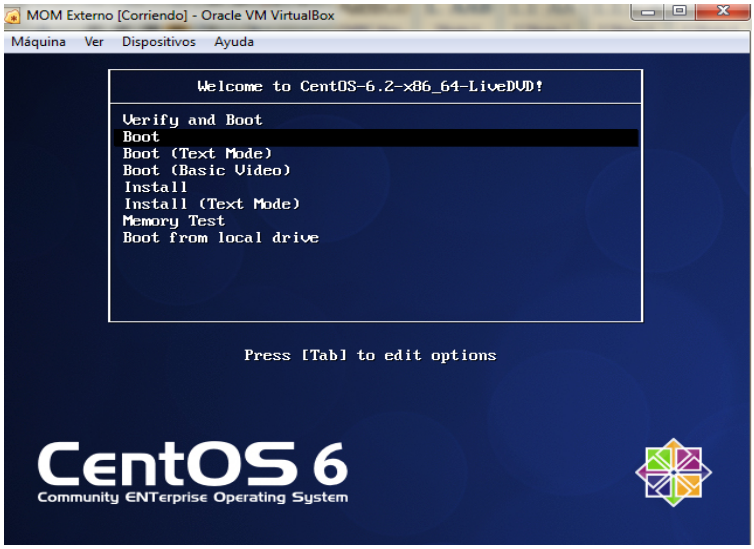


Figura. 13: *Imagen de la información otorgada por el sistema Opeativo CentOS v6 en el proceso de instalación.*

CentOS (**C**ommunity **ENT**erprise **O**perating **S**ystem) es una bifurcación a nivel binario de la distribución Linux Red Hat Enterprise Linux **RHEL**, compilado por voluntarios a partir del código fuente liberado por Red Hat.

El entorno de red a simular entre las dos máquinas, constará de dos interfaces. Por parte del MOM en su interfaz de red Eth0 se conectarán ambas máquinas simulando una LAN (Local Area Network, red de área local), no forzando la creación de un servidor DHCP (Dynamic Host Configuration Protocol, protocolo de configuración dinámica de host) sino otorgando una dirección Ip fija y su puerta de enlace correspondiente, simulando así el entorno de red de área local en la que se gestionarán las peticiones de ejecución de los trabajos a través de la instancia del servidor de TORQUE.

En el lado donde correrá el demonio pbs_server, se simulará el despliegue de una máquina que realice las misiones propias de un frontend, con la única peculiaridad a destacar, en un principio, que constará de dos interfaces de red diferenciados **Eth0** que corresponderá a la dirección Ip 80.0.0.1 mediante la cual conectará con la máquina dedicada al cálculo ,MOM, la cual forma parte de una red local a la cual se la denominada “clúster”.

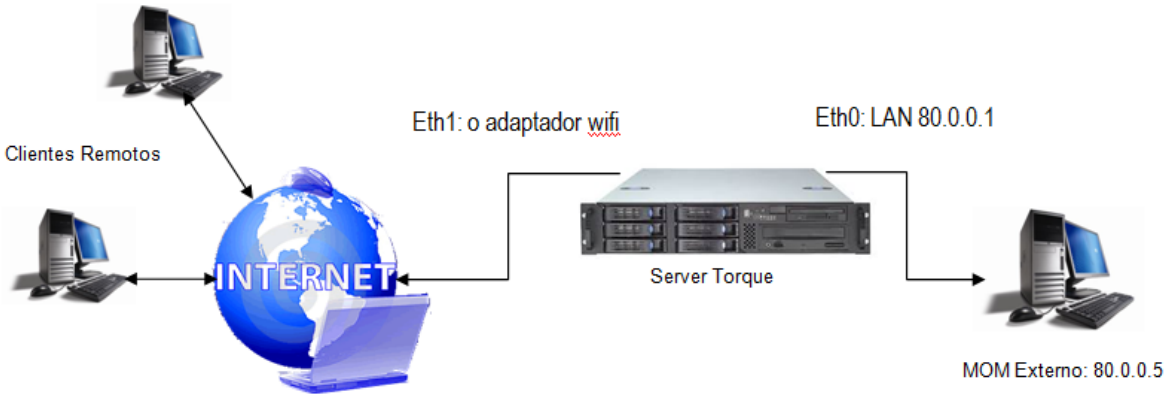


Figura. 14: *Arquitectura montada con TORQUE como gestor único.*

Por otro lado el interfaz Eth1 será el que provea al frontend de su capacidad para salir a la web, otorgándole así de funcionalidad de comunicación para interactuar con cualquier usuario en cualquier situación geográfica.

Como se puede apreciar, esta representación es el esquema básico tomado como referencia de partida, como arquitectura principal y así ir expandiéndola en función a las necesidades que se vayan planteando.

```
[centoslive@livedvd ~]$ ping 80.0.0.1
PING 80.0.0.1 (80.0.0.1) 56(84) bytes of data.
64 bytes from 80.0.0.1: icmp_seq=1 ttl=64 time=1.09 ms
64 bytes from 80.0.0.1: icmp_seq=2 ttl=64 time=0.568 ms
64 bytes from 80.0.0.1: icmp_seq=3 ttl=64 time=0.618 ms
64 bytes from 80.0.0.1: icmp_seq=4 ttl=64 time=0.694 ms
64 bytes from 80.0.0.1: icmp_seq=5 ttl=64 time=0.616 ms
64 bytes from 80.0.0.1: icmp_seq=6 ttl=64 time=0.736 ms
64 bytes from 80.0.0.1: icmp_seq=7 ttl=64 time=0.624 ms
^C64 bytes from 80.0.0.1: icmp_seq=8 ttl=64 time=0.627 ms
64 bytes from 80.0.0.1: icmp_seq=9 ttl=64 time=0.516 ms
^C
--- 80.0.0.1 ping statistics ---
9 packets transmitted, 9 received, 0% packet loss, time 8109ms
rtt min/avg/max/mdev = 0.516/0.676/1.092/0.161 ms
[centoslive@livedvd ~]$ ^C
[centoslive@livedvd ~]$
```

Figura. 15: *Imagen obtenida tras el lanzamiento del comando ping a través de la Shell para la comprobación de la conexión en red entre las diferentes máquinas componentes.*

Al instalar las dos máquinas y realizar las pruebas de conexión se observa que la red de área local se encuentra perfectamente implementada.

Como consecuencia de esta arquitectura la máquina denominada "Frontend2" será la que soporte la arquitectura de servidor, corriendo en ella el demonio pbs_torque (administrador de jobs).

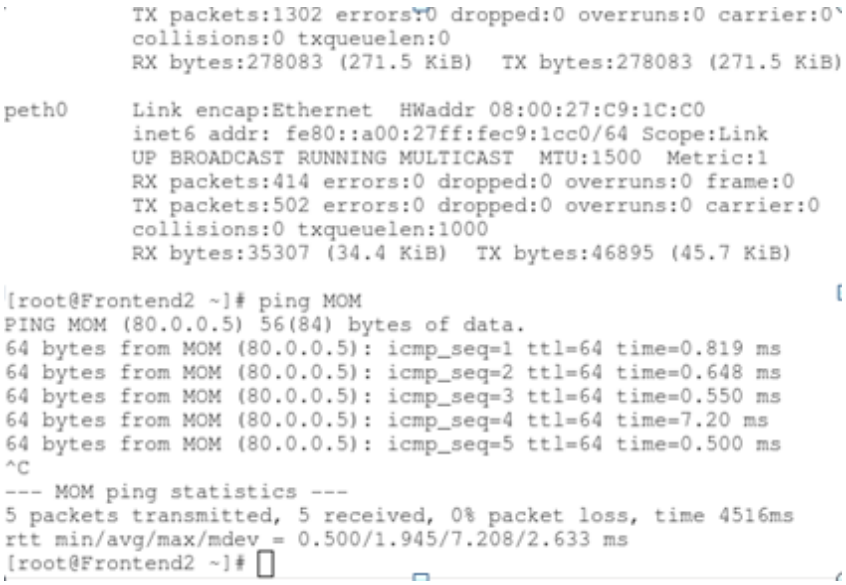


Figura. 16: *Imagen obtenida tras el lanzamiento del comando ping a través de la Shell del lanzamiento de solicitudes para la comprobación de la conexión en red de los nodos de cálculo (MOM's).*

Como complemento a este esquema, y dentro de una de las funciones encomendables a un frontend y a través de la instalación de un servidor web Apache en la máquina principal CentOS (distribución del sistema operativo Linux) se proporciona el soporte de seguridad suficiente para albergar los portales web, bases de datos y demás software necesario para el correcto funcionamiento del entorno propicio para el envío de trabajos a través de internet.

5.4 Instalación de TORQUE (plataforma multimodal).

Adapatative Computing es el proveedor en donde se alojan los elementos necesarios para la instalación de la arquitectura aquí presentada.

En 1966 David Jackson, fundador y director de Adaptative Computing creó el software de gestión de computación inteligente que permitía a varios equipos independientes trabajar de forma colaborativa, permitiendo a las diferentes empresas controlar y optimizar sus valiosos recursos informáticos. A

lo largo de los años ha ido evolucionando y abarcando más sectores de consumidores de supercomputación como por ejemplo la NASA, (*National Aeronautics and Space Administration, Administración Nacional de la Aeronaútica y del Espacio*) llegando hasta el reconocimiento por parte Gartner *[17]* que identificó a Adaptative Computing como uno de los pocos fabricantes mundiales que han demostrado ser capaces de ofrecer tanto la infraestructura en tiempo real como las soluciones en la nube para el supercómputo.

5.4.1 Componentes.

En un primer lugar, paso a especificar los cuatro integrantes básicos que componen esta arquitectura:

1) **Nodo Maestro:** Es donde correrá el demonio pbs_server. Dependiendo de la complejidad y necesidad de la arquitectura cumplirá de manera exclusiva esta función u otras en paralelo, como será nuestro caso ya que se instalará (como instancia definitiva y más adelante se detallará) en una máquina que desarrollará otras funciones no propias de TORQUE.
2) **Nodos interactivos para trabajos:** Estos permitirán a través de ellos el envío y uso de comandos para los clientes y usuarios de la arquitectura.
3) **Nodos de Cómputo:** Estos serán los verdaderamente importantes en este software ya que son los que verdaderamente llevarán la carga computacional. En ellos correrá el demonio de pbs_mom e interactuarán directamente con el servidor pbs_server. Dependiendo de las necesidades en una máquina física podrá haber implementado un solo MOM o varias instancias de ellos.
4) **Recursos:** Dependiendo de la arquitectura/necesidad, dentro de esta definición se encuentran englobadas las redes de alta velocidad, los dispositivos de almacenamiento dedicados etc.

El flujo básico del trabajo pasa por las siguientes etapas: *Creación, envío, ejecución y finalización*, siendo en todo momento posible su monitorización y seguimiento, radicando en este concepto uno de sus principales potenciales de trabajo.

5.4.2 Primeros pasos.

En una primera aproximación el montaje del sistema va a consistir en un *Server* y una sola máquina de cálculo (MOM) en comunicación a través de una red LAN, realizada a través de Oracle VirtualBox.

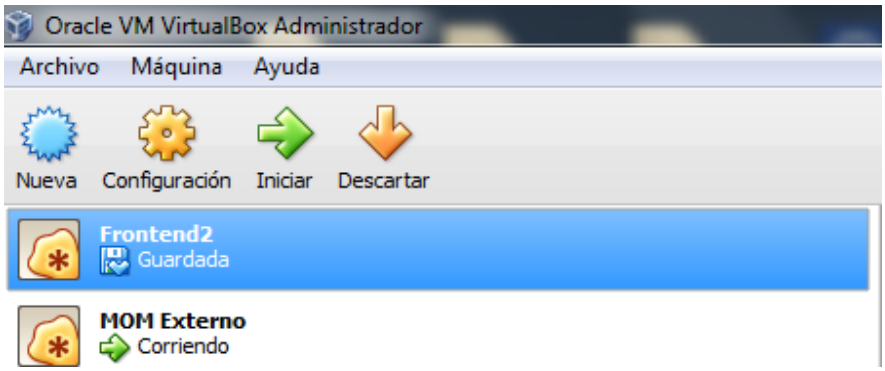


Figura. 17: *Imagen de las máquinas virtuales creadas (FrontEnd2, será la que albergará la instalación del server de TORQUE).*

Las máquinas creadas son *Frontend2* como nodo en el que se instalará el Server de TORQUE y la otra con nombre MOM Externo, que es donde correrá el demonio “pbs_mom” y realizará las funciones de nodo de cálculo.

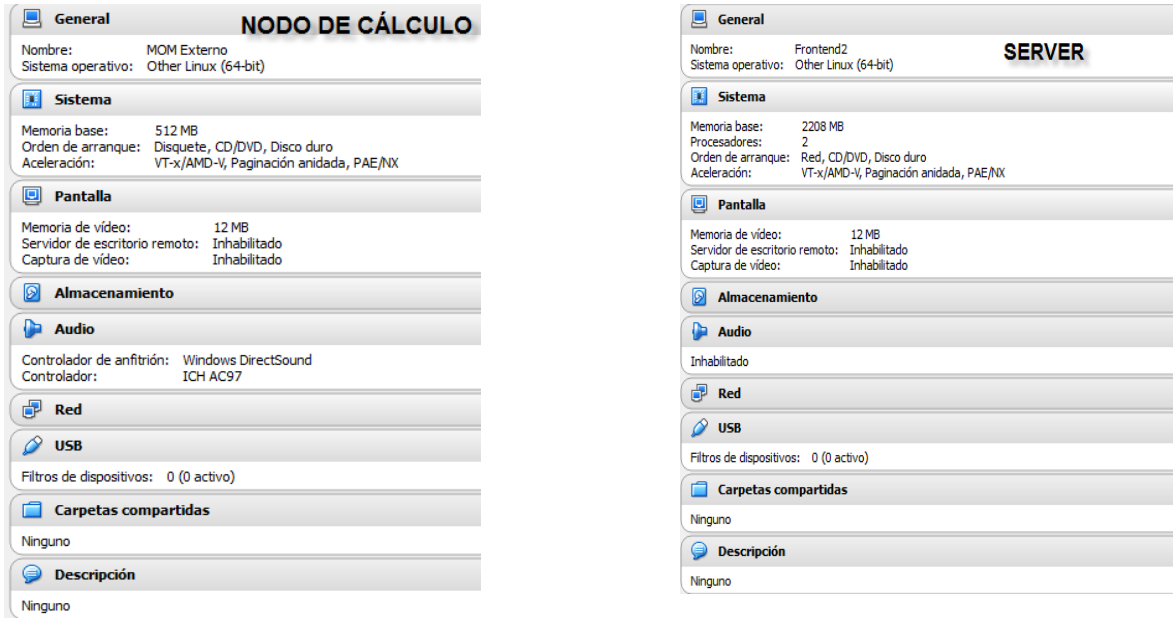


Figura. 18: *Características técnicas de las máquinas virtuales instaladas y van a ser el nono de cálculo (MOM) y el Server de TORQUE, respectivamente.*

La figura 18, refleja las características características técnicas generales de las máquinas que se han instalada dentro de OracleVM. Lo más significativo de esta instalación, va a ser que ambas máquinas son diferentes y a su vez se encuentran conectadas a una misma LAN.

Se ha seleccionado un S.O. Linux de 64 bit y de arquitectura Homogénea. *[18]*.

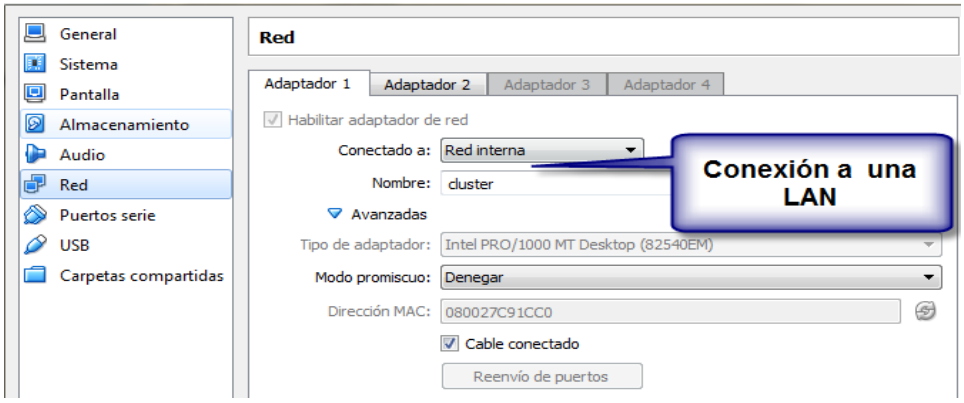


Figura. 19: *Configuración de red de la máquina virtual instalada.*

5.4.3 Instalación PBS_SERVER.

Lo primero que hay que hacer es descargase desde Adaptative Computing, bien directamente desde la página o a través de los comandos de instalación del S.O. CentOS, los paquetes de instalación de TORQUE.

Una vez descargado los paquetes de instalación se descomprimen e instalan. ./configure

make

make install

Se comprueba que se han generado las carpetas dentro de /var/spool/torque que será el $HOME_TORQUE.

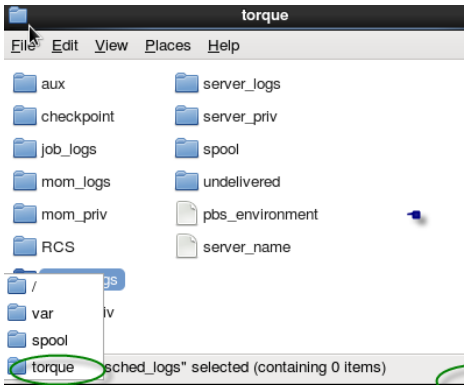


Figura. 20: *Imagen obtenida en la que se muestra la ubicación del directorio de instalación de TORQUE, en la que se albergan los archivos ya instalados en el sistema (conocido como $HOME_TORQUE).*

Importante verificar que las variables de entorno se encuentran bien definidas y ubicadas sus localizaciones.

Acto seguido se deben añadir en torque/server_priv/nodes el nombre de los nodos que formarán parte de la estructura de cálculo a definir.

Estos nodos de cálculo se pueden especificar de dos formas diferentes:

1) A través de una dirección IP que corresponderá con la definida para esa máquina (80.0.0.5)

2) O a través de un nombre de host (MOM).En este caso se debe poner dentro del fichero host de la máquina en donde corra el server de TORQUE, la equivalencia de la dirección IP con el nombre.

De igual modo la definición de estos nodos llevan asociados varios parámetros, los cuales son opcionales siendo éstos:

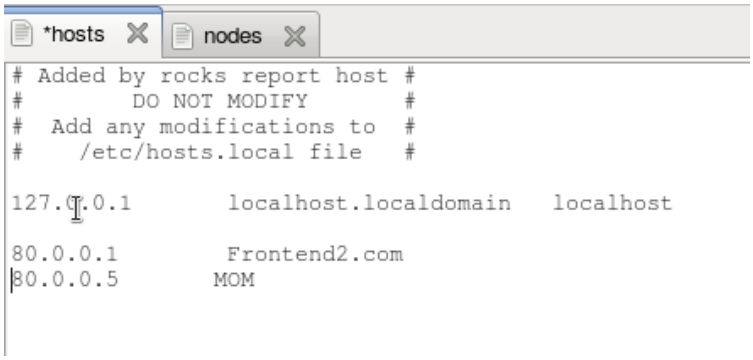


Figura. 21: *Contenido del archivo hosts de la máquina en la que se encuentra instalado el Server de TORQUE (Frontend2).*

:ts (timeshared). Este tiempo compartido es enumerado por el servidor en los informes de estado que emite, no asignando trabajos a ellos.

np= Número de procesadores virtuales otorgados, pudiendo ser menor, igual o superior a los reales.

gpus= Número de Gpus disponibles en cada nodo (fuerte rama de especificación para ganancia en potencia computacional.

Lo anteriormente detallado,se puede especificar para que de manera automática sea reconocido, a través del comando **"**qmgr -c set server auto_node_np = True"

Una vez realizadas estas modificaciones, se comprueba que las máquinas son visibles y accesibles a través de ssh (Secure Shell) *[19]*, habiéndose creado en la primera conexión y la clave de conexión entre ambas máquinas denominado como "keyfinger".

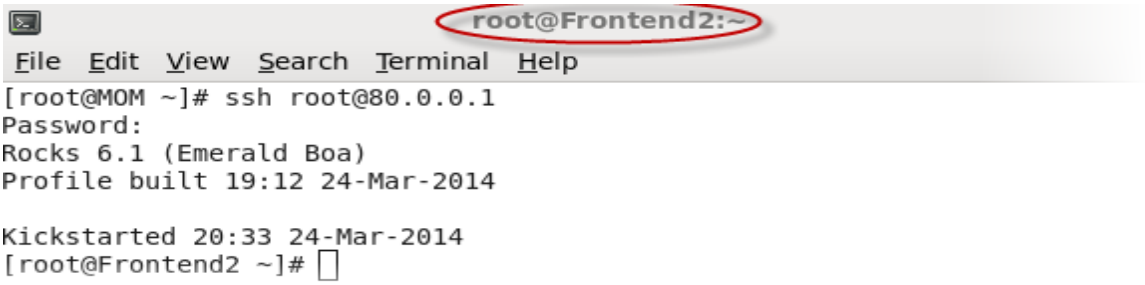


Figura.22: *Comprobación de la visibilidad de las máquinas (MOM y Frontend2) a través de SSH, con la correspondiente creación en primera instancia del keyfinger. (desde el Frontend2 hacia el MOM).*

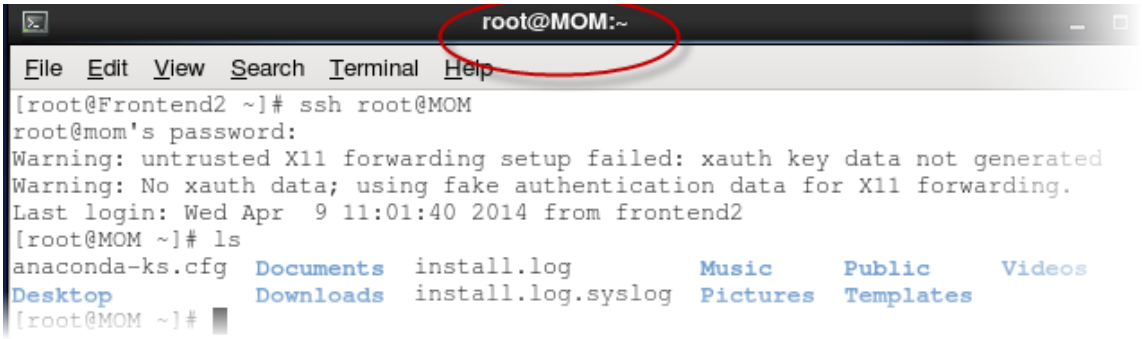


Figura. 23: *Comprobación de la visibilidad de las máquinas (MOM y Frontend2) a través de SSH, con la correspondiente creación en primera instancia de keyfinger (desde el MOM hacia el Frontend2).*

5.4.4 Creación Packages.

Desde la ubicación en donde se descomprimieron los paquetes de TORQUE, a través de la instrucción make packages, se crearán los paquetes necesarios para su instalación en aquellas máquinas que sean dedicadas como elementos de cálculo (MOM), sin necesidad de formateo u otros tipos de adaptaciones.

Los únicos paquetes necesarios para su instalación en las mencionadas máquinas son:

- torque-package-clients-linux.
- Torque-package-mom-linux.

5.4.5 Inicialización de TORQUE.

Principalmente existen dos alternativas para una inicialización principal del servidor, a través de la instrucción ./torque.setup pbs_server –t create teniendo que reiniciar el servidor (qterm,pbs_server) tras su realización.

Una vez inicializado el servidor, instalación de los paquetes en los nodos de cálculo, se procede a la configuración de estos, añadiendo en "TORQUE_HOME/mom_priv/config" el nombre del host donde corre el servidor ($pbsserver) entre otros parámetros opcionales como por ejemplo $usecp <HOST>:<SCRDIR> <DSTDIR>.

5.4.6 Configuración/creación de colas.

El concepto de cola aquí expuesto se corresponde con una estructura de datos donde los elementos contenidos se organizan y procesan según el orden de llegada. La configuración de la cola (prioridad, recursos asociados, tiempo de ejecución) se aplica a los elementos que contiene, otorgando estas características el gestor TORQUE, en función a la demanda del sistema implementado o incluso a las necesidades requeridas para los trabajos ejecutados.

Para comenzar a definir el concepto de "colas" serán aquellas especificaciones que permitirán que los trabajos enviados para su ejecución al server de torque, puedan ser recepcionados, almacenados y posteriormente distribuidos a diferentes ubicaciones en función a las distintas prioridades para las que han sido parametrizadas.

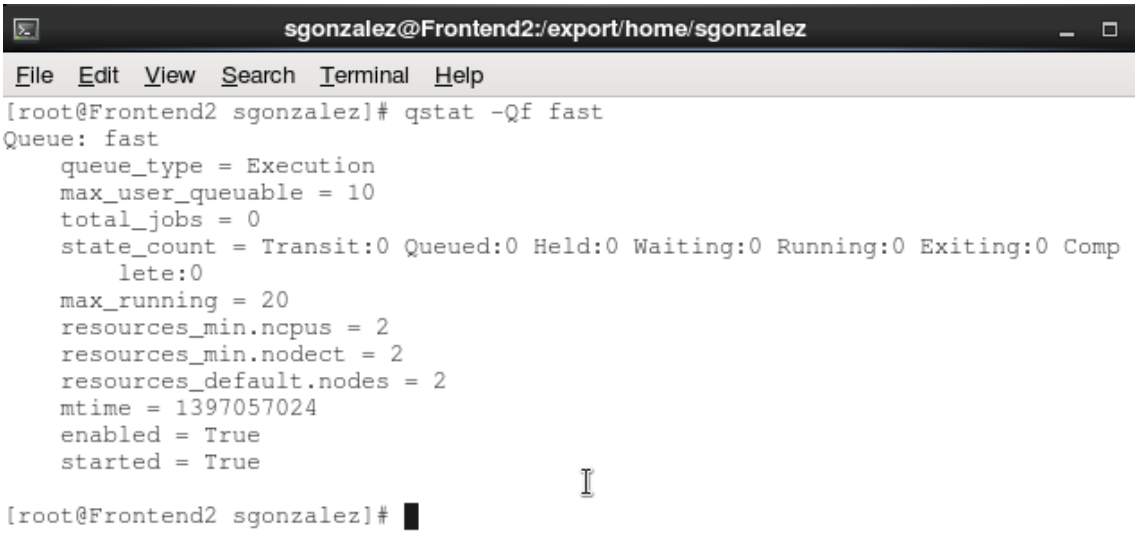


Figura. 24: *Imagen por la que a través del comando qstat –Q [nombre_de_la_cola], ordenada mediante consola en el Frontend2 se muéstralas características otorgadas a la ola de ejecución denominada "fast".*

Como se puede observar se ha creado a través del comando Qmgr una cola de ejecución denominada fast la cual ha sido configurada con diferentes parámetros, entre los que figuran el número mínimo de cpu, de nodos, número máximo de usuarios a los que se les permite tener trabajos encolados, así como el número de nodos máximo dedicados al trabajo encolado.

Evidentemente estos parámetros son todos configurables e irán en concordancia con la prioridad que tenga otorgada. En paralelo a todo esto y de gran importancia, es, como parte de los parámetros configurables, el número de usuarios que puedan acceder a los recursos otorgados por las diferentes colas, por lo que es otra vía de restricción a las mismas y a las propiedades del server en sí.

Otro tipo de colas son las de tipo "enrutamiento" siendo las encargadas de dirigir un trabajo a otra cola destino sobre las que están definidas otras

peculiaridades y limitaciones. Con este encaminamiento se consigue un gran nivel de encapsulamiento de tipos y direcciones de colas.

5.4.7 Comandos principales Torque.

En esta sección se pretende otorgar de conocimiento de una manera rápida y concisa de una serie de comandos de gran utilidad que son los que se van a utilizar para la creación, el envío y monitorización de los diferentes trabajos y colas de ejecución ó enrutamiento.

- **momctl**: Administración y diagnóstico de los nodos MOM.
- **pbsdsh**: Inicia tareas en trabajos paralelos.
- **pbsnodes**: Ver el estado de todos los nodos del sistema.
- **qalte**r: Modificación de los trabajos encolados.
- **qdel**: Borra cancela trabajos encolados.
- **gpumode:** Especifica un nueva forma utilización de GPU.
- **gpueset**: Resetea las GPUS.
- **qhold**: Bloqueo de trabajos encolados.
- **qmgr**: Para administrar/crear/modificar colas.
- **qrun**: Comienzo de un trabajo.
- **qrerun**: Re arranque de un trabajo.
- **qrls**: Lanzamiento de trabajo por lotes.
- **qstat**: Visualización de trabajos y colas.
- **qsub**: Envío de trabajos.
- **qterm**: Shutdown de pbs_server.
- **pbs_server**: Lanzamiento del demonio server.
- **pbs_mom**: Reinicio del demonio MOM (nodo de ejecución).

5.4.8 Problemáticas y puntos a destacar de esta configuración:

Tras las configuraciones, pruebas y manejo de diferentes combinaciones del sistema desarrollado y todo en el entorno virtual que se ha presentado, se han producido una serie de errores y de problemas, los cuales se detallan en secciones que a continuación se especifican, constituyéndose como puntos de partida de resoluciones que han forjado una fuente de buenas prácticas para el desarrollo de otra plataforma (también virtualizada) más óptima.

Los principales problemas encontrados y más tediosos en su resolución, fueron dos principalmente:

1) *Configuración de puertos de comunicación.*
2) *Sistema de tránsito de ficheros entre unidades de cómputo y servidor.*

5.4.8.1 Configuración de puertos de comunicación:

La comunicación entre nodos y servidor/servidores necesita la configuración de los puertos de comunicación (a partir del 15001,) siendo esta situación compleja de configurar manualmente, aumentando esta dificultad según se incremente la red de computadores dedicados al cálculo en el sistema (MOM´s).

Consecuentemente se debe tener en cuenta la configuración del contenido del firewall de la organización, el router y la propia asignación otorgada a los puertos de comunicaciones dentro de la configuración para los dispositivos de software y hardware ya existentes.

5.4.8.2 Tránsito de ficheros.

Cuando se inicia un trabajo de proceso por lotes, el archivo estándar de entrada si se especifica, se copia en el mismo directorio desde el que se ordenó su ejecución y a medida que el trabajo se va ejecutando los archivos stdout y stderr se generan y se colocan en este directorio utilizando la nomenclatura $JOBID.OU y $JOBID.ER.

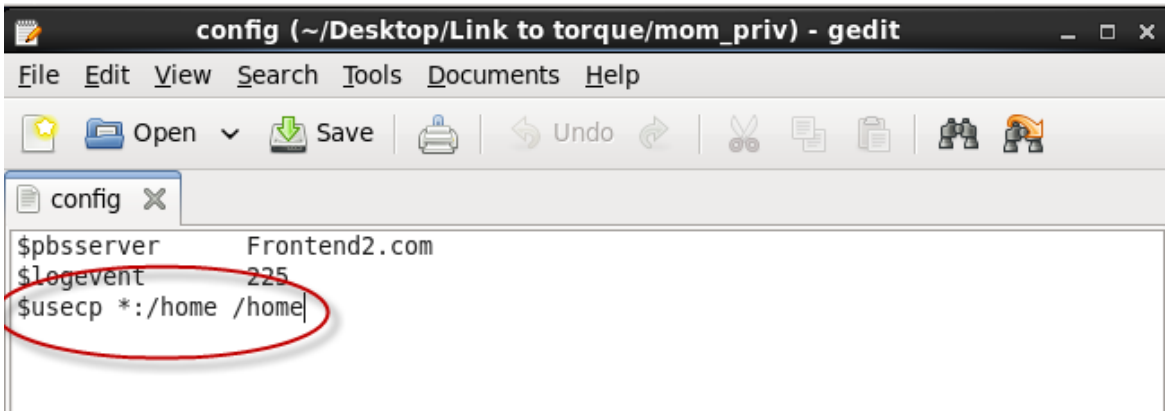


Figura. 25: *Contenido del archivo config contenido en la máquina de cálculo MOM por el que se especifica el nombre del host que aloja el server TORQUE, el nivel de registro del log de errores y el directorio utilizado por el parámetro usecp.*

Existe un parámetro en los scripts de ejecución PBS –W stagein= “ruta” y PBS –W stageout=”ruta” que lo que hace es copiar la lista de ficheros en el nodo de ejecución antes de que el trabajo empiece. Por lo que toda esta comunicación se debe generar a través de una copia remota de archivos por ejemplo utilizando scp. Si un sistema compartido de ficheros (NFS,DFS o AFS), se debe especificar en las directiva $usecp.(figura 25.)

Por lo que configurar estas directivas manualmente genera complicaciones que van en aumento en función al número de nodos MOM según se vayan agregando a la red de cálculo, por lo que se necesitaría ayuda externa por parte de los propios administradores de sistemas, inyectando en la arquitectura otras dependencias las cuales en un momento determinado volverían menos eficiente su gestión.

5.4.8.3 Ventajas ante el desarrollo de esta tecnología.

Las obtenidas de la propia virtualización, la portabilidad, la seguridad en situaciones como imágenes obtenidas a través de snapshots (instantánea tomada de un sistema en un momento determinado), de gran ayuda por la fiabilidad que ofrece para la restauración del sistema ante una eventualidad poco agraciada de una mala configuración, por lo que aporta una gran ventaja ante el ensayo de pruebas de diferentes arquitecturas y configuraciones.

La exportación de las máquinas virtuales se facilita en gran medida dada a su interfaz propia para poder trasladar los archivos.vdi (archivo que contiene a la máquina virtual completa) a diferentes soportes físicos con arquitecturas hardware diversas, por lo que se obtiene un alto grado de portabilidad, independiente de la arquitectura de soporte proporcionada por el hardware.

5.4.8.4 Inconvenientes encontrados.

Tras varios intentos de desarrollos con diferentes distribuciones de sistemas operativos (KUbuntu, Linux, CentOS, etc.) se han observado varias dificultades provenientes del tipo de virtualización ejecutado (virtualización completa).

Oracle VM VirtualBox realiza una virtualización completa, esto es así debidos a que la máquina virtual simula un hardware suficiente para permitir un sistema operativo “huésped” se ejecute sin ser modificado, pasando a ser

ejecutado de forma aislada, pudiéndose ejecutar al mismo tiempo muchas instancias del mismo.

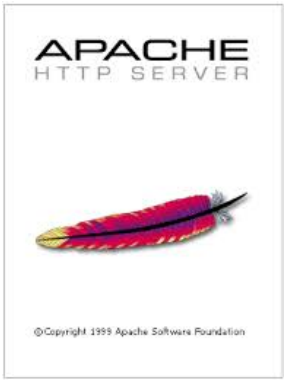


Figura. 26: *Imagen del servidor web HTTP de código abierto APACHE.*

En el momento de la puesta en funcionamiento, principalmente el inconveniente hallado ha sido en el instante del empaquetado, configuración manual de la instalación y posterior configuración de sistema, ya que con este modelo de virtualización desarrollado se debe tener una especial atención con la configuración de puertos de comunicación que permiten interconectar la máquina anfitriona (que soporta el entorno de virtualización) con recursos compartidos (dispositivos de red a través de switches, routers) así como la configuración de los puertos utilizados por los cortafuegos instalados en la máquinas virtualizadas.

De la misma manera que la asignación de los puertos debe adquirir un papel importante, no se debe dejar en un segundo plano la configuración del servidor web (Apache) que se debe instalar dentro de la máquina que va a alojar el servidor TORQUE (según la arquitectura diseñada) ya que es de vital importancia la excepcional configuración del archivo de configuración del servidor web (httpd.conf) en la máquina que soporte el portal web de autenticación de usuarios y envío de scripts hacia el Servidor. En esta instancia se ha optado para que la misma máquina que aloje el demonio pbs_server, albergue el portal web de autenticación y creación de scripts que se deben enviar al encolador (propio de TORQUE, MAUI ó Moab).

De esta manera se deja para futuras líneas de trabajo, la interacción de los usuarios-sistemas consumidores de TORQUE a través de un portal web

específicamente definido por el que los usuarios debidamente identificados en el sistema, tengan acceso al perfil de creación de scripts de ejecución o de direccionamiento para su envío a los recursos para los que estén autorizados según su rol, otorgando de este modo un nivel de abstracción sobre el sistema de tratamiento de procesos batch creado.

5.5 Instalación de TORQUE (sistema cluster de servidores).

En este apartado se va a describir punto a punto los pasos llevados a cabo para la instalación de TORQUE bajo la arquitectura de un clúster, en un principio bajo un entorno virtualizado para observar su evolución y desarrollo y así no asumir los peligros que conlleva implantarlo sobre un clúster que a día de hoy se encuentra operativo 100% en la UNED y en donde se han realizado las pruebas de envío, administración, ejecución y comunicación de diferentes ejecuciones de scripts diseñados al efecto.

5.5.1 ¿Qué es un cluster?.

Un clúster es un sistema de procesamiento de tipo paralelo o distribuido, que está formado de computadoras independientes, interconectadas entre sí, trabajando juntas como un solo recurso de cómputo intensivo, de ahí su gran potencia y gran nivel de encapsulamiento hacia los usuarios del mismo.

5.5.2 Elementos funcionales de un cluster.

5.5.2.1 Procesadores y tipologías.

Hoy en día es completamente factible la adquisición de un amplísimo catálogo de modelos de microprocesadores, dando cabida a cualquier modelo de arquitectura de los mismos (X86-64, AMD-XX, etc.). La tecnología actual de los procesadores de una maquina convencional nos da un rendimiento similar a los procesadores de una supercomputadora.

Otorgando cada procesador una diversa y amplia cantidad de memoria caché así como altas velocidades de procesamiento y sobre todo y de especial importancia, su bajo coste.

5.5.2.2 Interfaces de Comunicaciones.

Existen soluciones que necesitan pocos recursos económicos para interconectar los equipos que formarán parte del clúster. Se puede utilizar cualquier tipo de tecnología para la interconexión entre los equipos ya sea la utilización de redes, Myrinet ó Gigabit Ethernet. Con el que se obtiene un gran ancho de banda disponible para la comunicación y a su vez se consiguen bajas latencias.

5.5.2.3 Sistema Operativo.

Originalmente llamado NPACI Rocks, es una distribución de Linux para clusters de computadores de alto rendimiento. Fue iniciada por la NPACI y la SDCS en el año 2000.

Rocks se basó inicialmente en la distribución Red Hat Linux, sin embargo las versiones más modernas de Rocks están basadas en CentOS. Es una de las distribuciones más empleadas en el ámbito de clusters, por su facilidad de instalación y en lo referente a la incorporación de nuevos nodos de cálculo al cluster.

Se puede utilizar cualquier sistema operativo para la creación de un clúster sin embargo la implementación aquí presentada se va a realizar bajo Rocks en su versión 6.1 (Emerald Boa) con una serie de roles que se detallarán más adelante. Rocks clúster, está basado en el sistema operativo de CentOS junto al instalador anaconda el cual es muy útil por su gran ayuda a la hora de la simplificación de la instalación masiva de los nodos de cálculo componentes del clúster. La instalación se puede realizar de dos formas diferentes:

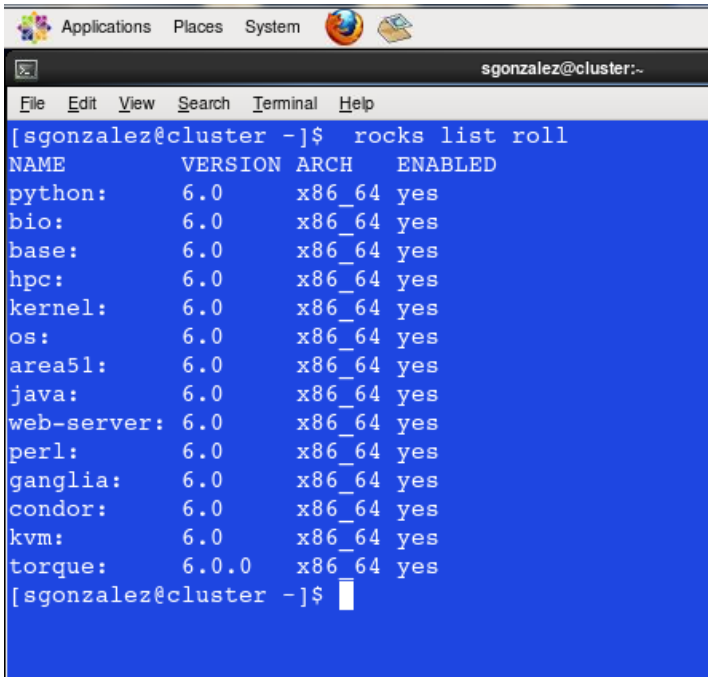


Figura. 27: *Listado obtenido a través del comando Rocks list rol en el cluster por el que se obtienen los roles instalados en el frontend.*

1) *De forma genérica ó estándar,* en la que se agregarán los componentes básicos para su correcto funcionamiento.
2) *De forma personalizada*, la cual, como su propio nombre indica, nos permite "especializar" a nuestro clúster, a través de la incorporación de roles.

En este proyecto y como queda reflejado en la figura 27, se han instalado varios roles, destacando entre todos ellos:

1. **Torque:** Que será el que aporte la funcionalidad de control, monitoreo, seguimiento y ejecución de los trabajos requeridos a la arquitectura del sistema.
2. **OS**: Sistema Operativo.
3. **Web-server**: Servidor web de Rocks cluster distribution.
4. **Kernel:** Núcleo del s.operativo.
5. **Java:** Java (sun).
6. **Ganglia**: Sistema de monitorización del clúster en conjunto y de cada nodo de forma individualizada.

La configuración en conjunto de todos estos roles y funcionalidades de las que está dotada la arquitectura otorga una tremenda dificultad en su configuración (como así quedó demostrado en el primer intento de desarrollo a

través del sistema heterogéneo de máquinas virtualizadas), por lo que se debe mantener una especial atención en el momento de su desarrollo.

5.5.2.4 Software complementario al sistema operativo.

Existe una gran cantidad de software que ya está preparado para funcionar en un clúster, desde la aparición de los procesadores con HiperThreading (HT) *[20],* la programación y la proliferación de software se ha desarrollado exponencialmente (todo aquel de gran exigencia en carga computacional, tratamiento de imágenes, juegos, gestión de servidores, comunicaciones etc,). Esta tecnología es capaz de desactivar los núcleos del procesador que se encuentren inactivos ganando así en eficiencia y aportando esta tecnología una mayor cantidad de posibilidades para las diferentes disciplinas científicas.

5.5.2.5 El factor humano y la escalabilidad del sistema.

El elemento más importante y a su vez el eslabón más débil de la cadena para el funcionamiento de cualquier sistema, corresponde al factor humano, que estando capacitado en la administración y manejo necesario de recursos proveerá de la herramientas de software necesarias para la generación de un ambiente más amigable para aquellos usuarios que pretendan utilizar el clúster, así como de su perfecta funcionalidad.

El clúster es fácilmente escalable a comparación de las supercomputadoras en donde la escalabilidad depende de una gran cantidad de recursos económicos. Con la facilidad de extender el clúster con equipos de bajo costo la escalabilidad no representa una gran limitación en el momento de agregar recursos necesarios para incrementar el poder de cómputo.

Existen además muchas herramientas en la actualidad para la administración y manejo del clúster, tanto en herramientas de monitoreo, así como de herramientas para la administración de trabajos y recursos (detallados en el punto 5.5.6 Configuraciones de interés).

El soporte en librerías para programación en paralelo están altamente desarrolladas, lo cual permite que la programación de nuevas aplicaciones que

puedan funcionar en multiprocesamiento sea pueda realizar de una manera bastante “amigable”.

5.5.3 Instalación del sistema operativo ROCKS.

Antes de proceder con la instalación del frontend es necesario asegurarse que las conexiones de la red externa y la red interna del clúster se hagan a la interfaz de red correcta. Rocks asume que la interfaz identificada como 'Eth1' por el kernel será aquella que está conectada a la red externa y la 'Eth0' a la red privada del clúster.

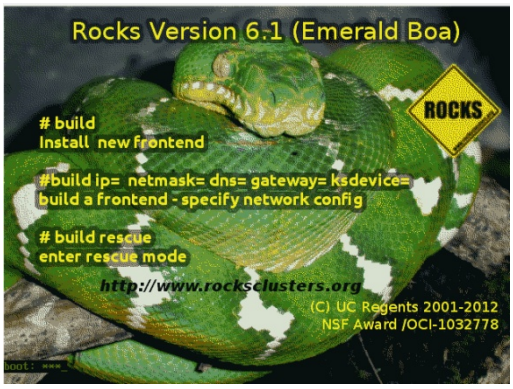


Figura.28: *Distribución de Linux empleada en la virtualización de un cluster (ROCKS v.6.1. Emeral Boa).*

Es importante esta puntualización sobre todo a la hora de su instalación a través de máquinas virtuales, ya que el orden dado por las interfaces de red del Oracle VirtualBox no corresponde con el gestionado y otorgado por Rocks lo que ocasionará problemas de alta complejidad a la hora de redirigir los tráficos de red (procedentes del exterior “Internet” y el generado en el propio clúster).

5.5.4 Instalación y configuración del frontend.

En un principio y tras haber descargado la versión apropiada del sistema operativo así como los .iso de los roles correspondientes de la página www.rocksclusters.org se debe insertar el DVD con la imagen del sistema operativo.

La figura 29 muestra las opciones que oferta el S.O., la intención es la de la instalación por lo que se debe teclear en el terminal “build”.

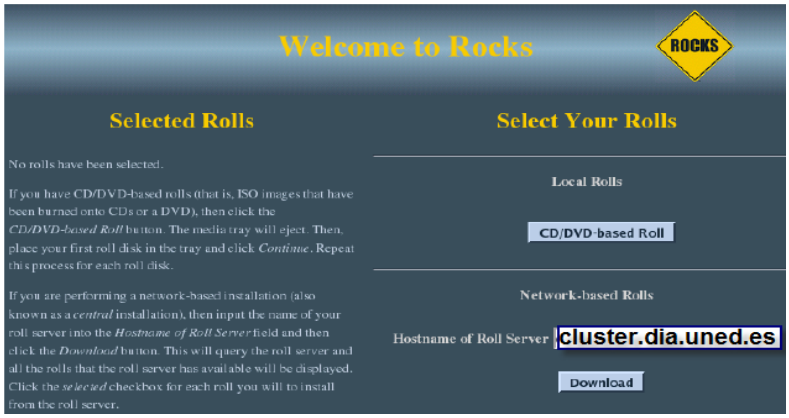


Figura. 29: *Pantalla de instalación ROCKS v.6.1. Emeral Boa.*

Tras la instalación del S.O. y la adición de los diferentes roles instalados, python, bio, base, hpc, kernel, os, area51, java, web-server, perl, ganglia, condor, kvm y TORQUE, se puede dar por finalizada la instalación del frontend.

Es importante puntualizar que al instalar el rol de TORQUE *no se debe instalar en paralelo el rol de Rocks denominado SGE (Sun Grid Engine)* puesto que al ser un encolador de trabajos propio del S.O. interfiere en el correcto funcionamiento de TORQUE, produciendo errores en las comunicaciones entre los nodos de cálculo y el server de TORQUE.

5.5.4.1 Problemas con las interfaces de red (Eth0 y Eth1).

El problema principal del sistema implementado, es que utiliza 4 interfaces de red. Las tarjetas son vistas por Linux como “dobles” de manera que generan conflictos para Rocks.

Rocks asigna por defecto la Eth0 a la red interna del clúster y Eth1 para la red externa. Por lo tanto se tiene que activar y configurar la Eth0 y la Eth1. Para ello se debe realizar lo siguiente:

En la carpeta etc/sysconfig/network-scripts se abren los ficheros de configuración de Eth0 y Eth1, en ellos es posible que se observe una línea referente a la etiqueta TYPE. Lo que se debe hacer es eliminar la entrada con la etiqueta TYPE del archivo si la había y poner al principio de los scripts :
TYPE=”Ethernet”

A continuación se abre un terminal y se escribiré System-config-network.

Una vez dentro se selecciona la interfaz 1 (Eth1) y se fuerza a que sea estática. A continuación se activa la interfaz Eth0 (que obtendrá la Ip estáticamente) y la interfaz Eth1 haciéndolas así operativas.

También se tiene que activar la casilla “activar cuando se inicie el ordenador” en las dos interfaces.

Por último hay que asegurarse de que las direcciones MAC (*media access control*, control de acceso al medio) de los dispositivos Eth0 y Eth1 se corresponden con las del sistema.

Se puede volver volver a reiniciar o ejecutar el “script-restarting”y se observará como ya funciona toda la red, Ganglia , ntp y además se puede acceder a internet.

Tras esta importante puntualización se procederá a la creación de usuarios, y opcionalmente aunque altamente recomendado es la creación de llaves privadas a través de ssh para una comunicación segura entre los nodos y el frontend.

5.5.5 Anexión de nodos de cálculo.

Una vez realizados los pasos anteriores se hace necesario reconstruir la distribución para que todos los nodos queden actualizados con los cambios de software realizados en el servidor hasta este momento, ejecutando el comando:
#cd /export/Rocks/install #Rocks create distro.

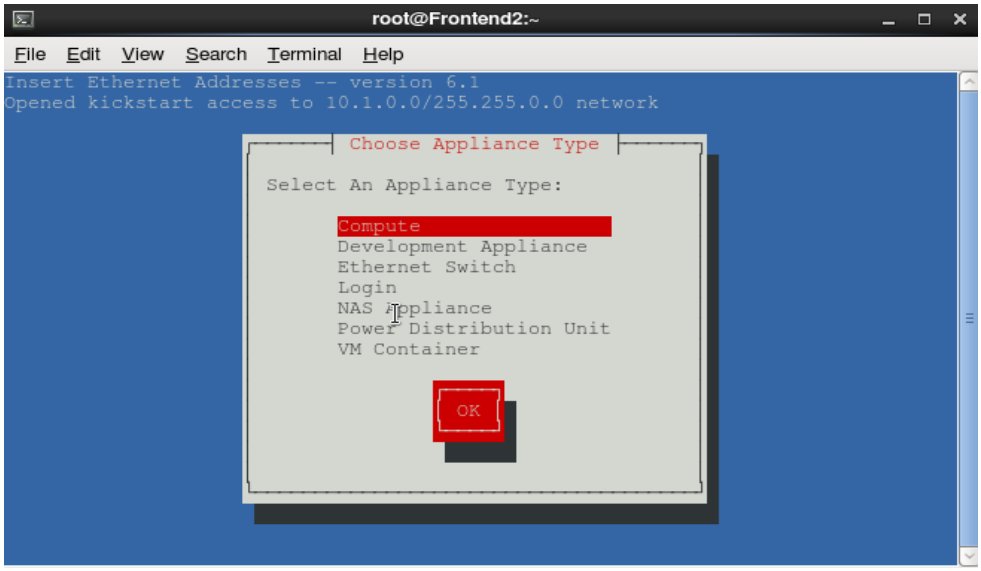


Figura. 30: *Pantalla de instalación ROCKS v.6.1. Emeral Boa, en la que se muestra las diferentes acciones a realizar (anexión de nodos al cluster).*

A continuación se estará en disposición de instalar los nodos de cálculo. Para hacerlo es necesario estar logueado en el frontend como *root* y ejecutar desde un terminal:

@ insert-ethers para agregar un nodo.

En este momento el sistema operativo se queda en escucha para que a través de la red detectar una a una las máquinas que van a formar parte del clúster.

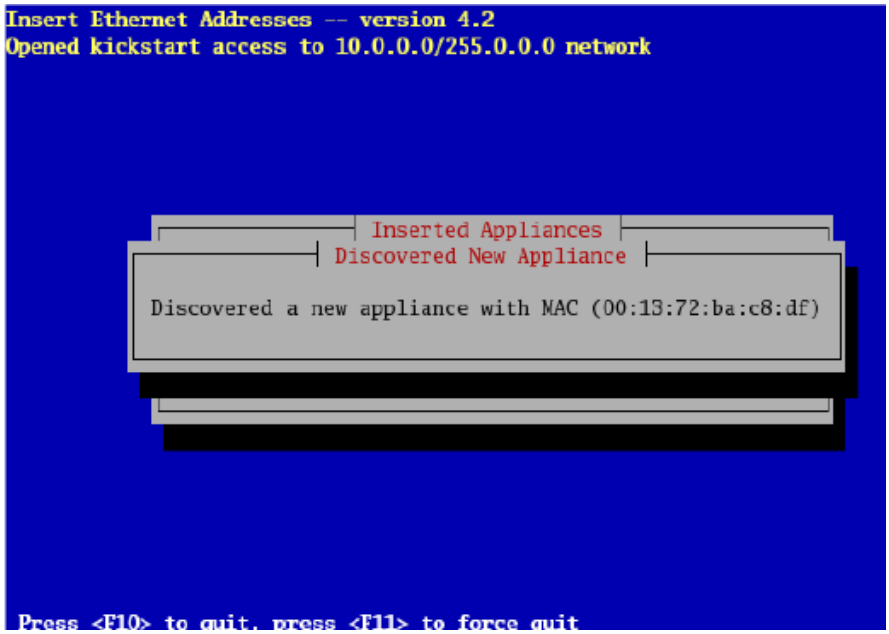


Figura.31: *Pantalla de detección de la dirección MAC de la tarjeta de red por la que se conecta al Frontend la nueva máquina.*

***Es importante puntualizar que el arranque configurado en la BIOS de estos futuros nodos será en primer orden a través del interfaz de red (PXE, network boot) **.*

Así será detectada la tarjeta de red del equipo que se desee agregar al sistema, ya que recibirá una señal DHCP, mostrándonos una pantalla como las que en la figura 32, siendo incluidas en la base de datos y actualizadas con la asignación de su correspondiente número de computadora en el clúster.

El asterisco indica la consecución del archivo kickstart y que el nodo ha empezado a adquirir los archivos necesarios procedentes del frontend, para ser añadido y configurado como parte del clúster.

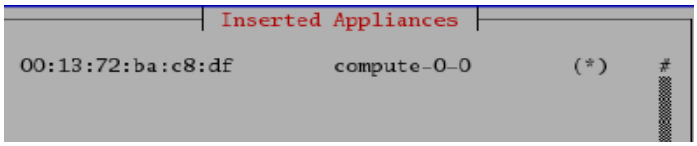


Figura. 32: *Imagen por la que se muestra la obtención del archivo kickstart por parte del Frontend y otorgado el permiso para la transferencia Frontend-Nodo de los archivos necesarios para la instalación del sistema.*

Se debe recordar, que una vez instalados todos los nodos se deben realizar sincronizaciones y actualizaciones en el clúster, todas ellas orientadas a que los archivos de configuración por defecto, se vean actualizados (usuarios, password, etc/passwd, /etc/group, etc.) y otros archivos que pueda ser de utilidad compartir a través del clúster (p.e. /etc/profile, /etc/bashrc, /etc/ld.so.conf, etc.).

A partir de este momento las instrucciones que amplían las funcionalidades del sistema operativo instalado se dejan un poco al margen ya que en la página www.rocksclusters.org se dispone de una amplia información detallada de los diferentes parámetros de configuración, así como de documentación específica de los mismos.

5.5.6 Configuraciones de interés.

En este apartado y aunque resulte algo discordante con lo expuesto en la sección anterior, se muestra la manera de complementar el clúster para ofrecer todos aquellos extras que junto con el rol de TORQUE harán más fácil su manejo.

Cambiar la dirección del mail de información del sistema tripwire.

Podemos designar las direcciones de mail donde queremos que tripwire nos envíe información diaria. Para hacerlo ejecutaremos el comando:

Rocks set tripwire mail address1 [address2] .

Por ejemplo, para enviar el mail a sgonzalez7@*alumno.uned.es* y *root*, el comando sería el siguiente:

Rocks set tripwire mail sgonzalez7@alumno.uned.es root@‘hostname‘.

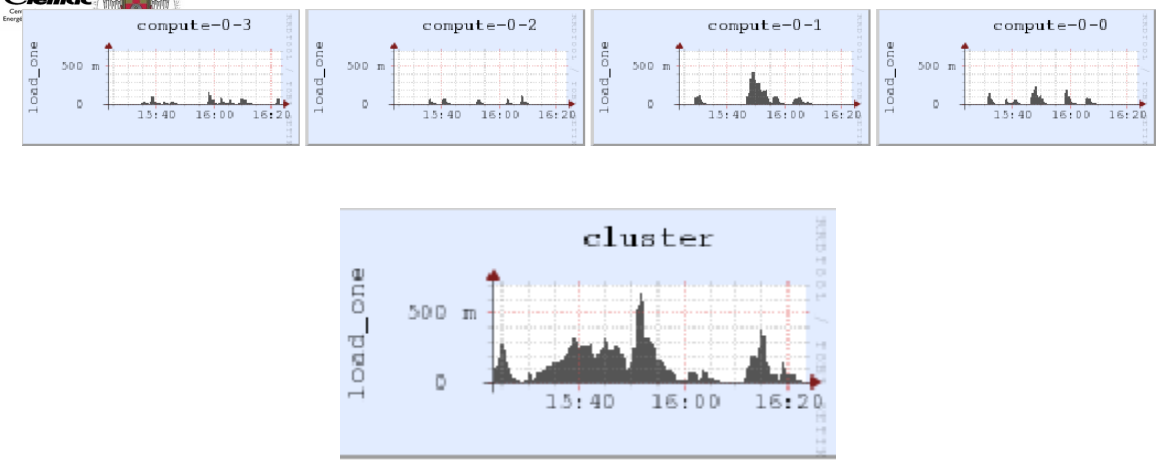


Figura.33: *Monitorización ofertada por Ganglia (Roll instalado en el Sistema Operativo) de recursos a monitorizar, siendo esta configurable*.

Con Ganglia se puede observar el estado del frontend en conjunto así como las diferentes opciones existentes para cada uno de los nodos que componen el conjunto, todas ellas configurables desde la propia aplicación.

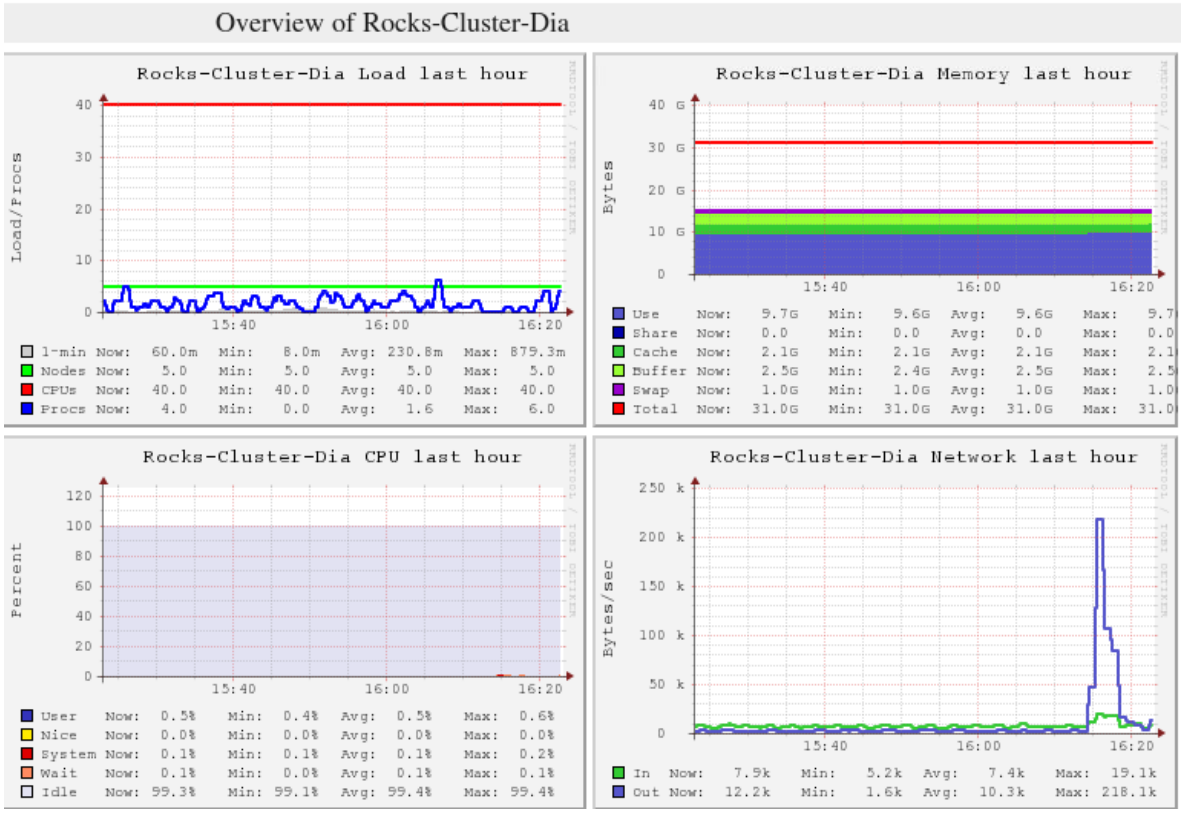


Figura. 34: *Otros recursos del sistema monitorizados por Ganglia.*

5.6 TORQUE en producción.

Como ya se comentó en apartados anteriores, en este punto se va a detallar paso a paso el despliegue del sistema procesamiento TORQUE que en este proyecto y tras haber sido desarrollado bajo diferentes entornos virtuales, como así ha quedado reflejado en secciones anteriores, está en modo producción en un frontend de la UNED llamado **cluster.dia.uned.es**

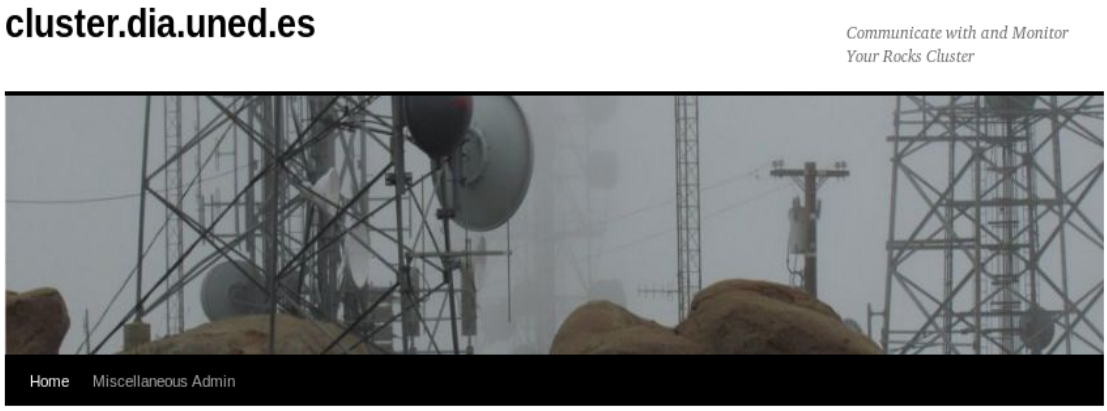


Figura. 35: *Imagen obtenida del sistema de navegación del Rocks Cluster (cluster.dia.uned.es).*

La configuración específica no varía de la instalación en un entorno virtualizado a excepción de algunos problemas detectados que se irán analizando en las secciones posteriores. Por tanto esta sección se centra en el funcionamiento y configuración específica para TORQUE como rol instalado a través de Rocks clúster, sin entrar en los detalles propios del despliegue que ya han sido desglosados con anterioridad.

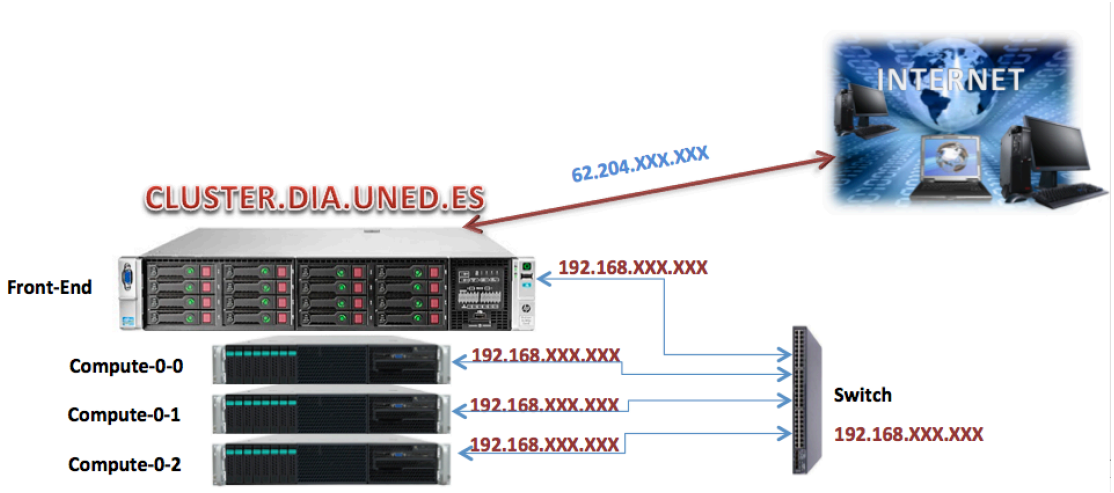


Figura. 36: *Arquitectura montada con TORQUE en el cluster de la U.N.E.D.*

Como se puede observar en la figura 36, el gráfico representa el despliegue montado en la U.N.E.D., diferenciándose claramente con la arquitectura descrita en la figura 14.

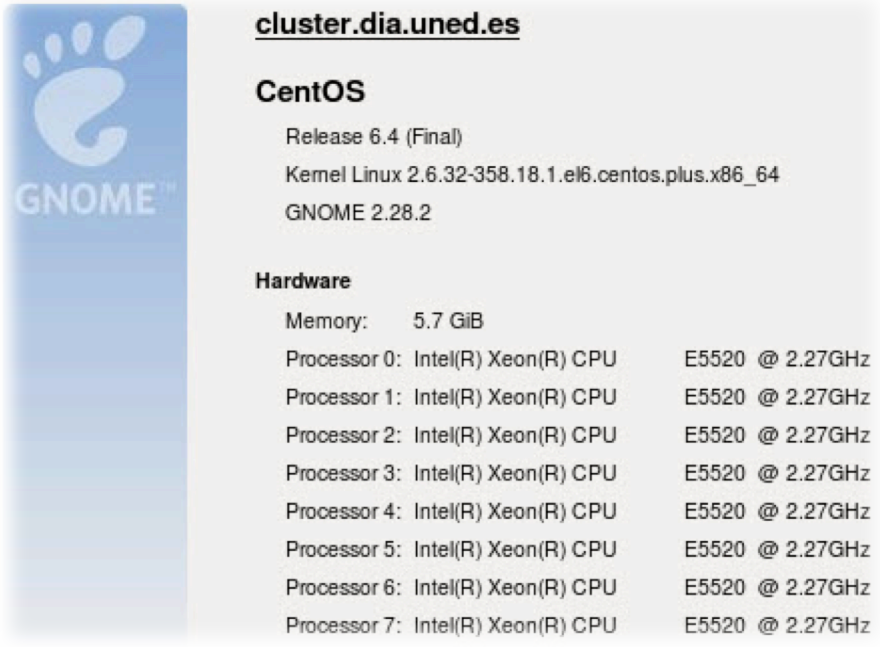


Figura. 37: *Hardware componente del cluster.dia.uned.es.*

5.6.1 Entorno de conexión.

La tecnología utilizada para poder realizar conexiones remotas al clúster de la UNED para evitar los desplazamientos al lugar de ubicación ha sido “NX Client for Windows”.

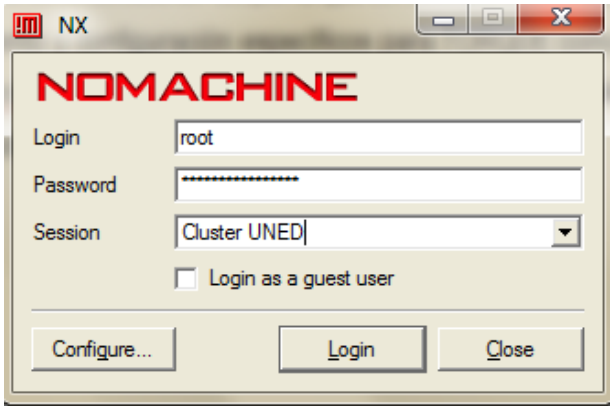


Figura. 38: *Pantalla de conexión e inicio de sesión remota a través de Nx Clients for Windows contra el cluster de la U.N.E.D.*

NoMachine utilizaba la licencia GNU General Public License para la tecnología del núcleo de NX, ofrecía un cliente libre y versiones de servidor para Linux y Solaris y clientes libres para Microsoft Windows, Mac OS X y sistemas embebidos a la vez que estaba ofreciendo una versión comercial no gratuita para las empresas.

NX es un programa informático del lado del cliente, que realiza conexiones remotas X11(conexiones bajo tecnología de ventanas) muy rápidas, lo que permite a los usuarios acceder a escritorios remotos de Linux o Unix incluso bajo conexiones lentas como las realizadas con módem. NX realiza una compresión directa del protocolo X11, lo que permite una mayor eficiencia que VNC. La información se envía mediante ssh, por lo que toda la información que se intercambian servidor y cliente está cifrada.Al cliente que se conecta al servidor NX se le considera cliente liviano.

Este tipo de conexión permite la administración y configuración del clúster desde diferentes lugares y en diferentes horarios otorgando disponibilidad total sobre la administración del mismo y con las medidas de seguridad en las

comunicaciones garantizadas ya que al realizar la conexión de manera visual como VNC, aunque lo realiza de una forma más rápida, ésta la realiza mediante ssh.

5.6.2 Usuarios creados.

Para la correcta interactuación y el correcto logueo por parte de los usuarios que estén debidamente autorizados tanto para el envío de trabajos al servidor de TORQUE, como para la administración del mismo e incluso para otras disponibilidades ofertadas por la plataforma multiusuario propias del sistema operativo, se deben crear usuarios con los perfiles que el administrador del sistema estime oportuno.

Para la realización de las pruebas del sistema implementado, se ha trabajado con dos usuarios creados para el envío y monitorización de los trabajos de TORQUE, que han sido sgonzalez y user1**,** ambos pertenecientes al mismo grupo, usuarios.

Se debe comentar que a la hora de la instalación de TORQUE hay que crear un usuario con privilegios de administración para el entorno de TORQUE independiente de los dos anteriores que corresponden a los que el sistema permitirá el envío de trabajos.

A través de torque.setup que utiliza el script de uso pbs_server –t create inicializa la base de datos del gestor serverdb , pudiendo añadir usuarios para administrar y operar con TORQUE. La sintaxis de utilización es la siguiente:

- ./torque.setup user
- Qmgr –c ‘p s’

Mediante la creación de los diferentes usuarios se pretende demostrar como la interactuación con múltiples usuarios actuando a la vez sobre el mismo gestor de trabajos, es independiente, siendo monitorizados los envíos, ejecuciones y seguimientos de los mismos de una manera completamente eficiente, durante toda la vida del proceso.

5.6.3 Configuraciones de colas y sus tipos.

Una de las posibilidades que ofrece TORQUE es crear nuestras propias colas (“queues”) según las necesidades de la arquitectura disponible. Estas colas permitirán al sistema (server) la distribución de los trabajos y la asignación

de recursos de hardware para el correcto y óptimo funcionamiento según las necesidades.

Estas colas pueden ser de dos tipos:

1) *De enrutamiento*.
2) *De ejecución.*

A continuación se detallan las características de cada una de ellas.

5.6.3.1 Colas de enrutamiento.

Una cola de enrutamiento dirigirá un trabajo a una cola de destino bajo la base de los atributos de trabajo y las limitaciones de la propia cola. Se establece mediante la creación de una cola de tipo queue_type "Ruta" con route_destinations. A través del comando qmgr, se crea la cola, definiendo su tipo, la cola destino, es decir el encaminador/es si se quiere que esté activa y si se desea que dicha actividad esté disponible des el mismo momento en el en que el servidor TORQUE se inicie.

```
qmgr

# routing queue
create queue route
set queue route queue_type = Route
set queue route route_destinations = reg_64
set queue route route_destinations += reg_32
set queue route route_destinations += reg
set queue route enabled = True
set queue route started = True
```

Figura. 39: *Creación de una cola de "enrutamiento" a través del comando qmgr.*

Una de las utilidades fundamentales del uso de este tipo de colas es el poder redirigir cargas de trabajo a sistemas externos (hablando en términos de ubicación física).

5.6.3.2 Colas de Ejecución.

Exactamente igual que en la sección anterior, la creación de las colas de ejecución se realiza a través del comando qmgr, lo que variará serán los parámetros específicos de configuración. El tipo que en la figura 38 se muestra

es de ejecución con unas restricciones como argumentos que darán "tipo" a la cola denominada "reg_32".

```
# queue for jobs using 16-31 nodes
create queue reg_32
set queue reg_32 queue_type = Execution
set queue reg_32 resources_min.ncpus = 31
set queue reg_32 resources_min.nodes = 16
set queue reg_32 resources_default.walltime = 12:00:00
set queue reg_32 enabled = True
set queue reg_32 started = True
```

Figura. 40: *Otorgación de diferente valores para los parámetros configurables de la cola batch denominada reg_32.*

Como continuación a estas creaciones, se pasa a detallar los elementos parametrizables a la hora de la creación de colas de ejecución o enrutamiento que nos ofrece TORQUE:

- **acl_groups** Sólo permitirá y admitirá trabajos enviados de los grupos de usuarios especificados.

```
> qmgr -c "set queue batch acl_groups=staff"
> qmgr -c "set queue batch acl_groups+=ops@h2"
> qmgr -c "set queue batch acl_groups+=staff@h3"
```

- **acl_group_enable** Determina si la restricción al envío de trabajos por parte de diferentes grupos de usuarios estará activa o no.

```
qmgr -c "set queue batch acl_group_enable=true"
```

- **acl_group_sloppy** Si se encuentra activa esta opción se chequearán todas las listas de las que el usuario que envía el trabajo es miembro.

- **acl_hosts** Comprueba la lista de los nombres de los hosts a los que les está permitido el envío de los trabajos hacia la cola en cuestión.

```
qmgr -c "set queue batch acl_hosts=h1+h2+h3"
```

- **acl_host_enable** Determina si la restricción del envío procedente de una lista determinada de hosts está activa o no.

```
qmgr -c "set queue batch acl_host_enable=true"
```

- **acl_logic_or** Si se encuentra activa realizará la operación lógica OR para la permisividad de usuario y host, permitiendo el envío del trabajo desde un usuario en concreto ó un hosts en concreto.

```
qmgr -c "set queue batch acl_logic_or=true"
```

- **acl_users** Especifica la lista de usuarios capacitados para el envío de trabajos.

```
> qmgr -c "set queue batch acl_users=john"
> qmgr -c "set queue batch acl_users+=steve@h2"
> qmgr -c "set queue batch acl_users+=stevek@h3"
```

- **acl_user_enable** Es importante destacar que si esta opción se encuentra activada, únicamente los miembros pertenecientes a la lista especificada de usuarios mediante acl_users estarán capacitados para el envío de jobs.

```
qmgr -c "set queue batch acl_user_enable=true"
```

- **disallowed_types** Especifica el tipo de trabajos que no les está permitido ser lanzados hacia esa cola de ejecución, por ejemplo arrays de ejecuciones.

```
qmgr -c "set queue batch disallowed_types = interactive"
qmgr -c "set queue batch disallowed_types += job_array"
```

- **enabled** Especifica si la cola está disponible para admitir o no trabajos.

```
qmgr -c "set queue batch enabled=true"
```

- **features_required** Especifica que todos los nodos de trabaja que pertenezcan a esta cola requerirán todas las características para las cuales hayan sido diseñadas, siendo las características sinónimos de cualidades.

```
qmgr -c 's q batch features_required=fast'
```

- **keep_completed** Específica el número de segundos en el que el trabajo una vez finalizado se debe mantener en este estado.
- **kill_delay** Especifica el número de segundos entre el envío de un SIGTERM y SIGKILL a un trabajo en una cola específica que desea cancelar.

```
qmgr -c "set queue batch kill_delay=30"
```

- **max_running** Número máximo de trabajos que son permitidos estar siendo ejecutados a la vez en dicha cola.

- **max_user_queuable** Permite el número máximo de usuarios que pueden mandar trabajos, incluyendo los activos, inactivos y bloqueados.

```
qmgr -c "set queue batch max_user_queuable=20"
```

- **max_user_run** Indica el número máximo de trabajos por usuario enviados y que son permitidos correr al unísono enviados por un mismo usuario.

```
qmgr -c "set queue batch max_user_run=10"
```

- **priority** Valor de prioridad asociado a cada cola.

```
qmgr -c "set queue batch priority=20"
```

- **queue_type** Tipo de cola “R” routing, “E” Execution.

```
qmgr -c "set queue batch queue_type=execution"
```

- **required_login_property** Añade la propiedad para el inicio de sesión, como requisito indispensable para todos los trabajos de la cola.

```
qmgr -c 's q <queuename> required_login_property=INDUSTRIAL'
```

- **resources_available** Especifica los recursos acumulados disponibles para todos los trabajos que se ejecutan en la cola. El comando qsub no permitirá la presentación de trabajos que solicitan más procesadores que los definidos para esa cola.

```
qmgr -c "set queue batch resources_available.nodect=20"
```

- **resources_default** Especifica los recursos de manera predeterminada para los trabajos enviados a esa cola.

```
qmgr -c "set queue batch resources_default.walltime=3600"
```

- **resources_max** Especificará los recursos máximos disponibles en esa cola para la ejecución de los trabajos.

```
qmgr -c "set queue batch resources_max.nodect=16"
```

- **resources_min** Especifica los recursos mínimos para los trabajos

```
qmgr -c "set queue batch resources_min.nodect=2"
```

- **route_destinations** Especifica los destinos potenciales de las colas para los trabajos enviados.

```
> qmgr -c "set queue route route_destinations=fast"
> qmgr -c "set queue route route_destinations+=slow"
> qmgr -c "set queue route route_destinations+=medium@hostname"
```

- **started** Indica si estará activa la cola en el momento del inicio del pbs_server.

```
qmgr -c "set queue batch started=true"
```

- **max_queuable** Número de trabajos máximos que la cola admite en espera, incluyendo en este total los que se encuentran ejecutándose, en espera y cancelados.

```
qmgr -c "set queue batch max_running=20"
```

Figura. 41: *Las figuras anteriormente plasmadas y debidamente explicadas una a una (texto en negro sobre fondo gris) corresponden a recortes obtenidos del manual de cluster resources de TORQUE.*

Todos estos atributos, anteriormente presentados no son de obligado cumplimiento, sino que están a nuestra disposición para dar un toque más personalizado a nuestro gestor de colas dentro del server, facilitando y optimizando su desarrollo y gestión, siempre en función a las necesidades que se planteen.

5.6.4 Recursos asociados a cada cola de ejecución.

Como tal los recursos son los especificados en los dos apartados anteriores, no habiendo entrado, en una primera instancia a especificar demasiados puesto que como se detalló en su momento, son de carácter

opcional y siempre deberán ir en completa armonía su definición con los requerimientos del sistema, para así no caer en la infravaloración ni como contrapartida sobredimensionamiento. Recuérdese que es uno de los puntos clave la optimización del sistema con sus recursos.

Como complemento, a continuación se muestran las colas creadas en el clúster de la UNED en dónde se ha realizado el proyecto.

```
[root@cluster ~]# qstat -Qf normal
Queue: normal
    queue_type = Execution
    total_jobs = 0
    state_count = Transit:0 Queued:-1 Held:0 Waiting:0 Running:1 Exiting:0
    resources_default.nodes = 1
    resources_default.walltime = 02:00:00
    mtime = 1364915713
    resources_assigned.nodect = 0
    enabled = True
    started = True
```

Figura. 42: *Características de la cola denominada "normal".*

```
[root@cluster ~]# qstat -Qf slow
Queue: slow
    queue_type = Execution
    total_jobs = 0
    state_count = Transit:0 Queued:0 Held:0 Waiting:0 Running:0 Exiting:0
    resources_default.walltime = 04:00:00
    mtime = 1364915713
    resources_assigned.nodect = 0
    enabled = True
    started = True

[root@cluster ~]#
```

Figura. 43: *Características de la cola denominada "slow".*

```
[root@cluster ~]# qstat -Qf fast
Queue: fast
    queue_type = Execution
    total_jobs = 0
    state_count = Transit:0 Queued:-4 Held:0 Waiting:0 Running:4 Exiting:0
    resources_default.walltime = 01:00:00
    mtime = 1364913739
    resources_assigned.nodect = 0
    enabled = True
    started = True
```

Figura. 44: *Características de la cola denominada "fast".*

Estas tres colas se han denominado:

- **Fast**: De tipo "Ejecución" y con un walltime de 1 hora. No se ha especificado el número de nodos que de manera inicial tomaría como recursos de cálculo (resources_default.nodes).
- **Slow:** De tipo "Ejecución" con un walltime de 4 horas. A destacar que la diferencia de configuraciones entre las colas en su parámetro walltime indica la prioridad de ejecuciones de los trabajos encolados en las mismas; a mayor walltime, menor es la prioridad (detallado en el punto anterior).
- **Normal:** De tipo "Ejecución" y el walltime de 2 horas. Todas ellas están habilitadas al arranque del sistema y activas (enabled and started = True).

5.6.5 Repaso de los comandos principales empleados.

Como en la sección anterior, 5.4.7., en la que se han detallado algunos comandos útiles para la correcta administración de TORQUE, sólo se pretende refrescar la utilidad de alguno de ellos, mostrando la funcionalidad específica de la que gozan en su implementación real en el clúster de la UNED.

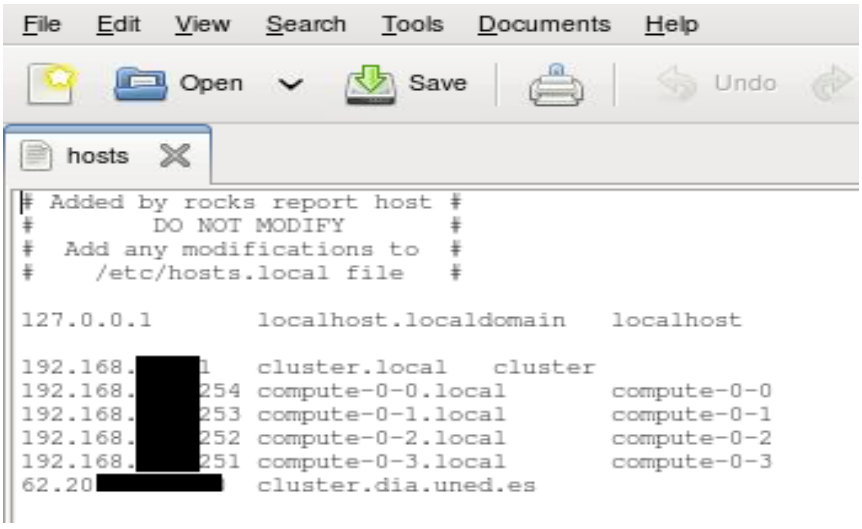


Figura. 45: *Contenido del fichero hosts, en donde se resuelven las direcciones de las máquinas identificadas en el sistema como MOM's.*

En primer lugar y tras ver la configuración del archivo etc/hosts del sistema, figura 43, se observa cómo están definidas las direcciones Ip´s de las diferentes máquinas que componen los MOM de cálculo de esta arquitectura, así como la dirección de salida a internet. Como se deduce, las direcciones Ip's corresponden, en una primera instancia a un rango correspondiente a una dirección pública de salida a internet 62.20.XXX.XXX y la otra a un rango de una red local 192.168.XXX.XXX. Estas últimas son las utilizadas por los diferentes MOM para comunicarse entre ellos y con el server de TORQUE todas encuadradas dentro de lo que se podría decir una misma red.

Uno de los puntos fuertes de este sistema es el cómodo aporte al sistema de recursos de cálculo para potenciar nuestro entorno.

Como se ha ido demostrando a lo largo de los capítulos, las posibilidades principalmente son bien mediante el aumento en el clúster de más nodos de cálculo, o bien mediante la expansión a otras redes y entornos debidamente configurados.

El pbs_server debe reconocer en la red, cuáles son los nodos de computación. Se debe especificar cada nodo en una línea en el archivo de nodos del servidor. Este archivo se encuentra en TORQUE_HOME / server_priv / nodos. En la mayoría de los casos, es suficiente para especificar sólo los nombres de los nodos en líneas individuales. Sin embargo, diversas propiedades se pueden aplicar a cada nodo e incluso que su identificación sea a través de la dirección Ip asociada. Por experiencia, hay que revisar exhaustivamente los parámetros de los cortafuegos y firewalls de todas las organizaciones implicadas en el esquema del sistema para que no corten, eviten o solapen puertos ya designados para el server de TORQUE.

```
compute-0-0
     state = free
     np = 8
     ntype = cluster
     status = rectime=1398700180,varattr=,jobs=391[499].cluster.dia.uned.es 404[
1].cluster.dia.uned.es,state=free,netload=3284808585,gres=,loadave=0.00,ncpus=8,
physmem=6113784kb,availmem=5653152kb,totmem=7137776kb,idletime=17212252,nusers=0
,nsessions=0,uname=Linux compute-0-2.local 2.6.32-220.13.1.el6.x86_64 #1 SMP Tue
 Apr 17 23:56:34 BST 2012 x86_64,opsys=linux
     mom_service_port = 15002
     mom_manager_port = 15003
     gpus = 0

compute-0-1
     state = free
     np = 8
     ntype = cluster
     status = rectime=1398700160,varattr=,jobs=,state=free,netload=1882515530655
,gres=,loadave=0.01,ncpus=8,physmem=6113784kb,availmem=5418432kb,totmem=7137776k
b,idletime=17211190,nusers=0,nsessions=0,uname=Linux compute-0-0.local 2.6.32-22
0.13.1.el6.x86_64 #1 SMP Tue Apr 17 23:56:34 BST 2012 x86_64,opsys=linux
     mom_service_port = 15002
     mom_manager_port = 15003
     gpus = 0


[root@cluster ~]# pbsnodes
compute-0-2
     state = free
     np = 8
     ntype = cluster
     status = rectime=1398700163,varattr=,jobs=,state=free,netload=888942662416,
gres=,loadave=0.00,ncpus=8,physmem=6113784kb,availmem=5476588kb,totmem=7137776kb
,idletime=17211539,nusers=0,nsessions=0,uname=Linux compute-0-1.local 2.6.32-220
.13.1.el6.x86_64 #1 SMP Tue Apr 17 23:56:34 BST 2012 x86_64,opsys=linux
     mom_service_port = 15002
     mom_manager_port = 15003
     gpus = 0

compute-0-3
     state = free
     np = 8
     ntype = cluster
     status = rectime=1398700200,varattr=,jobs=,state=free,netload=781071378705,
gres=,loadave=0.00,ncpus=8,physmem=8186740kb,availmem=7216440kb,totmem=9210732kb
,idletime=17210156,nusers=0,nsessions=0,uname=Linux compute-0-3.local 2.6.32-220
.13.1.el6.x86_64 #1 SMP Tue Apr 17 23:56:34 BST 2012 x86_64,opsys=linux
     mom_service_port = 15002
     mom_manager_port = 15003
     gpus = 0
```

Figura. 46: *Estado de los nodos de cálculo MOM`s (compute0-0, compute-0-1, compute-02 y compute-0-3) del sistema así como sus características principales.*

Nos ofrece una visión bastante completa de varios de los parámetros de cada una de las máquinas entre las que destacan, el estado, por el que se puede verificar si alguna de las máquinas sufre algún problema y está o no disponible. Los puertos de comunicación y administración de los MOM. Es importante destacar que aporta información sobre el número de procesadores reales de que disponen (np) para así, como se ha detallado con anterioridad,

crear las colas ejecutables con unos recursos acordes a las arquitecturas de las diferentes máquinas disponibles.

- **pbsnodes**: Administración y diagnóstico de cada uno de los nodos MOM
- **pbsdsh**: Inicia tareas en trabajos paralelos.
- **qalte**r: Modificación de los trabajos encolados.
- **qdel**: Borra cancela trabajos encolados.
- **gpumode:** Especifica un nueva forma utilización de GPU.
- **gpueset**: Resetea las GPUS.
- **qhold**: Bloqueo de trabajos encolados.
- **qmgr**: para administrar/crear/modificar colas.
- **qrun**:Comienzo de un trabajo
- **qrerun**: Re arranque de un trabajo.
- **qrls**: Lanzamiento de trabajo por lotes.
- **qstat**: Visualización de trabajos y colas.
- **qsub**: envío de trabajos.
- **qterm**:Shutdown de pbs_server
- **pbs_server**: lanzamiento del demonio server.
- **pbs_mom**: Reinicio del demonio MOM (nodo de ejecución).
- **XPBS:** Arranca un entorno gráfico de monitorización del SERVER TORQUE.

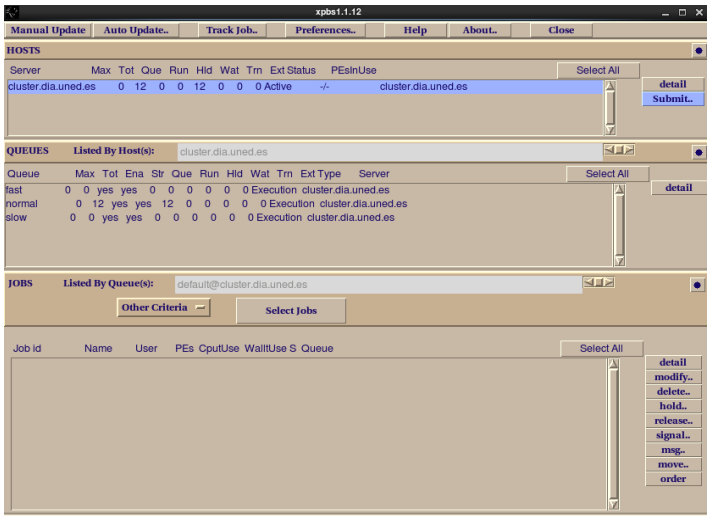


Figura. 47: *Imagen del gestor gráfico de TORQUE, invocado en línea de comando a través de la instrucción XPBS.*

5.6.6 Nodos de cálculo.

Como se ha detallado en la sección anterior y a través del comando pbsnodes, se ven los nodos de cálculo que componen el sistema, así como sus características (estado, puertos de comunicación, número de gpus, número de procesadores, etc).

De momento se disponen de de cuatro máquinas que forman parte del clúster y con visión en un futuro no muy lejano de ampliarlo a otras unidades de cálculo externas al clúster de la UNED.

Una ventaja que aporta la posibilidad de añadir máquinas externas al clúster actual es la facilidad con la que esta acción se puede llevar a cabo, creando y posteriormente instalando en los nodos nuevos de cálculo los auto extraíbles; torque_package_clients_linux
torque_package_momlinux,
pudiendo seguir expandiendo el sistema como bien ha quedado reflejado en el apartado 4.1.1.

5.6.7 Scripts de uso.

Para la comprobación del correcto funcionamiento del encolador y del gestor, se ha creado bajo lenguaje “C” un script de ejecución que lo que va a realizar básicamente es leer dos archivos de texto, debidamente formateados a tal efecto, línea a línea, creando otro de salida con el resultado de la suma de los números que estén contenidos en cada una de las líneas de los ficheros origen, independientemente del número de ellos.

```
**********************************************************************************************************************************************************
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define NUM_DATOS 100
int main()
{	FILE *archivo1;
	FILE *archivo2;
	FILE *archivoResultado;
	FILE *err_log;
	int caracteres[NUM_DATOS];
	int caracteres2[NUM_DATOS];
	int resultadoSuma[NUM_DATOS];
char tiempo[30];
	char extension[]=".txt";
	int i = 0;
	int dato = 0;
		for(i=0;i<100;i++){
		caracteres[i]=-1;
		caracteres2[i]=-1;
		resultadoSuma[i]=-1;
}	//abrimos y leemos primer fichero
	archivo1 = fopen("datos1.txt","r");
 if (archivo1 == NULL){
		printf("\nError de apertura del archivo1. Compruebe que existe/nombre/permisos \n\n");
		err_log = fopen("err_log.txt","a");
		fprintf(err_log, "%s","----->Error de apertura del archivo1. Compruebe que existe/nombre/permisos.\n");
		fclose(err_log);
		exit(1);
	}
	i=0;
	while (feof(archivo1) == 0)
	{fscanf(archivo1, "%d", &caracteres[i]);
		//printf("%d",caracteres[i]);
	i++;
	}
```

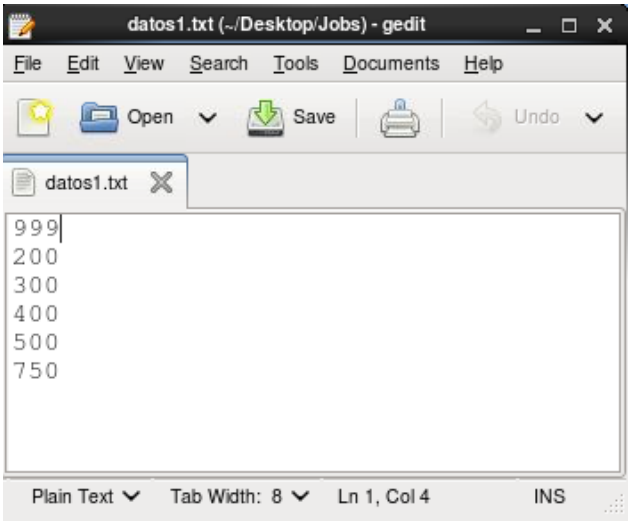
```
//printf("\nEl contenido del archivo de prueba es \n\n");
	fclose(archivo1);
	//abrimos y leemos segundo fichero
	archivo2 = fopen("datos2.txt","r");
	if (archivo2 == NULL){
		printf("\nError de apertura del archivo2. Compruebe que existe/nombre/permisos \n\n");
		err_log = fopen("err_log.txt","a");
		fprintf(err_log, "%s","----->Error de apertura del archivo2. Compruebe que existe/nombre/permisos.\n");
		fclose(err_log);
		exit(1);
	}
	i = 0;
	while (feof(archivo2) == 0)
	{
		fscanf(archivo2, "%d", &caracteres2[i]);
		//printf("%d\n",caracteres2[i]);
		i++;	}
	fclose(archivo2);
		for(i=0; i<NUM_DATOS && caracteres[i] != -1; i++ ){
		//printf("%d\n",caracteres[i]+caracteres2[i]);
		resultadoSuma[i]=caracteres[i]+caracteres2[i];
	}
		//abrimos y escribimos resultado primer fichero
		sprintf( tiempo, "%d", (int)time(NULL) );
		archivoResultado = fopen(strcat(tiempo,extension),"w"); // OJO SOBREESCRIBIRÁ EL FICHERO SI YA EXISTE
		if (archivoResultado == NULL){
		printf("\nError de creacion del archivo de suma. Compruebe que permisos \n\n");
		err_log = fopen("err_log.txt","a");
		fprintf(err_log, "%s","---->Error de apertura del archivo de suma. Compruebe que existe/nombre/permisos.\n");
		fclose(err_log);
		exit(1);
	}
	i = 0;
	for(i=0; i<NUM_DATOS && resultadoSuma[i] != -1; i++ ){
		fprintf(archivoResultado, "%d\n",resultadoSuma[i]);
	}
fclose(archivoResultado);

	//system("PAUSE");

	return 0;

}
```

Figura. 48: *Contenido del ejecutable para la realización de las pruebas de lanzamientos de trabajos realizado en lenguaje C.*

5.6.7.1 Ficheros de Datos.

Estos, como ya se ha comentado con anterioridad, son dos ficheros de texto que contienen números.

Datos1.txt :



Estas imágenes han sido capturadas directamente del clúster en donde se encuentra desplegado el sistema.

Datos2.txt:

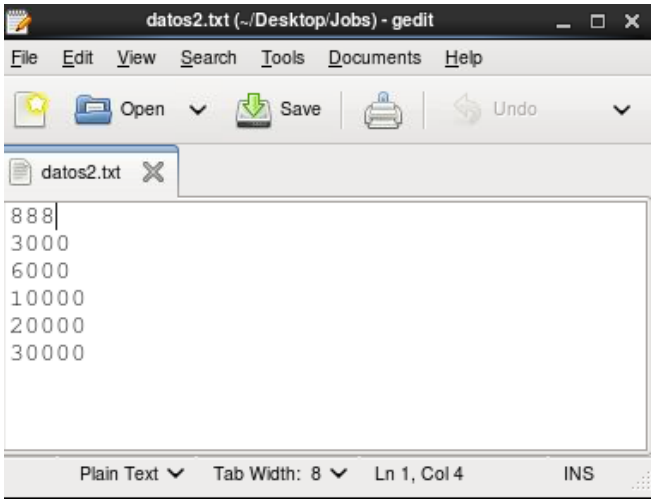


Figura. 49: *Contenido de los diferentes ficheros de pruebas empleados en las simulaciones.*

Suma2.c:

Este archivo corresponde al código que realiza la apertura, lectura, suma y creación del archivo resultado de la operación aritmética. Cabe destacar que para evitar la duplicidad de ficheros resultados o incluso que llamados a su

ejecución por diferentes usuarios que compartieran entorno, el script se encuentra debidamente diseñado para la evitar la pérdida de datos por ejecuciones paralelas de los mismos y diferentes usuarios. Está compilado en lenguaje C, pero admite cualquier fichero debidamente formateado y siempre que la máquina en la que está instalado el server tenga a su vez instalado el correspondiente entorno.

5.6.8 Script de ejecución.

Para proceder al lanzamiento de los trabajos (de naturaleza secuencial o paralela), a ejecutar hacia el gestor TORQUE, bajo línea de comandos a través de una consola y la debida conexión se realizaría de la siguiente manera:

$qsub-lnodes=<DESCRIPCION-NODOS>,walltime=<TIEMPO-LIMITE>
<SCRIPT>

- **DESCRIPCION-NODOS.**

Indica cómo distribuir su tarea entre los nodos disponibles en el clúster.:

-l nodes=4 -> un proceso en cada nodo

-l nodes=4:ppn=4 -> 4 nodos, 4 procesadores por nodo

-l nodes=2:ppn=2+2:ppn=4 -> 2 nodos de 2 procesadores y 2 de 4 procesadores

-l nodes=2:ppn=8+4:ppn=4+1:ppn=32 ->

- **TIEMPO-LIMITE.**

Un ejemplo de TIEMPO-LIMITE sería:

walltime=01:00:00 -> duración de una hora

walltime=03:00:00:00 -> duración de tres días

- **SCRIPT.**

Nombre del programa o *script* de descripción de su tarea. La manera más sencilla de crear este script es crearlo en el mismo directorio donde se encuentra su ejecutable. Un *script* básico podría ser:

```
#!/bin/sh
# Ver num MOM que se encuentran en el sistema:
NP=$(wc -l $PBS_NODEFILE | awk '{print $1}')
cd $PBS_O_WORKDIR
# Ejecutar la aplicación paralela
prun -n $NP ./<aplicación>.
```

Como se puede observar sería una manera muy básica de lanzamiento de un trabajo.

El script que se muestra a continuación y el cual se ha utilizado para la práctica, oferta de una manera más profesional la forma de lanzar un trabajo así como la parametrización del mismo.

script_suma2:

Sobre el mismo, no se realiza ninguna aclaración puesto que se encuentra debidamente documentado.

```
1  ##############################################################################
2  #                                                                            #
3  #                                                                            #
4  # Host        : cluster.dia.uned.es                          #
5  # Titulo      : Ejemplo de script de Ejecución para Torque          #
6  # Descripcion : Este script se utiliza para enviar a ejecutar programas     #
7  #               en batch por medio de Torque.                  #
8  #               El comando 'qsub' acepta parametros por linea de comando    #
9  #               o bien pueden definirse dentro de este script. Las lineas   #
10 #               que comienzan con '#PBS' son los parametros que tomara el   #
11 #               comando 'qsub', NO son comentarios.            #
12 #                                      #
13 #                                              #
14 # Ejecucion   : qsub script_suma2.sh                       #
15 #                                      #
16 ##############################################################################
17
18 ##############################################################################
19 #                 Asigna un nombre al trabajo                                #
20 ##############################################################################
21 #!bin/sh
22 #PBS -N suma_de_ficheros
23 #PBS -q fast
24
25 ##############################################################################
26 # Configura la cantidad de nodos a utilizar.                                 #
27 # nodes indica la cantidad de maquinas y ppn la cantidad de procesos         #
28 # por nodo.                                  #
29 # Si queremos solicitar 3 procesos con 1 por nodo.                           #
30 # EJ. -l nodes=3                                                             #
31 # Si queremos solicitar 8 procesos con 2 procesos por nodo.                  #
32 # EJ. -l nodes=4:ppn=2                                                       #
33 ##############################################################################
34
35 #PBS -l nodes=2:ppn=4
```

```
#################################################################################
# Deja la salida de 'standard output' y 'standard error' en diferentes     #
# archivos. Para que guarde todo en un unico archivo se define             #
# la opcion -j donde:                                                      #
# oe :  'std output' y 'std error' como output (*.o*)                      #
# eo :  'std output' y 'std error' como error (*.e*)                       #
#################################################################################

#PBS -o $PBS_JOBID/Recursos_utilizados.hard
#PBS -e $PBS_JOBID/Fichero.err

#################################################################################
# El 'qsub' envia un mail segun el/los evento/s definido/s:                #
#     a abort                                                              #
#     b begin                                                              #
#     e end                                                                #
#     n never                                                              #
#################################################################################

#PBS -m abe

#################################################################################
# Tiempo maximo de ejecucion del trabajo. cuando se cumpla el parámetro    #
# reflejado en walltime, si el trabajo no se ha ejecutado se matará el     #
# proceso encolado.     Los trabajos que declaren menor tiempo de walltime #
# tendran mayor prioridad.                                                 #
#################################################################################

#PBS -l walltime=00:03:00

#################################################################################
#               Mando ejecutar 8 veces seguidas el script                  #
#################################################################################

#PBS -t 1-8


#################################################################################
# Especificar direccion de e-mail donde se enviaran los mail               #
#################################################################################

#PBS -M sgonzalez7@alumno.uned.es

#################################################################################
# La variable PBS_NODEFILE es una variable de entorno del PBS, hace       #
# referencia al archivo que contiene la lista de nodos que se utilizara   #
# para esta ejecución dependiendo de la definicion anterior '-l nodes=...'#
# La variable NP guardara la cantidad de nodos elegidos                    #
#################################################################################

NP=$(wc -l $PBS_NODEFILE | awk '{print $1}')

#################################################################################
# Las  siguientes lineas imprimiran en el archivo de output los nodos que #
# fueron utilizados en el lanzamiento.                                     #
#################################################################################

echo "nodes ($NP cpu total):"
sort $PBS_NODEFILE | uniq
echo

#################################################################################
# La variable PBS_O_WORKDIR es una variable de entorno del PBS, indica     #
# el PATH absoluto del directorio de trabajo corriente                     #
# y es en este directorio donde quedaran los archivos de salida.           #
# con mkdir creo una carpeta con identificador unico correspondiente a la  #
# ID de cada instancia de ejecucion, solventando asi el que sobre escriban #
# los ficheros resultados por muchas ejecuciones se produzcan con los      #
# mismos ficheros y scripts fuente.
#################################################################################
```

```
107
108  cd $PBS_O_WORKDIR
109  mkdir $PBS_JOBID
110
111
112 ################################################################################
113 #                              Ejecuta el script suma                          #
114 ################################################################################
115
116  ./suma2
117
118
119 ################################################################################
120 #                            Muevo el archivo resultado suma                   #
121 #                            al directorio de ejecución                        #
122 ################################################################################
123
124  mv /home/sgonzalez/Desktop/Jobs/*.txt /home/sgonzalez/Desktop/Jobs/$PBS_JOBID/
125
126
127 ################################################################################
128 #                            Copio fichero de datos en el directorio           #
129 #                            creado para la ejecuación del                     #
130 ################################################################################
131
132  cp /home/sgonzalez/Desktop/Jobs/$PBS_JOBID/datos1.txt /home/sgonzalez/Desktop/Jobs/
133  cp /home/sgonzalez/Desktop/Jobs/$PBS_JOBID/datos2.txt  /home/sgonzalez/Desktop/Jobs/
134
135 ################################################################################
136 #                             Duermo la ejecución 90 segundos                  #
137 # para poder monitorizar su encolamiento, y lanzamiento, (todo ello desde      #
138 # una consola de un administrador de torque, ya que el usuario no tiene        #
139 # permitido por seguridad propia del gestor                                    #
140 ################################################################################
141  #sleep 90
```

Figura. 50: *Contenido del script de ejecución, el cual será interpretado y posteriormente ejecutado por el Server TORQUE.*

5.6.8.1 Ejecución del script.

Una vez explicadas las diferentes partes de las que se compone el sistema desarrollado, así como el desarrollo de su comprobación de funcionamiento, se va a proceder al lanzamiento por parte del usuario sgonzalez del script anteriormente detallado, mostrándose los efectos y resultados del mismo.

Se va a realizar de dos formas diferentes, a través de la conexión mediante escritorio remoto NX y otra a través de línea de comandos mediante conexión ssh.

En primer lugar se realiza la conexión utilizando el entorno, anteriormente explicado, *Conexión NX.*

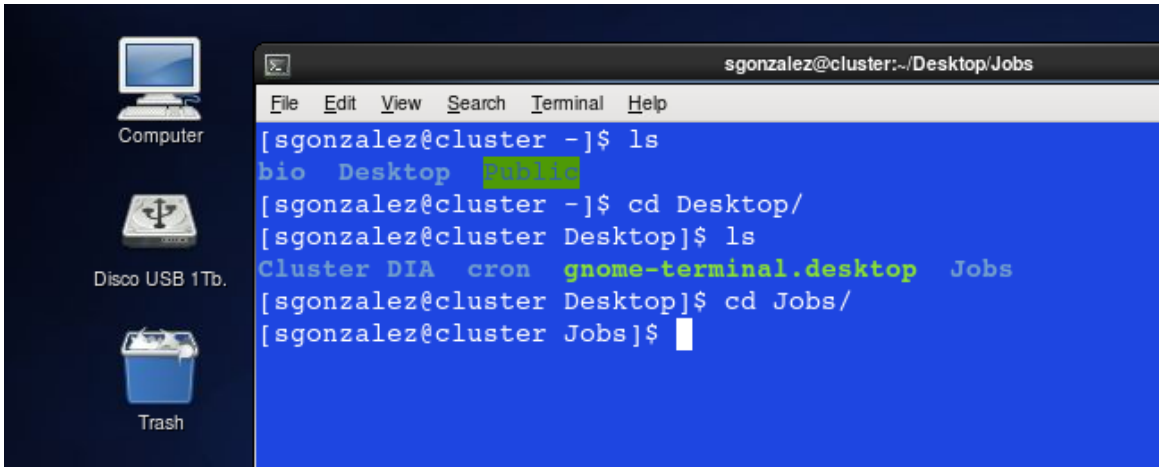


Figura. 51: *Imagen obtenida de una conexión realizada a través de NX y navegación a través de consola hacia el directorio de interés.*

Al efectuar la conexión y a través de una consola se realiza la navegación hacia el directorio en el que se encuentran los archivos necesarios (/Jobs).Una vez allí mediante la instrucción qsub [nombre_script], se lanza el trabajo a ejecutar**.**

El resultado de ello es que se inicia la ejecución del script dejando a TORQUE la gestión del trabajo encomendado. En este momento entran en juego la combinación de todo aquello que se ha definido dentro del script de uso (número de cpus, Gpus, cores, memoria RAM asignada como requerimiento mínimo/máximo del recurso, usuarios o grupos permitidos para la ejecución etc.)

```
[sgonzalez@cluster Jobs]$ qsub script_suma2.sh
405[].cluster.dia.uned.es
[sgonzalez@cluster Jobs]$ qstat -n

cluster.dia.uned.es:
                                                                  Req'd
Req'd   Elap
Job ID               Username    Queue    Jobname          SessID NDS   TSK    M
emory Time  S Time
-------------------- ----------- -------- ---------------- ------ ----- ------ -
----- ----- - -----
405[].cluster.di     sgonzalez   fast     suma_sgonzalez      --      2      8
  --  05:00 Q   --
    --
```

Figura. 52: *Comprobación del estado de los trabajos enviados para su ejecución al server TORQUE a través del comando qstat-n.*

A través del comando qstat -n se muestra como el trabajo ha sido enviado correctamente, le ha sido asignado un número de trabajo (JOB ID), el usuario

que lo ha enviado, el nombre con el que queda identificado el Job (este último es un parámetro configurable dentro del script de uso).

Es de destacar que, se observa que el JobID queda definido mediante 405[], indica que el trabajo con identificador asignado 405 es un array de ejecuciones, es decir que el mismo script de ejecución se ha ordenado su lanzamiento "x" veces, en este caso en concreto 8 (parámetro indicado en la figura como TSK).

Por cada ejecución y dentro de cada directorio donde se haya ordenado la ejecución se crean tantas carpetas como instancias de ejecución (ordenadas en el array). En este caso en particular corresponde al Job ID 307 [8] (ocho ejecuciones del mismo) por así estar definido en el script de ejecución.

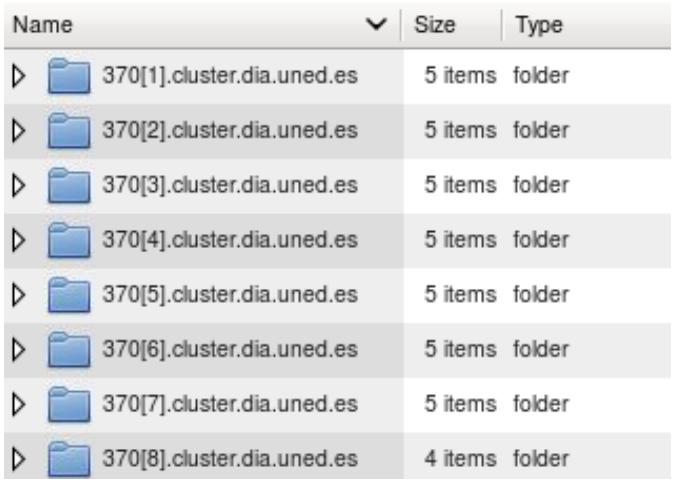


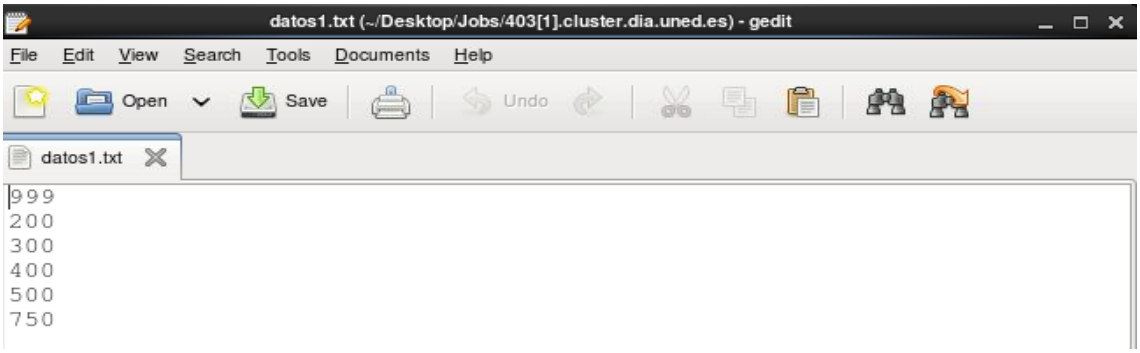
Figura. 53 : *Resultado obtenido de la ejecución del array de trabajos.*

A su vez se puede comprobar que según lo definido en el script a través del parámetro #PBS -m abe**,** el sistema a través que servicio de correo electrónico que tenga debidamente configurado y activo activo, realizará el envío de un email a la cuenta que tenga especificada dentro del script de ejecución cuando el trabajo aborte, comience y /o finalice (abort, begin o end).

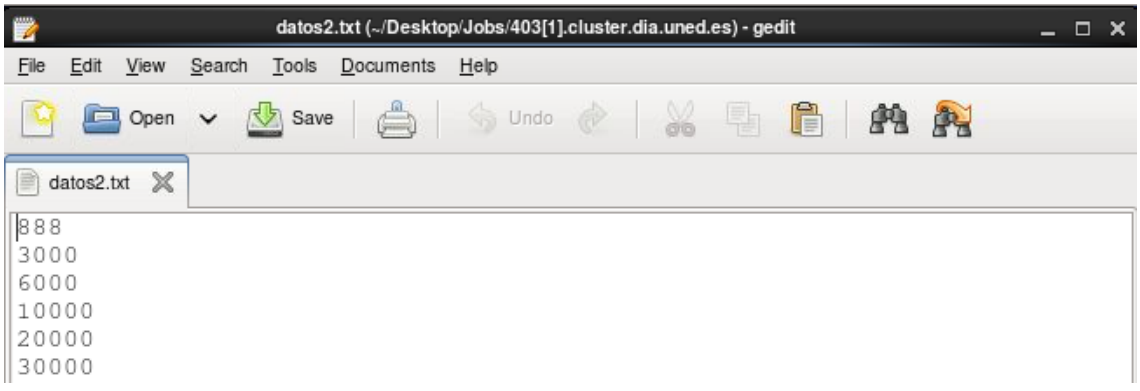
El contenido de cada carpeta son cinco archivos:

Figura.54 : *Resultado de cada ejecución del trabajo, con sus correspondientes datos de entrada, archivos de resultados, archivo de errores y de recursos empleados para la ejecución del mismo.*

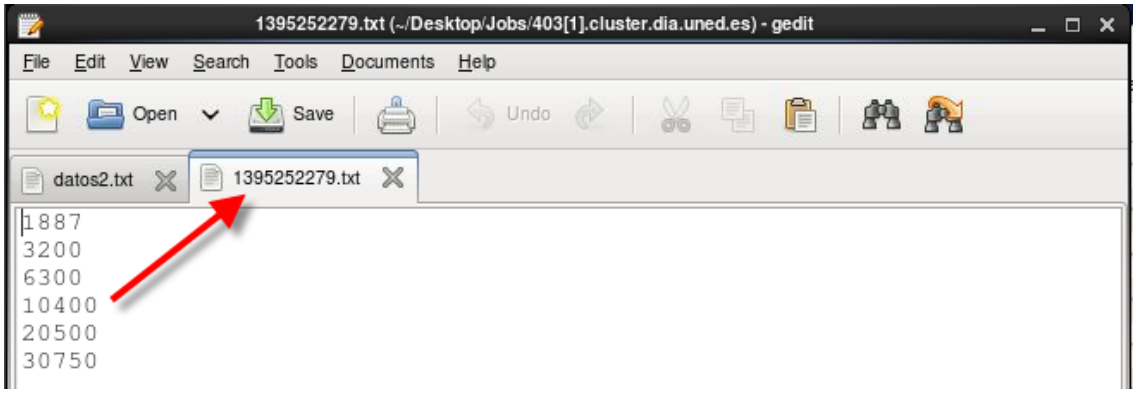
Datos1.txt: Primer fichero de origen de datos para el cálculo requerido.



Datos2.txt Segundo fichero de origen de datos para el cálculo.



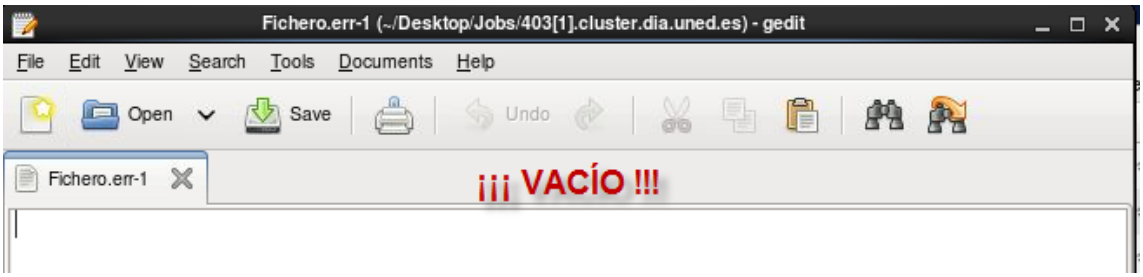
Resultado:



En este caso 1395252279.txt, Fichero que contiene el resultado de lo ordenado en el script. En este caso en particular al fichero se le ha otorgado el nombre en función a una conversión de la fecha del sistema en el momento en el que se lanzó su ejecución.

Fichero.err-1.txt

Fichero de Error: Guardará el resultado de los errores, en caso de que los hubiera, acaecidos durante la ejecución.

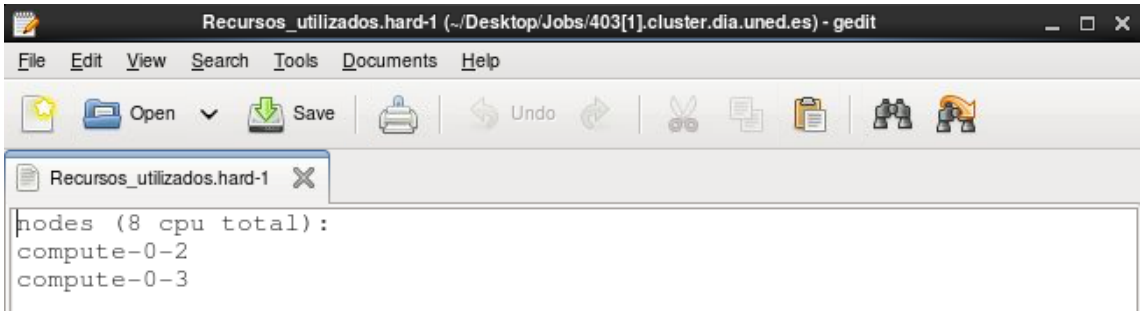


Recursos_utilizados.hard-1.txt:

Recursos utilizados.txt:

Breve reseña de los recursos de hardware que se han empleado para la ejecución del job.

```
echo "nodes($NP cpu total):"
sort $PBS_NODEFILE |uniq
echo
```



5.6.8.1.1 Conexión SSH.

Esta conexión, permite la realización de las mismas acciones que se han realizado a través de la consola gráfica de la misma manera, destacando que la conexión ssh a través de un gestor como puede ser putty, *[21]* crea una menor carga en la conexión porque simplemente es un entorno bajo consola.

A continuación se pasa a mostrar las imágenes obtenidas y correspondientes todas ellas a los resultados obtenidos con la ejecución de las mismas instrucciones como en el apartado anterior a través de NX.

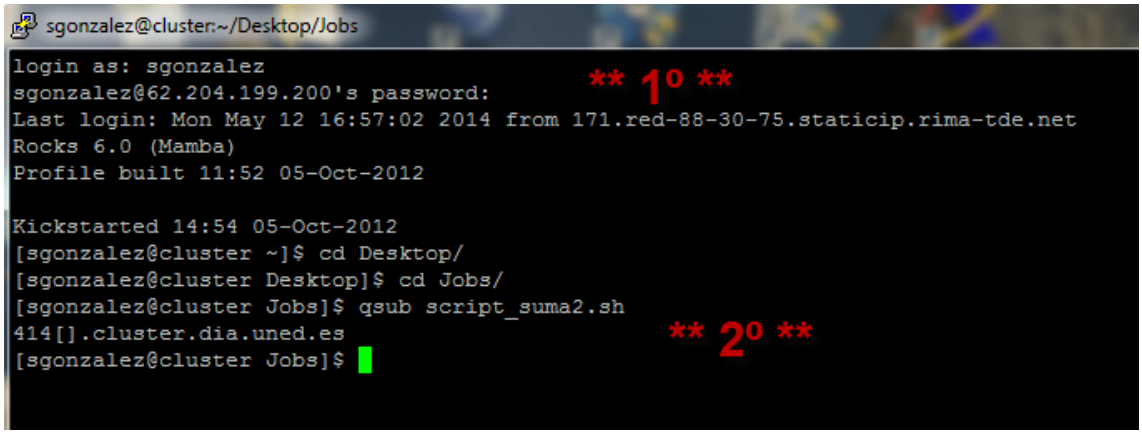


Figura.55: *Conexión realizada a través de SSH utilizando el software "putty", todo ello bajo un entorno de consola.*

En el primer apartado (en la figura 53 como "**1º")**, muestra los datos del logueo correspondiente al usuario que está debidamente autorizado para la realización de la conexión.

El segundo apartado (en la figura 53 como **"2º")** son las instrucciones necesarias para que el usuario navegue dentro del espacio para el que le está permitida la ejecución de sus jobs.

Más adelante en el apartado "líneas futuras de investigación", se detallarán las posibilidades al respecto.

5.7 Otras opciones a destacar.

Como se ha ido comentando a lo largo de toda la exposición de este proyecto, uno de los puntos fuertes de este gestor, es indudablemente las posibilidades ofertadas para su adaptabilidad, no sólo en arquitecturas sino en las diversas configuraciones propias como gestor de jobs, entre las que se destaca su potencia en la definición para cada usuario de su directorio de scratch , no teniendo la obligación, por parte del administrador del sistema, de la asignación obligatoria de un /home/user_torque, por cada usuario autorizado para lanzar trabajos, sino que simplemente le reasigna un "espacio" de intercambio en donde se le permite lanzar trabajos y posteriormente recogerlos.

Unas de las opciones configurables dentro de un script de ejecución de TORQUE son; #PBS –W stagein=file_list y #PBS –W stageout=file_list.

- #PBS –W stagein=file_list.

Esta opción copia los archivos en el host en donde se va a ejecutar el script, antes de que el propio trabajo comience su ejecución.

- #PBS –W stageout=file_list.

Esta opción copia los archivos desde el host en donde se ha ejecutado el script, una vez finalice la ejecución del script que le inició.

En general lo que viene a mostrar, es la puesta en escena de cuáles son los archivos se deben copiar en el sistema en el que se vayan a ejecutar antes de que comience el job de ejecución y que archivos se deben copiar una vez finalice la ejecución del mismo.

Lo denominado como **file_list,** será el path absoluto del archivo/s necesario/s para la ejecución del script de uso, independientemente de su ubicación (interna ó externa al propio sistema), siendo su sintaxis;

local_file@hostname:remote_file.

- local_file**:** Corresponde al nombre del archivo en el sistema en donde se ejecutará el trabajo.
- @hostname: Nombre del host o dirección IP.
- Remote_file: Corresponde al path completo, incluido el nombre, del archivo que una vez ejecutado el script, se generará.

  Un ejemplo de lo anteriormente expuesto sería:
- Stagein=archivo.entrada@cluster.dia.uned:/home/sgonzalez/Desktop/Jobs/archivo.resultado.
- Stageout=archivo.salida@cluster.dia.uned:/home/sgonzalez/Desktop/Jobs/archivo.resultado.

Cuando los trabajos se pueden presentar desde hosts diferentes, estas máquinas deben ser de confianza para el sistema a través de comandos tales como rsh y rcp (remote Shell y remote copy, respecivamente).

La confianza de envío y recepción entre diferentes máquinas es posible activarlo mediante la adición de los diferentes anfitriones en el archivo /etc /hosts.equiv de la máquina en donde se ejecute el demonio pbs_server.

La especificación exacta depende del sistema operativo para.

En la mayoría de los casos, la configuración de este archivo es tan simple como la adición de una línea al archivo /etc/hosts.equiv, como en el siguiente ejemplo:

/etc/hosts.equiv:

[+ | -] [hostname] [username]

cluster.dia.uned.es

node03 +

+ sgonzalez

Computenode-0-4

La permisividad para los accesos al host en donde corre el demonio pbs_server se puede especificar directamente sin utilizar la autenticación RCMD *[22],* estableciendo el parámetro submit_hosts en el servidor a través qmgr como en el siguiente ejemplo:

> qmgr -c 'set server submit_hosts = compute-0-0'

> qmgr -c 'set server submit_hosts += compute-0-1'

> qmgr -c 'set server submit_hosts += compute-0-2'

> qmgr -c 'set server submit_hosts += compute-0-3'

> qmgr -c 'set server submit_hosts += compute-0-4'

*** *El uso de* submit_hosts *es potencialmente vulnerable a un DNS spoofing, no debiéndose emplear fuera de zonas completamente confiables y debidamente controladas* ***.

Por lo que en caso de que se requiera que el sistema goce de un punto extra de seguridad, esta opción es de gran utilidad puesto que permite que el emplazamiento del server de TORQUE sea completamente distinto del entorno de trabajo, llamémoslo, máquina, servidor de ficheros o repositorio de archivos, en donde se permite subirlos para la ejecución de los scripts así como su recogida, puesto que se puede instalar un TORQUE en una única máquina, dedicada exclusivamente a la realización de cálculos, como pudiera ser un super-computador ó más correctamente hablando computadora de alto desempeño , ya que por definición las supercomputadoras son un conjunto de poderosos ordenadores unidos entre sí para aumentar su potencia de trabajo y desempeño.

Actualmente, la computadora de alto desempeño más rápida del mundo llamada **Tianhe-2,** se encuentra en China. Funciona a más de 33.83 petaFLOPS, esto se traduce a que puede realizar 1,000,000,000,000,000 de operaciones de punto flotante por segundo *[23].*

De lo expuesto en el párrafo anterior se puede deducir la importancia que se debe otorgar a la seguridad requerida para el acceso de usuarios a estas máquinas de alto rendimiento de ahí la importancia de esta opción de accesos a ficheros remotos.

6. ANÁLISIS E INTERPRETACIÓN DEL PROYECTO

Una vez dada una breve introducción a modo de barniz genérico de, alguna de las posibilidades ofertadas en el mundo de la supercomputación aplicada a entornos colaborativos y planteando como escenario de partida, la creación y desarrollo de este proyecto como un sistema, a la vez que eficaz, barato y completamente escalable, se ha desarrollado en las diferentes secciones que componen la documentación de este proyecto lo que en la justificación planteada al principio se detallada.

- Se ha conseguido un sistema potente de cálculo por lotes y ejecuciones en paralelo, completamente monitorizado, vigilado y controlado su estado, a lo largo de toda la vida de todos los procesos intervinientes;

1) Creación.

2) Envío.

3) Ejecución.

4) Finalización.

No permitiendo la aparición de situaciones en las que se pudieran producir posibles pérdidas de información y/o control, por el simple hecho del ordenamiento de ejecuciones individuales ó múltiples desde un mismo emplazamiento o distinto a éste, cosa que cuando se empezó a gestar el planteamiento de la realización de este proyecto, al comienzo de esta andanza, se esbozaba como el intento de la unificación de varias tecnologías que por separado permitirían la materialización de las diferentes fases de los procesos intervinientes (uso de servidores ftp, conexiones remotas a los servidores a través de consolas, software para rastreo de archivos y sus time lines, gestores de correos electrónicos, etc).

- Otra de las ventajas ha sido el poderlo llevar a cabo a través de entornos virtualizados, aprendiendo de esta tecnología y obteniendo un aprendizaje que sin él, hubiera sido muy difícil llevarlo a un entorno de producción en la máquina emplazada en el Departamento de Automática e Informática de la UNED.
- A su vez se ha cumplido la condición inicial de máxima austeridad puesto que todo lo utilizado, desde la versión de prueba y experimentación bajo

entornos virtualizados hasta el despliegue en el clúster físico ha sido software de distribución open source y software libre.

De acuerdo a la definición de la Free Software Foundation, el Software Libre es software que le da la libertad al usuario de estudiarlo, modificarlo y compartirlo ya que considera al usuario Libre. El "Free" o el "Libre" de la definición del Software Libre, según la FSF, tiene que ver más con la "libertad" de uso y no del "precio" del software. *[24]*.

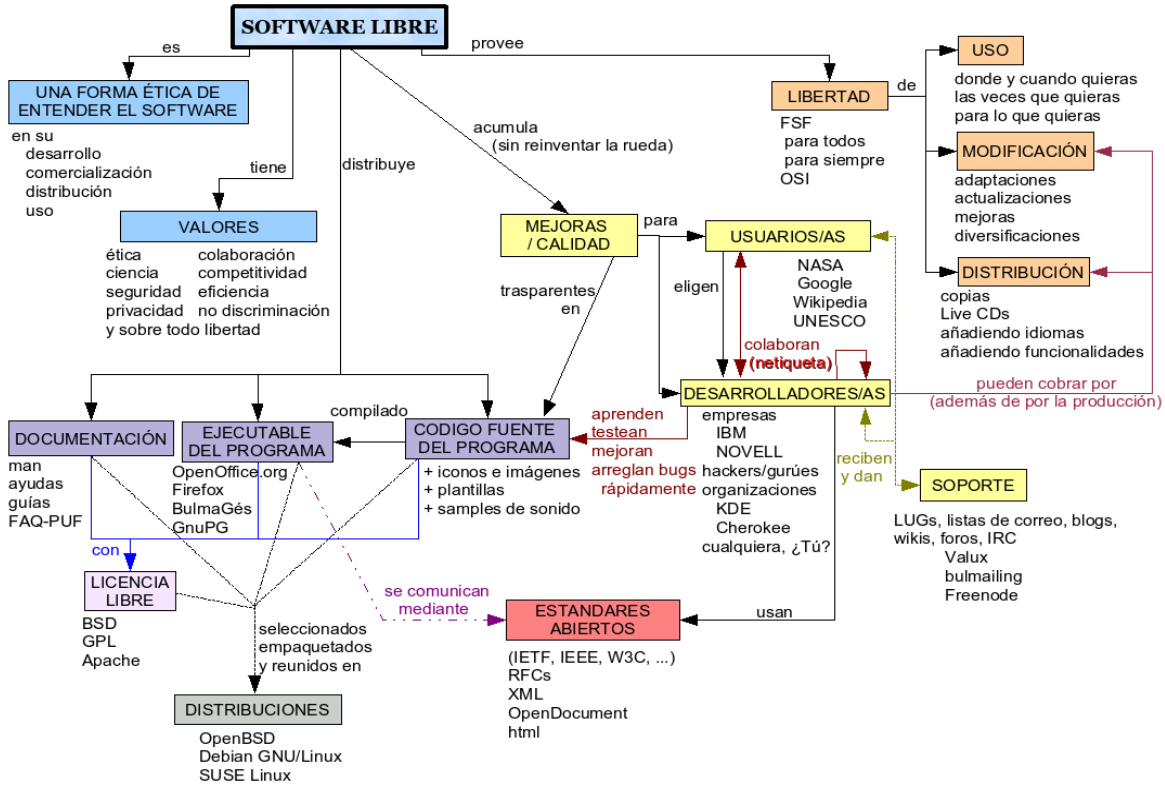


Figura.56: *Organigrama composicional, funcional y estructural de la definición de software libre según Free Software Foundation.*

O sea, no hay que confundir el término "Free" como que tiene que ser obligatoriamente "Gratis", por eso no hay que asociar el "Software Libre" con "Software Gratuito" ya que algunas veces no es lo mismo software.

El Código abierto (Open Source) tiene un punto de vista más orientado a los beneficios prácticos de compartir código que a cuestiones morales y/o filosóficas a las que apunta el Software Libre.

El concepto de Código abierto nació en el año 1998 para poder utilizarlo como reemplazo, en aquellos años, al ambiguo nombre del Software Libre.

Figura. 57 : *Logo distintivo de Open Source Initiative.* La **Open Source Initiative** (**OSI**, en español Iniciativa para el Código Abierto) es una organización dedicada a la promoción del código abierto. Fue fundada en febrero de 1998 por Bruce Perens y Eric S. Raymond.

Como se dijo anteriormente el término “Free” significa distintas cosas dependiendo el contexto en donde se lo emplee. Puede ser libre de uso pero no gratuito o puede ser gratuito y libre de uso.

Más que nada el término “Código Abierto” se refiere a la posibilidad de ver el código fuente del programa. Pero el concepto va mas allá del simple hecho de ver el código sino que la idea de que todos puedan ver el código fuente es que este evolucione, mejore y se desarrolle según las necesidades de los usuarios. Se corrigen los errores de manera mucho más rápida que si lo hiciera una empresa con software privativo y por ende una producción de software mejor. *[25]*

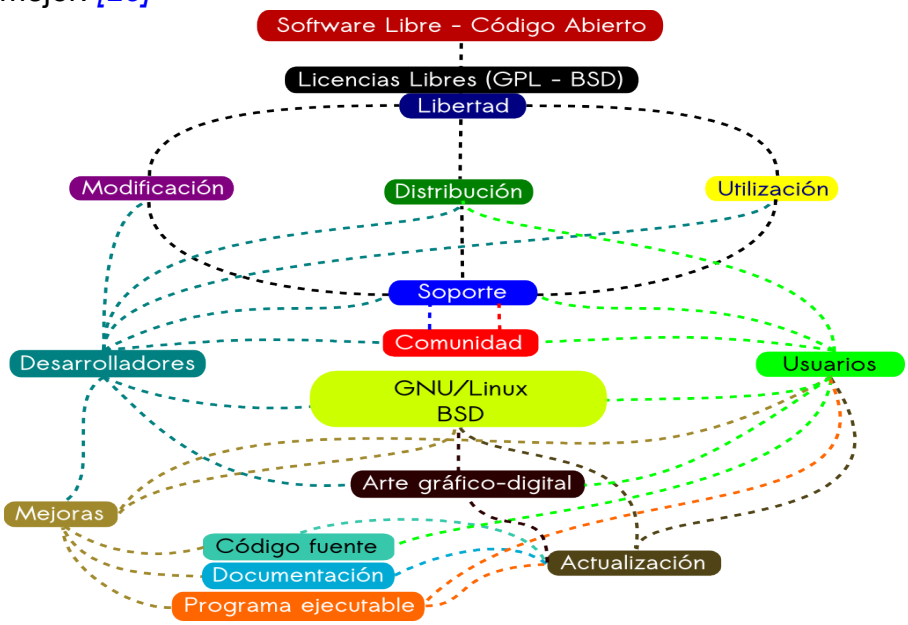


Figura.58: *Gráfico de interconectividad de definiciones que se encuadran dentro del marco de software libre-código abierto y licencias GPL-BSD.*

- Otro aspecto destacable ha sido el perfecto acoplamiento de TORQUE en una arquitectura no de uso específico para él, sino compartida con otros programas, usuarios y recursos de cálculo, viniendo a aportar un extra al aprovechamiento de recursos y por consiguiente de optimización.

7. CONCLUSIONES.

Tras el estudio y desarrollo aquí presentado, en el que se ha trabajado aplicando la tecnología de TORQUE (Tera-scale Open-source Resource and Queue manager) se ha llegado a las siguientes conclusiones.

- ✓ La curva de aprendizaje inicial objeto de la implantación de esta tecnología ha resultado ser de un alto nivel de coste en la obtención de resultados en función al tiempo invertido.
- ✓ Una vez obtenido el desarrollo correcto, la curva de aprendizaje pasa a ser de carácter exponencial a través de una representación resultados obtenidos-tiempo invertido.
- ✓ Gran facilidad de portabilidad de su instalación/configuración a través de entornos virtualizados.
- ✓ Posee una gran cualidad que es su fácil adaptabilidad a las diferentes distribuciones del sistema operativo Linux probadas en este proyecto (Ubuntu-server, Ubuntu desktop, CentOS y Red Hat, en sus versiones de 64bits).
- ✓ Es completamente transparente para el funcionamiento de los diferentes procesos intervinientes en los sistemas operativos, no desvirtuando ninguno de ellos.
- ✓ Gran aporte en el control y por consiguiente en la gestión de un trabajo desde que es creado hasta su finalización, pasando por todos sus estados.
- ✓ Descubrimiento de un gran campo de aplicación de esta tecnología en el área del cálculo computacional en entornos centralizados, distribuidos, homogéneos y heterogéneos, sin necesidad de arquitecturas de sistemas específicamente creados al efecto.

8. LÍNEAS FUTURAS DE INVESTIGACIÓN

Como futuras líneas de investigación para la ampliación de lo aquí expuesto, se debería dotar a la arquitectura implementada de un sistema adecuado de envío de los scripts de uso y ficheros que hagan la labor de parámetros de entrada para los cálculos requeridos, sin necesidad de que los usuarios debidamente autorizados, deban navegar a través de conexiones remotas por los sistemas en donde se despliegue TORQUE, proporcionando así otro nivel de abstracción al sistema y un entorno más amigable.

Otro punto interesante sería el acoplamiento de un encolador y gestor de trabajos como ***Maui*** para una mejora sustancial en el gestor de trabajos, dotando a la arquitectura aquí presentada de un punto extra de calidad.

Para mejorar a su vez el sistema de identificación de credenciales de usuarios y por consiguiente la ampliación de seguridad del entorno distribuido, se podría dar cabida a lo expuesto en la Tesis Doctoral de Dr. D. Rodrigo Castro Rojo *[26]*.

1. BIBLIOGRAFÍA.

- [1-2]C.m. Braamas y P.E. Stot. " Nuclear Fusion. Half a Century of Magnetic Confinement Fusion Research". Plasma Phys, Control and Fusion.
- [3]"En física nuclear, fusión nuclear es el proceso por el cual varios núcleos atómicos de carga similar se unen y forman un núcleo más pesado. Simultáneamente se libera o una cantidad enormedeenergía http://www.fnuc.es/estructura_grupos.php?grupo=CIEMAT.
- [4]**El Deuterio** es un isótopo estable de hidrógenos, de gran utilidad en la fusión nuclear por sugran sección eficaz de reacción.
- http://www-fusion.ciemat.es/New_fusion/es/Fusion/basica.shtml
- [5]***El Tritio*** un isótopo natural de hidrógenos, es radiactivo y de gran utilidadenlafusiónnuclearporsugransecciónеficazde reacción.http://www-fusion.ciemat.es/New_fusion/es/Fusion/basica.shtml.
- [6]***Espectrómetro de masas.*** Un espectrómetro de masas es un dispositivo que se emplea para separar iones dentro de una muestra que poseen distinta relación carga/masa.
- [7]***Plasma.***En física y química, se define plasma al cuarto estado de agregación de la materia, un estado parecido al gaseoso, poseyendo un número de partículas cargadas eléctricamente y no poseen equilibrio eléctrico.
- [8]J.D. Lawson, "Some criteria for a power producing thermonuclear reactor".
- [9-10] A.R. Bell, "Laser Produced Plasmas", Plasma Physics: An Introductory Course. Cambridge UniversityPress, Cambridge.
- [11]LymanSpitzer, astro físico creador del primer estellarator.
- [12]Primeras descargas con plasma.
- [13] Una **supercomputadora** o un **superordenador** es aquella con capacidades de cálculo muy superiores a las computadoras corrientes y de escritorio y que son usadas con fines específicos.

- [14]Como instancia de PBS_Server, se considera aquella ubicación en la que se encuentre desplegado el Servidor de TORQUE, para administración y monitorización de los Jobs a ejecutar.
- [15] - Networks Instruments, “Interop snapshot: virtualization deployments and hallenges top of mind for attendee”, 2009.

  Popek, Gerald J.; Goldberg, Robert.“Formal requirementsforvirtualizable third generation architectures”, Communications of the ACM, 1974, v. 17, n. 7, pp. 412–421.

  http://www.cs.berkeley.edu/~brewer/cs262/VM-requirements.pdfVMware Inc. How VMware virtualization right-sizes IT infrastructure to reducepowerconsumption,2008ahttp://www.vmware.com/files/pdf/WhitePaper_ReducePowerConsumption.pdfVMWare Inc. Reduce energy costs and go green with VMWare Green IT Solutions, 2010.

  http://www.vmware.com/files/pdf/VMware-GREEN-ITOVERVIEW-SB-N.pdf

  http://www.vmware.com/files/pdf/green_solutionbrief.pdfVMware Inc.
- WMware customersnapshot:IBM,

  http://www.vmware.com/files/pdf/customers/07Q2_ssvmw_ibm_english.pdf VMware
- WMware customer snapshot: Telefónica,2008b.

  http://www.vmware.com/files/es/pdf/08Q2_cs_vmw_telefonica_spanish.pdf
- [16] Oracle Virtual Box Technical Documentation

  https://www.virtualbox.org/wiki/Technical_documentation
- [17] ***Una arquitectura homogénea*** es aquella en la que la composición de sus elementos de hardware son idénticas, compartiendo así los mismos patrones. Esta definición es extrapolable a su vez al sistema operativo y elemento extras de software.
- [18] Gartner, Inc. Es la compañía de investigación y asesoramiento en materia de Tecnologías de la Información principal a nivel mundial.
- [19] ***HyperThreading*** , también conocido con H Technology) es una marca registrada para denominar su implementación de la tecnología Multithreading Simultáneo (SMAT). Permite a los programas preparados para ejecutar múltiples hilos(multi-threaded) en paralelo dentro de un único procesador, incrementando el uso de las unidades de ejecución del procesador. [wikipwedia.org].

- [20] La función **rcmd** es utilizada por el súper usuario para ejecutar una orden en una máquina remota usando un esquema de autenticación basado en números de puertos reservados. La función **rresvport** devuelve el descriptor de un enchufe (socket) cuya dirección cae dentro del espacio de puertos privilegiados. Los servidores utilizan las funciones **iruserok** y **ruserok** para autenticar a los clientes que solicitan servicios mediante **rcmd**
- [21] Definiciones e imágenes obtenidas, como complemento a las definiciones del software empleado. http://tuxfiles.wordpress.com/free-software-vs-open-source/