UNIVERSIDAD NACIONAL DE LA PLATA

FACULTAD DE INFORMATICA

Tesis presentada para obtener el grado de

Magíster en Ingeniería de Software

Integración de múltiples herramientas de Software Open Source para la Gestión y Análisis de Rendimiento de Clúster en Ambiente HPC

Autor: Esp. Leopoldo José RIOS

Director: Dr. Fernando G. TINETTI

INDICE GENERAL

Agradecimientos.

A mi esposa, compañera, colega y sponsor principal de esta experiencia. De no contar con su visión y paciencia, nada de esto hubiese ocurrido.

A mi director, un ser humano que supo apaciguar en mí la tentación de ir al código 'sin escalas' . Su pasión por el cálculo, me ayudó a recorrer un camino desconocido: componer material en el ámbito de la investigación y "hacer las cosas bien". Esta experiencia produce hoy satisfacción y mayor interés en la búsqueda de nuevos desafíos.

A mis padres, por permitirme ser parte del 7% de la población mundial, que confía en que el camino hacia el progreso, debe estar acompañado de sacrificio, preparación y visión social para intentar lograr un mundo mas justo.

1 INTRODUCCIÓN

En este capítulo se introduce el tema elegido, con la descripción de objetivos y metodología adoptada. Se enumeran trabajos previos tomados como referencia, y se brinda en detalle la estructura del documento.

1.1 Motivación.

En los últimos tiempos se ha experimentado un importante progreso en la construcción, integración y mantenimiento de software para el área específica de cómputo de altas prestaciones (o HPC, por las siglas en inglés de High Performance Computing). La industria en general y un importante número de actores varios, convergen en nuevos espacios dedicados a la investigación y mejoramiento de software, dirigidos especialmente a usuarios y organizaciones que requieren software de código abierto. Es posible mencionar a la fundación Linux [Linux], que ha propiciado la generación de un nuevo e importante proyecto denominado “OpenHPC” [OpenHPC], el cual abarca componentes de todo el ecosistema de software de HPC.

Un centro de computación de alto rendimiento, en muchos casos, utiliza software de código abierto para soportar el funcionamiento de toda su infraestructura. Los productos que emergen, es posible clasificarlos en: sistemas operativos, administradores de gestión remota, sistemas de archivos locales y en red, compiladores y librerías, módulos de autenticación de usuarios, gestores de trabajos del clúster, gestores de bases de datos, entre otros. Esta pila de software puede ser más o menos amplia en función al grado o calidad de servicio requerido, más aún si se trata de un ambiente crítico de computación de alto rendimiento. Existe en el mercado una importante oferta de paquetes de código abierto personalizadas que, en muy poco tiempo, permiten lograr un clúster básico y dejarlo en funcionamiento.

Una vez puesto en funcionamiento el clúster, se hace necesario mecanizar la gestión de los trabajos, enviados por los usuarios para su ejecución. Básicamente, el número de hosts de cálculo y el número de usuarios registrados, determinan la granularidad que se requiere para lograr eficiencia, planificación y gestión integral de los recursos. Un aspecto de importancia es conocer acerca del uso de los recursos del clúster, la generación y difusión de estadísticas; situación que se logra en parte con herramientas aportadas por la comunidad, y que en muchos casos es necesario aplicar modificaciones y adaptaciones.

De la interacción entre los componentes de software es posible obtener información de estado de los recursos, que es posible gestionar y poner a disposición para su difusión. El propio sistema operativo Linux ofrece importantes funciones a través de comandos de sistema operativo que aportan en este sentido. La aplicación que se desea construir a partir de la integración de herramientas de código abierto, permitirá gestionar recursos del clúster a través de interfaces web que no son parte de paquetes de distribución conocidos, y utilizará para su funcionamiento datos que surgen de la interacción misma de todo el sistema.

Las soluciones de software pagas disponibles en el mercado para la gestión de clúster HPC, similares a esta propuesta, hacen uso de esta idea. El sistema de información es alimentado con datos que provienen de varias fuentes: el sistema operativo, gestores de base de datos que registran la interacción de los usuarios con el sistema, y en particular, información provista por los contadores de hardware al momento de ejecutar programas de usuario, aspecto de utilidad para el logro de objetivos de rendimiento.

Las instituciones como Conicet y los centros de investigación que la integran, en gran número trabajan de manera vinculada y es común compartir experiencias en el uso de herramientas de software para HPC, y de los resultados obtenidos en sus procesos, aspecto que acompaña perfectamente la filosofía de código abierto. El hecho de desarrollar esta tesis y proponer un producto de software en esta modalidad, determina que el mismo podrá ser compartido a toda la comunidad científica y que ellas puedan hacer uso del mismo con total libertad [Conicet].

1.2 Objetivos.

El presente trabajo de tesis tiene como objetivo el diseño, desarrollo y evaluación de un software de gestión de clúster HPC a través de la integración de herramientas de software de código abierto. Se espera que la aplicación a construir permita programar y planificar los recursos computacionales para asegurar su disponibilidad, ejecutar programas de usuario en un ambiente controlado, almacenar los resultados logrados, y gestionar estadísticas de uso.

Objetivos específicos.

A partir del objetivo principal enunciado, surgen los siguientes objetivos específicos:

- En base al estado del arte determinar los requerimientos desde el punto de vista computacional, su interacción con el usuario, disponibilidad y resultados esperados y proponer una solución acorde.
- Investigar, probar e integrar herramientas de código abierto existentes que se adapten a la solución propuesta, considerando que existe una importante comunidad de usuarios en la administración de clúster HPC que pueden aprovechar este trabajo, tal es el caso instituciones públicas que llevan adelante proyectos de investigación, y que utilizan un clúster HPC para realizar cálculo numérico.
- Establecer una modalidad de acceso al sistema más segura por parte de los usuarios locales y externos.
- Desarrollar métricas específicas para el monitoreo de usuarios y software utilizado en la ejecución de trabajos y la obtención de estadísticas de uso para la elaboración de reportes e informes.
- Analizar la solución propuesta en un escenario real, y determinar la efectividad en cuanto al objetivo general, es decir, la facilidad de administración de ambientes HPC.

1.3 Metodología.

En base al problema y a los objetivos establecidos, la metodología adoptada es la siguiente:

- Estudio del estado del arte de herramientas de software que componen un clúster HPC.
- Formulación de las tareas a desarrollar por los usuarios, y su organización mediante roles.
- Identificación de datos a ser recolectados del sistema.
- Clasificación de la información a visualizar, organización en vistas según roles.
- Diseño y desarrollo de la plataforma web.
- Aplicación sobre el clúster del caso de uso.
- Análisis de resultados a través de la experiencia de usuario.
- Documentación.

1.4 Trabajos previos.

En la elaboración de esta tesis, se incorporan los siguientes trabajos:

1. Trabajo final para la obtención del grado de “Especialista en Ingeniería de Software” de la Facultad de Informática, Universidad Nacional de La Plata. En él, se describe el uso de la herramienta RRDtool [RRDtool], utilizada en la recolección de datos en los nodos de cálculo y su posterior visualización en reportes e informes [Sedici].
2. Publicación en el marco del 13th International Conference on Grid, Cloud, and Cluster Computing (GCC'17: July 17-20, 2017, Las Vegas, USA). En la cual, fue posible presentar y exponer esta propuesta.
3. V Jornadas de Cloud Computing & Big Data, desarrolladas en la Facultad de Informática de la UNLP, del 26 al 30 de junio de 2017. Trabajo “Acceso, Monitorización y Asistencia para el Desarrollo en Clúster para HPC: posible utilización en múltiples clústeres y clouds”. En esta oportunidad, la aplicación fue presentada como posible herramienta para gestión de recursos computacionales localizadas en cloud, para múltiples clústeres.
4. Publicación Año 2016, "Round Robin Data Bases for Performance Evaluation of High Performance Applications and Clusters", Fernando G. Tinetti, Leopoldo J. Rios, The 2016 International Conference on Grid, Cloud, and Cluster Computing (GCC’16), July 25-28, 2016, Monte Carlo Resort, Las Vegas, USA., Eds. Hamid R. Arabnia, Ashu M. G. Solo, Fernando G. Tinetti, ISBN: 1-60132-436-7, CSREA Press, pp 45-48.
5. Publicación Año 2015, “Instalación, Configuración y Evaluación de un Cluster HPC en una Nube Pública”, Fernando G. Tinetti, Leopoldo J. Rios, III Jornadas de Cloud Computing & Big Data, 29 de junio al 3 de julio de 2015, Fac. de Informática de la UNLP.

1.5 Organización del trabajo.

Este documento de tesis se encuentra organizado de la siguiente manera:

Capítulo 2: revisa el estado del arte de temas incluidos y de herramientas de código abierto disponibles, productos de software pagos del mercado, describe la problemática a resolver y los temas de investigación abordados en este trabajo.

Capítulo 3: describe el análisis de la propuesta de software, el diseño a partir de la herramienta seleccionada. Describe la organización de usuarios en grupos y se enumeran tareas a ser ejecutadas en función a roles establecidos. Describe las variables del sistema a ser monitoreada, las formas de gestión de datos y las posibilidades de difusión.

Capítulo 4: describe la propuesta de solución, el funcionamiento de la aplicación creada para usuarios del sistema, sus opciones de menú y secciones. Describe el monitoreo de trabajos y las formas de acceso al sistema.

Capítulo 5: enumera los resultados obtenidos, con información recogida de la interacción con los usuarios y directivos del clúster tomado como caso de uso.

Capítulo 6: brinda conclusiones acerca del trabajo, propone trabajo a futuro.

Lista Figuras: describe y localiza las figuras distribuidas en el documento.

Lista de Referencias: muestra el enlace de las referencias incorporadas.

Lista de Scripts: describe el contenido de cada script utilizado en la solución.

2 ESTADO DEL ARTE y PLANTEO DEL PROBLEMA.

Este capítulo revisa el estado del arte de aspectos que se consideran relevantes, se busca establecer el contexto y marco de trabajo para los siguientes capítulos.

Existe una importante variedad de recetas y fórmulas que permiten la instalación, despliegue y monitoreo de un clúster HPC. El andamiaje principal de un clúster HPC es sostenido en partes iguales por soluciones de hardware y software. Mientras el hardware se encarga del cálculo y de la interconexión y comunicación de los hosts, el software planifica, regula y monitorea su funcionamiento. De los productos de software que componen un clúster HPC, se necesita identificar y determinar los siguientes componentes:

- El sistema operativo se localiza sobre el hardware en forma directa. En el proceso de instalación del software, se reducen los módulos innecesarios, para lograr un funcionamiento holgado de los procesos más importantes de E/S y de acceso a redes. En todos los casos, los equipos tienen activada la funcionalidad de Hyper-Threading (equipos Intel) que utiliza recursos de procesamiento con mayor nivel de eficiencia, por lo cual pueden ejecutarse varios subprocesos en cada núcleo [Intel-ht]. En muchos casos se da que, en el host de gestión conocido como headnode del clúster, además de los módulos básicos, se instalen librerías adicionales, compiladores, generadores de impresión, gestión basada en web, módulos de autenticación, visualizadores, entre otros. El ambiente OpenHPC propone como plataforma operativa para el despliegue de sus distribuciones a CentOS74 y a SLES12 SP3 (Suse Linux Enterprise System).

  OpenHPC. Es una comunidad de reciente creación. Proporciona una colección de software precompilado común en HPC, es fundamentalmente un repositorio de paquetes. Los repositorios son de acceso público para su uso directo con los gestores de paquetes de Linux. Posee una orientación de construcción en bloques: los administradores pueden elegir paquetes de interés relevantes. OpenHPC proporciona recetas validadas para instalaciones de sistemas bare-metal, organizadas por elección de sistema operativo, arquitectura, componentes clave (proveedor, administrador de recursos).

- El sistema de procesamiento de trabajos en lotes (PBS) es un subsistema de red muy popular para enviar, supervisar y controlar cargas de trabajo organizadas en lotes, en uno o más sistemas. PBS posee una larga e importante historia, actualmente disponible en tres sabores:
  - OpenPBS: es el modelo PBS original desarrollado para la NASA a principios de la década de 1990, disponible como código abierto, todavía se encuentra disponible.
  - PBS Pro: una versión comercial de PBS de Altair Engineering. [PbsPro][Altair].
  - Torque: sucesor de código abierto de OpenPBS.
  - SLURM: es un sistema reciente de programación de trabajos y administración de clústeres de código abierto, tolerante a fallas y altamente escalable para clústeres de Linux grandes y pequeños. La comunidad OpenHPC adoptó SLURM, sin embargo utiliza una versión PBS como alternativa en sus recetas de instalación.

Altair Engineering ofrece el producto PBS Works, como un conjunto de tecnologías de computación basado en nube bajo demanda, más ampliamente utilizado para computación grid, cluster y on demand HPC. La suite integra las siguientes capacidades:

- Gestión de la carga de trabajo y programación de trabajos
- Presentación y seguimiento de trabajos basados en la Web
- Portal de visualización remota
- Portal de análisis y contabilidad
- Optimización de activos de software

De las características más sobresalientes, se pueden mencionar:

- Programación basada en políticas, basada en topologías
- Escalabilidad masiva a millones de trabajos (Jobs) al día
- Framework con integración de plugins potente y flexible para personalizaciones sencillas

El producto estrella de la suite PBS Works es PBS Professional, proporciona un entorno de computación flexible por demanda, permite compartir fácilmente diversos recursos de computación (heterogéneos) distribuidos, constituye la interface propia entre el usuario y el clúster. Pensada como arquitectura orientada a servicios, distribuye eficientemente cargas de trabajo a través de configuraciones de clúster, SMP e híbridas, escalando fácilmente a miles de procesadores y millones de trabajos. Según lo publican en su sitio web, una licencia de PBS Professional Floating Annual License asciende a u$s 18,00 por CPU Core. Se trata de una licencia anual por 365 días, el precio incluye el costo del software y soporte para el mismo período de tiempo [Altair-Price].

El producto “Compute Manager” se ofrece como un portal basado en web para el lanzamiento, monitoreo y gestión de trabajos simplificada. Los usuarios pueden configurar sus trabajos a través de PBS Professional utilizando “PBS Application Services”, monitorear y administrar cargas de trabajo y visualizar datos y resultados de forma remota. Se ofrecen funcionalidades de "arrastrar y soltar" con valores predefinidos, minimizar el esfuerzo necesario para escribir, modificar y probar secuencias de comandos -scripts- de aplicaciones complejas, el almacenamiento de scripts de uso común como plantillas. Sin duda, la facilidad en la edición de scripts de lanzamiento de trabajos, es una característica sobresaliente de este producto. Los requerimientos de instalación, para ediciones Linux son: SLES 10/11, RHEL 5/6 y CENTOS 5. Posee un sistema de configuración centralizado para definir usuarios y grupos, los accesos a las aplicaciones mediante perfiles de aplicaciones [AltairCM].

El producto “Display Manager” basado en JAVA, también parte de la suite, constituye un portal de visualización remoto de datos. Permite a los usuarios visualizar y colaborar grandes conjuntos de datos, consolidando de manera eficiente en gráficos y tablas, eliminando la necesidad de mover archivos fuera del centro de datos. Es posible resaltar que la funcionalidad de reportes, posee amplias opciones, que lo hacen una herramienta de excelencia; sin embargo, existe el detalle en la versión 13.1, que para instalaciones sobre Linux: solamente son aceptadas distribuciones ‘Enterprise’ de Red Hat o de Suse con librería GLIBC 2.3.2, con el agregado que se admiten únicamente interfaces gráficas Nvidia y ATI [AltairDM].

El producto “PBS Analytics” es ofrecido para tareas de visualización y análisis de datos, fácil de usar que soporta planificación y toma de decisiones basada en datos. Agrega datos de varios servidores en una base de datos común para obtener una vista consolidada de la productividad de los recursos. Constituye, desde el punto de vista de gestión de recursos, una funcionalidad que se debe poseer para la toma de decisiones y planificación.

El producto “Altair SAO” es una herramienta para medir y analizar el uso de aplicaciones para dimensionar las inversiones realizadas. SAO funciona perfectamente con gestores de licencias más populares para proporcionar vistas globales de activos de software, para que los administradores de la organización pueden mejorar la planificación de la capacidad.

La propuesta de Altair contiene productos de alta calidad comprobados en muchas partes del mundo, la forma de uso es en base a licencias que deben ser adquiridas en función a la cantidad de CPU/Cores a utilizar, y está claro que el entendimiento del modus operandi del licenciamiento requiere especial atención, dadas las múltiples opciones. Especial cuidado se requiere, cuando se debe contemplar el gerenciamiento de presupuestos basado en ciclos, y de cómo sea negociada la renovación de servicios y precios asociados. En el precio, según lo manifiestan en su sitio web, incluye soporte durante el tiempo de vigencia de la licencia, sin especificar el nivel o profundidad del soporte.

Torque PBS. Es una herramienta que proporciona control sobre trabajos por lotes y recursos informáticos distribuidos [Torque]. Producto de código abierto avanzado basado en el proyecto original de PBS, abarca mucho trabajo de una comunidad de programadores como del desarrollo profesional. A menudo incorpora avances significativos en las áreas de escalabilidad, confiabilidad y funcionalidad, y actualmente se usa en decenas de miles de sitios gubernamentales, académicos y comerciales líderes en todo el mundo. Es posible su uso, modificación y distribución libremente bajo las restricciones de la licencia incluida. Esta herramienta de software, como medida básica, identifica a los usuarios que darán uso al sistema, reconoce las capacidades computacionales con que se cuenta al momento de ejecutar los programas, organiza el inicio y la finalización de ejecución de aplicaciones de los usuarios y elabora informes sobre el uso de los recursos, entre otras tareas importantes.

Torque gestiona la ejecución de trabajos de usuario a partir de un concepto de colas FIFO -First In, First Out- donde ingresan los pedidos de ejecución de los usuarios, y a medida en que los recursos de computación de la cola de trabajo son liberados, da inicio a la ejecución de una nueva aplicación. Es posible poder comunicar mediante un mensaje de correo electrónico, el momento en que la ejecución de la aplicación es iniciada, y el momento en que finaliza.

Slurm. Trata de una propuesta reciente, diseñado a partir de nuevos paradigmas que permite un funcionamiento de mayor granularidad. Entre sus principales características es que no requiere modificaciones del kernel para su funcionamiento y es relativamente

autónomo. Como administrador de la carga de trabajo del clúster, Slurm tiene tres funciones clave. En primer lugar, asigna acceso exclusivo y / o no exclusivo a los recursos (nodos informáticos) a los usuarios durante algún tiempo para que puedan realizar el trabajo. En segundo lugar, proporciona un marco para iniciar, ejecutar y supervisar el trabajo (normalmente un trabajo paralelo) en el conjunto de nodos asignados. Finalmente, arbitra la disputa por los recursos al administrar una cola de trabajo pendiente. Su arquitectura, posee un administrador centralizado para supervisar los recursos y el trabajo. Puede ser configurado con un un administrador de respaldo para asumir esas responsabilidades en caso de falla. Cada servidor de cómputo (host) corre un demonio, que se puede comparar con un intérprete de comandos remoto: espera el trabajo, lo ejecuta, devuelve el estado y espera más trabajo. Estos demonios proporcionan comunicaciones jerárquicas tolerantes a fallas.
Este sistema ha sido pensado para una infraestructura de no menos a 20-30 hosts organizados en un número importante de colas de trabajo, dada la complejidad que se alcanza en tareas de programación y ajuste del clúster con ese volúmen. Slurm es software libre; puede ser redistribuido y/o modificarlo bajo los términos de la Licencia Pública General de GNU publicada por la Free Software Foundation; ya sea la versión 2 de la Licencia, o (a su elección) cualquier versión posterior. Slurm se distribuye SIN NINGUNA GARANTÍA; sin siquiera la garantía implícita de COMERCIABILIDAD o IDONEIDAD PARA UN PROPÓSITO PARTICULAR.

- El sistema de monitoreo del clúster. Ganglia es un sistema de monitoreo distribuido y escalable para sistemas informáticos de alto rendimiento, como clústeres y grids, se basa en un diseño jerárquico. Aprovecha tecnologías ampliamente utilizadas como XML para la representación de datos, XDR para transporte de datos compacto y portátil, y RRDtool para almacenamiento y visualización de datos. Utiliza estructuras de datos y algoritmos cuidadosamente diseñados para lograr muy bajos gastos generales por nodos y alta concurrencia. Su implementación es robusta, se lo ha incorporado a un conjunto extenso de sistemas operativos y arquitecturas de procesador, y actualmente se usa en miles de clústeres en todo el mundo. [Ganglia]

- Perfilado de aplicaciones. El objetivo de perfilar aplicaciones se asocia con lograr mejores niveles de rendimiento en el uso de aplicaciones de usuario de cálculo intensivo. El rendimiento es una medida de la cantidad de cómputo lograda por sobre los recursos utilizados y el tiempo. Perf es una herramienta de perfilado para sistemas basados en Linux 2.6+ muy básica, que abstrae las diferencias de hardware de CPU en las mediciones de rendimiento de Linux y es presentada como una interfaz de línea de comandos simple. [Profiling]. El diseño y funcionamiento de este componente introduce a perf como herramienta, sin embargo, es posible el uso de ‘gprof’, ‘PAPI’ u otra solución indistintamente.

2.1 El problema a resolver.

Por lo expuesto hasta acá es posible determinar los siguientes problemas que se intentará dar respuesta:

- Si bien existen productos competitivos en el mercado de software para clúster HPC, la comunidad de usuarios de código abierto, genera importante expectativa en torno al desarrollo de software basado en la experiencia compartida, la reusabilidad, y el no pago de licencias de uso. Las instituciones públicas que llevan adelante proyectos de investigación, y que utilizan un clúster HPC para realizar cálculo numérico, no siempre cuentan con suficiente presupuesto económico para solventar licencias onerosas, y que ellas, no siempre resuelven el 100% de la problemática.
- El software de monitoreo de recursos Ganglia, si bien describe mediante gráficas bien logradas, el uso de Memoria, CPU, Red, y Clúster en general, no incorpora métricas para el monitoreo de usuarios y software utilizado en la ejecución de trabajos. Para ello, será necesario desarrollar métricas específicas, y un procedimiento para su visualización por parte de usuarios y administradores del sistema, basado en tecnologías web-responsive.
- Acceso al sistema por parte de usuarios locales y usuarios externos a la institución. Por default, el acceso es mediante sesiones SSH para lograr interacción con el sistema. Esta manera tradicional es considerada insegura, y se propone una modalidad web, con autenticación basada en credenciales almacenadas en una base de datos, desligada del core del sistema.
- Organización de los recursos computacionales. Es necesario organizar los hosts de cálculo en colas de trabajo, bajo demanda de los usuarios, o por indicaciones de la administración. Es importante establecer las capacidades que se deben asociar a los distintos grupos de trabajo a generar, que el software generado por un usuario no impacte por sobre el de otros.
- Estadística. Se considera de importancia que cada usuario del sistema pueda generar estadísticas de uso del sistema, y utilizarlas en informes y reportes. Se prevé que cada usuario pueda contabilizar aspectos como tiempo de cpu, memoria utilizada, entre otros.
- Perfilado de aplicaciones de usuario. Los usuarios del sistema, tendrán a disposición, modelos de scripts para utilizar en el lanzamiento de aplicaciones en modalidad ‘profiling’, para descubrir su funcionamiento en asociación al uso de recursos y estrategias de comportamiento. Los resultados de los scripts de profiling, serán registrados en una base de datos como historial, disponible a los usuarios.

2.2 Temas de investigación abordados.

El presente trabajo integra varias áreas de conocimiento desarrolladas durante el cursado de la Maestría en Ingeniería de Software. Con el objeto de organizar la información en base a las áreas elegidas, los mismos se exponen de acuerdo a los siguientes temas:

2.2.a Base de Datos

En la propuesta de trabajo se analiza el diseño y despliegue de un sistema de información que tiene por objeto lograr la integración de herramientas de software de código abierto, para la organización y gestión de recursos computacionales de un clúster de alto rendimiento. Esta idea, supone la gestión de datos que provienen dinámicamente de múltiples fuentes, y que requieren ser transformados a lo largo del tiempo, y registrados

para su posterior difusión. Los datos identificados que se requieren almacenar para el objeto de gestión, son:

a) Gestión de usuarios. Las cuentas de usuarios de este sistema serán registradas en tablas de una base datos relacional, incorporando parámetros necesarios para la identificación de usuarios y de los permisos asignados a las funcionalidades del sistema.
b) Contadores de Hardware. Al usuario que utilice el sistema, se le proveerá asistencia para el despliegue de "profiling de aplicación", a los efectos de conocer en grado de detalle, cómo funciona el programa que desea ejecutar. A medida que el usuario prueba el código, será posible registrar los parámetros utilizados en cada caso, para usarlos como historial.
c) Estadística. Tanto para usuarios de administración del clúster, como para usuarios finales, estará disponible una herramienta para la registración de datos de series numéricas, necesarios para informes de uso de los recursos computacionales del clúster.

En este trabajo, se integran dos herramientas de gestión de datos -RRDtool y MySQL, con el objeto de dar uso específico a cada una de ellas. En el ámbito HPC donde se desarrolla esta propuesta, y específicamente en cuanto al objeto de programación, se hace necesario recolectar datos que ocurren cada minuto de tiempo o en fracciones menores, y que se requiere sean almacenados teniendo como índice de registro, el momento preciso en que fue colectado. La herramienta RRDtool es utilizada para la registración de datos de series de tiempo -exclusivamente números-, y el gestor MySQL será utilizado en la gestión de datos general. [MySQL]

El patrón elegido como estrategia de infraestructura del modelo de software, es el Modelo Vista Controlador (MVC), y para su despliegue es necesario gestionar datos. El sistema se encuentra programado para trabajar con sesiones de usuario a nivel de lenguaje PHP, y no está previsto el acceso a formularios sin estar autenticado. Durante el proceso de autenticación de usuarios que acceden al clúster, y en función a las credenciales ingresadas (username y password), se retornan roles, tareas y permisos generales, siempre que la cuenta se encuentre habilitada.

2.2.b Métricas.

Esta propuesta de software tiene por objeto la difusión de estadísticas de uso en tiempo real del clúster HPC. En este contexto, la estadística es una herramienta importante en el ámbito de gestión administrativa y de proyectos de investigación. Se intenta con ello, interpretar cómo son utilizados los recursos computacionales de memoria RAM, procesadores, interfaces de red, espacio en disco. Al final de un ciclo de gestión o quizás periódicamente, dependiendo de las políticas de administración y/o análisis de uso de recursos, se hace necesario brindar informes sobre el uso de estos recursos, y con ello, poder sostener nuevos proyectos que demanden mayores capacidades de infraestructura. Se considera importante, el aporte de las métricas en los informes de resultados en diferentes períodos de tiempo.

En el contexto del caso de estudio, el nodo Headnode es el encargado de recolectar datos, propios y de los nodos de cálculo. Los nodos generan una importante cantidad de datos relacionados a estados de CPU, memorias, interfaces de red, programas utilizados, usuarios involucrados. El sistema operativo a través de funciones incorporadas, recolecta estos datos, que se registran en archivos para su posterior uso. Una funcionalidad prevista en la aplicación que se propone, es la de permitir la visualización de esta información dentro de una ventana de tiempo. Las métricas utilizadas se incorporan una vez instaladas las aplicaciones específicas, que se despliegan en forma de módulos escritos en lenguaje C y Python. Es necesario, que los nodos que van a ser supervisados, están conectados a la red del clúster, ya que la topología prevista determina que todos los nodos se encuentran preparados para enviar y recibir métricas de todos los nodos del clúster.

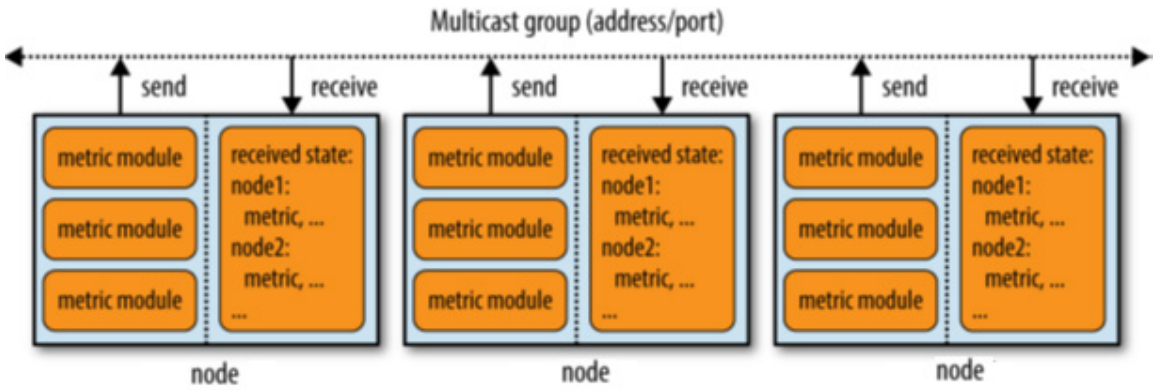


Fig-1. Topología del despliegue de métricas

Las métricas desplegadas sobre la red del clúster del caso de estudio tomado, trata de una red Gigabit Ethernet. A este momento, se encuentra en proceso de compra, dispositivos y servicios para el despliegue de una red de alta velocidad y capacidad en cuanto a transporte y comunicación de datos “Infiniband” [Infiniband].

Métricas de uso de software y actividad de usuarios.

En el ámbito del escenario del caso de estudio mencionado, se logró establecer de parte de sus autoridades, requerimientos de información en relación al uso del clúster HPC. El resultado del estudio de requerimiento, determinó que una parte importante no es provista de manera automática por ninguno de los paquetes de software involucrados, y para su obtención, se hace necesario procesar datos de varias fuentes. Las métricas solicitadas por las autoridades son:

- De Usuarios:
  - CPU utilizado durante una ventana de tiempo
  - memoria RAM utilizado durante una ventana de tiempo
  - software (de una lista pre-establecida) utilizado durante una ventana de tiempo
  - espacio en disco por cuenta de usuario durante una ventana de tiempo

- de paquetes de software utilizado:

- Tiempo de ejecución utilizado por cada producto durante una ventana de tiempo
- Cantidad de procesos en paralelo durante una ventana de tiempo
- Cantidad de usuarios que utilizan los productos durante una ventana de tiempo

Para reunir esta información, es necesario desplegar una estrategia que integre múltiples fuentes de datos, y una forma de lograrlo, es mediante la ejecución programada de scripts. La idea es monitorear comandos de sistema operativo y aplicaciones de usuario mientras son ejecutados en segundo plano, de esta interacción surge una importante cantidad de información útil.

Es posible determinar que una gran parte de la información sobre el uso de recursos computacionales del clúster, resultan de métricas propias del sistema operativo, y otras más específicas se logran mediante la programación de scripts que se lanzan y ejecutan de forma periódica para el registro de información de estado.

2.2.c Rendimiento de software.

Las pruebas de rendimiento de software sirven para múltiples propósitos. Los propósitos que nos interesan son los criterios de rendimiento y las comparaciones de dos sistemas o versiones de sistemas y encontrar cuál de ellos funciona mejor. En programas de cálculo numérico, es importante el poder identificar las partes del sistema o de carga de trabajo que provocan que el conjunto rinda mal. Es posible realizar un diagnóstico de medición para establecer umbrales que determinen tiempos de respuesta aceptables.

La evaluación de rendimiento de software, puede ser llevada a cabo mediante el uso de herramientas Linux a modo de librería. Deben permitir la implementación de contadores de rendimiento de CPU, herramientas que se encuentran incluidas en el kernel de Linux y suelen ser actualizadas con buena frecuencia y con importantes mejoras. Al desarrollar software, se busca realizar profiling de los contadores de hardware, cuando por ejemplo cambiamos de arquitectura de computación, cuando llegan nuevos servidores de cálculo, se intenta encontrar mejoras en el rendimiento general de la aplicación en sintonía con funcionalidades de software desconocidas [Profiling].

Utilizando el doble de recursos de hardware, se puede esperar razonablemente que un programa se ejecute dos veces más rápido. Sin embargo, en programas paralelos típicos, esto raramente es el caso, debido a una variedad de gastos generales asociados con el paralelismo. Una cuantificación precisa de estos gastos generales es fundamental para la comprensión del rendimiento paralelo del programa [IntroParallel].

Los contadores de rendimiento son registros de hardware de CPU que cuentan sucesos de hardware, que ocurren a medida que el programa es testeado, por ejemplo: cantidad de instrucciones ejecutadas, “cache-misses” sufridos o ramas “mispredicted” obtenidas. La información colectada de estos contadores puede ser utilizada en la creación de un perfil – o profile- de aplicación, para permitir el rastreo del flujo de control dinámico y poder identificar puntos de acceso. Es posible obtener contadores por tarea, por la CPU y por carga de trabajo y se acceden a través de descriptores de archivos utilizando llamadas al sistema. En

el caso de Linux versión de kernel 2.6.31 en adelante, se utilizaba la llamada "sys_perf_event_open" renombrada como "perf_event" en la versión 2.6.32 [Profiling].

El subsistema eventos de performance de Linux, provee una manera de medir el flujo de trabajo con baja sobrecarga, sea de una sola aplicación en particular como del sistema operativo completo. En muchos casos, los registros de salida de un evento a la vez, no son suficientes para determinar de manera eficiente, por ejemplo, cuellos de botella en el sistema.

En la aplicación que se propone, se incluye un grupo de tareas en forma de menú, con nombre "Profiling", que será dispuesta a los usuarios, integra básicamente una serie de scripts predefinidos, para que sus programas puedan ser testeados en los nodos de cálculo, a efectos de conocer el comportamiento real del software y poder registrar el resultado. La registración del resultado, determina la generación de un "historial" de ejecuciones, al que el usuario tendrá acceso para revisión.

2.2.d Gestión y visualización de recursos computacionales.

El aspecto de gestión y visualización de estado de recursos computacionales, es importante tanto para usuarios del clúster como para autoridades del instituto, quienes ejercen funciones administrativas y de control de estado del equipamiento en general. El sistema y la estadística generada durante su uso, colaboran en el sostenimiento de estrategias de expansión. Del sistema de estadística, se podrá obtener el uso en tiempo real, durante una ventana de tiempo de días, semanas o meses; y los elementos a involucrar serán los nodos de cálculo y sus capacidades de memoria RAM, CPU y red.

El monitoreo de recursos pretende dar respuesta a:

- Usuarios que demandan conocer el estado de sus trabajos
- Directivos, que gestionan proyectos a largo plazo

La funcionalidad de visualización de monitores de estado de recursos será realizada mediante una solución web-responsive, para poder determinar con especificidad, el estado real de ocupación de los recursos computacionales de la institución. Esta forma de monitoreo se estima de utilidad, porque implica la posibilidad de planificación, el poder conocer cuándo un recurso será liberado, y partir de ello, tener la posibilidad de asignárselo a otro usuario; o también, la posibilidad de modificar el orden establecido en una cola de trabajo.

Para el rol de usuario está previsto la disponibilidad de monitoreo de:

Detalle	Granularidad
Actividad de su cuenta de usuario en el clúster	(última hora, día, semana)
Colas de trabajo asignadas con actividad en el clúster	(última hora, día, semana)
Programas de cálculo utilizados	(última hora, día, semana)
Estado de los nodos de cálculo asignados	Diario
Estado de servicios adicionales (correo interno, internet, copias de seguridad)	Diario

Para el rol de Dirección, además de los ya mencionados, estarán disponibles:

Detalle	Granularidad
Todos los usuarios con actividad en el clúster	(última hora, día, semana)
Todas las colas de trabajo con actividad en el clúster	(última hora, día, semana)
Nodos de cálculo más utilizados del clúster	(última hora, día, semana)
Trabajos encolados y fecha de posible inicio	Diario
Estado de todos los nodos de cálculo	Diario
Programas de cálculo más utilizados en el clúster	(semana, mes)

2.2.e Protocolos de comunicación de datos y redes.

Quizá sea la capa de infraestructura desplegada para la comunicación y transporte de datos, la que más aporta al logro de objetivos de rendimiento en el ambiente de ejecución de trabajos de clúster HPC. Tal como se describen en muchos documentos en cuanto al estado de arte, es necesario contar con una infraestructura básica para la conexión de los elementos que intervienen en la configuración del clúster: el Headnode, los nodos de cálculo, como así también los switches y routers de comunicación, sistemas de copia de seguridad, almacenamiento de datos, entre otros [OpenHPC].

En un sentido técnico, se hace necesario separar el tráfico a nivel de capa de enlace, de acuerdo al modelo capas OSI de la ISO [ISO]. El escenario básico se logra conformando una “red de acceso” y una “red de datos”. En la “red de acceso” existe tráfico de múltiples protocolos, en la “red de datos” hay tráfico con protocolos controlados bien definidos. Desde un punto de vista estricto no es conveniente que sean “mezclados” en la misma red.

En cada red definida, se implementan protocolos de comunicación, de sesión, de autenticación y de aplicación totalmente distintos. Otro aspecto importante es el de las capacidades de transmisión en cada segmento de red, las redes de alta velocidad se guardan para la conexión de dispositivos de almacenamiento y nodos de cálculo, que requieren respuestas rápidas para dar continuidad a los trabajos en ejecución; las redes de acceso son redes desplegadas sobre conexiones a 100 Mbits o 1000 Mbits (1Gb), son más que suficientes para lograr objetivos de conexión. Las redes de datos, alcanzan a 100Gb/s de enlace en la actualidad, para cuando los volúmenes de datos a transportar desde-hacia los nodos de cálculo son de gran tamaño, se requiere una red de velocidad y capacidad de transmisión de al menos 10Gbe, y con circuitos virtuales simultáneos, por ejemplo los implementados con la tecnología Infiniband.

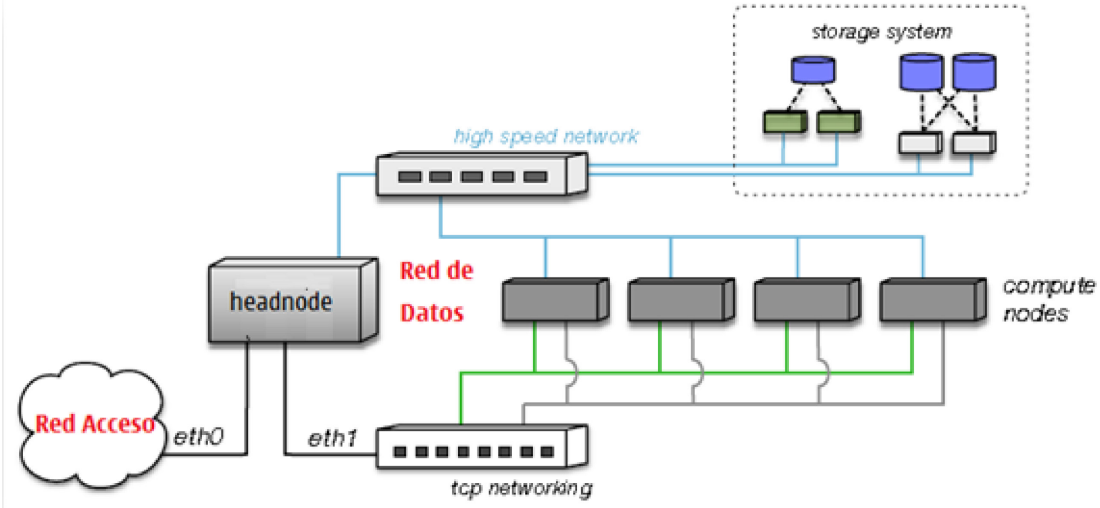


Fig-2. Esquema de conexión de redes.

Nivel de capa 2. La segmentación de red en redes virtuales o vlan"s es una técnica que favorece los tiempos de respuesta de una red, que puede ser implementada a nivel de switch de conexión. Se debe lograr de manera dinámica, establecer redes "pequeñas" cuando la circunstancia lo amerita: por ejemplo, cuando establecemos un agrupamiento de, digamos, 2 hosts para correr un trabajo en particular; los puertos que conectan los hosts elegidos se vinculan a un ID de vlan distinto, y el tráfico de capa 2 es comunicado únicamente entre ellos y no sobre los demás puertos de switch. Esta configuración es realizada en el módulo de software del propio switch de conexión, y puede ser vuelto atrás una vez finalizado el trabajo en particular.

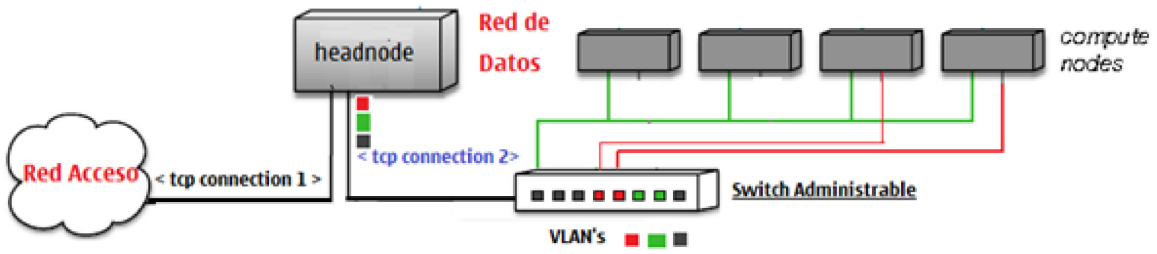


Fig-3. Esquema de comunicación de redes virtuales

El hecho de contar con switches de conexión con módulo de software de gestión en un ambiente de clúster HPC, suma a los objetivos de mayor rendimiento en el uso general de los recursos computacionales. La gestión de múltiples VLAN"s es una de las funcionalidades, entre muchas otras, que tienen aplicación frecuente en este ambiente HPC. Otras, pueden ser: generación y transporte de paquetes Ethernet "grandes" de 10 kbytes en lugar de 1500 bytes, otra, link agregattion -suma de puertos- donde (2) puertos RJ45 de 1Gbyte Ethernet se convierten en un único enlace virtual de 2Gbytes.

Nivel de capa 3. De esta capa, se debe determinar la estrategia de numeración IP versión 4 o versión 6, adecuada a la conformación física de los elementos: cantidad de hosts, físicos y virtuales, dispositivos de almacenamiento, switches de conexión, computadoras personales. A partir de este primer estudio, se desprende la cantidad de hosts a direccionar dentro de la red de Acceso y de la red de Datos, y con ello la máscara de red IP adecuada, ajustando a

un valor que contenga un margen de posible crecimiento durante los dos primeros años de funcionamiento; por ejemplo, contemplar un 25% adicional de direcciones IP para nuevos equipos, es suficiente.

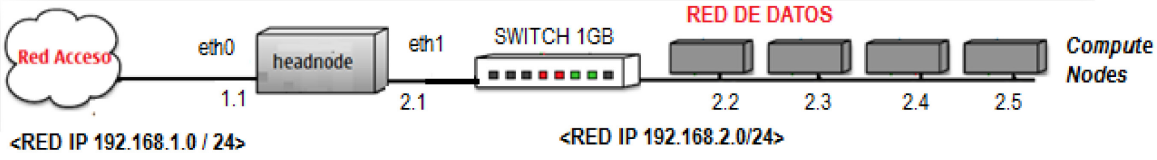


Fig-4. Esquema de direccionamiento IP.

En la figura 7, se aprecia que el nodo Headnode interconecta dos redes IP, a través de dos interfaces de red, cada una conectada a un switch de conexión distinto. Este esquema de conexión permite el despliegue de distintas funciones para cada red y es considerado básico en mucha bibliografía específica. Para este escenario, es necesario recurrir a procesos internos del sistema operativo, llamado “IP forwarding” para el envío de paquetes de datos de una red a otra; y de otros como “firewall” que, en base a políticas definidas, establecen orden y prioridad en la ejecución de reglas de pasaje de datos [OpenHPC].

2.2.f Protocolos de seguridad, acceso seguro.

En muchos casos, se hace necesario incorporar hardware (Firewall) de control y filtrado de paquetes de datos, comunicados entre la “red de Acceso” y el Headnode, con el objeto de brindar protección a la red del clúster ante posibles ataques e intentos de accesos no autorizados. Esta situación es ideal para un ambiente de clúster HPC donde se prevé la administración de datos críticos con una magnitud de importancia superior; que amerita la incorporación, con mayor presupuesto mediante, de dispositivos basados en hardware adicionales. Ver figura 8.

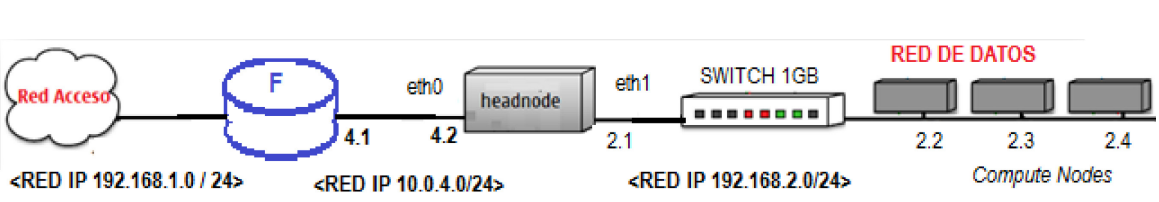


Fig-5. Esquema de conexión seguro.

Comunicación de datos. En ambiente de clúster HPC, se hace necesario desplegar protocolos de almacenamiento de datos, con el objeto de centralizar carpetas con archivos a ser utilizados por muchos nodos en forma simultánea. Existen muchas soluciones en cuanto a este requerimiento, la idea, es dar análisis a aquellas que se caracterizan por ser de simple instalación y mantenimiento operativo. Para que el acceso sea asegurado únicamente a usuarios autorizados, es necesario desplegar como primera medida una estrategia de autenticación de cuentas de usuario.

Nuestra propuesta, en el aspecto del despliegue de procesos de autenticación de usuarios, requiere tener una base de datos con la información acerca de las cuentas de usuario

habilitadas para el acceso a los recursos del clúster. Esa base de datos será consultada tanto por el host Headnode, como por los demás nodos de cálculo y dispositivos de la red de datos, por ello, la base de datos debe estar disponible todo el tiempo e integrada a la red de Datos. Ver figura 8.

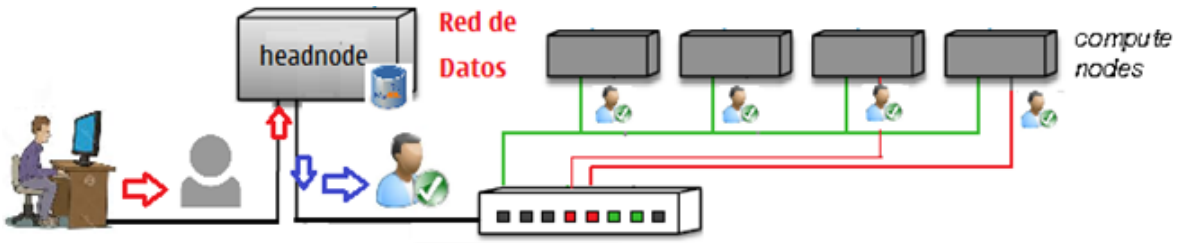


Fig-6. Comunicación de datos

3 CASO DE ESTUDIO Y PROPUESTA.

Este capítulo describe el caso de estudio tomado, la propuesta de solución en un esquema basado en etapas, enumerando las herramientas utilizadas en cada caso.

La estrategia para el trabajo de tesis, en particular para el aspecto de diseño, fue pensada para que la aplicación sea funcional, que integre facilidades de movilidad, que esté provista de elementos que la hagan segura y como producto, que pueda ser mejorado y desarrollado por otros miembros de la comunidad científica que lo requiera. El aspecto funcional y de movilidad, lo aporta el hecho de que pueda correr sobre un navegador web, y que parte de las vistas de informes, posean características web-responsive.

Las tecnologías y herramientas de software utilizadas para la gestión de usuarios, permiso de accesos y gestión de datos, determina la parte de infraestructura requerida para dar soporte a las funcionalidades previstas; sumado a la idea de que sean herramientas de código abierto, asegura que el producto pueda ser compartido libremente con la comunidad científica.

El sistema propone los siguientes pasos para que un usuario pueda hacer uso del clúster y ejecutar sus aplicaciones de cálculo:

1. Poseer una cuenta de usuario, con credenciales válidas, y perfil de rol asignado.
2. Hacer login para confirmar el acceso y verificar las opciones de menú disponibles.
3. Levantar (upload) la aplicación de cálculo de usuario y sets de datos en formato comprimido, utilizando la opción “Projects/Add”, para luego ser descomprimida y compilada.
4. Compilar la aplicación, previendo la existencia de librerías.
5. Testear la aplicación, utilizando el grupo de tareas “Debug”. La aplicación será ejecutada en el nodo Headnode, con un set de datos mínimo, con el objeto de verificar el funcionamiento integral del sistema.
6. Generar un perfil de aplicación, utilizando la opción “Profiling”, mediante la selección adecuada de puntos de control a verificar. Iniciar la ejecución y verificar los resultados. Los resultados son almacenados y registran en forma de historial las ejecuciones hechas.
7. Lanzar la ejecución de la aplicación de cálculo al clúster, utilizando la opción “RUN”, el número de Job asignado por el gestor de trabajos será requerido para tareas de monitoreo.
8. Verificar el ingreso del job a la cola de trabajo asignada, utilizando la opción “Workload/Jobs queded”, debería existir un registro con el número de Job asignado.
9. Verificar el estado de trabajos en ejecución con la opción “Workload/Jobs running”.
10. Verificar el estado de trabajos completados, pulsando la opción “Home” del menú, el cual muestra el historial organizado según el número de Job, las fechas recientes primero.

El punto 3 prevé el uso de una vista en la cual el usuario levanta la aplicación en formato comprimido (no permite otros formatos). El archivo es alojado en la carpeta de home de usuario, la descompresión del archivo podrá ser realizada mediante sesión SSH, o la ejecución de una secuencia de comandos en un script. Es posible la descompresión del archivo, generando una subcarpeta en la misma carpeta de usuario.
El punto 4 prevé la compilación de la aplicación con los compiladores provistos por el sistema. De ser necesarias librerías especiales, el usuario debería requerir al administrador del clúster ajustes de PATH para su acceso o la instalación de las mismas.

Punto 5: se sugiere realizar un test de funcionamiento de la aplicación compilada, que será hecha sobre el nodo “Headnode” y no sobre el clúster; para lo cual, se requiere el uso de un set de datos mínimo a los efectos de acortar al máximo el tiempo de ejecución, con el objetivo de comprobar la integridad de la aplicación. Esta funcionalidad se encuentra implementada en un nivel mínimo, posee múltiples puntos a mejorar, en función a lo que pueda requerir el usuario final. Lo hecho, tiene por objetivo el monitoreo de recursos y evaluación de rendimiento/perfilado básico.

Punto 6: esta funcionalidad otorga al usuario, una serie de modelos de scripts para realizar el perfilado de la aplicación a través de “perf”, y obtener resultados que pueden determinar la línea base de funcionamiento. Es posible programar otras herramientas de perfilado como “gprof” o “papi”.

Punto 7: el sistema ofrece al usuario modelos de script para lanzar su trabajo al clúster. El grupo de tareas “Scripts”, permite clonar template de scripts, con un nuevo nombre de archivo y palabra clave. El nuevo script será registrado en la base de datos, y el usuario dispondrá de el en todo momento. El lanzamiento del script se realiza desde la opción “RUN/Execute Scripts”, en la cual el usuario elige el script a ser ejecutado por el clúster, luego el sistema asigna un número de job, dato que se debe tomar nota para tareas de mantenimiento.

Las tareas de monitoreo son desplegadas en los puntos 8, 9 y 10. El ingreso del job a la cola de trabajo se debe verificar en primera instancia, dado que suele ocurrir errores en la descripción del script, por ejemplo, en la mala asignación de recursos de memoria RAM y CPU, o cuando la aplicación del usuario no puede ser accedida por motivos de permisos de acceso u otros. Cuando el sistema de gestión de trabajos verifica que están dadas las condiciones para el inicio de una aplicación, copia la aplicación del usuario al nodo programado, y prepara todo lo necesario para dar inmediatamente inicio; si la opción de dar aviso por correo electrónico está configurada, el sistema genera un mensaje advirtiendo al usuario del hecho, con detalles informativos. Los trabajos completados informan al usuario las tareas finalizadas por el clúster, pueden ser completadas satisfactoriamente, como no, dada la ocurrencia de algún error, que será informado por separado.

3.1 Caso de estudio.

El caso de estudio que se describe, debe ser considerado como un caso testigo y que podría ser el de cualquier institución que posea un sistema en clúster HPC con objetivos de gestión y planificación de sus recursos. En los detalles que se aportan, se intenta describir

la situación de trabajo real de los usuarios a diario y de la gestión de los recursos del clúster; para pasar después a la propuesta de software que pretende cubrir estos aspectos de manera simple y sistémica. Se utilizará este caso de estudio como representativo de varias instalaciones/instituciones con centros de cómputo desde dos puntos de vista: administrativo en cuanto a usuarios/programadores/ científicos que hacen uso de las instalaciones y su acceso habilitado, y computacional en lo referente a las características del cómputo llevado a cabo en los recursos disponibles.

Caso de estudio: Instituto IMIT Conicet/UNNE.

Es una institución de doble dependencia, depende del Consejo Nacional de Investigaciones Científicas y Técnicas (Conicet) y de la Universidad Nacional del Nordeste (UNNE). Posee un centro de cómputos configurado como clúster de alto rendimiento, utilizado por sus becarios e investigadores como parte de su actividad, en la resolución de problemas de cálculo computacional. El instituto se encuentra localizado en la Ciudad de Corrientes, Departamento Capital, Provincia de Corrientes.

La Facultad de Ciencias Exactas y Naturales y Agrimensura de la Universidad Nacional del Nordeste alberga en sus instalaciones al Instituto IMIT, y comparte el espacio de su centro de cómputos para alojar los equipos servidores y demás accesorios que integran el Clúster del Instituto IMIT. Los usuarios del clúster del Instituto IMIT, becarios e investigadores, son docentes y desarrollan tareas de postgrado/investigación en el contexto de proyectos de la propia Facultad/Universidad.

Los usuarios del clúster.

El Instituto cuenta unos diez (10) usuarios que interactúan, en mayor o menor medida, a diario con el clúster. Los usuarios acceden al clúster desde oficinas localizadas en distintas áreas del edificio, a través de una red cableada, a la que denominamos “Red de Acceso”. La arquitectura de conexión desplegada en el Instituto IMIT, se corresponde con la siguiente figura:

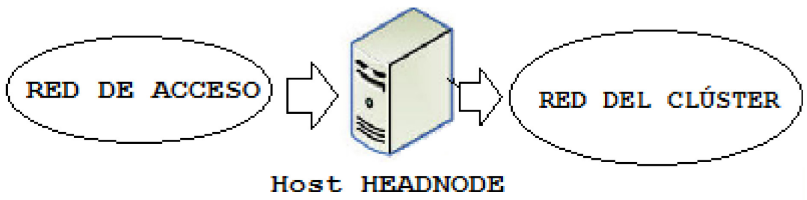


Fig-7. Arquitectura de conexión, caso de estudio.

El host Headnode es el anfitrión y atiende “asuntos” del clúster. Los usuarios solicitan acceso a los servicios del clúster utilizando algún cliente SSH y mediante credenciales propias se le permite el inicio de sesión en un servidor Linux. Todo el software desplegado en este host, es de código abierto.

El Instituto cuenta con acceso a internet utilizado tanto por la parte administrativa para actividades de gestión, como por becarios e investigadores para el acceso desde Internet a los servicios que difunde el Clúster. Posee un servicio de conexión propio a través de un ISP nacional y un acceso propiciado por la Universidad (UNNE), en este caso, con

restricciones propias gestionadas por sus autoridades. El enlace con Internet es utilizado, como se menciona, para el acceso de usuarios a los servicios del clúster, dada la característica de movilidad que despliegan mucho de ellos en otros centros de investigación, dentro y fuera del país.

Infraestructura - El software de base utilizado.

El sistema operativo que corren los servidores del instituto, tanto el nodo "Headnode" como los nodos de cálculo, es Linux distribución OpenSUSE versión 13.2 a la fecha. Si bien, en el ámbito HPC es posible encontrar otras distribuciones como ser "Red Hat", Debian, Ubuntu, se determinó su uso en base a la experiencia personal de varios años en el uso de esta distribución [OpenSuse].

Red del clúster.

La separación de las redes informáticas, como lo muestra la figura 1, en la cual se observa al nodo Headnode como elemento de interconexión entre la "Red de Acceso" y la "Red del Clúster". Esta estrategia responde a principios de arquitectura para un correcto funcionamiento. El nodo Headnode recibe las peticiones de conexión a los servicios del clúster de los usuarios, para lo cual es necesario estar autenticado [TCP-Ilustrated].

La "Red del Clúster" conecta múltiples nodos de cálculo independientes con el nodo Headnode a través de una red cableada. Es común tener en el ambiente del Datacenter, hardware heredado con varios años de uso, junto a otros equipos de última generación; con ello, se hace necesario organizar el hardware según su capacidad y nivel de rendimiento, para lo cual se crean particiones o colas de trabajo, que involucran hosts con iguales capacidades. Esto permite orientar la ejecución de código, según la capacidad requerida por el usuario, se envía lo más básico -poca carga de computación- sobre los nodos 'antiguos' y lo más 'pesado' -importante carga de computación- sobre los nodos más nuevos. Hasta este momento, el Instituto cuenta con diez (10) hosts dedicados a cálculo y un (1) host dedicado a la administración (headnode).

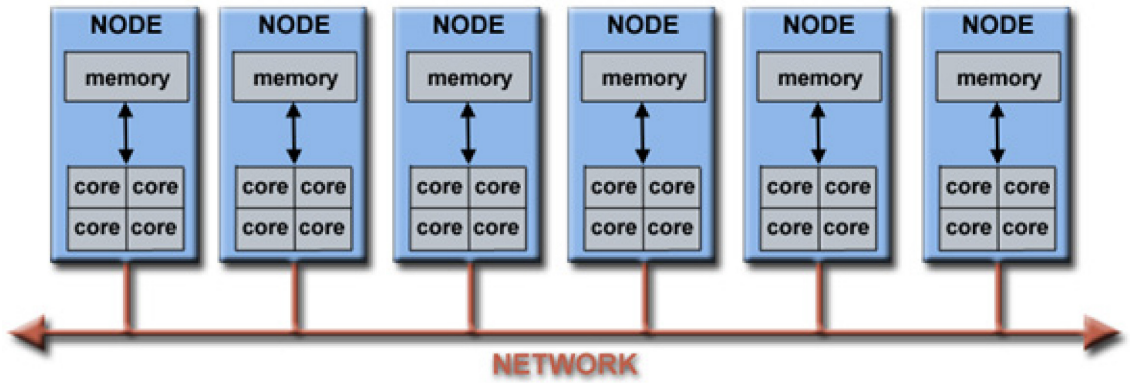


Fig-8. Conexión de nodos en la red del clúster

Gestión del Clúster.

Los trabajos de cálculo que se vuelcan al clúster se gestionan con una herramienta de código abierto, de arquitectura Cliente-Servidor, conocida como Torque PBS. Si bien SLURM es una herramienta estudiada, se considera mejor opción continuar utilizando

Torque, debido a la baja complejidad en la operación de los módulos de software que componen el clúster. En los estudios hechos, quienes se vuelcan a SLURM, poseen no menos de 20-25 nodos de cálculo, con una mayor complejidad operativa.

La parte “Servidor” de Torque PBS es implementada en el nodo Headnode, quien aguarda los comandos que son enviados desde las sesiones de usuario. Los usuarios escriben sus scripts, frecuentemente con un editor de textos y en ellos describen variables e invocan a los programas que desean ejecutar. Luego lanzan los scripts mediante comandos provistos por el gestor Torque, que son encolados en modalidad batch y aguardan turno para su ejecución [Torque].

El gestor de colas, a través del componente “Scheduler”, determina el momento justo en el cual el script de usuario debe ser iniciado, en función a los requerimientos de memoria y CPU solicitados. El componente “Server”, copia a los nodos de cálculo los programas a ejecutar y los parámetros asociados y les ordena su ejecución. Al finalizar el trabajo, los resultados son alojados en carpetas personales de cada usuario. Ver figura 3, gestión de trabajos con Torque.

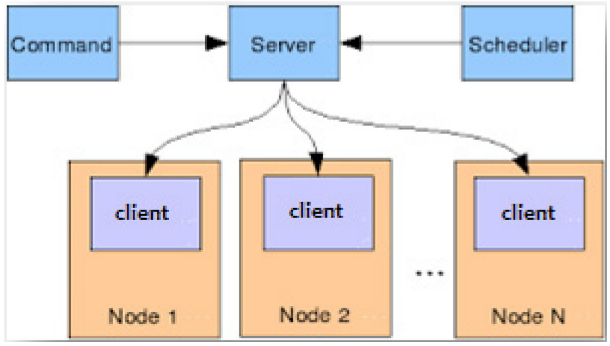


Fig-9. Gestión de trabajos en el clúster

Autenticación y Acceso.

El clúster del Instituto cuenta con un servicio de autenticación y acceso de usuarios, de arquitectura cliente/servidor, el cual corre sobre el sistema operativo del nodo Headnode y la parte cliente sobre los nodos de cálculo del clúster. El objeto es poder autenticar por igual a los usuarios a través de una única base de datos de cuentas de usuarios que se comparte y difunde sobre la red del clúster.

Para el acceso a los servicios del clúster, primero se accede al host Headnode con una sesión SSH. Una vez logueado, el usuario tiene acceso a sus scripts y documentos alojados en una carpeta propia. Las cuentas de usuarios pueden ser desactivadas a pedido de las autoridades del Instituto.

Computación Paralela.

En computación, SIMD (del inglés Single Instruction, Multiple Data, en español: "una instrucción, múltiples datos") es una técnica empleada para conseguir paralelismo a nivel de datos. El paralelismo de datos es un tipo de paralelismo en programas con ciclos, con el objeto de distribución de datos entre los diferentes nodos computacionales que deben

tratarse en paralelo. La paralelización de ciclos conduce a secuencias de operaciones o funciones que se realizan en los elementos de una gran estructura de datos [Flynn].

El clúster del instituto ejecuta código en serie y en paralelo, y para ello son utilizadas librerías específicas en la compilación de programas con el objeto de obtener mejor provecho de ambas modalidades. El paralelismo se logra a través de la ejecución del programa de usuario, utilizando múltiples hilos de ejecución provistos por el sistema operativo. El código a ejecutar en los hilos siempre es el mismo, pero con distinto set de datos, es decir, se aplica aquí el concepto SIMD, de la taxonomía de Flynn [Flynn] dado el tipo de equipos de computación con que se cuenta.

Los programas de usuario, ya compilados, se disponen básicamente de dos maneras: cuando lo utilizan varios usuarios simultáneamente se dispone en una carpeta “cluster” del nodo Headnode; cuando lo utiliza únicamente el usuario, queda localizado en la carpeta del home de usuario. Desde el nodo Headnode, mediante un servicio de compartición de archivos, se comparten las carpetas sobre la “Red del Clúster”, aplicando un riguroso sistema de permisos de acceso.

Los usuarios, en sus scripts de lanzamiento, invocan a los programas localizados en las carpetas de la red y establecen como parámetro, capacidades -en número de cores o procesos- y memoria RAM -en megas o gigas-para el posterior proceso. El destino de ejecución puede ser un único host, en cuyo caso el límite de CPU será la cantidad de cores reconocidos por el sistema operativo; si el destino es una cola de trabajo, el límite será la sumatoria de cores de los nodos que la integran.

Visualización y estadística.

Las tareas de visualización de monitoreo, se soportan sobre un sitio web que se ejecuta sobre el host Headnode. En el sitio, se difunde el estado de los trabajos en ejecución en tiempo real y de ayuda en el diseño de scripts y de uso general del sistema. El acceso está previsto únicamente para usuario autenticados, los cuales deben ingresar sus credenciales individuales para la visualización de este contenido.

3.2 Descripción general.

El framework utilizado en el desarrollo y gestión de la aplicación web, Cakephp en su versión 2.0, aporta las bibliotecas necesarias para dar soporte al aspecto web-responsive, esto representa una importante funcionalidad al usuario, dado que la forma de acceso a la información es siempre igual, cambia el dispositivo utilizado para el acceso, y el sistema reconoce automáticamente la situación, y propone un formato preciso para cada caso.

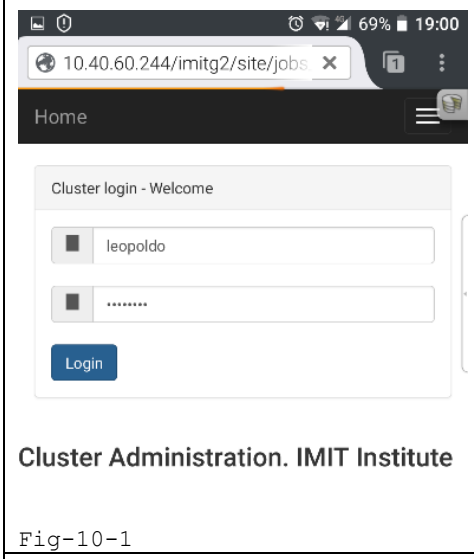


Fig-10-1

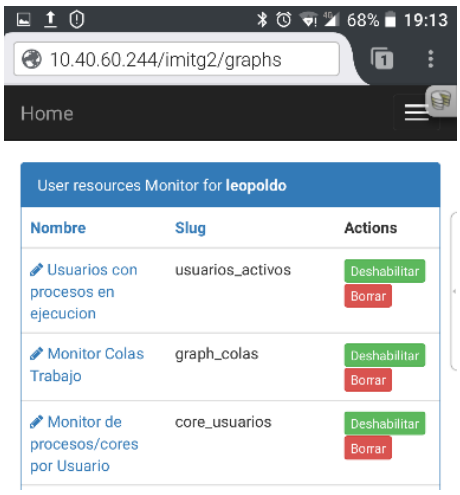


Fig-10-2

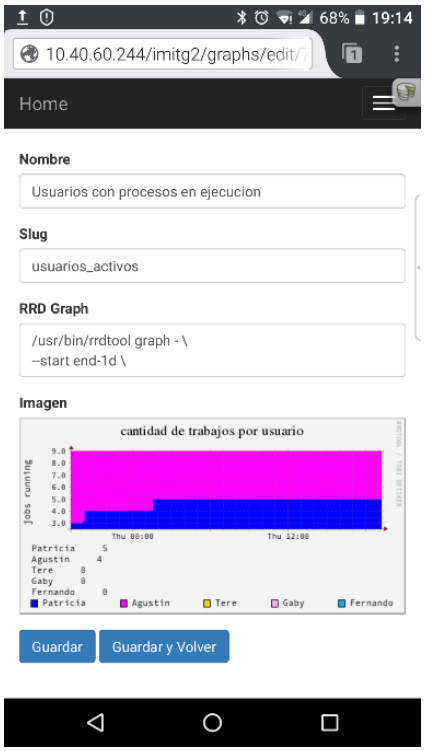


Fig-10-3

En la figura 10, es posible observar una secuencia de capturas de pantalla, el cual ejecuta el cliente Firefox sobre sistema operativo Android de un SmartPhone, para lo cual el dispositivo celular fue conectado por Wifi a la red de acceso de la Facultad. La figura 10-1, es la pantalla de Login, donde se incorporan las credenciales para el acceso a la aplicación. La 10-2, muestra un listado de enlaces de monitoreo ya configurados, los cuales pueden ser seleccionados. La 10-3, muestra el estado de un monitoreo seleccionado, particularmente, de la cantidad de procesos ejecutados en el clúster clasificado por usuario.

El core de la aplicación fue pensado para ser ejecutado en sistemas operativos Linux y desplegar tecnologías RBAC [Rbac] para programar capacidades de superusuario en perfiles. El esquema prevé que los perfiles se asignen a cuentas de usuario, para asumir un rol para realizar un trabajo que requiere algunas de las capacidades de superusuario. El sistema permite configurar diferentes perfiles de derechos para diferentes cuentas. Los perfiles de derechos pueden proporcionar capacidades flexibles para la ejecución de tareas. Al crear roles de usuario, es posible asignarlos a un rango de capacidades bastante amplio (como el de un superusuario), o capacidades más restringidas. Los permisos de usuario específicos se representan como la unión de concesiones de permisos que se asignan a las funciones de usuario.

La aplicación fue pensada, desarrollada y puesta en producción en las siguientes etapas, y las herramientas involucradas son:

DISEÑO	Desktop Windows 10 Eclipse Neon CakePHP(tm) Rapid Development Framework (http://cakephp.org) 0.2.9
DESARROLLO PRUEBAS	Máquina virtual Sistema operativo: Servidor Linux Opensuse 32 bits LAMP: Apache2, MySQL, PHP5 CakePHP 0.2.9 Ganglia + RRDtools Cliente Filezilla (ftp)
PRODUCCIÓN	Sistema operativo: Servidor Linux Opensuse 64 bits Apache2, MySQL, PHP5 CakePHP 0.2.9 Ganglia + RRDtools Torque PBS 6.1 (gestor de trabajos en batch) Clúster HPC del Instituto

La aplicación se soporta sobre un servidor HTTP Apache 2 de código abierto. Para el montaje, tanto en el desarrollo como en la producción, se utiliza la plataforma LAMP (Linux Apache MySQL y Php5). Debe ser ejecutada sobre el nodo anfitrión del clúster, y tiene la capacidad de interactuar con los demás subsistemas y servicios del sistema operativo Linux. El sistema web Apache es multiusuario, es capaz de atender peticiones de forma concurrente pudiendo realizar las siguientes funciones, entre otras, necesarias para soportar funciones previstas:

- Registro de actividad y errores
- Control de acceso basado en la dirección del cliente, contenido o usuario/contraseña
- Reescritura de URLs
- Alias o mapeados de rutas

Herramientas de desarrollo.

El core de la aplicación está hecho con PHP versión 5, lenguaje interpretado, que corre sobre servidor web Apache2. Como herramienta de diseño fue utilizado “Eclipse” [Eclipse], y el editor de textos “Notepad++”, ambas de código abierto. Durante el desarrollo, se optó por escribir el código inicial y realizar las pruebas sobre una máquina virtual ejecutada sobre una sesión de Windows como sistema operativo. Esto permitió un nivel de interacción importante en el uso de las distintas herramientas, luego de resistir las primeras pruebas de funcionamiento, el modelo fue pasado a un entorno de producción, localizándolo sobre un nodo anfitrión de clúster, para aprovechar e interactuar con el entorno de trabajo completo.

3.3 Framework CakePHP.

El Framework CakePHP fue utilizado para soportar modelo-vista-controlador “MVC” e importantes funcionalidades PHP, que colaboran en el logro de los objetivos funcionales previstos en esta aplicación. Proporciona una estructura organizativa básica que cubre los nombres de las clases, archivos, tablas de base de datos y otras convenciones. Como alternativa de framework CakePHP, podría ser utilizado “CodeIgniter” o “Symfony” [Frameworks].

La aplicación está pensada como una gran lista de tareas a ser ejecutadas, almacenadas en una tabla de base de datos. El acceso a las tareas está definido por roles, existe una tabla de roles, que relaciona una serie de tareas a cargo, que luego un usuario con ese rol instanciado, podrá utilizarlas. Los permisos, se encuentran registrados en una tabla adicional, y determinan acciones sobre objetos. Las cuentas de usuarios están registradas en una tabla de usuarios, la cual incluye información para identificar al usuario, para determinar si está habilitado para el acceso al clúster.
En la figura 11 es posible observar parte del relacionamiento establecido:

Users

#	Nombre	Tipo
1	id	int(10)
2	username	varchar(50)
3	name	varchar(500)
4	password	varchar(255)
5	created	datetime
6	modified	datetime
7	e-mail	varchar(60)

Permissions

#	Nombre	Tipo
1	id	int(11)
2	controller	varchar(255)
3	view	varchar(255)
4	created	datetime
5	modified	datetime
6	type	varchar(255)
7	status	varchar(100)
8	name	varchar(500)

Roles

#	Nombre	Tipo
1	id	int(11)
2	name	varchar(255)

Fig-11. Tablas RBAC

Este despliegue, es abordado mediante la modalidad denominada Control de acceso basado en Roles -RBAC- [Rbac], técnica para asignar el acceso a funciones y tareas de software a partir de la programación de roles y permisos, de manera muy exhaustiva. La

programación de la infraestructura RBAC utiliza una combinación de MySQL para la gestión de datos, y CakePHP como framework PHP, como se dijo. La siguiente figura muestra un esquema de la implementación de nuestra estrategia RBAC:

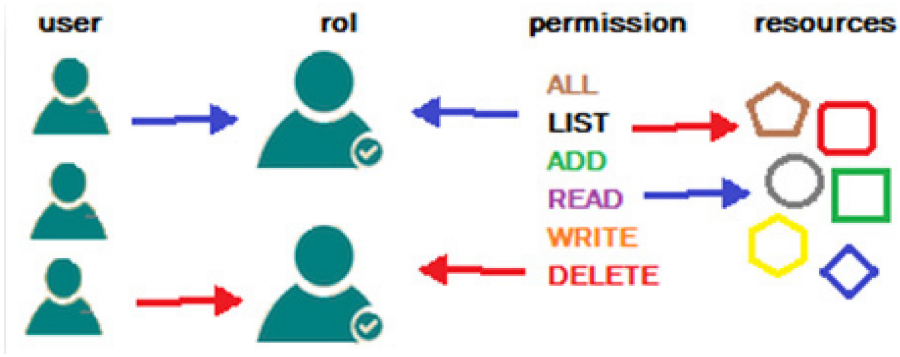
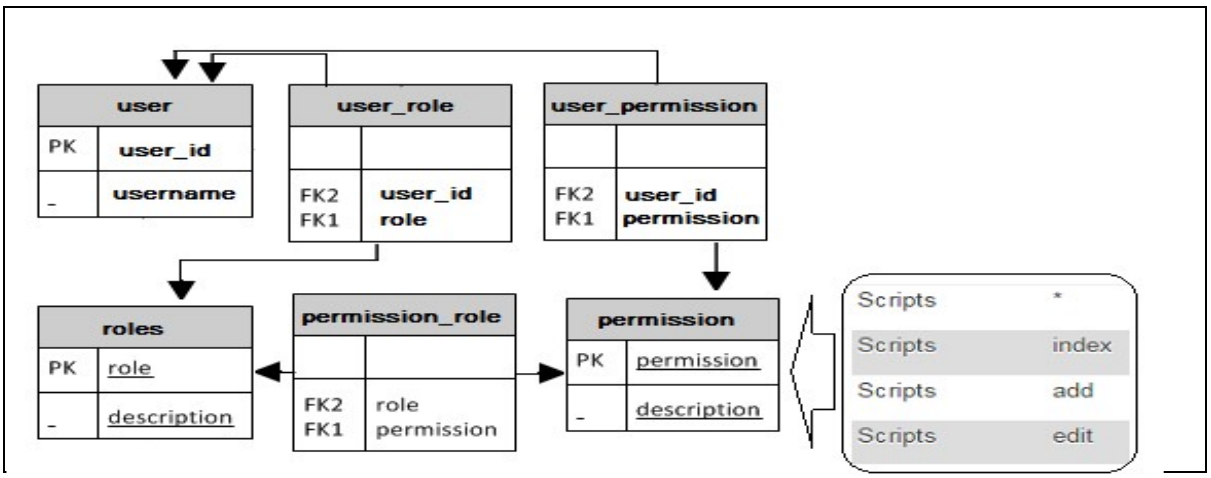


Fig-12. Esquema de implementación RBAC.

Las autorizaciones, privilegios y comandos con ciertos atributos de seguridad se asignan a los perfiles de rol. Los roles son, a su vez, asignados a cuentas de usuarios para el control de acceso. De forma predeterminada, cuando un nuevo usuario es agregado, la cuenta no posee funciones asignadas, lo que significa que por omisión los usuarios no tienen acceso a realizar ninguna tarea.



El sistema gestiona, define y hacer cumplir acciones estrictamente establecidas en los roles y permisos. En la figura 13 es posible observar el sistema de tablas y sus relaciones, lo cual durante su ejecución, no implica problema de rendimiento para el clúster dado el número relativamente pequeño de usuarios y datos involucrados en la gestión de permisos y control de acceso, todo el sistema puede ser implementado sobre el mismo nodo “Headnode”.

La implementación del sistema fue hecha con la herramienta MySQL, un gestor de bases de datos transaccional, con las siguientes funcionalidades:

- Las bases de datos relacionales proporcionan un modelo o estructura de datos fuerte, que aprovechamos en este escenario específico para la organización y el mantenimiento de los datos RBAC.
- Integración con otras herramientas de software de código abierto para el desarrollo de portales web para la difusión de información y despliegue de aplicaciones web.

El framework CakePHP organiza los archivos para la gestión de RBAC de la siguiente manera:

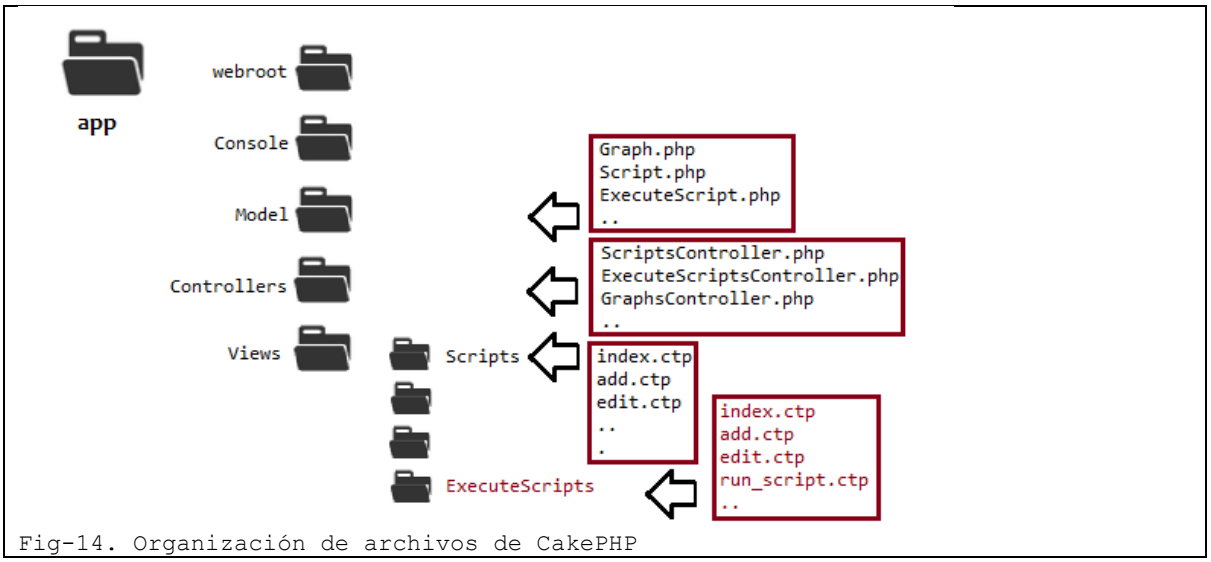


Fig-14. Organización de archivos de CakePHP

La figura 14 muestra la organización de carpetas y archivos a partir del nivel “app”, nombre dado para identificar el sitio raíz de la aplicación. Dependiendo de la versión, integra algunas carpetas más a las mostradas. La carpeta “webroot” es usada como espacio para el trabajo con archivos de datos, archivos temporales, carpetas de usuarios temporales, entre otros. En la carpeta “Model” se guarda la configuración de archivos modelo asociado a vistas y su controlador.

En la carpeta “Views” se guardan las vistas de la aplicación organizadas en subcarpetas, en función a los grupos de tareas o acciones. En la carpeta “Controller” se guarda la configuración de control de funciones programadas, a ser utilizadas en las vistas, para lo cual existe un archivo por cada grupo de tareas. Esta es la manera en que es desplegada la estrategia MVC -modelo vista controlador.

La carpeta “app” es gestionada por el servicio web apache, en la configuración se define su localización, y se determinan los permisos de acceso de usuario y grupo con el cual será ejecutada la aplicación. La carpeta “Console” posee el ejecutable propio “cake”, el cual es utilizado para la sincronización y coordinación del sistema a nivel del framework. Incorpora código PHP que es ejecutado de manera temporal.

3.4 Usuarios, roles, permisos, grupos de tareas.

Estrategia para autenticación de usuarios.

Cada usuario del clúster interactúa con el sistema a través de un proceso de inicio de sesión como se muestra en la figura 15, se refiere a la misma vista de la figura 10-1, ahora vista desde un cliente Firefox sobre sistema operativo Windows 10. Por configuración de ruteo de la aplicación, el sistema se encuentra programado para que, ante cualquier intento

de acceso a elementos del sitio sin sesión habilitada, siempre derive el pedido, a la página de inicio de sesión.

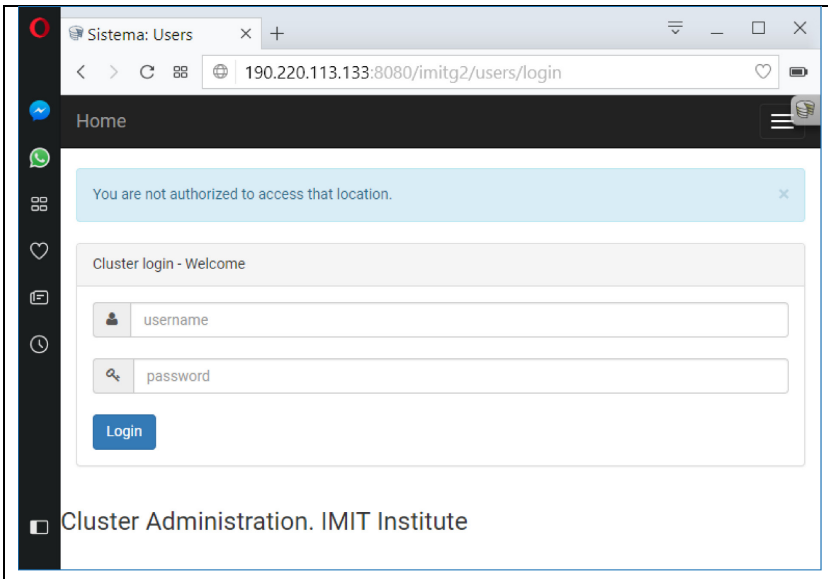


Fig-15. Login de usuario

El usuario “admin” corresponde al administrador general del sistema, su panel de control integra un menú completo de opciones para:

- agregar roles de usuario y agregar usuarios, asignación de roles a usuarios.
- asignación de permisos de acceso a recursos por rol de usuario.
- monitoreo de recursos del clúster, y de recursos de usuarios.
- monitoreo de historial de usuario.
- definición y asignación de colas de trabajo.

En la figura 16, es posible advertir que el menú desplegado para el usuario admin, le corresponden por omisión todas las opciones posibles de configuración:

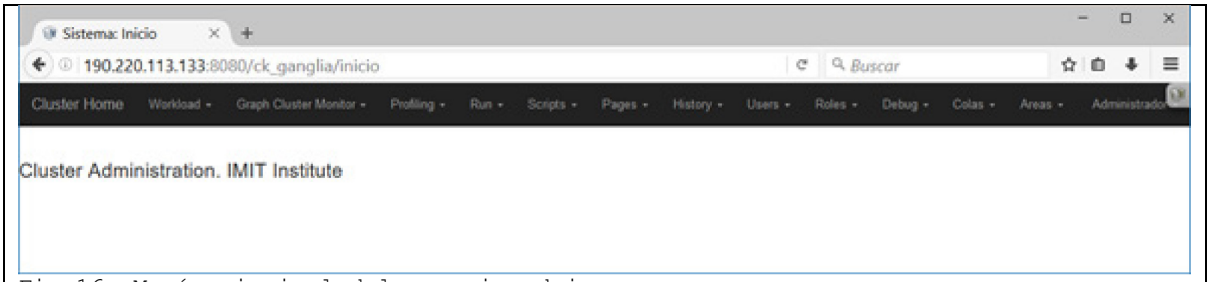


Fig-16. Menú principal del usuario admin.

Roles: El manejo de roles es una funcionalidad de administrador, y se utiliza para definir y asociar funciones a los usuarios del sistema. Los roles definidos en la propuesta de implementación actual son: "becario", “investigador”, "director" y "administrador". Cada rol integra permisos de acceso a acciones sobre los recursos, de menor a mayor medida.

Las cuentas de usuario son definidas mediante un nombre y sus roles asignados y pueden asumir más de un rol. El usuario puede editar su perfil de cuenta, como es habitual, pero no está autorizado a modificar los roles asignados. La definición de un rol requiere un nombre, los permisos sobre objetos y tareas, y un área al que se aplica.

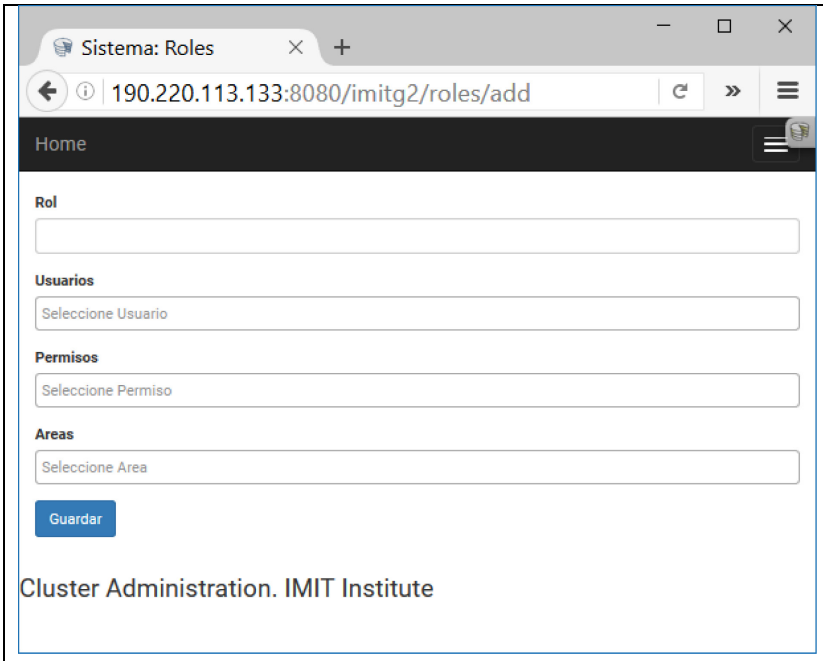


Fig-17. Definición de rol.

A partir del listado de roles, es posible seleccionar un rol y editar sus propiedades, por ejemplo, para adicionar o quitar permisos sobre objetos o tareas. Esta actividad solamente la puede hacer el usuario “admin”:

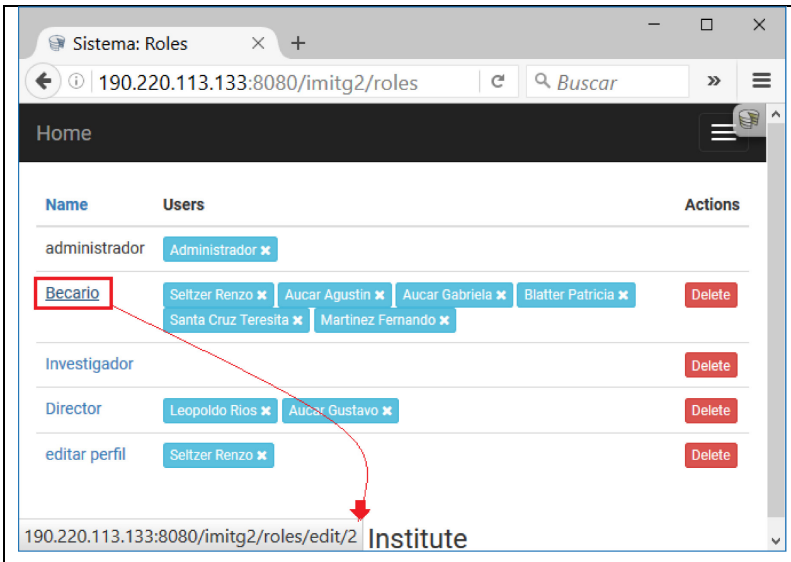


Fig-18. Lista de roles definidas por omisión.

Una vez elegido el rol, por ejemplo, para la instancia “Becario”, tenemos la posibilidad de editar sus propiedades, como asignar o quitar usuarios vinculados, adicionar o eliminar permisos sobre grupos de tareas, asignación de áreas, como se aprecia en la figura 19.

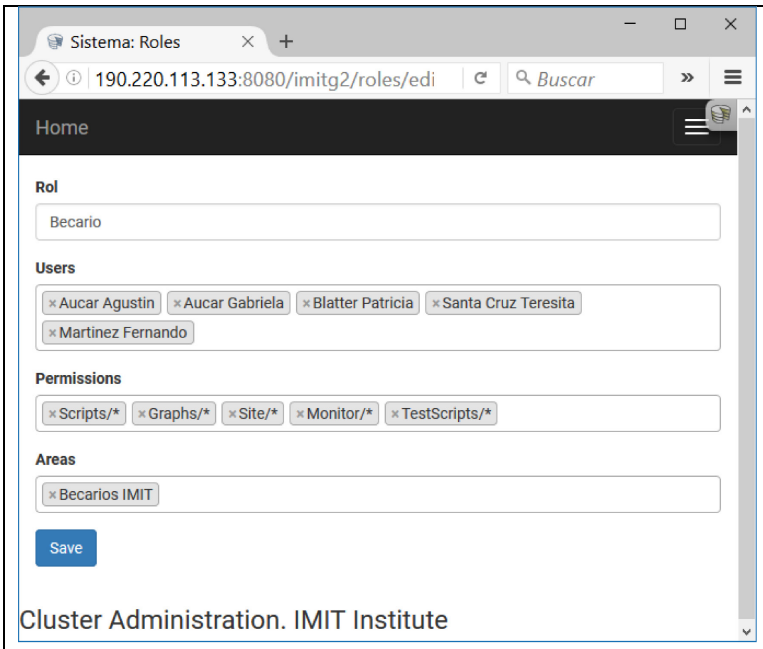


Fig-19. Edición de Rol.

Permisos sobre tareas: La aplicación permite al administrador del sistema, asignar permisos sobre grupos de tareas a roles de usuario definidos. Por ejemplo, al rol “Becario” se le asigna el permiso “Scripts/*”, este asterisco implica la asignación de todos los permisos sobre el grupo de tareas “Scripts”: el acceso al índice de scripts, adicionar scripts y editar scripts existentes. De lo contrario, si un rol tiene asignado solamente el permiso "Scripts/index", los usuarios con este rol asignado, solo tendrán acceso al índice de scripts, sin poder editar ni borrar elementos.

El control de acceso a los elementos de menú, está asegurado por el modelo-vista-controlador mencionado, las acciones previstas para el caso -representadas por funciones específicas-, se encuentran programadas en la sección “Controller”, a través código PHP.

La figura 20 muestra la edición de permisos para un rol, el sistema lee los permisos actuales almacenados en la base de datos y permite al administrador cambiarlos, por ejemplo, para adicionar un nuevo permiso. Se puede observar, que se escriben las primeras cuatro letras del grupo de tareas “Script”, es decir “Scri” y el sistema devuelve los recursos y permisos que contengan esa etiqueta, resta que el administrador seleccione el permiso adecuado. De esta manera se asignan permisos a grupos de tareas. La sección “Áreas” es una clasificación que se utiliza para la organización de roles y la posterior generación de estadísticas de uso.

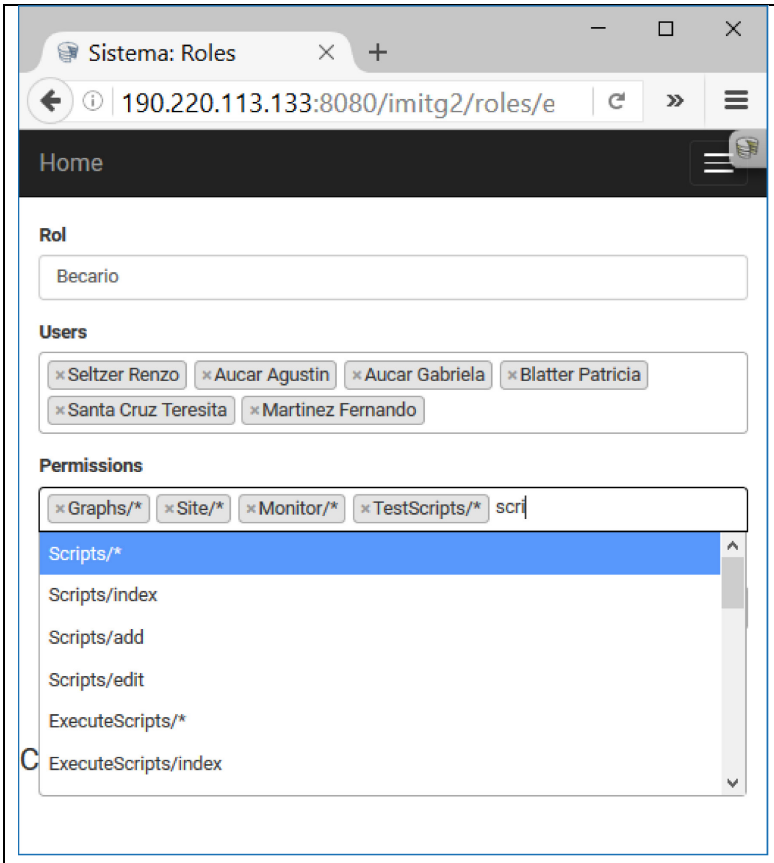


Fig-20. Adicionar permisos a un rol.

Una vez que se incorporan las modificaciones, los cambios se hacen permanentes almacenándolos en la base de datos. A partir de ese momento, los usuarios con el rol "Becario" que ingresen al sistema, podrán acceder al grupo de tareas "Scripts" con todos sus permisos habilitados, es decir, agregar, modificar o eliminar elementos.

Grupos de Tareas.

Este concepto, representan en la aplicación a un grupo de acciones organizadas en un mismo elemento de menú, que al usuario le pueden aparecer o no, en función al rol asignado. Por ejemplo, mencionamos el grupo "Scripts", elemento de menú previsto para reunir scripts de usuario a ser ejecutados en el clúster. Por ejemplo, un usuario con el rol "investigador" puede generar scripts, y otorgar permiso de uso y ejecución de ese script a otro usuario con el rol "becario".

Esto asegura que el script a ser utilizado sea el definido por el usuario investigador. Si fuese necesario que el usuario con rol "becario" requiera editar los scripts, será suficiente con incorporar el permiso "Script / edit" en el perfil de rol. Con esta secuencia se intenta demostrar la técnica utilizada para la asignación de permisos a roles de usuario.

El acceso a cada grupo de tareas, a nivel de programación PHP, están pensados con la siguiente estrategia:

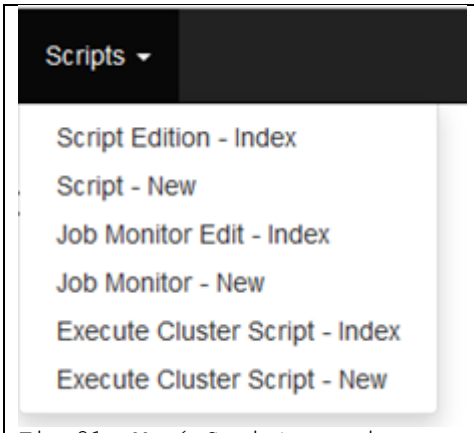
<Referencia HTML>	<permiso requerido>	<función del controlador>	<Nombre del menú>

/view/scripts/add.ctp	“scripts/add”	“function add()”	“Script new”
/view/scripts/edit.ctp	“scripts/add”	“function edit()”	“Modify Script”
/view/monitor/add.ctp	“monitor/add”	“function add()”	“Job Monitor new”
/view/execute/edit.ctp	“execute/*”	“function edit()”	“Execute Script”

La columna “permiso requerido” se corresponde con el permiso requerido por el rol instanciado, la columna “función del controlador” determina la función a ejecutar para dar curso al pedido siempre que el permiso esté otorgado.

El framework CakePHP utilizado, define en el archivo “backend.ctp” localizado en “Views/Layout”, los permisos requeridos para el acceso a las vistas, y configuran el menú de acciones de cada cuenta de usuario. Cuando un usuario hace “login”, los permisos asignados según el rol vinculado, son cargados a un array para su uso. El sistema chequea los permisos, y de tener el permiso, se habilitan las acciones programadas (“index”,”add”,”edit”,etc), que determinan el acceso a vistas específicas para cada acción.

En la siguiente figura 21 se observa el despliegue del menú “Scripts”, cada opción determina una acción específica y requiere para ello, un permiso específico asociado al rol vinculado a la cuenta de usuario. En caso de que el usuario seleccione una acción para lo cual no posee permisos, el sistema mostrará un cartel, dando aviso de la falta de permisos.

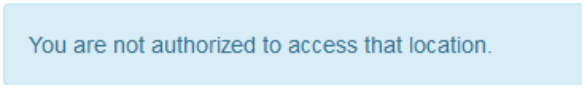


<- Permiso “Scripts/index”
<- Permiso “Scripts/add”

<- Permiso “Graphs/index”
<- Permiso “Graphs/add”

<- Permiso “Execute_Scripts/index”
<- Permiso “Execute_Scripts/add”

Fig-21. Menú Script, acciones.



Las tareas y acciones se asignan a roles, no a usuarios. Cada línea del menú Scripts de la figura 21, determina el acceso a una vista particular. El sistema organiza el acceso a las vistas en función a las opciones de menú, por ejemplo: “Scripts” es una subcarpeta asociada al grupo de tareas Scripts, y en ella, existen archivos de vista, con el mismo nombre de la acción determinada: “index.ctp”, “add.ctp”, “edit.ctp”, y otros. Estas vistas son llamadas por el framework y presentadas al usuario, siempre que el usuario posea permiso de acceso.

Detalles de implementación.

La estrategia de implementación de RBAC está soportada combinando las siguientes herramientas de software, toda de código abierto:

- Gestor de base de datos relacional MySQL
- Servidor Web Apache2
- Lenguaje de programación PHP, Framework CakePHP
- Sistema operativo Linux distribución OpenSuse,
- Gestor de trabajos Batch Torque PBS
- Gestor de métricas Ganglia y RRDtool

La integración y coordinación, junto con la programación específica de funciones basadas en scripting, determinan el funcionamiento del sistema. El acceso a los recursos a través de la web está protegido por las reglas definidas para el servidor Apache. El acceso a la aplicación informática está controlado por el framework CakePHP a través del modelo "Modelo-Vista-Controlador" desplegado. No es posible conectar a ningún recurso del portal web sin hacer primero el procedimiento de inicio de sesión. Este modelo asegura el acceso específico a vistas predefinidas en la programación del "controller" y del "modelo". Se puede decir que, si algo no está programado en el modelo del controlador, no será posible mostrarlo ni usarlo.

El escenario de ejecución de este programa, prevé el acceso de usuarios mediante sesiones SSH para funciones de compilación y ajustes de la aplicación y para gestión de archivos de input y de resultados; la idea es que los usuarios interactúan en mayor parte con el sistema web. El acceso SSH es necesario y requerido porque constituye la manera de trabajar más afianzada en los usuarios HPC, conocedores natos de esta modalidad basada en "comandos" e instrucciones.

Los usuarios podrán bajar sus programas a las carpetas del servidor, y generar los binarios necesarios para su ejecución vinculando las librerías disponibles en nodo Headnode. Los scripts de usuario, a ser utilizados en la ejecución de los programas de cálculo en clúster HPC, son almacenados en la base de datos RBAC; así, cuando un usuario inicia sesión, la información de scripts y monitoreo son leídos y quedan a disposición para su ejecución o modificación. El usuario dispone de una carpeta temporal con seguridad, para que sus archivos y trabajos estén por fuera del alcance de otros usuarios del sistema.

3.5 Variables a monitorear, lectura y registración de datos.

Monitoreo de trabajos de usuario.

Constituye una práctica generalizada, realizar el seguimiento de la ejecución de un Job de usuario, una vez que el script de lanzamiento ha sido aceptado por el gestor de trabajos. El sistema otorga un número de trabajo -Job ID- necesario para tareas de monitoreo de estado. En muchos casos, el usuario recurre a la observación constante del archivo de salida que genera la aplicación en ejecución, con la idea de seguir su evolución, sobre todo ante casos de ejecuciones que duran semanas o meses.

La solución prevista en la aplicación propuesta, es utilizar la serie de comandos provistos por la herramienta de gestión de trabajos “Torque PBS”, para interactuar con el sistema de gestión, lanzar trabajos al clúster y observar su evolución, por ejemplo:

- “*qsub -q cola*”, lanza un trabajo a una cola o partición. El demonio gestor de trabajos es quien recibe el pedido, le asigna automáticamente un número de Job, el cual le es informado al usuario inmediatamente. El usuario, con este número podrá realizar consultas acerca del estado de su trabajo mientras este sea ejecutado en el clúster.
- “*qstat -r*”: informa los trabajos de usuario que se encuentran en ejecución.
- “*qstat -i*”: informa los trabajos encolados, a la espera de ejecución.
- “*qstat -f*”: más el número de Job, reporta en gran detalle la ejecución en curso.
- “*qstat -an*”: reporta detalles de los trabajos en ejecución, de los nodos de cálculo utilizados, y de recursos involucrados como RAM y CPU.
- “*tracejob nro_job*”: reporta detalles específicos de trabajos de usuario en ejecución: memoria real y virtual utilizada, tiempo utilizado y restante, variables de estado, entre otros.

En la aplicación web, el usuario del clúster dispone de una serie de vistas, en las cuales se dispone la información específica en tiempo real, es decir, lo que está ocurriendo en el sistema, es transferido como respuesta a las vistas. La implementación de esta modalidad supone:

- Recolección de datos, tarea a realizar mediante scripts de sistema operativo a ser ejecutados en forma periódica, almacenados en localizaciones locales.
- Proceso de la información previo a su difusión, mediante scripts de sistema operativo. Los resultados pueden ser almacenados en la base de datos de RBAC.
- Difusión de la información mediante tecnología PHP-Web, organizándola en niveles y otorgando acceso a cuentas habilitadas con permisos específicos.

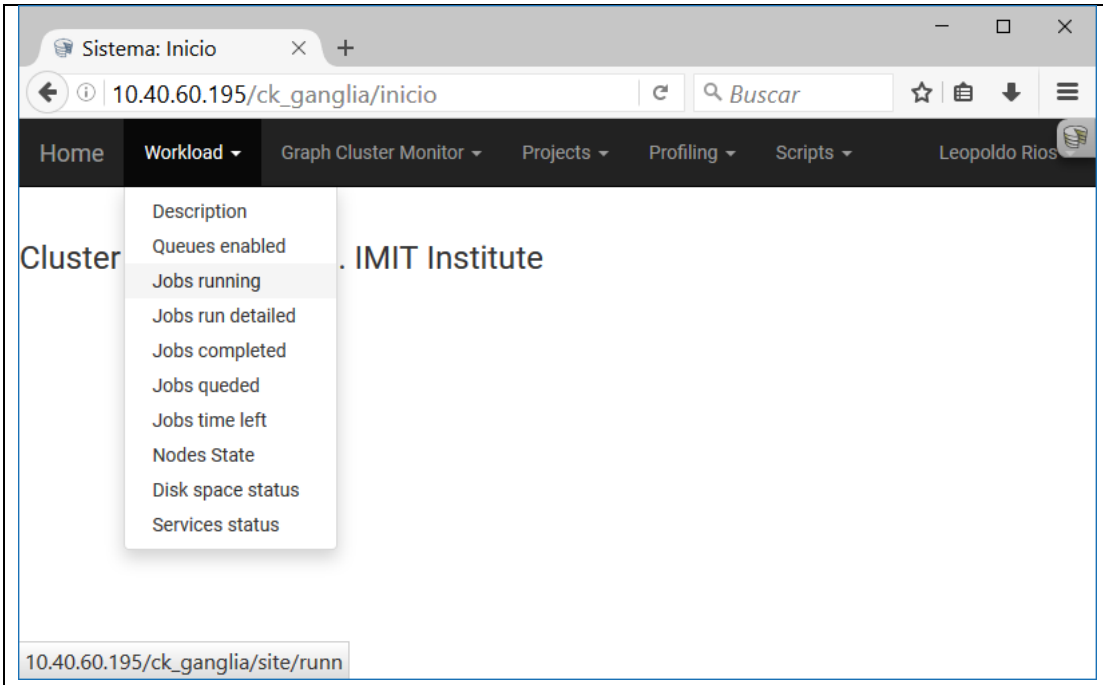


Fig-22. Monitor del sistema mediante web.

El sistema posee un grupo de tareas llamado “Site”, con etiqueta de menú “Workload”, que agrupa acciones que requieren información del estado de la carga de trabajo del clúster y de estados de servicios del nodo Headnode. La opción “Nodes state”, permite observar en detalle los nodos de cálculo configurados: estado, disponibilidad, capacidad en procesadores y cola de trabajo asignada:

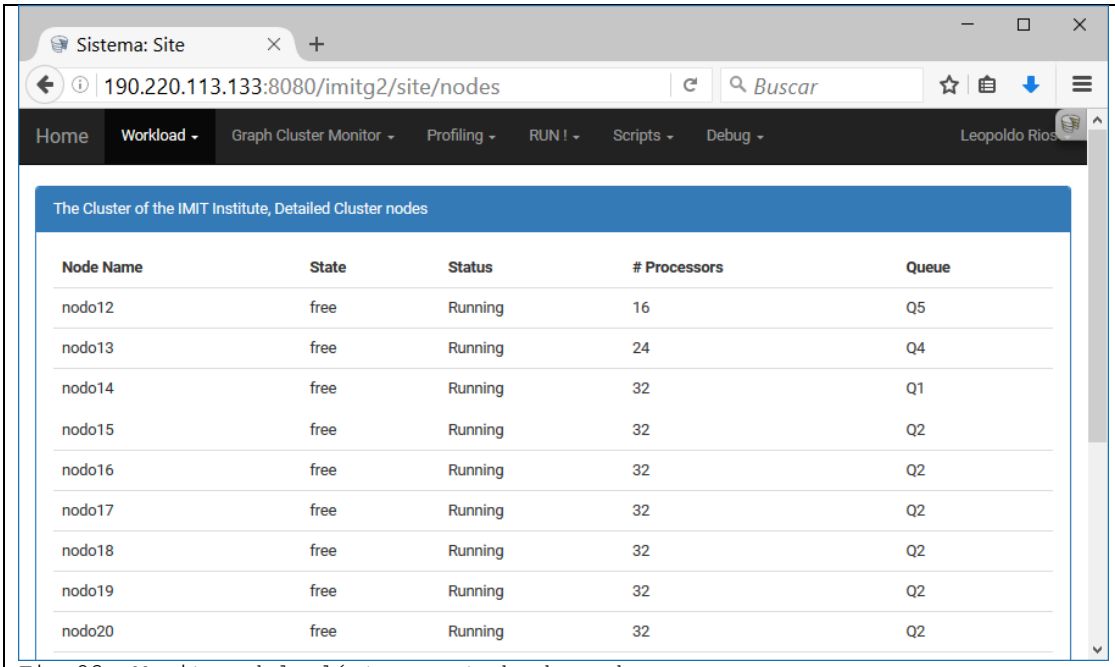


Fig-23. Monitor del clúster, estado de nodos.

La vista de la figura 23, es generada por un script que es ejecutado en forma periódica, con el objeto de reunir en un archivo la información de estado de los nodos, que luego son expuestos en la vista. El código, en resumen, que genera esta vista es el siguiente:

```
<div class="panel-body">
<?php $registros = array();?>
<?php if( ($handle = fopen('nodesrun.csv', 'r' )) !== false ) {?>
    <table id="runjobs" class="table">
    <thead>
        <tr>
           <th> <?php echo "Node Name"?></a> </th>
           <th> <?php echo "State" ?> </th>
           <th> <?php echo "Status" ?> </th>
           <th> <?php echo "# Processors" ?>    </th>
           <th> <?php echo "Queue" ?>    </th>
        </tr>
    </thead>
       <?php while( ($data = fgetcsv( $handle )) !== false )
        {?>
        <tr>
           <?php foreach( $data as $value )
               {?> <?php echo sprintf( '<td>%s</td>', $value );?>
               <?php } ?>
        </tr> <?php } fclose( $handle); ?>
     </table> </div></div></div>
```

El código del script que genera el archivo de datos 'nodesrun.csv', es descripto en la lista de scripts "nodes.sh".
Con estos ejemplos, se describe la metodología utilizada en la generación de información de estado del clúster a través de la aplicación web, en cuanto al uso de scripts que son ejecutados automáticamente en períodos de tiempo regulares. Hay scripts que con la información que obtienen, colaboran con otros scripts en la elaboración de información más específica. La ejecución de los scripts es realizada mediante el sistema cron de Linux, el cual permite que un usuario mantenga una lista de scripts, cuya ejecución es programada en función al tiempo del sistema. En el script número 6 de la Lista de Scripts, es posible observar un modelo de crontab.

Monitoreo operativo del clúster.

Uno de los objetivos de gestión más importantes es el de brindar información en tiempo real acerca del uso de los recursos computacionales del clúster, en ventanas de tiempo ajustables. Este concepto ha sido desarrollado en el punto 2.2.b., gran parte de la tarea de reunir datos y presentarlos de manera organizada lo hace el framework "Ganglia". La información que provee Ganglia, refiere a valores cuantitativos sobre el uso de memoria RAM, CPU, interfaz de Red, espacio en disco. La información acerca de usuarios que utilizan el clúster, de paquetes de software utilizados, entre otras, se reúnen en la aplicación mediante scripting, como se ha mencionada en el punto anterior, y es integrada posteriormente a Ganglia [Ganglia].

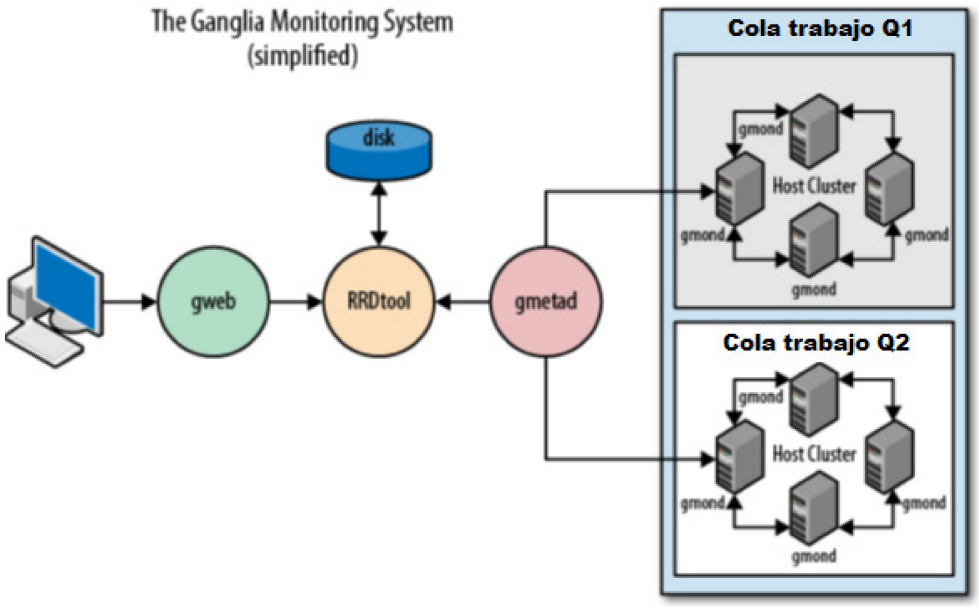


Fig-24. Sistema de monitoreo Ganglia.

La figura 24 muestra un esquema de funcionamiento de los módulos y procesos que constituyen la herramienta "Ganglia. El paquete es instalado por default como lo sugieren sus desarrolladores para lograr:

- Reunir datos de uso de CPU, memoria, disco, red, de cada nodo de cálculo del clúster, y registrarlo en un archivo local.

- Reunir datos de todos los nodos de cálculo, organizados en colas de trabajo, y registrarlos en un archivo local.
- Reunir datos de los nodos cabecera de las colas de trabajo en una base de datos de serie de tiempo con la herramienta RRDtool.
- Procesar, bajo demanda, los datos almacenados en las bases de datos, y difundir mediante distintas técnicas web la información obtenida.

Es posible obtener resultados en detalle de métricas de CPU, RAM, Red, Clúster, los cuales pueden ser reproducidos en los siguientes formatos:

- gráficos (png) histogramas, barras, etc.
- con el soporte de PHP XML adecuado, es posible exportar en formato CSV y JSON.

Esta última funcionalidad, permite al usuario optar la manera en que los resultados le aportan mayor utilidad, al poder migrarlos a otras herramientas para fines cuantitativos.

Funcionamiento de Ganglia.

La arquitectura de funcionamiento es Servidor-Cliente. La parte Servidor, está implementada en el demonio "gmetad" que corre en el nodo Headnode del clúster, y es quien obtiene datos de los nodos cabecera de las colas de trabajo configuradas. La recolección de datos es realizada en cada nodo de cálculo por un proceso cliente llamado "gmond". Las métricas se despliegan en forma de módulos escritos en lenguaje C y Python. Es posible utilizar la herramienta "gmetric", para desarrollar métricas personalizadas para casos especiales.

gmond: Agente de recolección de métricas, la parte cliente de la arquitectura.

El demonio "gmond", es un agente de software ligero cuya finalidad principal es reunir y comunicar los valores obtenidos de la ejecución de métricas. Una vez instalado y configurado, el demonio comienza a ejecutar las métricas de manera periódica y de acuerdo a lo configurado, los datos reunidos se almacenan en archivo local, a la espera de su difusión. Es utilizo el protocolo simple de escucha/anuncio (XDR) para recopilar y compartir información de estado con otros servicios "gmond" en una cola de trabajo, el nodo de cálculo configurado, reúne los datos de los demás nodos y queda en espera de la comunicación del nodo "Headnode" para su difusión.

Las métricas utilizadas se encuentran implementadas en módulos escritos en lenguaje C y Python. La mecánica que se reconoce, es mediante la lectura de datos del sistema "/proc" del sistema operativo Linux en tiempo real. Los archivos fuente de los módulos, se encuentran alojados en la subcarpeta del instalador de Ganglia:

```
…/ganglia-3.6.0/gmond/modules # ls
Makefile.am Makefile.in conf.d cpu disk example memory network perl php
python  status system
```

Cada subcarpeta, contiene el módulo en lenguaje C para un recurso, por ejemplo, módulo cpu:

```
…/ganglia-3.6.0/gmond/modules # ls cpu -l
total 56
-rw-r--r-- 1 root root   866 May  7  2013 Makefile.am
-rw-r--r-- 1 root root 22123 May  7  2013 Makefile.in
-rw-r--r-- 1 root root  3433 May  7  2013 mod_cpu.c
-rw-r--r-- 1 root root  1624 May  7  2013 mod_load.c
-rw-r--r-- 1 root root 17433 May  7  2013 mod_multicpu.c
```

Es posible con esta metodología, desarrollar módulos a medida, como también adaptar y mejorar los existentes. Es necesario, ante modificaciones hechas en los archivos, volver a recompilar el sistema para que los cambios surtan efecto.

gmetad: componente recolector de datos, la parte servidor de la arquitectura.

La parte servidor de Ganglia, en cuanto al aspecto de recolección de datos, es representado por el demonio “gmetad”, que corre comúnmente en el nodo “Headnode”, y se encarga de obtener datos de los nodos cabecera de las colas de trabajo del clúster, y registrarlos en una base de datos de series de tiempo, mediante la herramienta RRDtool. Las métricas se almacenan en bases de datos "round robin", que consisten en asignaciones estáticas de valores por espacio de tiempo.

Impacto en la implementación:

- Cuantos más nodos y métricas por nodo sean configurados, mayor será el consumo de memoria en los procesos “gmond” que están configurados para recibir métricas de los demás nodos.
- Cuanto más rápida sea la tasa de métricas que llegan de la red, más CPU es utilizada. La tasa de llegada de métricas depende del número de nodos, el número de métricas por nodo y la velocidad a la que los nodos están configurados para transmitir valores nuevos. Si el archivo RRD mantiene datos durante un período de tiempo prolongado o mantiene diferentes tipos de datos (valores MIN y MAX), puede requerir múltiples IO al producir una sola actualización.

Un archivo RRD.

El archivo se compone de un encabezado seguido de una serie de uno o más arreglos. Si todo el archivo RRD encaja en un bloque en el disco (4.096 bytes, normalmente), entonces todas las modificaciones en el archivo pueden, en el mejor de los casos, realizarse con una única petición de I/O. La cantidad real de datos que son almacenados para un único punto de datos en el archivo RRD es de 8 bytes. Si los puntos de datos se almacenan, por ejemplo, a intervalos de 60 segundos, será necesario un único bloque de disco de 4,096 bytes para almacenar un poco más de ocho horas de datos. [RRDtool]

Información de monitoreo.

La información obtenida por Ganglia es desplegada en el sistema, a través de la interfaz web del servidor Apache. El componente web de Ganglia -denominado gweb- solicita datos almacenados en archivos de bases de datos RRDtool, quien aporta los resultados en varios formatos y para determinadas ventanas de tiempo. La información visualizada en un gráfico

es posible que sea guardada en formato JSON o CSV. Ganglia crea un archivo RRD diferente para cada métrica a desplegar, en lugar de crear múltiples fuentes de datos dentro de un solo archivo RRD, el siguiente listado es solo un ejemplo breve de los archivos que por omisión son creados en una instalación estándar de Ganglia, para obtener:

cpu_num.rrd	Número de cpus en uso
disk_free.rrd	Espacio en disco libre en disco principal
mem_free.rrd	Espacio libre en memoria expresada en bytes
proc_run.rrd	Cantidad de procesos ejecutándose

Obs: estos archivos RRD de datos, se localizan por omisión en subcarpetas ubicadas en /var/lib/ganglia/rrds/..

En cuanto a la información de métricas de CPU, Ganglia mantiene una importante cantidad de información en los siguientes archivos RRD, que definen el siguiente gráfico:

cpu_user.rrd
cpu_nice.rrd
cpu_system.rrd
cpu_idle.rrd

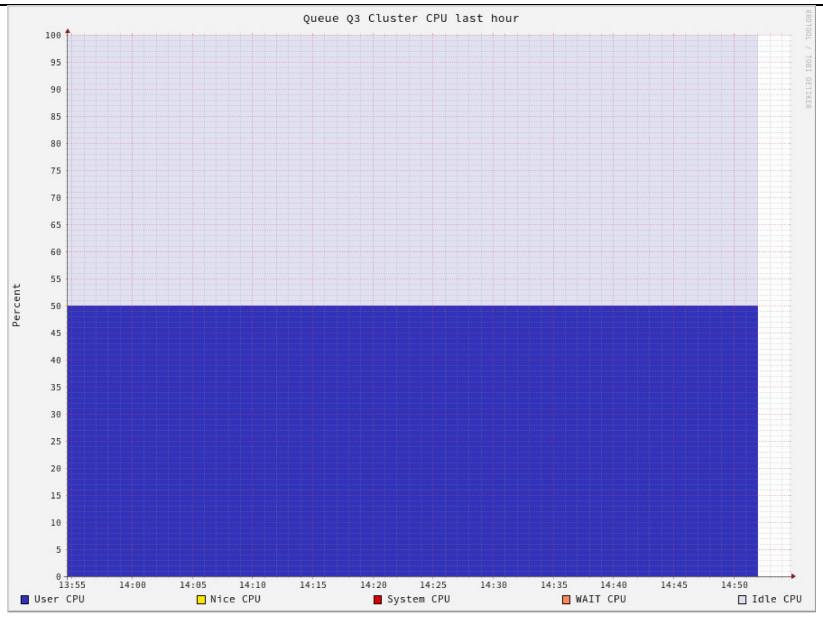


Fig-25. Reporte cpu_report de Ganglia.

Como alternativa de verificación, es posible consultar la última actualización de cada archivo, a través de la herramienta 'rrdtool' y opción 'lastupdate', el cual arroja los siguientes valores:

```
$ rrdtool lastupdate cpu_user.rrd
                sum   num
1497448875: 50.0    1
```

```
$ rrdtool lastupdate cpu_nice.rrd
                sum   num
1497448875: 0.0     1
```

```
$ rrdtool lastupdate cpu_idle.rrd
                sum   num
1497448875:  50.0   1
```

```
$ rrdtool lastupdate cpu_system.rrd
                    sum   num
1497448875:     0.1     1
```

Ganglia a través de la interfaz web, permite la visualización ampliada de variables del entorno operativo, denominada reporte de Grid:

- La figura 26-1, muestra el panorama completo del estado del Grid de la última hora de trabajo.
- La figura 26-2, muestra un sub-menú seleccionable, de las colas de trabajo configuradas.
- La figura 26-3, muestra en zoom, la información de configuración actual del grid: cantidad de cpus, de nodos de cálculo en línea y fuera de línea, con la fecha.
- La figura 26-4, muestra la carga del clúster con: el número de procesos en ejecución, número de CPU en uso y nodos de cálculo involucrados.
- La figura 26-5, muestra el uso de Memoria RAM: cacheada, en uso, en Swap, compartida.
- La figura 26-6, muestra el uso de CPU en porcentajes: user, system, wait, idle.
- La figura 26-7, muestra el uso de la interfaz de red activa sobre el clúster: In (input) y Out (output) en bytes por segundo.

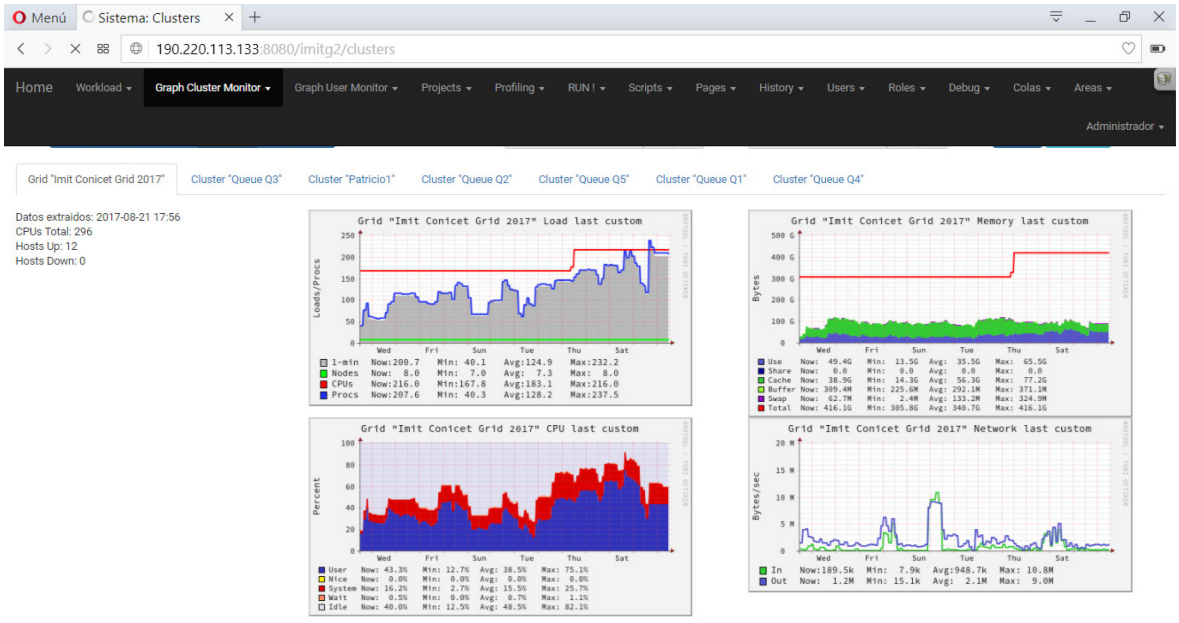


Fig-26-1. Ganglia, reporte general de Grid.

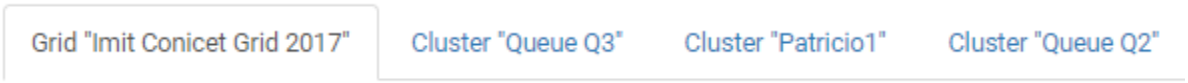


Fig-26-2

Fig-26-3

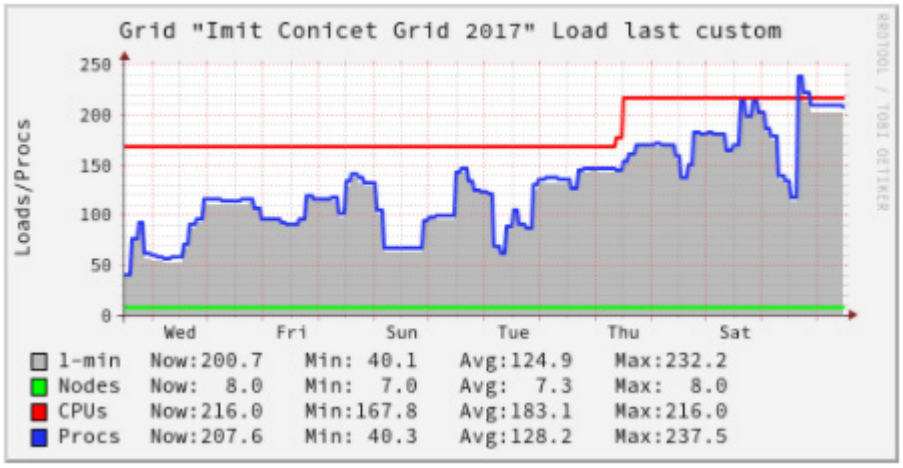


Fig-26-4

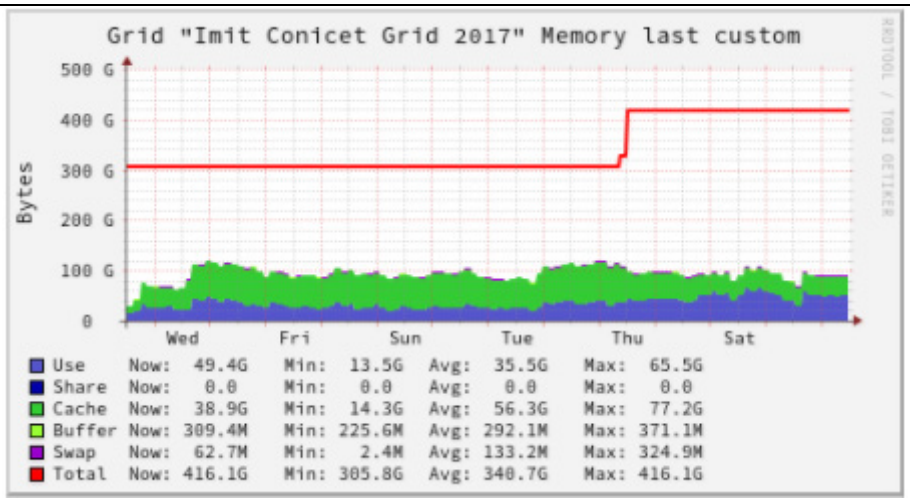


Fig-26-5

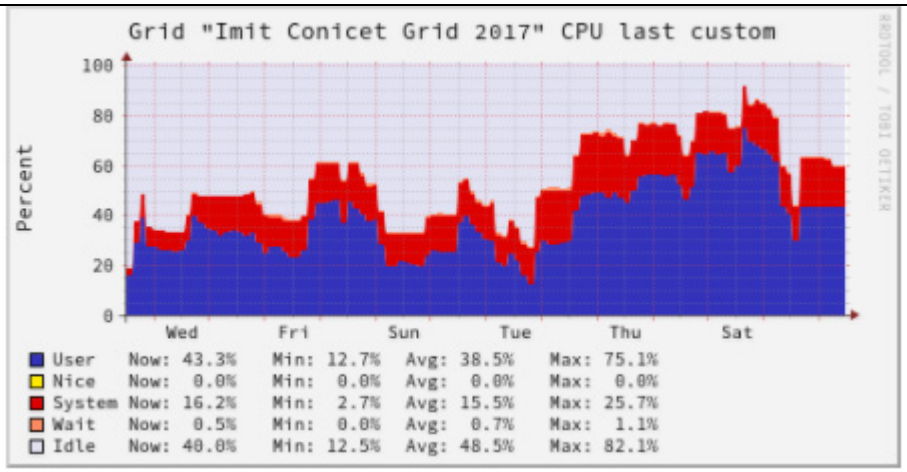


Fig-26-6

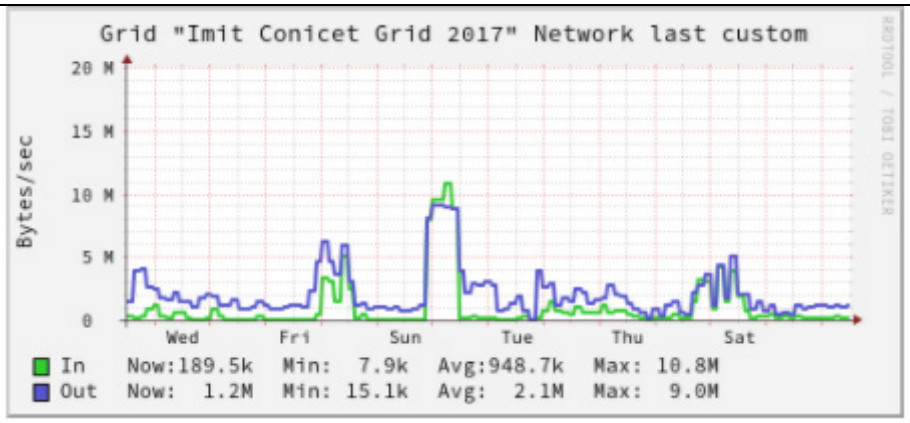


Fig-26-7

En la figura 26-1 es posible observar cajas de texto “Inicio” y “Fin”, donde es posible precisar tiempos específicos. Posee una botonera, en el margen superior izquierdo, para seleccionar ventanas de tiempo específicas: última hora, 2hs, 4hs, día, semana, mes, año; a partir de la selección, Ganglia reconfigura la información a mostrar con los valores almacenados en los archivos RRD para el período de tiempo elegido.

Es posible advertir el valor que tiene la visualización de estas variables, configurables bajo demanda, en ventanas de tiempo específicas. En este sentido, y en función al estudio realizado del producto, se debe mencionar que, a pesar del buen nivel de información en detalle que ofrece, Ganglia no brinda respuesta a consultas como las siguientes:

- ¿Qué usuarios hacen uso del CPU?, en ventanas de tiempo específico.
- ¿Qué software está siendo utilizado en ese período de tiempo en ese Nodo13?

Para dar respuesta a estos interrogantes, es implementado un sistema de ejecución de scripts, dado que Ganglia no proporciona este nivel de información de forma automática, aspecto desarrollado en el punto 3.b. Sin embargo, es posible adosar al framework de Ganglia toda información adicional, obtenida sean por métricas o por scripts elaborados a medida. En este sentido, Ganglia permite diseñar, si se quiere, un tablero de control de monitores a medida del usuario, modificando la programación PHP que trae por omisión.

4 ORGANIZACIÓN DE TRABAJOS EN EL CLÚSTER.

Este capítulo describe la vinculación de la aplicación con las herramientas de gestión de clúster y su interacción. Muestra en detalle el funcionamiento de la infraestructura completa. Describe la ejecución en distintas modalidades, en particular 'Profiling' para perfilado de aplicaciones. Enumera las formas de acceso a la aplicación.

4.1 Organización y ejecución de aplicaciones, generación de estadística.

En centros de computación con infraestructura de clúster HPC es necesario desplegar herramientas de software que colaboren en la administración y en el funcionamiento integral. Cuando es evidente que la cantidad de código a ejecutar supera a la capacidad de cómputo existente, se hace necesario desplegar software de gestión para la organización, priorización y coordinación en la ejecución de programas, a efectos de lograr resultados esperados por todos los usuarios. La solución de gestión, es desplegada en un formato de capas, que se montan por encima del sistema operativo del nodo anfitrión "Headnode" y de los nodos de cálculo. Cada una de las capas consume recursos de memoria RAM y CPU a la vez que logran sus resultados. En determinadas circunstancias, es necesario desarrollar un diagnóstico de la situación, que esté cercano a la realidad del grupo de usuarios que hará cálculo con un sistema en clúster, previendo funcionalidades de software que ayuden a lograr esos objetivos, y que no derive en un consumo desmedido en sus recursos de computación.

Para lograr la ejecución de aplicaciones de cálculo en un sistema en clúster de manera planificada y coordinada, existen técnicas que van de menor a mayor grado de complejidad. Quizá la más básica de todas, es el uso de scripts de sistema operativo, para organizar tanto el inicio, como el fin de las ejecuciones, y lograr con ello continuidad de trabajo. En la etapa de inicio, es posible consultar el estado de un nodo de cálculo acerca de sus recursos: memoria, cpu, disco y red; una vez chequeada la disponibilidad, se lanza la ejecución de la aplicación sincronizando con el tiempo del sistema para lograr, por ejemplo, métricas de tiempo de ejecución. El script podría incorporar comandos que informen mediante algún servicio de comunicación, como ser correo electrónico o simplemente difusión web, la finalización de la ejecución, a los efectos de quedar disponible para otros trabajos.

La experiencia en el uso de las aplicaciones de cálculo, nos da idea del tiempo en que demanda una aplicación y un set de datos en ser ejecutados; sin embargo, suele ocurrir que la ejecución de la aplicación se detiene de forma anormal, y el sistema permanece inactivo hasta recibir nuevas directivas de usuario, situación que puede demandar altos costos operativos en cuanto al consumo de recursos como ser energía eléctrica y de aire acondicionado, convirtiéndose así en una situación no prevista para los usuarios. En este contexto, quizá un indicador que es posible utilizar, es el número de usuarios del clúster que requieren ejecutar código; si el número es mayor a cuatro (4) personas, será entonces necesario integrar al clúster alguna herramienta de software de gestión de trabajos. En determinados escenarios, puede que dos usuarios sean suficientes para recomendar el uso

de un gestor cuando, por ejemplo, el tiempo de ejecución que demanda la ejecución de los trabajos individuales es alto (semanas o meses).

Aspectos de organización: software de gestión de trabajos Torque PBS.

La aplicación que se propone utiliza “Torque PBS”, o Terascale Open-source Resource and QUEue, para la organización de ejecución de programas en modalidad batch, distribuidos sobre los nodos de cálculo del clúster. Se trata de una arquitectura Cliente-Servidor, que implementa una porción sobre el nodo “Headnode” y la otra en los nodos de cálculo. El programa “pbs_server”, es el demonio que corre en el nodo “Headnode”, quien se encarga de hacer ejecutar los programas de cálculo de usuario; una vez finalizada la ejecución, dispone el archivo de salida en las carpetas de usuario predefinidas en el script de lanzamiento.

El procedimiento la inicia el usuario, con la edición de un script, definiendo básicamente la cola de trabajo a utilizar para la ejecución, las capacidades de CPU y memoria RAM requeridas por el programa de cálculo, y la localización de la aplicación a ejecutar con su set de datos. El gestor de colas es representado por el demonio “pbs_sched” o “maui” que, dependiendo de la disponibilidad de los recursos invocados por el usuario, determinará cuándo dar inicio a la ejecución. El sistema reporta el estado de los trabajos a los usuarios, mientras son ejecutados en los nodos de cálculo. Una vez finalizado, los archivos de salida resultantes se almacenan en carpetas temporales disponibles para el usuario [Torque].

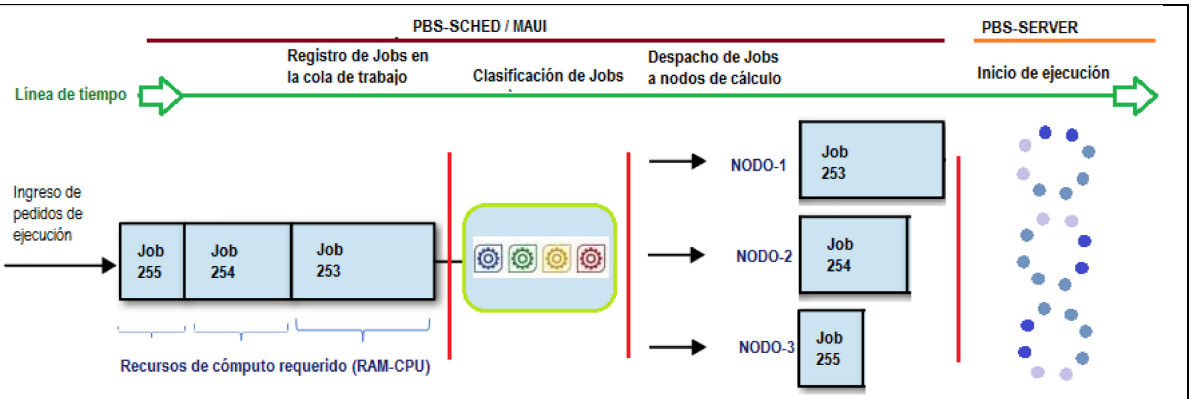


Fig-27. Ejecución de trabajos de cálculo con gestor de colas Torque PBS.

En Linux, una línea típica de comando de usuario para la ejecución de un programa, podría ser:

$ ./user_app.exe input1.dat > stat-out.log

El comando anterior lanza a ejecución la aplicación “user_app.exe”, la cual requiere un archivo de datos “input1.dat”, y define que el resultado sea guardado en un archivo de salida “stat-out.log”. Ahora bien, en la aplicación que proponemos, para ejecutar el programa de cálculo sobre el clúster, primero, el usuario debe configurar un script de lanzamiento, en el cual especifica parámetros de funcionamiento y capacidades requeridas.

Para ello, la herramienta “Torque-PBS”, integra comandos que son invocados en la forma “#PBS..”, que son dispuestos en el script sería como el siguiente:

```
#!/bin/bash
### Ejemplo de script Torque PBS – Los comandos PBS se interpretan según #PBS
#PBS -S /bin/bash              "shell a utilizar para ejecutar el script"
#PBS -N Vector_01 "Nombre del JOB – usado para identificarlo en el sistema de
colas"
#PBS -q Parallel_1 "Nombre de la cola o agrupamiento donde será ejecutado"
#PBS -m ae          "Requiere un correo electrónico al finalizar la ejecucion"
#PBS -M leopoldo@headnode.local
#PBS -o introduccion.$PBS_JOBID "Archivo de salida con el volcado del trabajo"
#PBS -e intro-error.$PBS_JOBID  "Archivo de salida con el volcado de error"
### requerir 2 nodos en total y 16 procesadores en cada nodo, y la cola de
trabajo
#PBS -l nodes=2
#PBS -l ppn=16:q_nodo12
WORKDIR="/home/$USER
### programa a ejecutar, el input y el archivo de salida
./user_app.exe input1.dat > $WORKDIR/stat-out.log
exit 0
### Fin del script PBS
```

El siguiente gráfico, ilustra sobre el ciclo de ejecución de programas en clúster HPC mediante software de gestión de trabajos, la función de cada componente en el sistema, y los pasos previstos: el lanzamiento del script, su análisis sintáctico, encolado para ejecución, la ejecución misma del trabajo en los nodos de cálculo, y el almacenamiento del resultado en la carpeta establecida por el usuario:

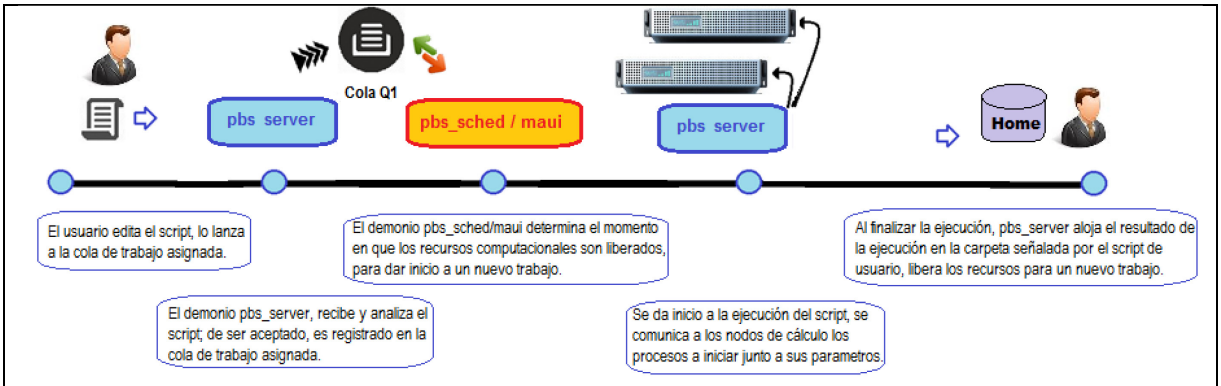


Fig-28. Colas, ciclo completo de ejecución de programas mediante gestor de trabajos.

4.2 Ejecución de trabajos de usuario.

En la aplicación que proponemos, la ejecución de programas de usuario es realizada en tres formas distintas, según la función específica de ejecución. Para el caso de actividades de test de la aplicación, se sugiere la utilización de la opción de menú “Debug”, puesto que el sistema direcciona la ejecución a una cola de trabajo prevista para test, y no sobre nodos en

producción. Esta cola, posee capacidades limitadas, y es usada con fines de prueba de I/O de archivos.

Las otras opciones corresponden a ejecución en modalidad “Profiling”, donde el usuario busca conocer a través de un perfil de aplicación, el funcionamiento de su programa; y la ejecución sobre el clúster mismo, cuando el programa de cálculo del usuario ha sido testeado y se encuentra en condiciones para ser lanzado sobre el clúster.

Ejecución en modo Debug:

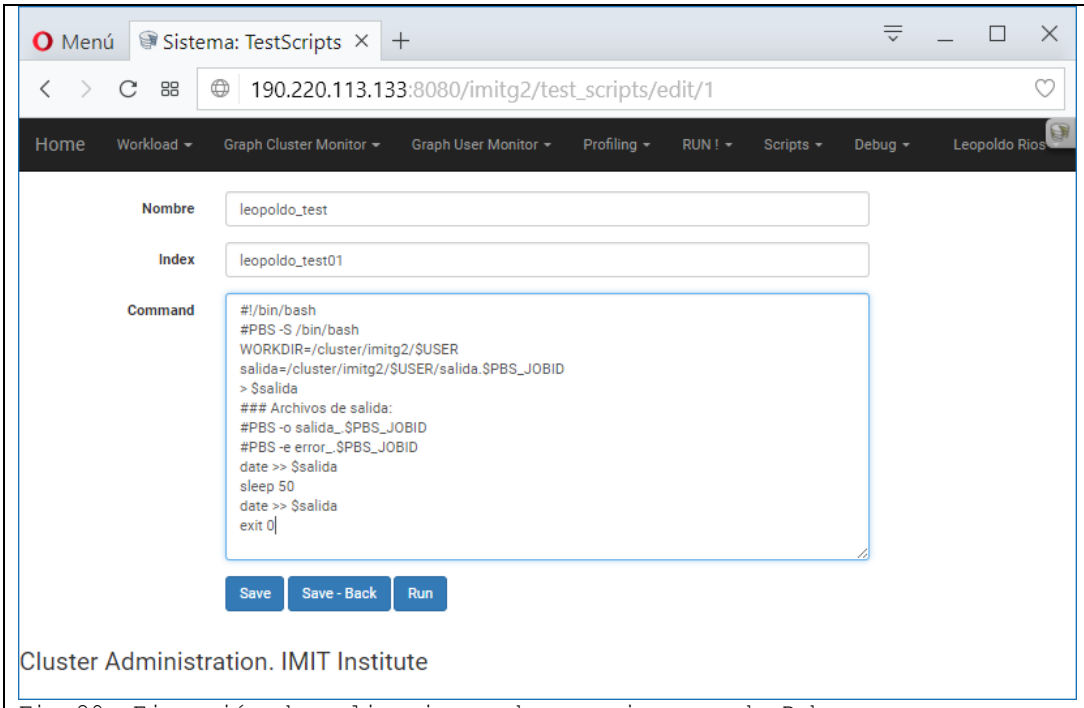


Fig-29. Ejecución de aplicaciones de usuario en modo Debug.

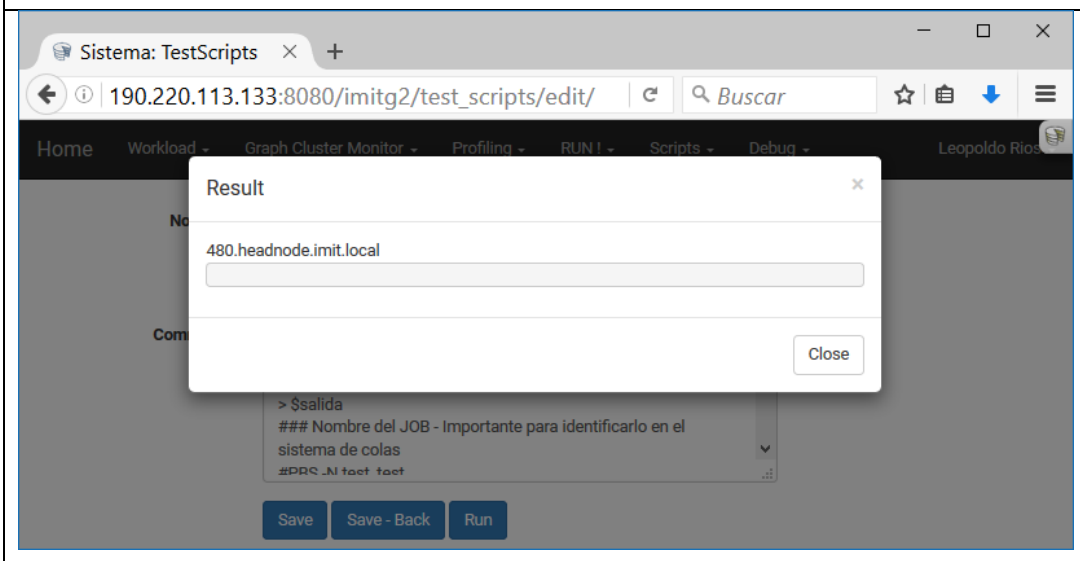


Fig-30. Ejecucion de aplicaciones de usuario en modo Debug.

Una vez que el script es generado y guardado, es posible desde la vista “Debug/Index”, modificar parámetros, y lanzarlo a ejecución con el botón RUN. El sistema, por omisión

lanza el trabajo a la cola ‘Test’, configurada especialmente para esta tarea de prueba. El resultado del trabajo, será alojado en la carpeta señalada por el usuario en el script.

El sistema prevé la ejecución de aplicaciones de usuario en modalidad ‘Profiling’, para ello se debe utilizar la opción “Profiling” del menú, el cual sugiere al usuario templates de scripts, los cuales integran variadas formas de profiling, basados en el programa ‘perf’. El usuario, debe modificar e incorporar al template, el nombre de su aplicación, y el archivo de datos -si sea el caso-, y el lugar donde alojar el archivo de reporte, para futuras observaciones.

Ejecución de aplicaciones en modalidad Profiling.

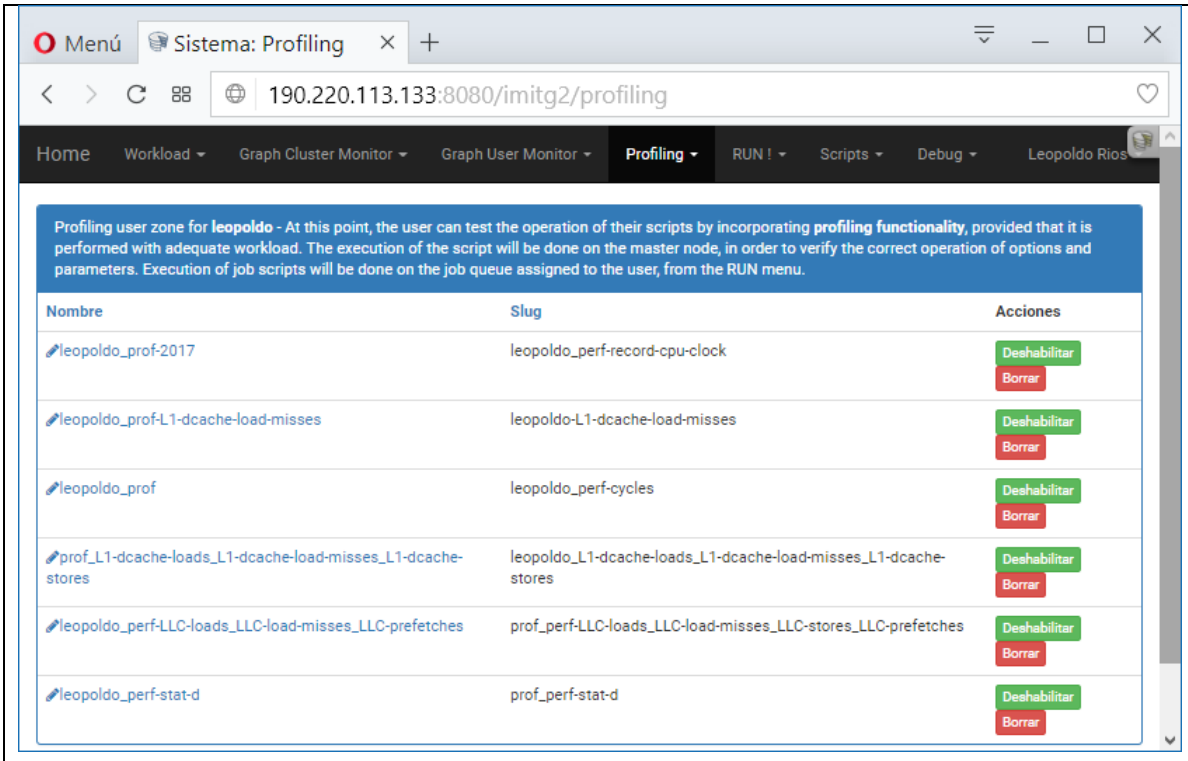


Fig-31. Ejecución de aplicaciones de usuario en modo Profiling.

La vista del índice de scripts de profiling, permite la selección de elemento para su ejecución, ver figura 31. Una vez seleccionado el script, es posible editar parámetros, guardarlo y lanzarlo a ejecución con el botón RUN. La vista genera un resultado, el cual informa la aplicación y los parámetros de ejecución.

La opción de profiling, incorpora una funcionalidad, que es considerada de suma importancia: el historial de profiling por usuario, generado a partir de los envíos que realiza el usuario. Se acceden desde el menú Profile / Historial, y es una vista diseñada para rescatar de una tabla, los registros de profiling de usuario, presentados por fecha, primero los más recientes. Puede resultar útil, para el estudio de funcionamiento de la aplicación de cálculo, a medida que es modificada, luego compilada, y ejecutada; puede llegar a presentar mejoras en los valores arrojados.

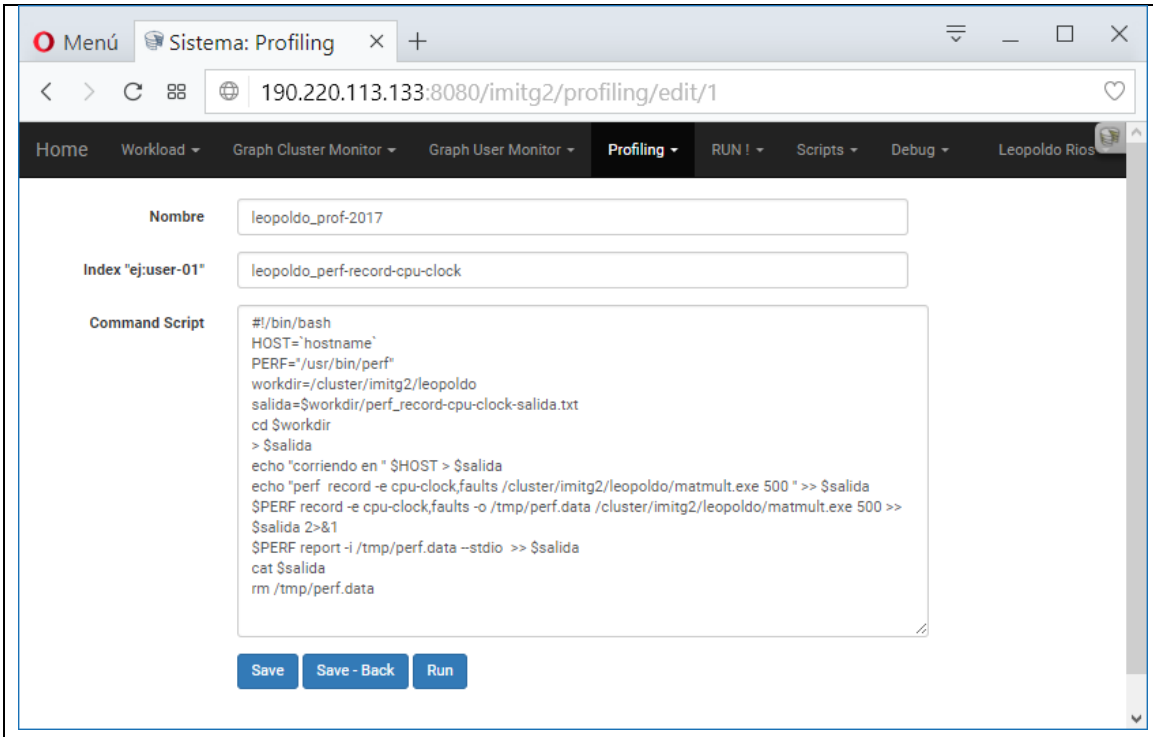


Fig-32. Ejecución de aplicaciones de usuario en modo Profiling.

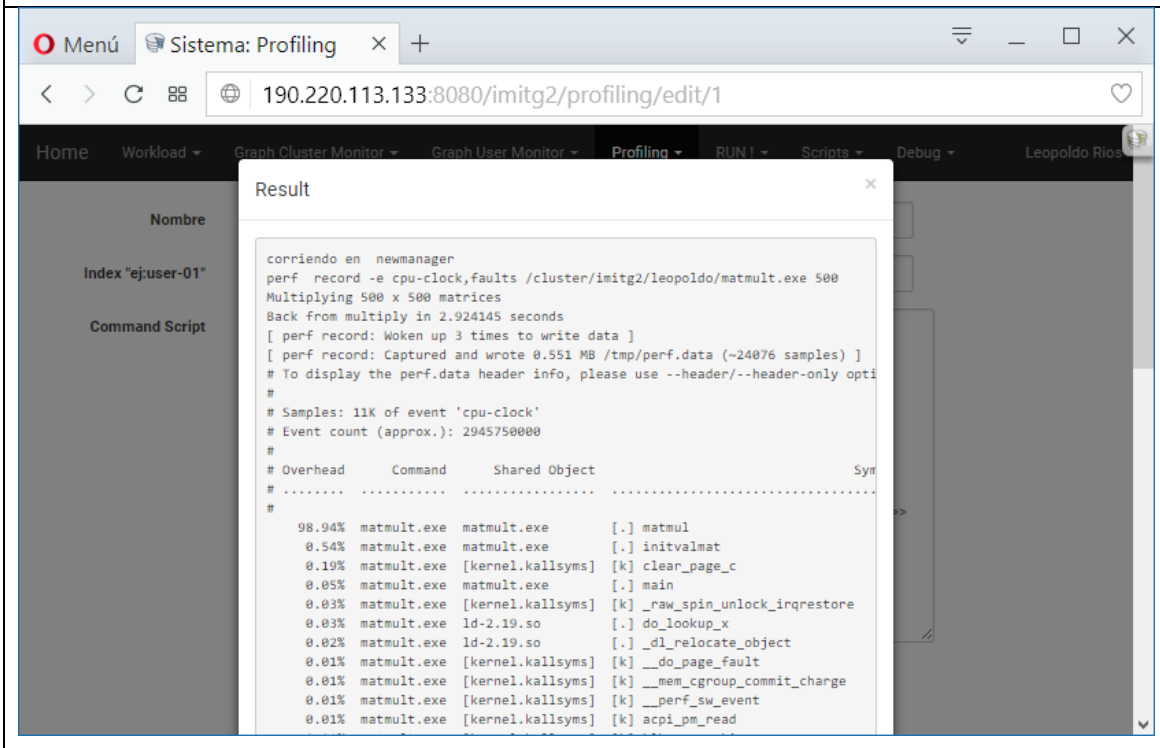


Fig-33. Ejecución de aplicaciones de usuario en modo Profiling.

Ejecución de aplicaciones de usuario en clúster.

Una vez que la aplicación a correr en el clúster, es testeada y se tiene idea del funcionamiento a partir del perfil arrojado en la vista 'Profiling', es posible dar paso a la ejecución del programa, con la opción del menú "RUN !". El sistema prevé que los scripts de

usuario se realicen sobre la opción de menú “Scripts”, que luego pueden ser usados para la ejecución sobre el clúster. El índice de scripts es presentado con la opción “RUN !”:

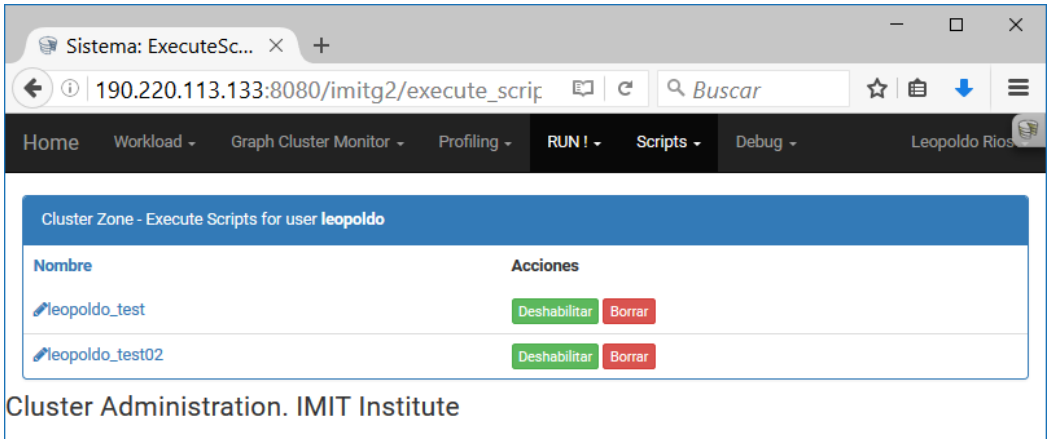


Fig-34. Ejecución de aplicaciones de usuario en clúster.

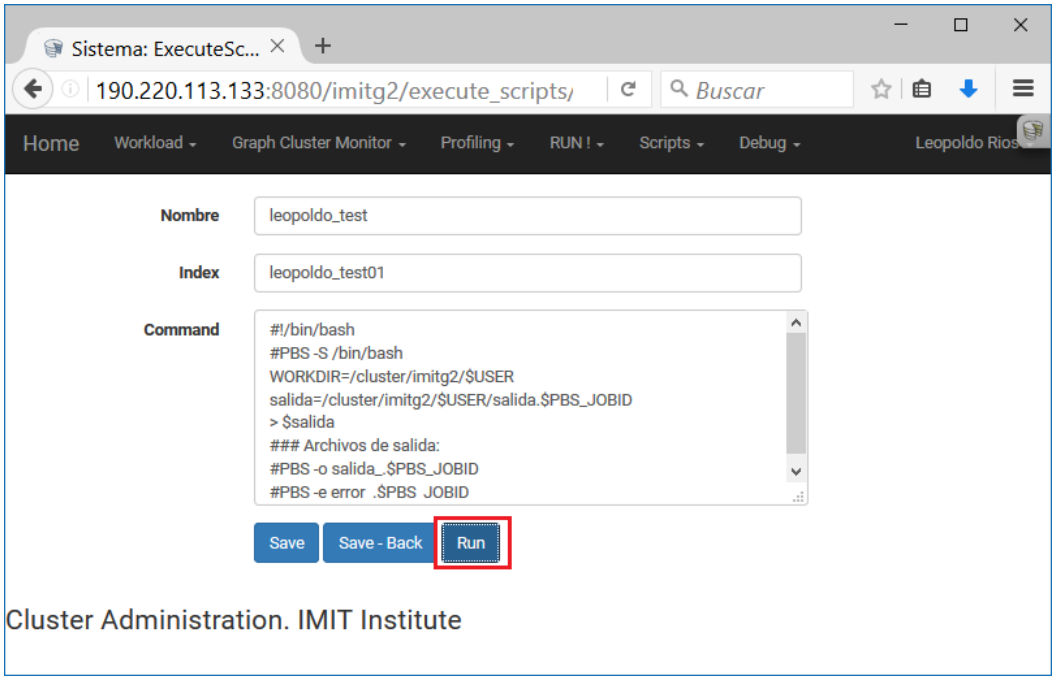


Fig-35. Lanzamiento de script de usuario.

En la figura 35, se observa la vista de edición del script a lanzar a ejecución. Aquí es posible modificar valores, lo que requiere primero se registren las modificaciones con “Save”, y luego lanzar a ejecución con “Run”.

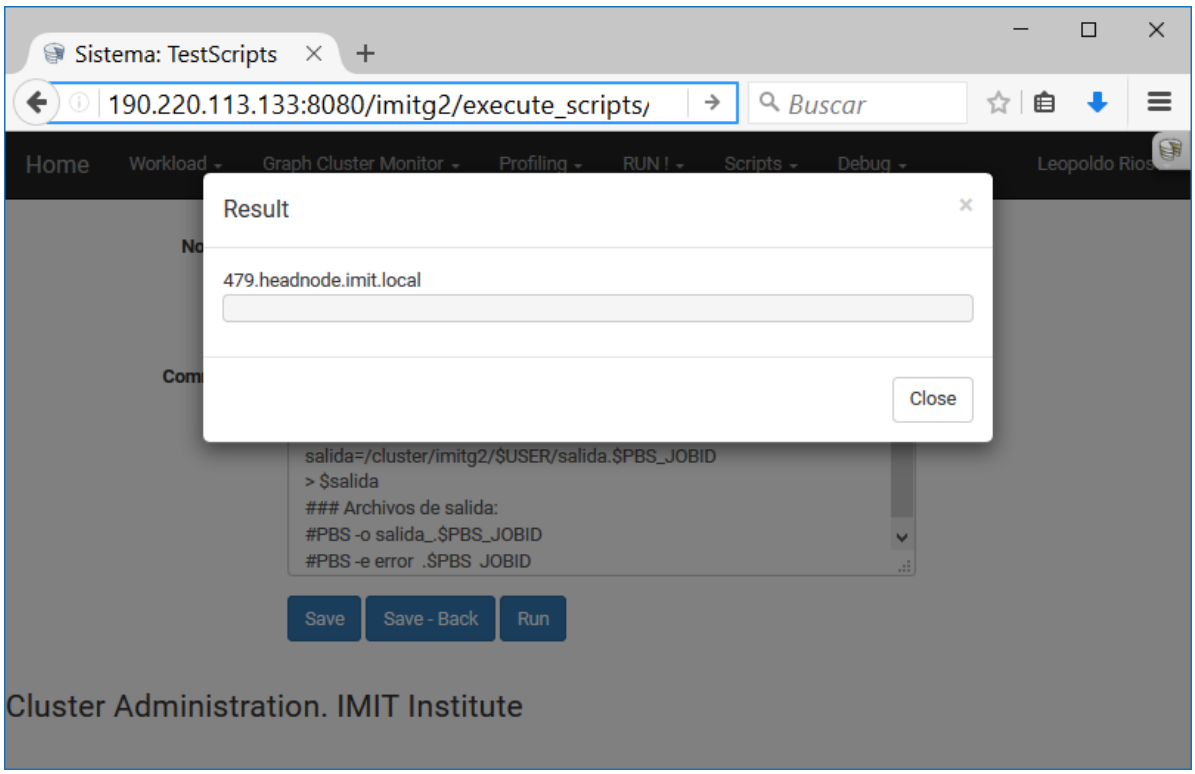


Fig-36. Lanzamiento de script de usuario.

La figura 36 muestra el resultado del proceso de despacho del script al demonio 'pbs_server', el cual informa el número de JobID con el cual podrá ser monitoreado. En caso de existir algún error, por motivos de sintaxis o de capacidad, el pop-up informará en detalle.

4.3 Monitoreo y seguimiento.

Una vez que el script es lanzado al clúster, es analizado por el demonio "pbs_server/maui", se verifican aspectos sintácticos y de capacidades, y en caso de pasar la verificación, el script es registrado en la cola de trabajo solicitada, y se le asigna un número de trabajo -JobID-, con el cual será identificado durante todo el proceso. Mediante el "JobID", es posible monitorear el estado del trabajo a través de vistas creadas especialmente.

En la figura 37, se observa información generada por el enlace de la opción de menú "Workload /Jobs running". La vista despliega información acerca de los trabajos que se encuentran en ejecución en el clúster, uno por línea, identificado por número de JobID, usuario que lanzó el trabajo, cantidad de nodos involucrados, procesadores requeridos, memoria solicitada, tiempo solicitado y tiempo transcurrido desde su inicio. La información es actualizada cada minuto.

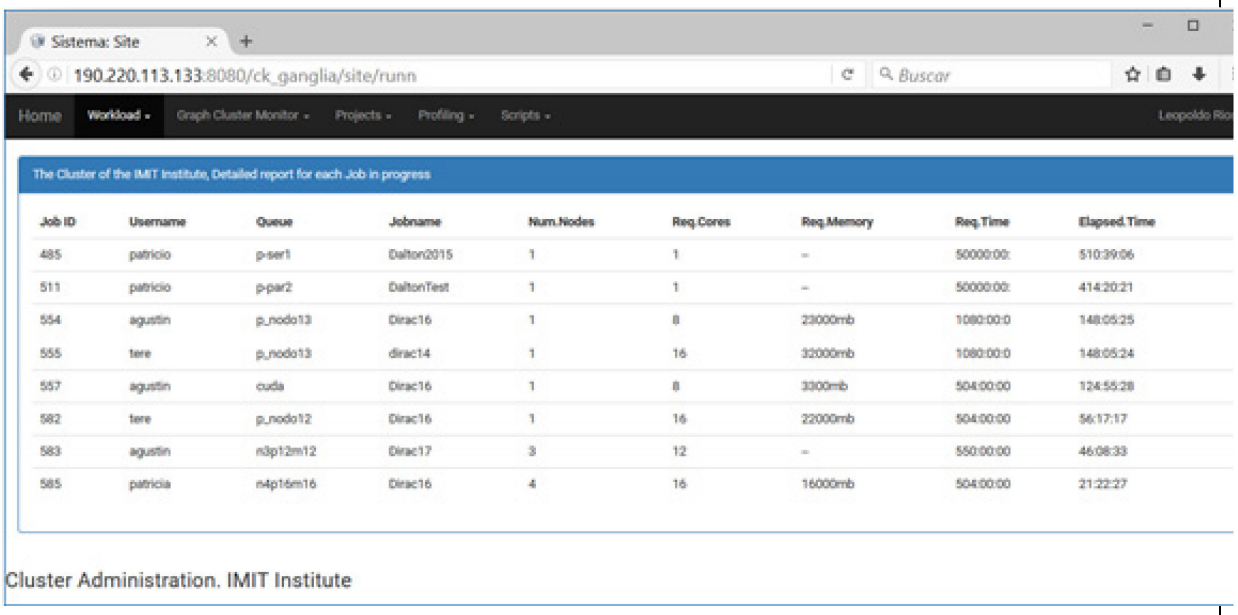


Fig-37. Monitor del clúster, trabajos en ejecución.

La vista de la figura 37, obtiene datos del archivo “jobs-csv.csv”, el cual es generado por un script -número 4 de la Lista de Scripts- ejecutado de manera periódica -cada minuto-, para la obtención de estos datos. Así, la vista reporta datos prácticamente en tiempo real.

El menú “Workload” posee la opción “Jobs running - details”, la cual carga una vista con los números de JobID que se encuentran en ejecución en el clúster.

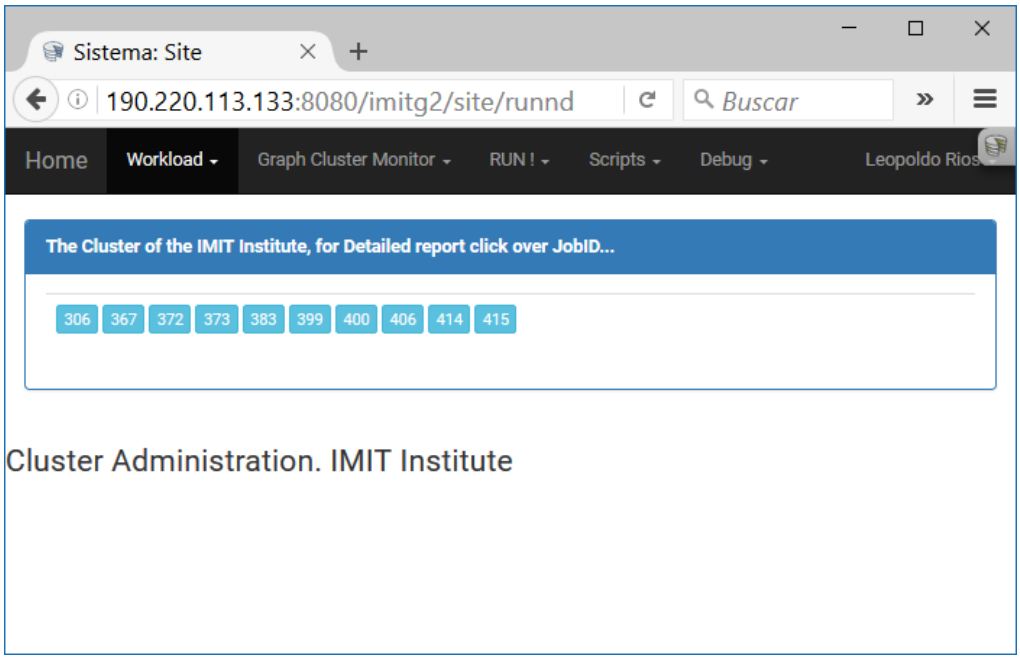


Fig-38. Monitor del clúster, información en detalle.

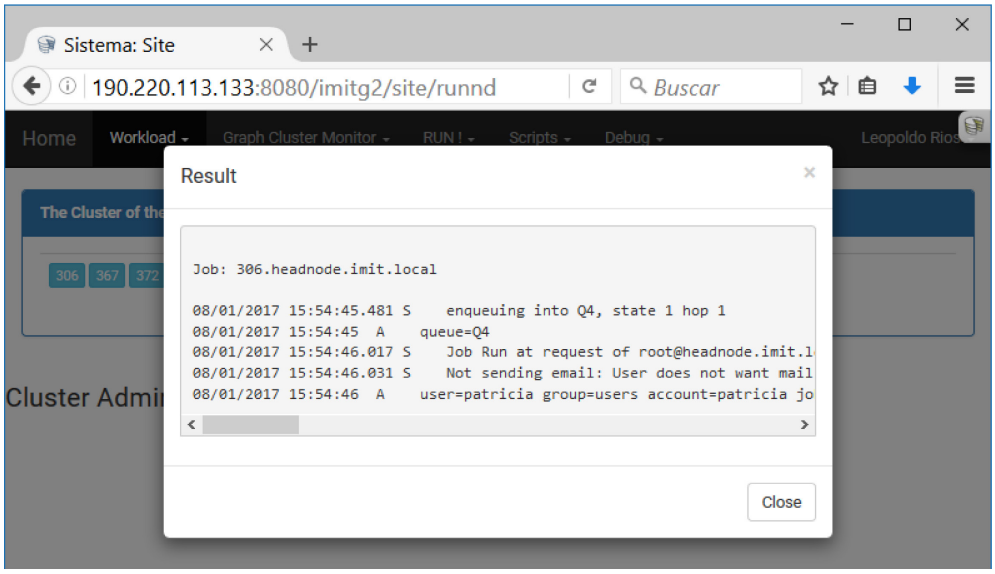


Fig-39. Monitor del clúster, pop-up con detalle.

En la figura 38 es posible observar la vista “Workload/Jobs detailed” con la lista de JobID en ejecución, y en la figura 39, es desplegado el pop-up generado por la selección de un JobID en particular. La información desplegada en el pop-up es generada por el script ”job-process-1.sh”, del listado de Scripts. El contenido, menciona fechas y estados que el Job ha tomado desde su lanzamiento: el momento en que fue ingresado a la cola de trabajo, momento en que fue habilitado para su ejecución, informa que el usuario no ha solicitado reporte mediante correo electrónico interno, entre otros.

Es posible, a través de la vista “Workload / Jobs queued”, observar los trabajos aceptados que esperan turno para su ejecución en el clúster. Esto se debe a que los recursos solicitados para su ejecución no se encuentran disponibles. El demonio “pbs_server o maui” determinan el momento en que esos recursos son liberados, y proceden a lanzar a ejecución el trabajo encolado, pasando al demonio “pbs_server” la información necesaria. En la siguiente figura xx es posible observar un ejemplo:

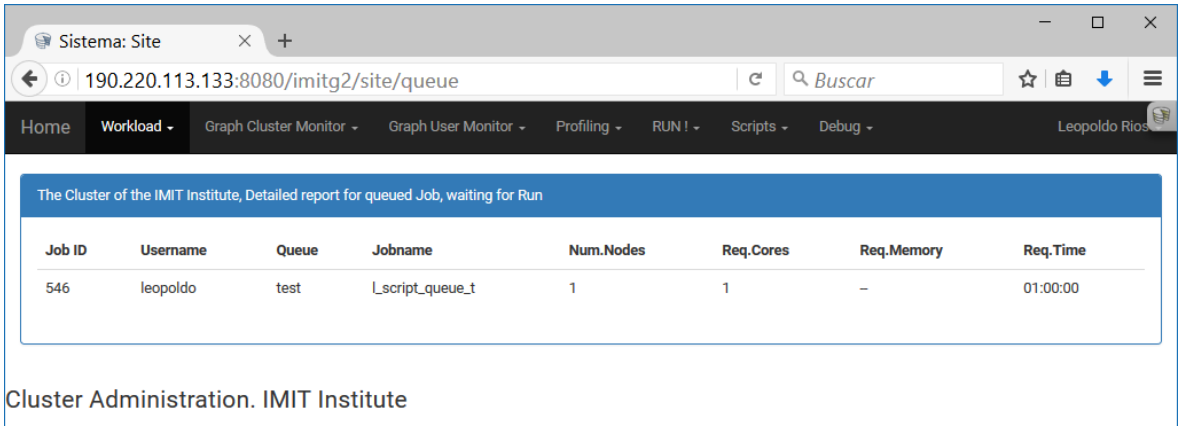


Fig-40. Monitor del clúster, trabajos en cola de espera.

Otra vista que ofrece el menú “Workload”, es la vista de “Job end prediction”, con información acerca de los trabajos en ejecución y la planificación hecha para su culminación. Todas las colas de trabajo son configuradas con un tiempo de espera máximo a través de la variable “resources_default.walltime=horas:minutos:segundos”, el script que procesa la información de los trabajos en ejecución, toma en cuenta el valor de walltime y el valor de “Elapsed time”, por lo que su diferencia determina el tiempo de fin de ejecución del trabajo. Esta vista se corresponde con la siguiente figura:

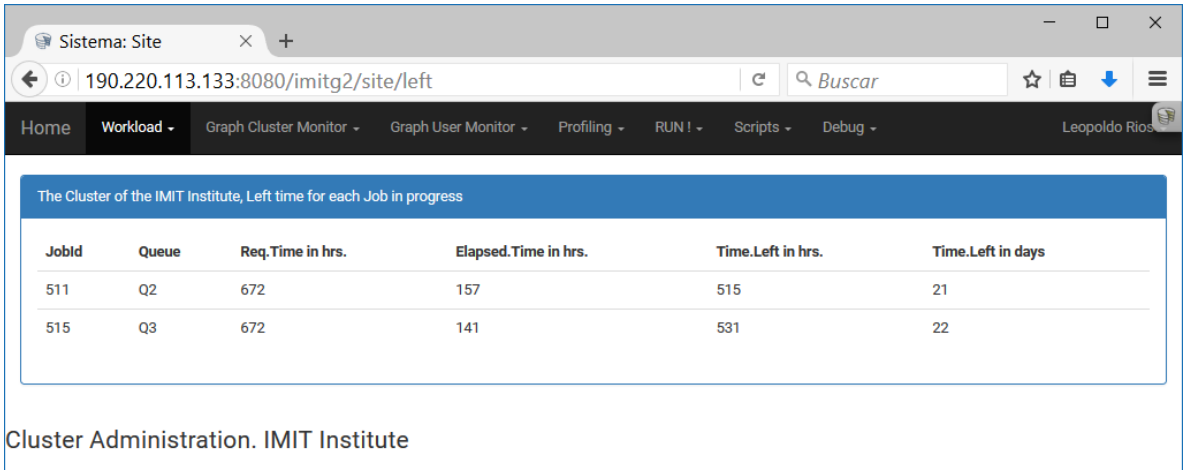


Fig-41. Monitor del clúster, predicción de finalización de trabajos.

4.4 Acceso al sistema, protocolos de seguridad.

Infraestructura requerida en la configuración de un clúster HPC. Los nodos de cálculo se encuentran localizados en una sala acondicionada especialmente. La instalación eléctrica es uno de los aspectos quizá más importantes en la estrategia a llevar adelante, y lo sigue la ambientación del lugar en cuanto a refrigeración y control de acceso de personal. Los equipos están organizados en un gabinete, acomodados de tal manera que facilita, tanto su acceso frontal, como su acceso posterior.

La conexión física demanda organización en la conducción de distintos modelos de cables que soportan los servicios: para la red de datos, el conductor es de cobre; y la red eléctrica, también en cobre, conecta los equipos al UPS, localizado en el mismo gabinete. En la siguiente figura 42, es posible apreciar el rack de servidores (derecha), y el rack de comunicaciones (izquierda) alojados en el Datacenter del caso de estudio de referencia.

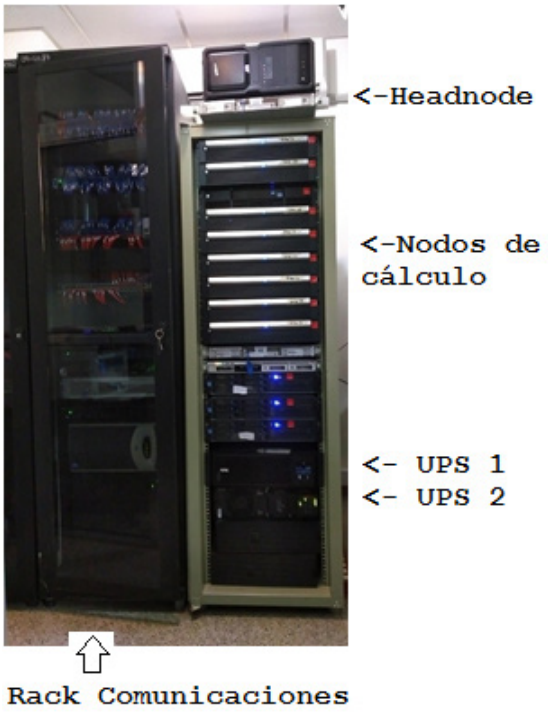


Fig-42. Racks del Datacenter.

En la figura 42 se observa, que se encuentran instaladas dos (2) unidades UPS, de tipo Online de doble conversión, que se reparten las cargas de los nodos de cálculo instalados. Estas UPS están conectadas a tomas de alimentación normalizados, vinculados a conductores que provienen del grupo electrógeno, instalado fuera del edificio. Esto permite dar capacidad de autonomía de alrededor de 6 horas a partir de un fallo eléctrico en el suministro de corriente.

La comunicación entre nodos está dada por una red cableada, basada en un esquema como lo muestra la figura 43, la cual permite especificar capacidades de ancho de banda y protocolos de servicios diferenciados para cada segmento. La estrategia elegida define la existencia de una “red de acceso”, desde donde los usuarios del sistema, acceden a los servicios publicados por el nodo “Headnode”, equipo anfitrión del clúster. A partir del inicio de sesión con credenciales habilitadas, es posible acceder a la “red del clúster”, donde se difunden los servicios del clúster.

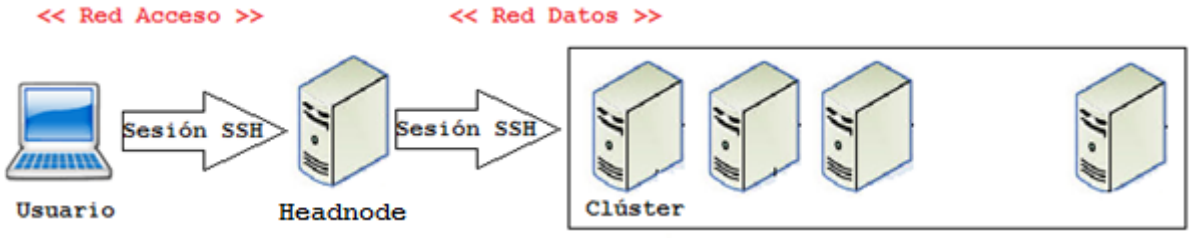


Fig-43. Conformación de Redes del clúster.

La estrategia de comunicación entre nodos de cálculo y el nodo Headnode, se realiza a través de paso de mensajes, en arquitectura Cliente-Servidor. El protocolo de transporte y comunicación de red es TCP/IP, lo utilizan los programas de acceso de usuario y el sistema operativo que corre en los nodos. La siguiente lista de servicios y software es utilizada en la implementación de la solución:

- El sistema operativo y servicios:
    - Protocolo TCP-IP, para direccionamiento IPv4 y transporte de datos.
    - Servicio NIS, de autenticación centralizada de usuarios.
    - Servicio NFS, de compartición de carpetas y archivos en red.
    - Servicio Web, para la difusión y gestión de contenido.
    - Servicio SSH, para la ejecución remota de scripts.
    - MySQL, gestor de bases de datos relacional.
    - Scripting Bash, obtención de información de estado de usuario y trabajos.
    - Librerías MPI, para la compilación de software de cálculo en modalidad paralela.
- Servicio Torque PBS, de gestión de trabajos en modalidad batch del Clúster.
- Ganglia y RRDtool, métricas de uso de recursos.
- Software de usuario, software de cálculo, representativo de lo que puede ser utilizado en clusters HPC:
    - Dirac Program (versiones 14, 15, 16 y 17)
    - Dalton Programa (versiones 2013, 2016)
    - Gaussian (versiones 03 y 09)

El software de cálculo que los usuarios corren en los nodos, comunica sus mensajes mediante protocolo TCP-IP, y tienen como base para el direccionamiento IPv4, la lista de hosts del archivo “/etc/hosts” del nodo “Headnode”, el cual se utiliza como resolver de la red. Sin TCP/IP no sería posible la ejecución de ninguno de los productos de software mencionados. En la figura 43 se aprecia el sentido de inicio de conexiones TCP, para lo cual se utiliza una tabla de direcciones como la siguiente:

```
#
# hosts         This file describes a number of hostname-to-address
#               mappings for the TCP/IP subsystem.  It is mostly
#               used at boot time, when no name servers are running.
#               On small systems, this file can be used instead of a
#               "named" name server.
# Syntax:
# IP-Address  Full-Qualified-Hostname  Short-Hostname
#
127.0.0.1       localhost
# IPv4 hosts - Clúster
192.168.10.10   headnode.local  headnode
192.168.10.20   nodo1.local nodo1
192.168.10.21   nodo2.local nodo2
#
```

Para cada nodo de cálculo, se inscribe una dirección IPv4 en el archivo “/etc/hosts” del nodo “Headnode”. Desde ese lugar, el sistema operativo resuelve la dirección IPv4 del nodo al

que se desee conectar por nombre. Se inician las conexiones TCP hacia los nodos para correr las aplicaciones de cálculo.

Un paso importante es, definir en el nodo “Headnode”, las claves SSH para cada usuario que requiera la ejecución remota de aplicaciones en los nodos de cálculo. Esta funcionalidad constituye un ambiente “password-less” o proceso de login “automático”. El proceso de login, requiere el ingreso del ID de la cuenta de usuario y su contraseña. La automatización de este proceso requiere, que una vez generadas las claves SSH, la parte pública sea copiada a los nodos remotos, proceso que se describe en los siguientes comandos:

```
leopoldo@headnode:~> ssh-copy-id -i .ssh/id_rsa.pub nodo13
The authenticity of host “nodo13” can”t be established.
RSA key fingerprint is db:d0:45:cc:78:cf:10:4a:c6:80:c5:ff:a6:10:3a:43.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added “nodo13” (RSA) to the list of known hosts.
leopoldo@nodo13”s password:
Now try logging into the machine, with "ssh “leopoldo@nodo13”", and check in:
~/.ssh/authorized_keys
to make sure we haven”t added extra keys that you weren”t expecting.

leopoldo@headnode:~> ssh nodo13 uptime
 10:35am  up 175 days  3:35,  0 users,  load average: 23.47, 23.61, 23.66

leopoldo@headnode:~> ssh nodo13 free -m
             total       used       free     shared    buffers     cached
Mem:         64458       9281      55176          0        305       3735
-/+ buffers/cache:       5241      59216
Swap:         2054        143       1911
```

En la figura anterior se observa primero que, desde el nodo “Headnode” se copia la parte pública -archivo id_rsa.pub- de la clave al nodo “nodo13” para el usuario “leopoldo”. La clave es registrada y define un ingreso de registro al archivo “authorized_keys” propio del usuario, el cual recordará la próxima vez que se intente acceder a dicho nodo. El siguiente comando invoca al programa “ssh” en dirección al nodo “nodo13” y se especifica la ejecución del programa “uptime”. Desde “nodo13”, se devuelve la salida del programa “uptime” en forma automática, sin solicitar credenciales de login. La siguiente salida, en este caso del programa “free -m”, muestra la configuración de memoria (RAM y Caché) en formato Megabits. Aquí, “free -m” representa la parte del comando remoto ejecutado en nodo “nodo13”, y el resultado es devuelto a consola del nodo “Headnode”. El proceso de generación y copia de clave pública en nodos de cálculo, debe ser hecho para cada usuario del sistema en clúster, y para cada nodo donde el usuario solicitará ejecución remota de aplicaciones.

La red de Acceso está definida por una Red IP versión 4, y la red de Datos del Clúster por otra, es decir, se encuentran en dominios de broadcast diferentes [Dominios]. Como un ejemplo, podemos definir que la red IP de Acceso es la red 10.20.30.0/24, para la cual el nodo “Headnode” utiliza la dirección 10.20.30.100/24 para la interfaz conectada; y para la red de Datos utilizamos la red 192.168.10.0/24, utilizando el nodo “Headnode” la dirección 192.168.10.5/24 en una segunda interfaz de red. Esta estrategia es realizada para limitar las

sesiones TCP hacia la red del clúster, es decir, que para interactuar con la red del clúster es necesario tener una sesión activa en el nodo “Headnode”, ver figura xx.

El protocolo TCP/IP se monta sobre tecnología Ethernet a velocidad de 1GB, de un único canal de comunicación compartido. Si bien es un único canal de comunicación para todos los nodos, el Ethernet dispone en cada puerto de conexión, un ancho de banda de 1GB de recepción y 1GB de transmisión. La matriz de conmutación del switch Ethernet, dispone de memoria suficiente para el caso de tener conectadas la totalidad de los puertos disponibles. En un contexto más general, la red del Instituto IMIT del caso de estudio, se corresponde con la siguiente figura:

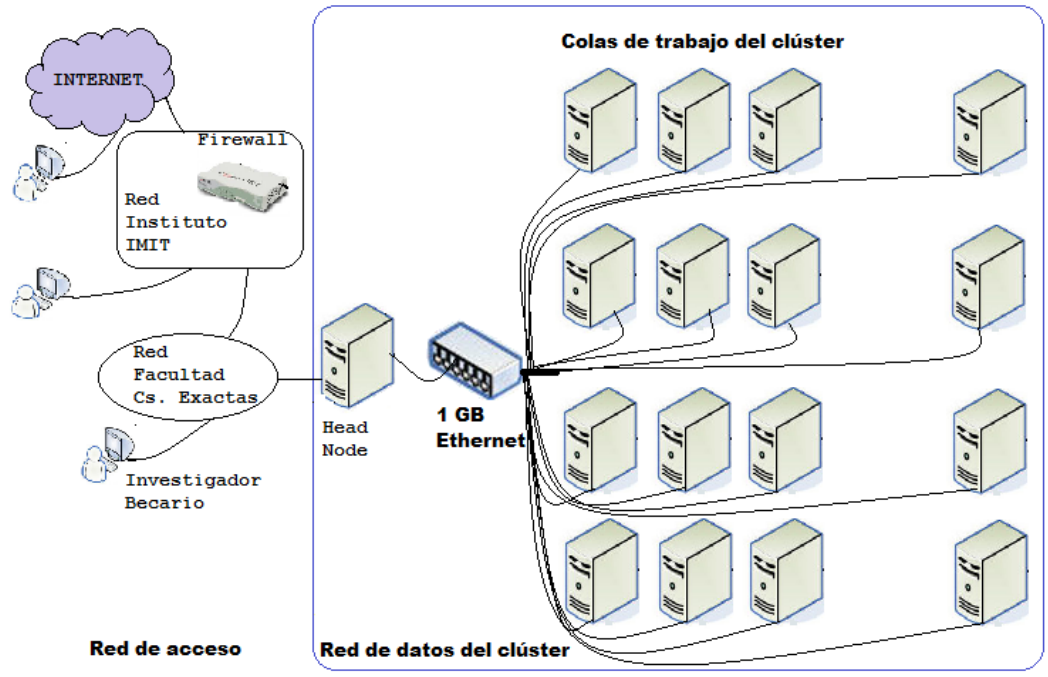


Fig-44. Red de Acceso al Clúster del caso de estudio.

Con el objeto de dar movilidad a Investigadores, Becarios y Administrativos, es posible iniciar sesión SSH al nodo “Headnode” desde múltiples localizaciones: La red de la Facultad de Ciencias Exactas donde se encuentran oficinas del Instituto IMIT, desde la Red propia del Instituto, o inclusive desde Internet.

Protocolos de la red de datos del clúster.

Los protocolos utilizados en el despliegue de la red de datos son:

- NFS, o Network File System Protocol. Es la implementación de un sistema de archivos distribuido. Permite la compartición de carpetas desde el nodo “Headnode” hacia los nodos de cálculo del clúster. Por ejemplo, la carpeta “/home” de las cuentas de usuario, es utilizada por todos los nodos del clúster, a los efectos de centralizar en una única localización los archivos de usuario. Este es un requerimiento, para el servicio NIS, implementado como forma centralizada de autenticación de usuarios. Es el sistema de archivos distribuido más utilizado en

ambientes Linux, por su facilidad y seguridad en la implementación, en este caso es utilizada la versión 4. El protocolo está diseñado para ser portable a través de diferentes máquinas, los sistemas operativos de red, arquitecturas y protocolos de transporte.

- NIS, Network Information Service. Es un protocolo de servicios de directorios cliente-servidor desarrollado por Sun Microsystems para el envío de datos de configuración en sistemas distribuidos tales como nombres de usuarios y hosts entre computadoras sobre una red. Es utilizado para la autenticación centralizada de usuarios y hosts en ambiente de clúster HPC. El nodo “Headnode” es el servidor NIS y los nodos de cálculo son los clientes NIS, proporciona prestaciones de acceso a los archivos passwd y groups a todos los nodos cliente de la red. Cada nodo cliente experimenta el uso de la misma carpeta “/home” de usuarios; cada usuario, sin importar el nodo donde se encuentre ejecutando su programa de cálculo, siempre utiliza la misma carpeta donde alojar los archivos tanto temporales como de resultados finales.

- TCP-IP, es el protocolo de transporte de datos y de red utilizado para la comunicación de mensajes entre los nodos del clúster. El nodo “Headnode” hace a su vez de router de la red del clúster, configurado como “reenviador IP”, aspecto necesario para que los nodos de cálculo puedan actualizar el sistema operativo desde Internet y otras funcionalidades de software. Esta desplegado un esquema de direccionamiento IPv4 en la red IP “192.168.10.0/24”, donde cada host utiliza una dirección, registrada en el archivo “/etc/hosts” del nodo “Headnode”.

- DHCP, Dynamic Host Configuration Protocol, permite la asignación dinámica de una lista de direcciones IP y las asigna a los clientes que lo requieren. Es utilizado en conjunto con el servicio TFTP, montados ambos sobre el nodo “Headnode” para brindar la funcionalidad de boot sobre red, para los casos en que un nodo de cálculo requiera la reinstalación de su sistema operativo por algún motivo. Es posible también utilizarlo para el montaje en red de imágenes de sistema operativo diversas, para casos de nodos de cálculo sin disco rígido local.

- DNS, o Domain Name Service, servicio utilizado en la traducción de nombre IP a número IP en redes de datos TCP-IP. Los hosts de la red de datos se encuentran configurados como clientes DNS de un servidor DNS que posee la red de la Facultad, al cual está conectado el clúster; consultan al servidor por nombres IP requeridos en la actualización de sistema operativo o de funcionalidades de software específicas.

Acceso seguro.

La dinámica de trabajo del Instituto tomado como caso de estudio, determinó la necesidad de contar con dispositivos que permitan flexibilizar la conexión de los usuarios al clúster, desde varias fuentes de acceso. Se cuenta con un dispositivo Firewall que interconecta las redes y permite entre otras cosas, acceso a Internet por doble vía. El ISP contratado para el servicio de acceso a Internet, provee una dirección IPv4 de tipo pública ruteada sobre el

sistema nacional, lo que permite que este dispositivo sea accedido desde cualquier punto de Internet. Está implementado servicios de Firewall, Proxy, Routing, NAT, VPN para ofrecer acceso seguro a los usuarios que requieren el ingreso a la red del Instituto desde Internet. Ver la siguiente figura:

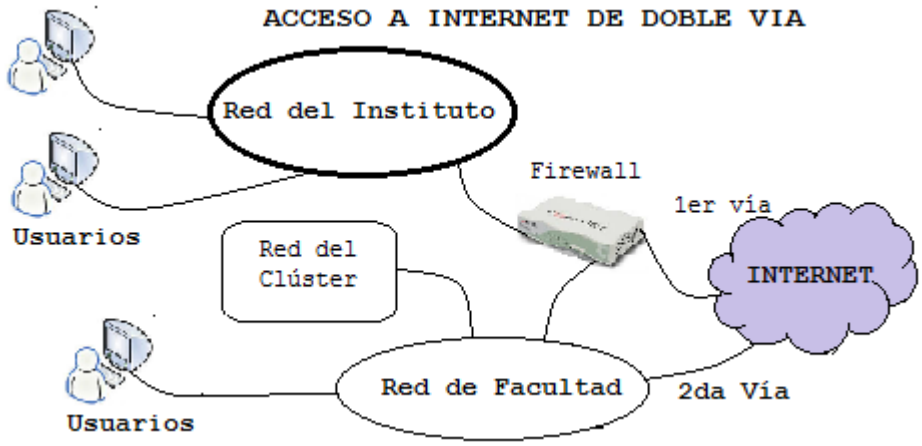


Fig-45. Acceso a Internet de doble vía.

El dispositivo Router + Firewall de la figura 33, interconecta las redes y establece políticas de tráfico entre ellas. Las sesiones iniciadas desde Internet hacia la red del Clúster, se establecen mediante el protocolo SSL, el cual encripta la sesión con una clave propia. Las cuentas de usuario se generan para los miembros que posee permiso de acceso. Posee también una funcionalidad importante: servicio VPN, para permitir que las conexiones remotas al clúster sean encriptadas y aseguradas desde los puntos de inicio de sesión. El dispositivo funciona como servidor VPN, y los usuarios “discan” una conexión desde sus equipos de acceso, con credenciales individuales de usuario y contraseña, para el acceso al clúster.

Las políticas de tráfico, se establecen en función a los usuarios con el objeto de asegurar calidad de servicio a todos los integrantes del Instituto. Es posible afirmar que el sistema de acceso es seguro, porque en el procedimiento de conexión que se ha intentado describir, son utilizadas herramientas de hardware y software que cumplen con estándares de funcionamiento normalizados.

Una funcionalidad provista en la aplicación, es que cuando el usuario se encuentre conectado a la aplicación mediante VPN, el sistema habilita un enlace para conexión SSH, implementado como un servicio en el nodo ‘Headnode’ sobre el servicio Apache, ver la siguiente figura:

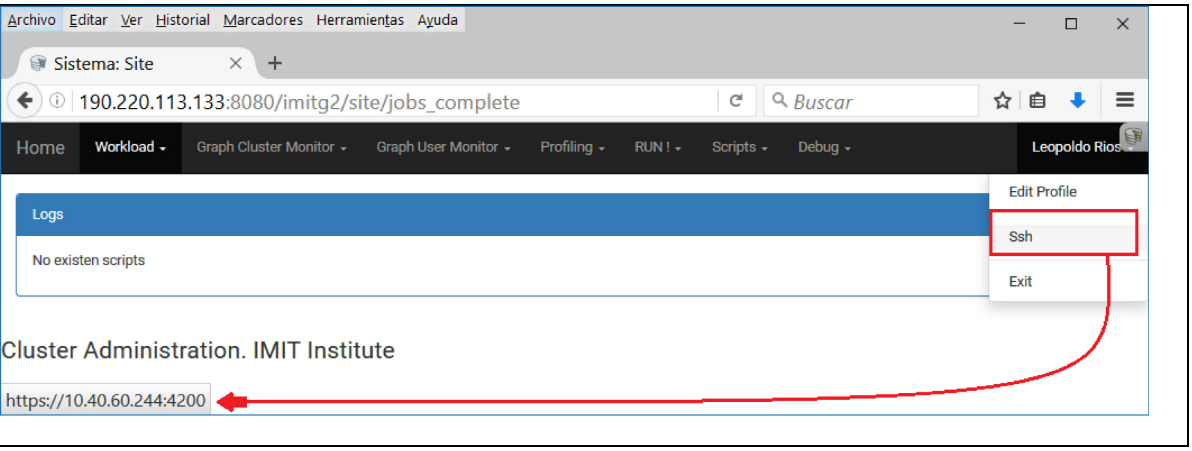


Fig-46. Conexión Ssh disponible.

La opción de menú “Ssh” ubicada en el extremo derecho de la figura 34, es habilitada cuando el sistema detecta que la dirección IP del nodo “Headnode” es alcanzable por el cliente web. Se considera una opción útil para los usuarios amantes del trabajo en línea de comando. En este caso, se trata de una configuración Ssh basada en cliente web, que resulta práctica y simple de utilizar.

Conexión desde Internet.

Para el acceso al clúster desde Internet, es decir, para el caso en que el usuario se encuentra conectado a una red remota al clúster, desde su casa o en estancia en otra institución del país o del exterior, será necesario informar la dirección IP pública otorgada por el ISP contratado. En cuanto al caso de estudio mencionado, el ISP otorga una dirección IPv4 de tipo pública, que es utilizada en la configuración de la interfaz “externa” del firewall.

Además de la configuración de la dirección IP, se sigue una estrategia de redirección de puertos -Port Forwarding- establecida en el firewall; lo cual aporta facilidad en la gestión de conexiones remotas al clúster. El firewall provee una vista en la cual es posible monitorear las conexiones entrantes al clúster. Esta técnica, admite conexiones remotas, y re-envía las peticiones a una dirección IP del nodo “Headnode”, quien es el que procesa y contesta las peticiones HTTP requeridas por el cliente web conectado desde Internet.

Una alternativa a la redirección de puertos, ya mencionada, es el servicio VPN -Red privada virtual- que permite autenticar sesiones de usuario remoto, y habilitar a las sesiones, acceso a la red de Acceso del Instituto mientras dure la conexión. El usuario debe poseer credenciales de login habilitadas para tal fin. Este tipo de sesión de usuario, es encriptada desde la conexión del usuario hasta el extremo de la interfaz del firewall, lo cual determina un alto grado de seguridad en la transmisión de datos a distancia.

5 EVALUACIÓN DE LA HERRAMIENTA.

En los capítulos anteriores se ha brindado una descripción en detalle acerca de la propuesta y desarrollo del trabajo, en este capítulo se presentan comentarios de evaluación en general: una descripción global, comentarios relevantes de los usuarios con oportunidad de conocer, usar y en cierta forma evaluar la herramienta.

5.1 Descripción de los resultados.

El resultado final de este trabajo, es consecuencia de un número importante de errores y también de aciertos. Tiempo atrás, observaba y evaluaba productos corporativos de gestión de clúster, y tiempo después, me encontraba iniciando las primeras pruebas de funcionamiento de la infraestructura de gestión de usuarios y roles. Rescato en este trabajo, la satisfacción personal por haber podido aplicar conocimientos obtenidos en los cursos de la Maestría, y un logro profesional de gran valor, por haberme integrado a un ambiente interdisciplinar y aportar una herramienta de gestión y monitoreo, de simple implementación y despliegue.

La actividad de desarrollo fue solapada con actividades de gestión propias del clúster, del instituto tomado como caso de estudio de este trabajo. Día a día, a medida que las ideas surgían, los avances se mostraban interesantes y consistentes con lo planificado. La interacción lograda con los usuarios permitió, que a medida que iba incorporando funcionalidades a la aplicación, éstos lo iban usando y a la vez, proponiendo mejoras y sugerencias. El planteo de las autoridades del ámbito del instituto, fue siempre que el sistema de gestión de trabajos del clúster funcione, independientemente del aporte hecho el desarrollo, para que la planificación de trabajos sea ejecutada de manera constante.

Durante el desarrollo de la propuesta, la etapa en determinar una modalidad acorde de autenticación de usuarios, fue la que demando más tiempo, porque se debía estudiar en detalle herramientas que hicieran posible su despliegue. La propuesta de gestión de usuarios basado en roles -RBAC (control de acceso basado en roles)-, permitió la ejecución de tareas sin tener que permitir acceso a objetos. Considero positiva la elección de la plataforma “CakePHP”, con un despliegue modelo vista controlador, lo cual constituye la base de funcionamiento del sistema, y es ampliable a tantas funcionalidades como requerimientos de los usuarios exista.

La asignación de tareas en función a roles de usuario, pudo ser establecida con un mecanismo muy simple. Se pudo demostrar que el acceso a las funcionalidades del sistema, están en función al rol del usuario vinculado. El administrador del clúster de manera muy simple, puede modificar las asignaciones bajo demanda, la experiencia indica que es necesario concentrarse en la identificación de roles, en la declaración y definición de permisos, y en la asignación de los roles a las cuentas de usuario, sin importar el volumen de usuarios.

La herramienta Torque PBS se integra a la aplicación propuesta, como el módulo de gestión de trabajos en modalidad batch del clúster. Se encarga de la organización, clasificación, ejecución y finalización de trabajos de usuario. Provee una serie de comandos que hacen

de interface al ser ejecutados, informan en tiempo real el estado de los trabajos de usuario; lo cual registrado por la aplicación para reunir información, registrarla y difundirla al usuario bajo demanda. La aplicación reúne una serie de vistas con la información provista por Torque PBS, en algunos casos se encuentra filtrada según el usuario que la utiliza, y en otras, es información general para los usuarios y administrativos. El aporte que se logra, es justamente la visibilidad a través de herramientas web, superando ampliamente a la interfaz por línea de comando, donde hay que recordar formatos de comandos y parámetros muchas veces complicados. El hecho de que la aplicación sea web-responsive, permite adecuar su formato según el dispositivo de acceso con que se cuente, aspecto valioso considerado por los usuarios.

En lo referente a la administración de datos, se ha logrado trabajar datos de usuario, sus scripts, los historiales, con la herramienta de base de datos relacional MySQL; y los datos puramente numéricos obtenidos de diversas métricas, son gestionados con la herramienta RRDtool. Es posible demostrar que las herramientas se complementan perfectamente bien, sin observar interferencias. Queda por hacer en este sentido, mejoras en cuanto a procedimientos para el mantenimiento de tablas y de la base de datos en general; aspecto no previsto en este trabajo de tesis.

La aplicación suma alrededor de treinta (30) megabytes de espacio en disco, ocupados por el código PHP, los archivos de datos RRD's, las librerías y framework CakePHP; y otros 12Mb la actual base de datos MySQL. Estos datos reportan la escasa capacidad requerida por el sistema para dar soporte al clúster HPC.

La gestión de un centro de procesamiento de datos configurado en Clúster HPC, requiere software de calidad, para asegurar el funcionamiento y continuidad de los trabajos en el tiempo. El software de calidad no siempre se obtiene por compra; sino más bien, es el resultado de procesos de desarrollo internos, de interacción con los usuarios, de la realización de pruebas y mejoramiento constante, de maduración de conceptos, y de ser disciplinado en el uso de modelos de trabajo conocidos.

Existen productos de software en el mercado que poseen muchas de las funcionalidades descriptas en la aplicación que se propone, y claramente tienen muchas más. Sin embargo, su acceso es mediante el pago de licencias, situación que todavía genera rechazo en los ambientes de investigación, tanto por el alto costo de las mismas, como por la indisponibilidad de los archivos fuente. El mantenimiento de software y el tratamiento de errores de programa, no siempre se encuentran incluidos en el precio de la licencia; estos aspectos constituyen puntos de negociación, resultan onerosos y muy difíciles de mantener bajo control para los usuarios finales.

El escenario en el cual el software propuesto ha sido instalado y testeado, es un Datacenter que se encuentra en producción, dado que los usuarios realizan sus actividades de cálculo de forma normal. La estrategia ha sido, integrar la aplicación al sistema en producción, a los efectos de probar las funcionalidades ya esgrimidas, en particular, el seguimiento y monitoreo de trabajos individuales de usuario, la generación de gráficas y el listado de trabajos completados. La siguiente figura 47, representa la idea de locación y funcionamiento de la aplicación propuesta, integra las herramientas que acompañan la

solución, a la vez que se esquematiza la relación entre los componentes, tanto de hardware, conexión de red, y software:

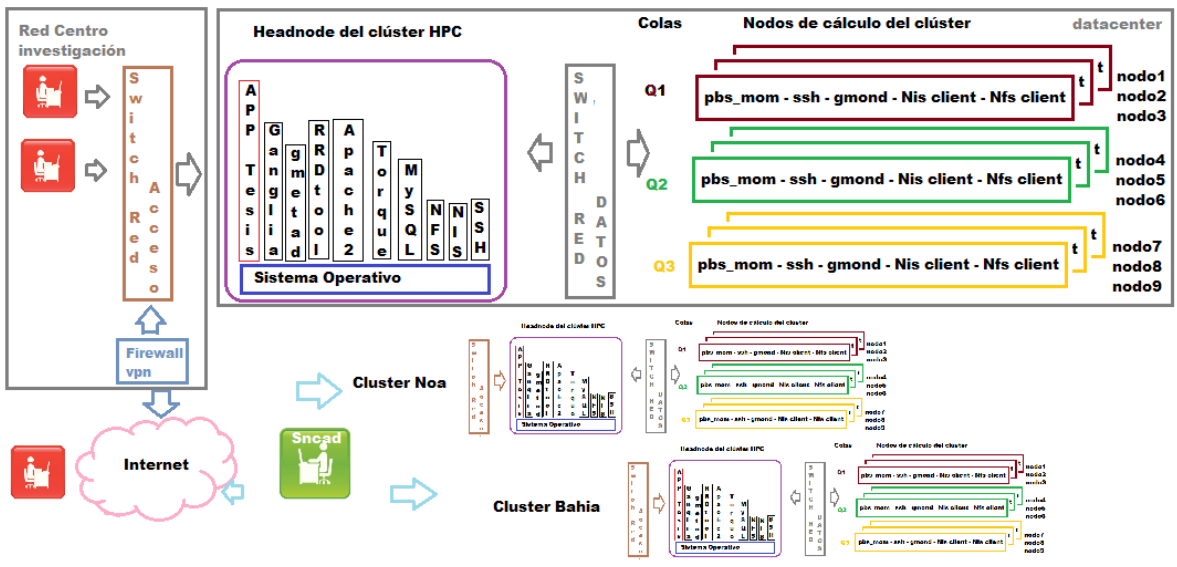


Fig-47. Vista general del sistema propuesto.

De la figura 47, lo hecho hasta la fecha, se encuentra desplegado dentro del Datacenter de la institución tomada como caso de estudio. Los usuarios -íconos rojos- del clúster, acceden tanto desde sus oficinas del Instituto, como de Internet, para interactuar con la aplicación propuesta. Del ícono verde 'Sncad' hacia la derecha, corresponde nada más a una propuesta a presentar ante autoridades de SNCAD, como se describe en el capítulo de conclusiones.

5.2 Respuesta de los usuarios.

En el mes de junio de 2017, fui convocado para brindar un seminario a la comunidad de Conicet Nordeste, en la ciudad de Corrientes. Fue posible describir esta propuesta de software, enumerando funcionalidades y objetivos para usuarios y para directivos del instituto. En términos generales, la propuesta fue aceptada, de la interacción lograda se puede mencionar:

1. El director del instituto se manifestó a favor de implementar y utilizar esta herramienta por encima de la situación si el equipamiento del instituto es prestado o no, valorando las funciones de monitoreo personalizado y bajo demanda, situación advertida durante el desarrollo del seminario. Solicitó poner en funcionamiento el sistema, con especial énfasis en el monitoreo de trabajos de usuarios, quizá el aspecto más valorado por quienes deben aportar a sus trabajos, material gráfico que acompañe la descripción de resultados.
2. Becarios del instituto, se manifestaron también a favor de la implementación, sugiriendo como primera medida, una capacitación en el uso específico de la herramienta

RRDtool, dadas las cualidades positivas advertidas en la exposición. Los aspectos valorados son:

a. Posibilidad de personalizar el monitoreo de trabajos del clúster, y en forma granular, el seguimiento de determinados procesos durante ventanas de tiempo acotadas.
b. Aplicación web-enabled, con soporte web-responsive. Fue probado el acceso con teléfonos inteligentes, pc's, notebooks y con tablets, la experiencia de poder acceder a información de estado con páginas web que se acomodan automáticamente al formato de la pantalla, fue positiva.

El siguiente cuadro, describe la opinión que arrojó una encuesta de satisfacción realizada en forma anónima a algunos usuarios mediante un formulario Google:

Marca temporal	Facilidad de uso	Lo que me aporta, para elaborar reportes e informes	Acceso a la aplicación (vpn + acceso web)	Interface Web (velocidad), el intercambio	Información que aporta sobre el estado del clúster
27/06/2017 12:45:46	4	3	Rebuscado	Por momentos demora...	Valiosa e incompleta, queda por mejorar
27/06/2017 16:35:19	3	2	Rebuscado	Por momentos demora...	Valiosa e incompleta, queda por mejorar
28/06/2017 09:40:05	5	4	Simple	Rápida, de simple interacción	Valiosa para el envío de nuevos trabajos
28/06/2017 11:24:15	5	3	Simple	Por momentos demora...	Valiosa para el envío de nuevos trabajos
28/06/2017 12:30:24	4	4	Simple	Rápida, de simple interacción	Valiosa e incompleta, queda por mejorar

Observaciones.

El ítem "Facilidad de uso", define un valor (1-complicado, 5-simple) que determina la facilidad en el uso de la aplicación propuesta.
Algunos becarios, no todos, utilizan los reportes gráficos para ilustrar informes que son requeridos con frecuencia, el valor 1-no es valioso, el 5-Muy valioso.
La conexión a la aplicación web es mediante número IPv4, cuando es realizada desde la red informática de la Facultad, el acceso es directo. Cuando el becario desea ingresar desde Internet, primero debe discar una conexión VPN, asociando un usuario y contraseña, y luego acceder mediante navegador a la aplicación. Se entiende por 'Rebuscado' el hecho de que para acceder a la aplicación, primero se tiene que establecer la comunicación segura, que en ciertos momentos falla, según comentan los usuarios.
La velocidad en el acceso a la aplicación, depende de si está conectada desde la red informática de la facultad, o desde Internet mediante VPN, en este último caso, es posible experimentar demoras, debido a la compresión y descompresión del tráfico.

En situación de las V Jornadas de Cloud Computing desarrolladas en la Facultad de Informática de la UNLP, a fines de junio de 2017, se tomó la decisión de exponer esta propuesta de software de código abierto para la gestión de clúster HPC en ambiente de Cloud. El sustento de esta idea es que, tarde o temprano, gran parte del caudal de procesamiento de datos de la comunidad científica nacional, por varios motivos, será mudado a un ambiente de Cloud. Se entiende que también en un ambiente de Cloud será

necesaria una herramienta para la gestión de recursos computacionales, la organización de trabajos en agrupamientos y el monitoreo en línea; esta aplicación puede resultar de utilidad. La aplicación que se propone, está preparada para trabajar tanto para ambiente Cloud, como para un escenario de red local. Luego de la presentación de la propuesta, el señor Decano de la Facultad consultó acerca de la factibilidad de la implementación y en cuanto al nivel de porcentaje de implementación de funcionalidades en el Instituto tomado como caso de uso; consultas que fueron contestadas y que considero de real importancia.

6 CONCLUSIONES.

Este capítulo intenta alinear objetivos propuestos y los logros obtenidos, y propone trabajo a futuro.

En el mercado existe software comercial que da soporte a las demandas que requiere hoy en día un clúster HPC del cual ya se ha brindado información. El enfoque de este trabajo de tesis fue elaborar una herramienta de gestión de clúster HPC basada en código abierto, para lo cual se ha incorporado otras herramientas de software que lo propician.

El resultado obtenido permite, de manera muy simple, programar, lanzar, ejecutar y monitorear tareas de usuarios en el clúster, mediante conexiones locales y remotas. Es altamente positivo el poder seguir la ejecución de trabajos a distancia, a través de la interface web. La gestión de usuarios, roles y programación de tareas requiere el conocimiento de programación PHP, MySQL para gestión de datos y RRDtool para el monitoreo.

El diseño previsto permite la interconexión entre las herramientas propuestas, de manera muy simple, con la posibilidad de reemplazo de alguna de ellas y re-configuración posterior. Es posible migrar a plataformas operativas alternativas como RedHat, Ubuntu, CentOS; y de herramienta de gestión de datos como PostgreSQL. Es posible el reemplazo de Torque PBS como gestor de trabajos por SLURM. La serie de scripts que se encargan de la recolección de datos del estado del clúster, presentes en la lista de scripts, deberán ser modificados según las plataformas elegidas.

La aplicación permite gestionar equipos de computación configurados en clúster HPC, como un todo, o como partes de un todo. En el escenario del caso de estudio, el ámbito es el clúster completo: todos los nodos son asignados a colas de trabajo, gestionados desde la misma aplicación. Quizá en otros escenarios, se determine el uso de solo una parte del clúster, entonces la aplicación puede ajustarse específicamente a esos recursos, que luego los usuarios darán uso.

La metodología de trabajo adoptada, en cuanto a desplegar la solución sobre un clúster en producción, si bien acarrea tiempo extra por diversos controles, fue positivo por la interacción lograda con los usuarios: a medida que las opciones de menú se iban incorporando, fue posible obtener el impacto que causaba de forma inmediata.

6.1 Descripción de lo obtenido.

Se asume que el objetivo de desarrollar software de código abierto para la gestión de un clúster HPC ha sido demostrado. La aplicación se encuentra en producción en el Datacenter del escenario tomado como caso de estudio, con una lista de diez cuentas de usuario, roles, grupos de tareas, permisos de acceso a grupos de tareas, y colas de trabajo en el clúster en producción.

En esta etapa de difusión de la aplicación, ha sido valorada por la facilidad en el seguimiento de trabajos, en la generación de estadísticas y en la visibilidad de trabajos completados y sus detalles. La edición de scripts, la generación de proyectos de software, la compilación y puesta a punto, todavía son tareas realizadas mediante línea de comandos -sesión SSH-, dada la familiarización que tienen los usuarios con esa modalidad de trabajo. Otro aspecto de valor a mencionar, es la disponibilidad de históricos, tanto de ejecuciones de trabajos en el clúster, como de profiling de aplicaciones.

6.2 Trabajo a futuro.

1. Propuesta a SNCAD.
   Se contempla la situación en que la SNCAD [Sncad], pueda propiciar el uso de esta aplicación con el objeto de estandarizar una herramienta para la gestión de Clúster HPC, para la obtención de métricas de uso de recursos computacionales de Datacenter distribuidos en el país. Con ello, sería posible:

   - Identificar al instante los Datacenters con mayor y menor capacidad de cómputo utilizado, pudiendo derivar en los últimos, trabajos de cálculo, con el objeto de bajar los tiempos de espera en la ejecución de programas.
   - Estandarizar la medición de recursos como memoria RAM, CPU, Threads, red.
   - Centralizar el diseño y recepción de reportes de uso.
   - Conocer los programas de cálculo utilizados con mayor frecuencia.
   - Asignar espacios de Datacenter con baja utilización a tareas de docencia por tiempo.

El monitoreo a realizar puede ser desplegado de las siguientes maneras:

1. Es posible generar vistas con información específica para el acceso remoto de usuarios de la SNCAD, con información en detalle a determinar por ellos mismos.
2. Comunicar los resultados de las métricas en equipos localizados en Datacenter propio de SNCAD, a efecto de procesarlos de manera local. Para ello, se deben poder transportar los archivos RRD's desde el Headnode del Datacenter local hacia los equipos de SNCAD.

La configuración del sistema de permisos de acceso, puede ser desplegado, tanto por la SNCAD, como por los usuarios administradores de los Datacenter de centros de investigación. Es posible programar de manera remota el ingreso de cuentas de usuario, la declaración de tareas, la asignación de permisos, y las vistas de resultados.

En la figura 48, es posible apreciar un ícono de color verde, que representa a un usuario que puede ejercer el rol de monitoreo de múltiples sistemas conectados a través de Internet. En este sentido, es posible configurar la aplicación para que conecte con diversos Datacenter distribuidos, en una misma vista centralizada de datos. Lo que se observa en la siguiente figura 49, es el aspecto que podría tener una vista de Ganglia, configurado en un nodo central de SNCAD, con agentes de métricas instalados en los nodos de cálculo remotos.

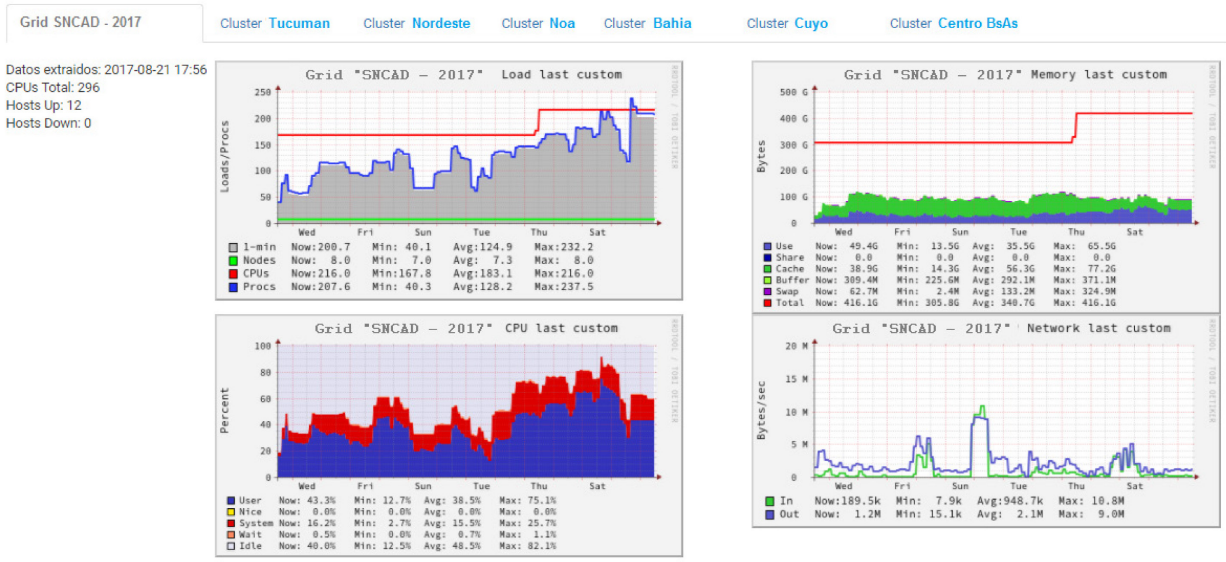


Fig-48. Vista de Ganglia configurado en SNCAD, trabajo a futuro.

6.1 Nuevos Desarrollos.

Una posibilidad adicional en la mejora a esta aplicación que se propone, es incorporar un módulo de gestión de proyectos de investigación, con el fin de documentar la interacción entre investigadores, becarios y administrativos de instituciones. Este nuevo módulo debería permitir planificar y ejecutar actividades en un contexto de intranet, con intercambio de correo electrónico, para obtener al final del ciclo, un historial con la interacción dada como documento de trabajo.
Esta posibilidad puede ser lograda con frameworks dedicados a la gestión de proyectos en modalidad open source, que permita gestión de cuentas de usuarios, grupos de usuarios, proyectos individuales y grupales, actividades, metas, y la comunicación de mensajes, a través de un sistema de correo electrónico interno, a los efectos de salvaguardar la información previa a la difusión definitiva.

Uno de los productos que puede ser tenido en cuenta, es Kanban “Kanboard”, el cual puede ser instalado sobre el nodo “Headnode” del clúster HPC, y programar y procesar desde allí todas las operaciones. Cada usuario de la aplicación, podría tener acceso a Kanboard, con el mismo usuario previsto en el sistema, y desde allí interactuar con los proyectos en los cuales se encuentre vinculado. El sistema prevé que el usuario incorpore material -documentos, tablas, dibujos, fotos-, que sea resultado de sus trabajos de investigación, y los comunique al grupo de usuarios o a al director del proyecto mediante correo electrónico ‘interno’, sin salida a Internet.

Las interfaces de Kanban Kanboard, pueden ser presentadas mediante difusión WEB, desde el mismo servicio Apache2 que ejecuta el nodo ‘Headnode’. Las pizarras de trabajo, integran la modalidad de tickets:

- Backlog: trabajos pendientes de hacer, requeridos por el director del proyecto.
- Work in progress: trabajos que se están haciendo, en forma individual o grupal.
- Ready: trabajos finalizados, pendientes de aprobación del director del proyecto, que pueden ser redirigidos al Backlog, para su revisión.

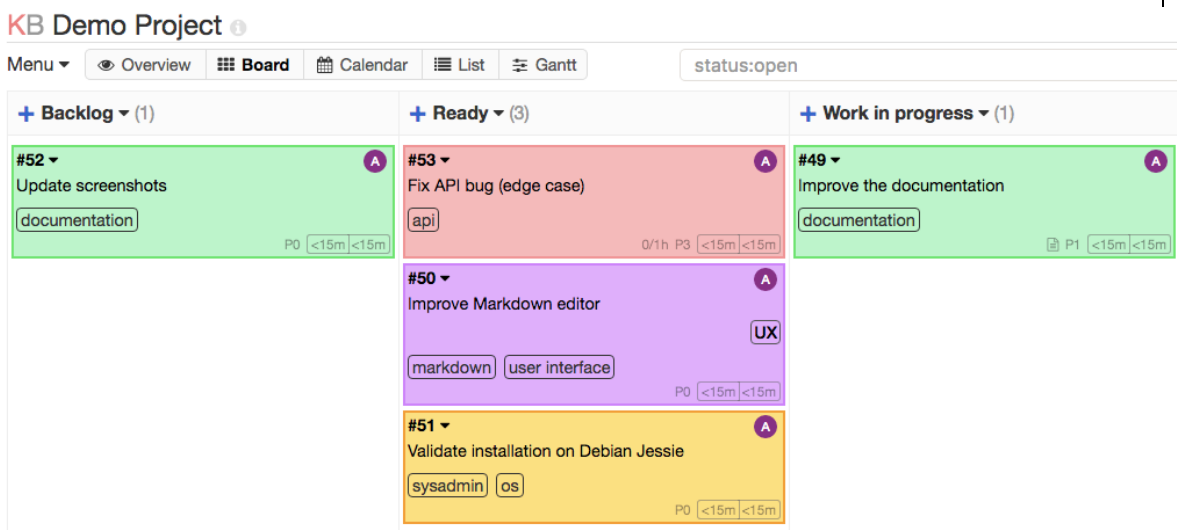


Fig-49. Modelo de pizarra electrónica, distribución de trabajos en un proyecto

Tópicos que pueden ser abordados en trabajo a futuro:

- Desarrollo de proyectos de investigación en escenario de nube privada
- Gestión de proyectos mediante herramientas informatizadas.
- Unificación del módulo gestor de proyectos con la aplicación de gestión de trabajos en clúster HPC.

APENDICES.

LISTA DE FIGURAS

LISTA de REFERENCIAS.

[Linux]	- https://www.linuxfoundation.org/about - GNU/Linux Command−Line Tools Summary, Gareth Anderson
[OpenHPC]	- Cluster Computing with OpenHPC, HPCSYSPROS '16 November 14, 2016, ISBN 978-1-4503-213 9. - http://openhpc.community/downloads/ - https://opensource.com/article/17/11/openhpc
[Opensource]	- Understanding Open Source and Free Software Licensing, Andrew St. Laurent, O'Reilly Media, Ebook: 2008 ISBN 978-0-596-15308-3
[Slurm]	- https://slurm.schedmd.com/overview.html - https://slurm.schedmd.com/disclaimer.html - Job Scheduling Strategies for Parallel Processing: 13th International Workshop JSSPP 2007 - Eit Frachtenberg – Springer – ISBN10: 3-540-78698-8 - Building Clustered Linux Systems - Robert W. Lucke – Prentice Hall – ISBN 0131448536, 9780131448537
[Conicet]	- http://www.conicet.gov.ar/conicet-descripcion/
[Sncad]	- http://www.supercalculo.mincyt.gob.ar/
[RRDtool]	- Tobias Oetiker, http://oss.oetiker.ch/rrdtool/
[Ganglia]	- Monitoring with Ganglia, by Matt Massie, Bernard Li, Brad Nicholes, and Vladimir Vuksan, O'Reilly Media. ISBN: 978-1-449-32970-9
[Sedici]	- http://sedici.unlp.edu.ar/pages/queEsSedici - RRDtool data management in HPC Environments, Trabajo de Especialización en Ing. de Software March 2016, http://sedici.unlp.edu.ar/handle/10915/53669.
[OpenSuse]	- https://es.opensuse.org/Informacion_en_general - https://www.opensuse.org/
[PbsPro]	- https://twiki.opensciencegrid.org/bin/view/Education/PbsSetup#Flavors_of_PBS
[Intel-ht]	- https://www.intel.la/content/www/xl/es/architecture-and-technology/hyper-threading/hyper-threadin technology.html
[TCP-Ilustrated]	- TCP/IP Illustrated, Second Edition, ISBN-10: 0-321-33631-3
[Torque]	- http://www.adaptivecomputing.com/products/open-source/torque/ - http://www.clusterresources.com/products/torque/docs10/a.hlicense.txt
[Flynn]	- Computer Architecture A Quantitative Approach, Fourth Edition, John L. Hennessy Stanford University, David A. Patterson University of California at Berkeley, ISBN 13: 978-0-12-370490-0
[MySQL]	- MySQL Cookbook , Third Edition, Paul DuBois, ISBN: 978-1-449-37402-0
[Profiling]	- Profiling and Tracing in Linux, Sameer Shende, Department of Computer and Information Science, University of Oregon, Eugene, OR, USA - "Optimizing a GPU Algorithm Through Hardware Profiling Analysis", Fernando G. Tinetti, Sergio M Martin, CSCI, The 2014 International Conference on Computational Science and Computational Intelligence (CSCI'14) March 10-13, 2014, Las Vegas, USA, ISBN 978-1-4799-3010-4/14, pp. 45-5 - W. E. Cohen, "Tuning Programs with OProfile", Wide Open Magazine, 2004, pages 53-62. https://people.redhat.com/wcohen/Oprofile.pdf - https://perf.wiki.kernel.org/index.php/Tutorial - http://www.brendangregg.com/perf.html - http://sandsoftwaresound.net/perf/perf-tut-profile-hw-events/ - Lilja D.J. Measuring computer performance. A practitioner's guide, ISBN 0-511-03627-2 eBook.
[IntroParallel]	- Introduction to Parallel Computing, Second Edition, Ananth Grama, Anshul Gupta, George Karypis Vipin Kumar, ISBN: 0-201-64865-2
[ISO]	- https://support.microsoft.com/en-us/help/103884/the-osi-model-s-seven-layers-defined-and-functio

	explained - Comunicaciones y Redes de computadores, William Stallings, Séptima edición, ISBN: 978-84-205 4110-5
[Rbac]	- http://csrc.nist.gov/rbac/rbac-std-ncits.pdf - https://docs.puppet.com/pe/latest/rbac_intro.html - https://en.wikipedia.org/wiki/Model-view-controller - https://www.asp.net/mvc
[Frameworks]	- R. Dāsa, Learn CakePHP: With Unit Testing, 2nd Ed., ISBN-10:1484212134, 2016. https://cakephp.org/ - https://book.cakephp.org/2.0/en/index.html - CakePHP Cookbook Documentation, Cake Software Foundation, 2017. - PHP MASTER: write cutting-edge code, By Lorna Mitchell, 2011 SitePoint - Building PHP Applications with Symfony™, CakePHP, and Zend® Framework, ISBN: 978-0-470-88734-9 - https://codeigniter.com/docs - http://symfony.com/doc/current/index.html
[Altair]	- https://secure.altair.com/onlinestore/
[Altair-Price]	- https://secure.altair.com/onlinestore/index.php?main_page=product_info&cPath=65&products_id=1 - http://www.altairhyperworks.com/newsdetail.aspx?news_id=183&news_country=en-US
[AltairCM]	- http://www.altair.com/compute-manager/
[AltairDM]	- http://www.altair.com/display-manager/
[Infiniband]	- http://journal.info.unlp.edu.ar/wp-content/uploads/JCST-Jul08-8.pdf - https://blogs.oracle.com/networking/infiniband-building-blocks
[Eclipse]	- www.eclipse.org/neon

LISTA de Scripts.

1 – ‘his-process.sh’

Obtiene mediante el comando ‘tracejob’ el estado de trabajos que se encuentran en ejecución, buscándolos por número de JobID. Genera un archivo por cada Job, como nombre utiliza su número de JobID. Los archivos son utilizados en la vista “runnd”, del menú “Jobs running detailed”. Auto ejecuta cada minuto.

```
#!/bin/bash
folder="/srv/www/htdocs/ck_ganglia/app/webroot/files/cluster/tmptrace"
cd $folder
lastnumber=`tail -n 1 jobs-r.csv`
firstnumber=`head -1 jobs-r.csv`
echo "Primer valor:$firstnumber"
echo "Last number:$lastnumber"
for (( thenumber = $firstnumber; thenumber <= $lastnumber; thenumber++));
do
if [ -f $thenumber ]; then
   echo "File $thenumber exists."
else
        tracejob -q -n 20 $thenumber > $thenumber
fi
done
exit 0
```

2 – ‘his-process-complete.sh’

Se busca en cada archivo generado en (1) los Jobs en estado COMPLETE, y se obtiene un archivo “CSV” con variables separadas por coma, a ser utilizados en la actualización de una tabla “history-completed” con el estado de trabajos de usuario, finalizados por el clúster.

```
#!/bin/bash
folder="/srv/www/htdocs/ck_ganglia/app/webroot/files/cluster/tmptrace"
cd $folder
lastnumber=`tail -n 1 /
srv/www/htdocs/ck_ganglia/app/webroot/files/cluster/csv/jobs-i.csv`
### la variable firstnumber debe contener el numero
firstnumber=$lastnumber-30
line=""
echo "Primer valor:$firstnumber"
echo "Last number:$lastnumber"
for (( thenumber = $firstnumber; thenumber <= $lastnumber; thenumber++));
do
if [ -f $thenumber ]; then
y=`cat $thenumber | grep dequeuing | cut -d "," -f 2`
if [[  $y == *"COMPLETE"* ]]; then
a=`cat $thenumber | grep Job: | sed "s/.n.*//"| cut -d ":" -f 2 `
b=`echo ", "`
c=`cat $thenumber.txt | grep jobname= | cut -d "=" -f 2 | head -1`
e=`cat $thenumber.txt | grep user= | cut -d "=" -f 2 | head -1`
g=""$(cat $thenumber.txt | grep start= | cut -d "=" -f 2 | head -1 |{read gmt ;
date -d "@$gmt" ; } )""
i=""$(cat $thenumber.txt | grep qtime= | cut -d "=" -f 2 | head -1 |{read gmt ;
date -d "@$gmt" ; })""
k=`cat $thenumber.txt | grep queue | cut -d "=" -f 2 | head -1`
m=`cat $thenumber.txt | grep total_execution_slots= | cut -d "=" -f 2`
o=""$(cat $thenumber.txt | grep exec_host= | cut -d "=" -f 2 | head -1)""
q=""$(cat $thenumber.txt | grep Resource_List.walltime= | cut -d "=" -f 2 | head
-1)" "
s=""$(cat $thenumber.txt | grep resources_used.walltime= | cut -d "=" -f 2 | head
-1)""
u=""$(cat $thenumber.txt | grep resources_used.cput= | cut -d "=" -f 2 | head -1
)""
w=`cat $thenumber.txt | grep resources_used.mem= | cut -d "=" -f 2 | head -1 `
```

```
y=`cat $thenumber | grep dequeuing | cut -d "," -f 2 | awk "{print $2}"`
line=$a$b$c$b$e$b$i$b$g$b$k$b$m$b$o$b$s$b$u$b$w
echo $line >> /srv/www/htdocs/ck_ganglia/app/webroot/files/cluster/hist-jobs-
complete.csv
line=""
else
        echo "File $thenumber does not exist!"
fi
fi
done
```

3 – 'sched-timeleft.sh'

Se buscan Jobs que se encuentran en ejecución, y se obtiene el tiempo requerido y el tiempo transcurrido desde su inicio, se calcula el tiempo restante, y se lo guarda en un archivo "CSV" con variables separadas por coma, a ser utilizados en la vista "Job end prediction".

```
#!/bin/bash
qmgr="/opt/torque/bin/qmgr"
qstat=/opt/torque/bin/qstat
#obtener los trabajos en ejecucion, y el "elapsed time"
# a = cantidad de colas habilitadas
a=$($qstat -r | sed 1,5d | awk "{ if ($10 =="R") print $1}" | wc -l)
folder="/srv/www/htdocs/ck_ganglia/app/webroot/files/cluster/csv/"
cd $folder
touch=runn-left.csv
> runn-left.csv
for (( thenumber = 1; thenumber < $a; thenumber++))
do
### primero guardo el nombre de la cola
lista1[$thenumber]=$(qstat -r | awk "{print $3}"|sed "s/:.*//" | sed 1,5d |sed -n
"$thenumber p")
#echo ${lista1[$thenumber]}
## segundo lugar, guardo el walltime programado en la cola
lista2[$thenumber]=$(qstat -r | awk "{print $9}"|sed "s/:.*//" | sed 1,5d |sed -n
"$thenumber p")
#echo ${lista2[$thenumber]}
### tercer lugar, guardo el elapsed time en total de segundos
lista3[$thenumber]=$(qstat -r | awk "{print $11}"|sed "s/:.*//" | sed 1,5d |sed -n
"$thenumber p")
#echo ${lista3[$thenumber]}
## cuarto lugar; calculo el tiempo restante!!!!
lista4[$thenumber]=`expr "${lista2[$thenumber]}" - "${lista3[$thenumber]}"`
lista5[$thenumber]=`expr "${lista4[$thenumber]}" / 24`
echo
"${lista1[$thenumber]},${lista2[$thenumber]},${lista3[$thenumber]},${lista4[$thenum
ber]},${lista5[$thenumber]}" >> runn-left.csv
done
exit 0
```

4 – 'jobs-csv.sh'

Obtiene Jobs que se encuentran en ejecución en el clúster con todo el nivel de detalle posible, y lo guarda en un archivo "CSV" con variables separadas por coma, utilizados en la vista "Jobs running".

```
#!/bin/sh
qstat="/opt/torque/bin/qstat"
#### DATOS PARA LA VISTA DE RUNN.CTP
$qstat -r | awk "{print $1,t,$2,t,$3,t,$4,t,$6,t,$7,t,$8,t,$9,t,$11}{t=","}" |
sed "s/.headnode.lo//" | sed "1,5d" >
/srv/www/htdocs/ck_ganglia/app/webroot/files/ cluster/csv/jobs-csv.csv
```

5 – ‘jobs-r.sh’

Obtiene información de jobs que se encuentran en ejecución en el clúster, y se lo guarda para a ser utilizados en otros scripts de esta lista. Obtiene y guarda únicamente el número del Job ID de los trabajos que son ejecutados al momento de su ejecución, los guarda en un archivo temporal.

```
#!/bin/sh
export SHELL="/bin/sh"
export
PATH="/sbin:/usr/bin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/bin:/opt/o
mpi/bin:/opt/torque/sbin:/opt/to
rque/bin"
qstat="/opt/torque/bin/qstat"
##### genera datos para la vista runnd.ctp, generar los botones para obtener
informacion en detalle.
$qstat -r | awk '{print $1}'| sed 's/.headnode.*//' | sed '1,5d' >
/srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/jobs-id.csv
```

6 – ‘jobs-q.sh’

Obtiene información de Jobs que se encuentran encolados, a la espera de ejecución en el clúster. La información es usada en la vista “Jobs queued”.

```
#!/bin/sh
qstat="/opt/torque/bin/qstat"
#### DATOS PARA LA VISTA DE RUNN.CTP
$qstat -i | awk '{print $1,t,$2,t,$3,t,$4,t,$6,t,$7,t,$8,t,$9,t,$11}{t=","}' |
sed 's/.headnode.imit.loca//' | sed '1,5d'
> /srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/jobs-q.csv
```

7 – ‘nodes.sh’

Obtiene información del estado de los nodos de cálculo configurados en la herramienta Torque PBS. La información es usada en la vista “nodes State”.

```
#!/bin/bash
export SHELL="/bin/bash"
export
PATH="/sbin:/usr/bin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/bin:/opt/o
mpi/bin:/opt/torque/sbin:/opt/to
rque/bin"
##### genera datos vista nodes.ctp.
# genero archivo
> /srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/nodesrun.csv
pbs="/opt/torque/bin/pbsnodes"
HOSTS="nodo12 nodo13 nodo14 nodo15 nodo16 nodo17 nodo18 nodo19 nodo20 nodo21
nodogpu patricio1 patricio2 patricio3"
#HOSTS="nodo12"

for myHost in $HOSTS
        do
        $pbs -q $myHost | sed '6,20d' >
/srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/$myHost-node.log
        d1=`cat /srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/$myHost-
node.log | grep "state =" | head -1 | awk '{p
rint $3}'`
        d2=`cat /srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/$myHost-
node.log | grep "power_state =" | head -1  |
awk '{print $3}'`
```

```
        d3=`cat /srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/$myHost-
node.log | grep "np =" | head -1  | awk '{pri
nt $3}'`
        d4=`cat /srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/$myHost-
node.log | grep "properties =" | head -1  | a
wk '{print $3}'`
        c=","
line=$myHost$c$d1$c$d2$c$d3$c$d4
echo $d1,$d2,$d3,$d4
echo $line >> /srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/nodesrun.csv
done
```

8 – 'jobs-soft.sh'

Obtiene información de estado de los trabajos en ejecución en el clúster. Obtiene del parámetro #PBS -N configurado en el script de lanzamiento, el nombre del programa utilizado en el presente trabajo. La información es registrada en una base de datos RRD, para monitoreo posterior.

```
#!/bin/bash
### set the paths
export SHELL="/bin/sh"
export
PATH="/sbin:/usr/bin:/usr/sbin:/usr/local/sbin:/root/bin:/usr/local/bin:/bin:/opt/o
mpi/bin:/opt/torque/sbin:/opt/torque/bi
n"
rrdtool=/usr/bin/rrdtool
grep=/usr/bin/grep
awk=/usr/bin/awk
cat=/usr/bin/cat
qstat=/opt/torque/bin/qstat
### Cuento la cantidad de veces que aparece los usuarios que deseo monitorear
$qstat -r | awk '{print $4,$7}'|grep 'dirac14' |awk '{ sum += $2 } END { print sum
}'

dirac14=`$qstat -r |awk '{print $4,$7}'|grep irac14 |awk '{ sum += $2 } END { print
sum }'`
dirac15=`$qstat -r |awk '{print $4,$7}'|grep irac15 |awk '{ sum += $2 } END { print
sum }'`
dirac16=`$qstat -r |awk '{print $4,$7}'|grep irac16 |awk '{ sum += $2 } END { print
sum }'`
dirac17=`$qstat -r |awk '{print $4,$7}'|grep irac17 |awk '{ sum += $2 } END { print
sum }'`
dalton15=`$qstat -r |awk '{print $4,$7}'|grep alton2015 |awk '{ sum += $2 } END {
print sum }'`
dalton16=`$qstat -r |awk '{print $4,$7}'|grep alton2016 |awk '{ sum += $2 } END {
print sum }'`
dalton13=`$qstat -r |awk '{print $4,$7}'|grep alton2013 |awk '{ sum += $2 } END {
print sum }'`
gaussian03=`$qstat -r |awk '{print $4,$7}'|grep Gaussian03 |awk '{ sum += $2 } END
{ print sum }'`
gaussian09=`$qstat -r |awk '{print $4,$7}'|grep Gaussian09 |awk '{ sum += $2 } END
{ print sum }'`
### collect the data
$rrdtool update /srv/www/htdocs/imitg2/app/webroot/rrds/procesos_db.rrd --template
\

dirac14:dirac15:dirac16:dirac17:dalton13:dalton15:dalton16:gaussian03:gaussian09 \

N:$dirac14:$dirac15:$dirac16:$dirac17:$dalton13:$dalton15:$dalton16:$gaussian03:$ga
ussian09
```

9 – ‘upd-queues.sh’

Obtiene información de estado de las colas de trabajo en ejecución en el clúster. Obtiene el número de procesos configurado en el script de lanzamiento. La información es registrada en una base de datos RRD, para estadísticas de usos de cada cola de trabajo.

```
#!/bin/bash
### set the paths
rrdtool=/usr/bin/rrdtool
grep=/usr/bin/grep
awk=/usr/bin/awk
cat=/usr/bin/cat
qstat=/opt/torque/bin/qstat
sed=/usr/bin/sed
log=/srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/colas.log
touch $log
echo "" > $log
## Obtengo la lista de colas
$qstat -q > $log
### Cuento la cantidad de corridas en las colas que deseo monitorear
GPU=$($cat $log | grep GPU | awk '{print $6}')
Q1=$($cat $log | grep Q1 | awk '{print $6}')
Q2=$($cat $log | grep Q2 | awk '{print $6}')
Q3=$($cat $log | grep Q3 | awk '{print $6}')
Q4=$($cat $log | grep Q4 | awk '{print $6}')
Q5=$($cat $log | grep Q5 | awk '{print $6}')
### collect the data
$rrdtool update /srv/www/htdocs/imitg2/app/webroot/rrds/colas_db.rrd \
--template gpu:q1:q2:q3:q4:q5 \
N:$GPU:$Q1:$Q2:$Q3:$Q4:$Q5
exit 0
```

10 – ‘queues.sh’

Obtiene datos del estado de las colas de trabajo, para presentar en la vista ‘Queues enabled’ del menúm ‘Workload’.

```
#!/bin/sh
##### genera datos vista colas.ctp de la vista SITE
# genero archivo
> /srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/queues-enabled.csv
> /srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/queues-log.log
qstat="/opt/torque/bin/qstat"
file=/srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/queues-log.log
salida=/srv/www/htdocs/imitg2/app/webroot/files/cluster/csv/queues-enabled.csv
lastnumber=`$qstat -Q | sed '1,2d' | awk '{print $1,$2,$3,$4,$6,$7}' | wc -l`
#lastnumber=1
firstnumber=1
$qstat -Q | sed '1,2d' | awk '{print $1,$2,$3,$4,$6,$7}' >
/srv/www/htdocs/imitg2/app/webroot/files/cluster/tmp/queues-log.log
for (( thenumber = $firstnumber; thenumber <= $lastnumber; thenumber++));
do
        lista0[$thenumber]=`cat $file | sed -n "$thenumber p" | awk '{print $1}'`
        lista1[$thenumber]=$(cat $file | sed -n "$thenumber p" | awk '{print $2}')
        lista2[$thenumber]=$(cat $file | sed -n "$thenumber p" | awk '{print $3}')
```

```
        lista3[$thenumber]=$(cat $file | sed -n "$thenumber p" | awk '{print $4}')
        lista4[$thenumber]=$(cat $file | sed -n "$thenumber p" | awk '{print $5}')
        lista5[$thenumber]=$(cat $file | sed -n "$thenumber p" | awk '{print $6}')
##echo
"${lista0[$thenumber]},${lista1[$thenumber]},${lista2[$thenumber]},${lista3[$thenum
ber]},${lista4[$thenumber]},${lista5[$t
henumber]}"
#echo  "${lista0[$thenumber]}"
echo
"${lista0[$thenumber]},${lista1[$thenumber]},${lista2[$thenumber]},${lista3[$thenum
ber]},${lista4[$thenumber]},${lista5[$the
number]}" >> $salida
done
```

crontab que ejecuta automáticamente.

Este ejemplo de archivo crontab, del usuario “leopoldo", muestra que, cada minuto de tiempo (indicado por la primer columna */1), ejecuta el script según su localización.

```
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/rrds/actualizar_usu_cores.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/rrds/actualizar_procesos.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/rrds/actualizar_usuarios.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/rrds/actualizar_nodos.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/rrds/actualizar_colas.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/jobs-r.sh >/dev/null
2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/jobs-q.sh >/dev/null
2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/jobs-i.sh >/dev/null
2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/jobs-csv.sh >/dev/null
2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/his-process-
complete.sh >/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/tracejob1.sh
>/dev/null 2>&1
*/1 * * * * /srv/www/htdocs/imitg2/app/webroot/files/cluster/scripts/sched-timeleft.sh
>/dev/null 2>&1
```