DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE TECNOLOGÍAS DE INFORMACIÓN (SISTEMAS E INFORMÁTICA)

TRABAJO DE TITULACIÓN PREVIO LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN SISTEMAS E INFORMÁTICA

TEMA:

"INTEGRACIÓN DE APLICACIONES DE PROCESAMIENTO EN PARALELO PARA PROVEER DE INFRAESTRUCTURA DE TI A LOS PROYECTOS DE INVESTIGACIÓN DE LA ESPE BASADO EN LA ARQUITECTURA HPC"

AUTORES:

ROCHA HOYOS LUIS GONZALO

VERA LUQUE FREDDY JOSÉ

DIRECTOR:

PHD. MARCILLO PARRA DIEGO MIGUEL

SANGOLQUÍ, 2017

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN SISTEMAS

CERTIFICACIÓN

Certifico que el trabajo de titulación, *"Integración de aplicaciones de procesamiento en paralelo para proveer de infraestructura de ti a los proyectos de investigación de la ESPE basado en la arquitectura HPC"* realizado por los señores *Luis Gonzalo Rocha Hoyos y Freddy José Vera Luque*, ha sido revisado en su totalidad y analizado por el software anti-plagio, el mismo cumple con los requisitos teóricos, científicos, técnicos, metodológicos y legales establecidos por la Universidad de Fuerzas Armadas ESPE, por lo tanto me permito acreditarlo y autorizar a los señores *Luis Gonzalo Rocha Hoyos y Freddy José Vera Luque* para que lo sustenten públicamente.

Quito, 7 de Agosto de 2017

PhD. Diego Miguel Marcillo

DIRECTOR

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

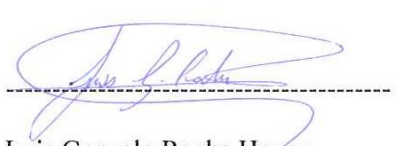
CARRERA DE INGENIERÍA EN SISTEMAS

AUTORÍA DE RESPONSABILIDAD

Nosotros, *Luis Gonzalo Rocha Hoyos y Freddy José Vera Luque*, con cédulas de identidad N° *1725412306 y 1312139866* respectivamente, declaramos que este trabajo de titulación *"Integración de aplicaciones de procesamiento en paralelo para proveer de infraestructura de ti a los proyectos de investigación de la ESPE basado en la arquitectura HPC"* ha sido desarrollado considerando los métodos de investigación existentes, así como también se ha respetado los derechos intelectuales de terceros considerándose en las citas bibliográficas.

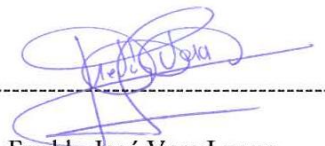
Consecuentemente declaramos que este trabajo es de nuestra autoría, en virtud de ello nos declaramos responsables del contenido, veracidad y alcance de la investigación mencionada.

Quito, 7 de Agosto de 2017



Luis Gonzalo Rocha Hoyos

1725412306



Freddy José Vera Luque

1312139866

ESPE
UNIVERSIDAD DE LAS FUERZAS ARMADAS
INNOVACIÓN PARA LA EXCELENCIA

DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN

CARRERA DE INGENIERÍA EN SISTEMAS

AUTORIZACIÓN

Nosotros, *Luis Gonzalo Rocha Hoyos y Freddy José Vera Luque*, autorizamos a la Universidad de las Fuerzas Armadas ESPE publicar en la biblioteca Virtual de la institución el presente trabajo de titulación *"Integración de aplicaciones de procesamiento en paralelo para proveer de infraestructura de ti a los proyectos de investigación de la ESPE basado en la arquitectura HPC"* cuyo contenido, ideas y criterios son de nuestra autoría y responsabilidad.

Quito, 7 de Agosto de 2017

Luis Gonzalo Rocha Hoyos

1725412306

Freddy José Vera Luque

1312139866

DEDICATORIA

A mis Padres Juan Rocha y María Hoyos, por darme la vida, creer en mí y por brindarme apoyo siempre, todo esto se los debo a ustedes.

Mis hermanos, Franklin Vinicio y Juan Carlos, por cuidarme y ayudarme a ser la persona que ahora soy, los quiero mucho.

Mi hermosa novia Adriana sin ella esto no fuese posible.

Todos mis amigos, Marcelo, Edison, Stalin, Cristóbal, Raúl, Damián, Eduardo, Iván, Andrés, Dennys, José, Ronald, Martin, Victoria, Edgar, Wilmer, Christian, Mauricio, por estar conmigo en los buenos y malos momentos.

Todos aquellos familiares y amigos que no recordé al momento de escribir esto, ustedes saben quiénes son.

Luis Rocha

A mi madre María del Carmen Luque, por siempre tener toda su fe depositada en mí, y brindarme su apoyo incondicional.

A mi hermana y prima María Teresa que siempre está presente para mí en todos los momentos de mi vida, te amo demasiado ñaña.

A mi familia por apoyarme en todas las decisiones que he tomado, sin ustedes no estaría donde estoy ahora.

A mis amigos Jorge, Omar, Ray, Ale, Dennys, Shane, Pao, Andrés, Bryan, Mario, millón gracias por estar conmigo en todo momento.

Freddy Vera

AGRADECIMIENTO

En el presente proyecto de investigación en primer lugar me gustaría agradecerle a Dios, Por haberme permitido llegar hasta este punto de mi vida, por brindarme la oportunidad de vivir experiencias incomparables, por fortalecer mi corazón e iluminar mi mente y por poner en mi camino a todas aquellas personas que han sido mi soporte y compañía durante todo el periodo de estudio.

A la Universidad de las Fuerzas Armadas ESPE por darme la oportunidad de estudiar y ser un profesional.

A mi director de proyecto de investigación, Dr. Diego Marcillo por su esfuerzo y dedicación, quien, con sus conocimientos, su experiencia, su paciencia y motivación ha logrado en mí que pueda terminar mis estudios con éxito.

También me gustaría agradecer al Ing. Santiago Salvador durante todo el proyecto de investigación porque ha aportado con un granito de arena a mi formación, y en especial a la Ing. Margarita Zambrano por su enseñanza y más que todo por darme la oportunidad de demostrarle mis talentos.

Y por último a Freddy Vera mi gran amigo y excelente compañero en este proyecto de investigación, gracias amigo por todo el apoyo y tu excelente trabajo. ¡Lo logramos!

Luis Rocha

Primeramente, agradezco a la Universidad de las Fuerzas Armadas ESPE por abrirme las puertas y darme la oportunidad de estudiar mi carrera, así como a los docentes que me brindaron sus conocimientos para ser el profesional que soy ahora.

Agradezco a mi tutor de Tesis el Ingeniero Diego Marcillo por su valioso tiempo y darnos la oportunidad de hacer un proyecto a su nombre. De igual manera al Ingeniero Santiago Salvador por brindarnos su apoyo en la realización de la tesis y darnos su tiempo cuando más lo necesitábamos.

Para finalizar agradezco a todos mis compañeros y amigos, en especial a mi compañero de tesis Luigi, hemos pasado por mucho amigo lo logramos.

Freddy Vera

INDICE

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

RESUMEN

En la mayoría de ingenierías y áreas científicas, se realizan investigaciones que involucran grandes cálculos matemáticos y numéricos utilizados para generar modelados, simulaciones basados en algoritmos de alta complejidad lógica donde el uso de tecnologías de la información es un factor determinante para el desarrollo Estos procesos consumen muchos recursos de hardware y software que las computadoras convencionales no pueden cubrir, impidiendo satisfacer las necesidades de los investigadores. Para ello existen laboratorios y centros de datos especializados que se enfocan a realizar dicha labor. La arquitectura HPC es una solución a este inconveniente, ha sido utilizada por décadas para brindar soporte a la investigación científica resolviendo algoritmos y problemas con alta disponibilidad y rendimiento El presente trabajo de investigación propone una alternativa para resolver dicho problema a través de la integración de aplicaciones en paralelo como es el caso de BEAST y su biblioteca Beagle en el clúster de alto rendimiento en la Universidad de las Fuerzas Armadas ESPE. Además, la creación de un micrositio informativo para la comunidad investigadora y un sistema para la gestión del HPC Rumiñahui utilizando un framework y una metodología de desarrollo ágil.

Palabras claves:

HPC

COMPUTACIÓN PARALELA

BEAST

BEAGLE

CLUSTER

ABSTRACT

In most engineering and scientific areas, research is carried out involving large mathematical and numerical calculations used to generate modeling, simulations based on highly complex algorithms where the use of information technology is a determining factor for development. These processes consume many hardware and software resources that conventional computers can not cover, thus preventing the needs of researchers. For this purpose there are laboratories and specialized data centers that focus on this work. HPC architecture is a solution to this problem, has been used for decades to support scientific research solving algorithms and problems with high availability and performance This research proposes an alternative to solve this problem through the integration of parallel applications such as BEAST and its Beagle library in the high performance cluster at the University of the Armed Forces ESPE. In addition, the creation of an informative microsite for the research community and a system for the management of HPC Rumiñahui using a framework and an agile development methodology.

Keywords

HPC

PARALLEL COMPUTING

BEAST

BEAGLE

CLUSTER

CAPITULO I
INTRODUCCIÓN

1.1 ANTECEDENTES

Las aplicaciones de procesamiento en paralelo son aquellas que demandan un alto consumo de recursos lógicos y físicos en su ejecución ya sean estos por simulaciones de modelos matemáticos, climáticos, económicos, etc. El hardware de la máquina entra en juego ya que es preciso maximizar la relación entre el tiempo de cálculo útil y el perdido en el paso de mensajes, parámetros que dependen de la capacidad de proceso de las CPUs y de la velocidad de la red de comunicaciones.

Por otro lado, a lo largo del tiempo las infraestructuras de TI han ido acoplándose y/o actualizándose para cumplir estos requerimientos, ya sean estos en procesamiento, así como en velocidad de respuesta. Esto se debe a las altas demandas de los usuarios al requerir con más prontitud sus respuestas lógicas en las aplicaciones antes mencionadas.

Una arquitectura HPC (High Performance Computing), es una infraestructura usada por décadas para dar soporte a la investigación científica, es un conjunto de hardware y software de alto rendimiento y alta disponibilidad para el tratamiento de grandes volúmenes de información. Se caracteriza por realizar todos cálculos basados en procesamiento en paralelo y disminuir el consumo de tiempo en procesos sumamente tardíos. HPC involucra el uso de plataformas de hardware y software avanzadas, que de forma colectiva permiten acelerar la ejecución de aplicaciones científicas.

La Universidad de las Fuerzas Armadas ESPE ya cuenta con la infraestructura que conforma un HPC en el campus Sangolquí que se encuentra instalada y en funcionamiento. Con el objetivo de fortalecer el desarrollo de la investigación de la institución, ésta pone a disposición para la comunidad el CLÚSTER DE SUPERCOMPUTACIÓN RUMIÑAHUI. Éste permitirá a los investigadores ejecutar tareas que requieren de gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez, con alta disponibilidad y confiabilidad.

1.2 PROBLEMÁTICA

Actualmente para el procesamiento de proyectos de investigación en la Universidad de las Fuerzas Armadas ESPE no se está utilizando un sistema HPC por lo cual los cálculos complejos pueden tardar horas en generar resultados. Éstos pueden tardar inclusive días para generar un reporte que los grupos de investigadores puedan utilizar para avanzar con su proyecto. El problema radica en que algunas de estas herramientas de software necesitan computadoras con recursos de hardware extremadamente altas que dificulta de esta manera la ejecución de procedimientos, simulaciones, modelados 3D, modelos matemáticos, etc. En el HPC Rumiñahui no se encuentran aplicaciones de software instaladas para apoyo a los requerimientos de los investigadores y de esta manera puedan realizar cálculos que requieren altos recursos de procesamiento.

1.3 JUSTIFICACIÓN

HPC ha tenido tremendo impacto sobre todas las áreas de las ciencias computacionales, gobierno e industria, entre otras. Muchos problemas podrían ser solucionados con técnicas propias de esta arquitectura que eran imposibles de solucionar con estaciones de trabajo o computadores personales.

Con el fin de impulsar el uso de una plataforma HPC, al momento la Universidad de las Fuerzas Armadas ESPE ya cuenta con esta arquitectura, la cual se encuentra en funcionamiento. Con la utilización del mismo y la integración de las aplicaciones necesarias se podrá dar solución a los proyectos de investigación de la universidad que requieran altos procesamientos en información para poder mejorar tiempos de respuesta de los cálculos requeridos.

Con respecto a las aplicaciones a necesitar, se optará por la utilización de software de licencia libre, los cuales serán instalados en la plataforma. De esta manera la universidad no manejara gastos adicionales en licenciamientos.

1.4 OBJETIVOS

1.4.1 Objetivo General

Integrar aplicaciones de procesamiento en paralelo para proveer de infraestructura de TI a los proyectos de investigación de la ESPE basado en la arquitectura HPC.

1.4.2 Objetivos Específicos

- Investigar e integrar una aplicación que permita proveer la arquitectura HPC a los investigadores de la ESPE.
- Desarrollar un micrositio y un sistema web de gestión de investigadores para proveer de información sobre el uso y la disponibilidad de la plataforma HPC.
- Ejecutar simulaciones y modelos matemáticos haciendo uso de la aplicación instalada para analizar las ventajas del procesamiento en paralelo sobre una arquitectura HPC en comparación a una arquitectura sin HPC.
- Elaborar un manual de políticas y procedimientos sobre el proceso de instalación, puesta en producción y acceso a investigadores para los servicios de HPC.

CAPITULO II
ESTADO DEL ARTE

En este capítulo se despliegan los temas tratados en la realización del proyecto de investigación. Se presenta el estudio de los recursos de hardware y software involucrados, se investiga la arquitectura de computación de alto rendimiento (HPC por sus siglas en ingles High Performance Computing) y sus tecnologías, se realiza el análisis de los lenguajes de programación, marcos de trabajo, protocolos que intervienen en el desarrollo. Además, se detalla el estudio de la metodología utilizada y cómo se acopla a las necesidades del proyecto.

2.1 Procesamiento en paralelo

De forma general, la eficiencia de una máquina, para resolver un problema dado, depende de la implementación de un algoritmo de resolución. En particular, las máquinas paralelas ofrecen un poder de cálculo enorme, pero explotar todas sus potencialidades no es fácil (Aguilar & Leiss, 2004).

Es importante tener en cuenta que la eficiencia de una máquina se ve afectada por la intervención de varios factores físicos como son la memoria, procesadores y el almacenamiento y de factores lógicos como son núcleos procesamiento y velocidad de respuesta.

También existen problemas los cuales se presentan al realizar aplicaciones paralelizables. La mejor forma de resolverlos no consiste en crear un buen algoritmo, sino conocer que fuente del paralelismo se ajusta a la resolución del problema. Es aquí donde entran en juego los siguientes modelos:

2.1.1 Modelo concurrente

Consiste en la ejecución de múltiples tareas de manera simultánea, estas pueden ser producto de algún hilo o proceso pertenecientes a un programa (Figura 1). Estos pueden ser ejecutados en uno o en varios procesadores, en algunos casos también se presentan en red. Este modelo se enfoca en la interacción de las tareas.

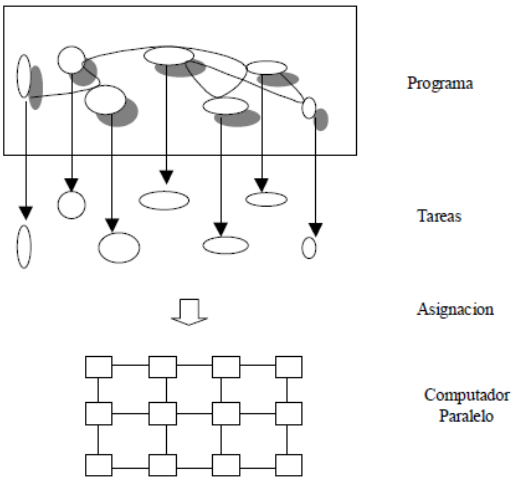


Figura 1 Modelo Concurrente

Fuente: (Aguilar & Leiss, 2004)

2.1.2 Modelo paralelo

Se trata de la ejecución de tareas simultáneamente en uno o en varios procesadores, se enfoca en el tiempo de ejecución de las tareas (Figura 2). Generalmente en este caso el paralelismo se presenta en forma implícita, es decir que el paso de mensajes entre los procesos permite que las tareas interactúen entre sí, generando mutua información para la resolución del problema presentado.

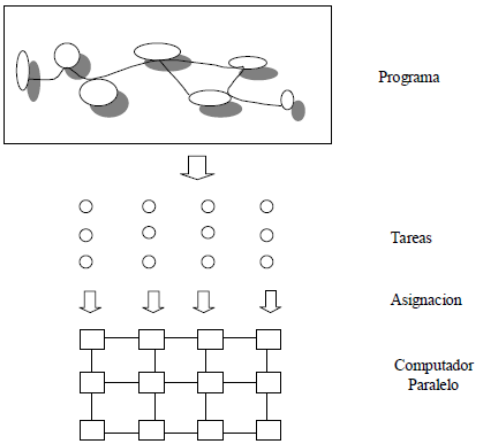


Figura 2 Modelo Paralelo

Fuente: (Aguilar & Leiss, 2004)

2.2 Clústeres

"Tipo de computación paralela o de procesamiento distribuido, formado por una colección de computadores individuales interconectados entre sí, trabajando conjuntamente en un objetivo unificado de cómputo" (Navas, 2006).

Un clúster se define como el conjunto de dos o más equipos de cómputo que se comportan como uno solo, compartiendo recursos de hardware y software a través de una conexión de red (Figura 3), además incorpora un sistema para la interacción del usuario permitiéndole utilizar de manera óptima los recursos del clúster.

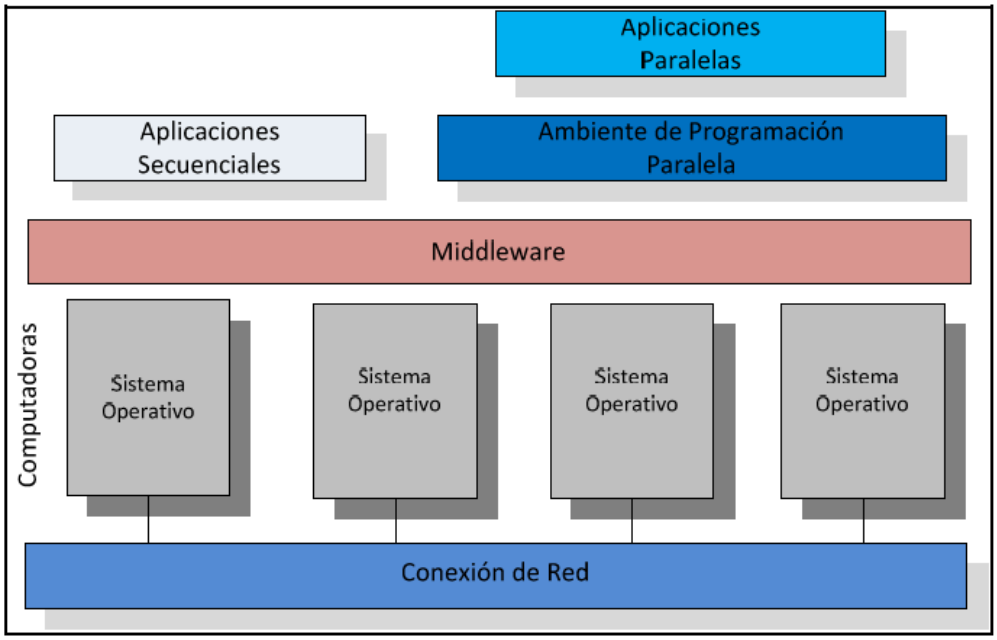


Figura 3 Arquitectura de un Clúster

Fuente: (Arellano & Fernández, 2014)

2.3 Componentes de un clúster

Esta infraestructura requiere de múltiples componentes tanto de hardware y software para poder funcionar, como se describen a continuación:

- **Nodos**

Son diferentes puntos que conformar el clúster, no tienen la necesidad de poseer hardware adicional como son mouse, teclado o monitor; sólo están vinculados compartiendo sus recursos con el nodo principal por medio de una red de conexión.

- **Almacenamiento**

Consiste en el lugar físico o lógico donde se guarda la información. Un clúster posee un almacenamiento centralizado, es decir todos los nodos archivan la información en un único dispositivo. Este almacenamiento puede ser a través de tecnologías como Red de Área de Almacenamiento (SAN), Almacenamiento conectado en Red (NAN) o interno del servidor.

- **Sistema operativo**

Software que permite controlar los recursos del computador, entre ellos tenemos la memoria, procesador, etc. Permite la intercomunicación con el usuario y la máquina.

- **Middleware**

"Un Middleware puede ser visto como un conjunto de servicios y funciones reutilizables, expandibles, que son comúnmente utilizadas por muchas aplicaciones para funcionar bien dentro de un ambiente interconectado." La organización IETF (Internet Engineering Task Force) en mayo de 1997.

Es un concepto normalmente utilizado en la informática, se refiere a la compartición del trabajo a realizar entre uno o más hardwares utilizando múltiples unidades de procesamiento. También se lo conoce como un software de conectividad que usa varios servicios permitiendo la interacción de múltiples procesos que se ejecutan en distintos equipos.

- **Ambiente de programación**

Para la programación paralela existen otros ambientes muy diferentes a los convencionales, estos son programas que permiten realizar aplicaciones exclusivas para sistemas con recursos compartidos donde el intercambio de mensajes, el uso de memoria compartida, el número de hilos de procesamiento y la velocidad del bus de datos entra en juego.

2.4 Computación de Alto Rendimiento (High Performance Computing)

"La computación de alto rendimiento es el uso de procesamiento paralelo para ejecutar aplicaciones avanzadas de manera eficiente, confiable y rápida." (UTIC, 2016). Mejor conocido como HPC o supercomputadora, está enfocado a unidades que trabajen por encima de los Petaflops (10^{15}) velocidad de procesamiento de información.

Se caracteriza por ejecutar programas paralelizables que demandan muchos recursos, entre ellos un alto grado de procesamiento y uso de memoria. Son utilizados en el área científica para resolver problemas o simulaciones que requieran grandes cálculos matemáticos, como puede ser en la renderización de gráficas y en mejorar el tiempo de respuesta en la solución de problemas.

2.5 Métodos de acceso a un clúster HPC

2.5.1 VPN (Virtual Private Network)

Tecnología que permite la extensión de redes locales o LAN sobre una red pública como es el caso del internet, hacia una red remota. Esto lo realiza mediante procesos de encriptación y de encapsulamiento de paquetes de datos sobre varios puntos remotos haciendo uso de infraestructuras públicas de transporte. (Alvarez, Jorquera, Sepúlveda, & Zamora, 2014). Es una de las tecnologías más utilizadas que permite a usuarios o terceros acceder a redes corporativas y a sus diferentes servicios mediante el internet.

Figura 4. Muestra el diagrama básico de funcionamiento de una VPN desde el acceso de un host remota hasta conectarse a un cliente local utilizando un túnel encriptado por medio del internet.

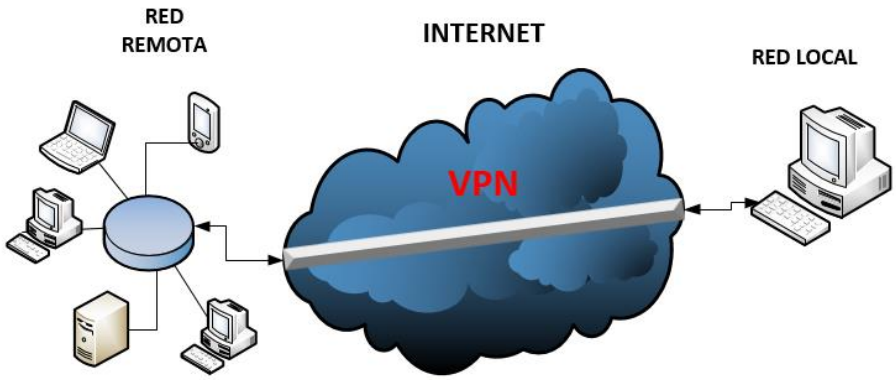


Figura 4 Diagrama Básico VPN

2.5.2 Tipos de VPN

En los sistemas de información actualmente se encuentran varios tipos de VPN, los cuales son utilizados para acceder a redes basados en la arquitectura cliente/ servidor. Existen 3 tipos como como son:

- **VPN de acceso remoto**

Consiste en la conexión remota de usuarios hacia una red cualquiera haciendo uso del internet como medio de acceso. Una vez realizado el enlace, los usuarios tienen los mismos privilegios como si se encontraran en la red local.

- **VPN punto a punto**

Permite la conexión de diferentes redes mediante el internet, posee un acceso constante, es utilizado para enlazar oficinas remotas con la central como una de sus varias aplicaciones.

- **VPN interna**

Mantiene el concepto de VPN de acceso remoto, a diferencia que este utiliza solo una red local, aumentando la seguridad en un sistema en redes WIFI[1]. Esta VPN es utilizada en centros o empresas para proporcionar seguridad adicional y evitar instrucciones en el sistema.

2.5.3 Protocolos y estándares

Al presentarse el uso de VPN entran en juego varias opciones en la elección de protocolos para realizar los enlaces remotos. Existen varios entre los más utilizados son:

- **Protocolo SSH (Secure Shell)**

Según Smaldone el protocolo SSH permite la comunicación, ejecución y logueo para procesos remotos (Smaldone, 2004) . Se caracteriza por ser una aplicación cliente/ servidor haciendo uso del puerto 22 de TCP. Este protocolo genera un túnel para la gestión de equipos haciendo uso de la consola mediante comandos, realiza la conexión entre las dos redes permitiendo así la comunicación (Figura 5).

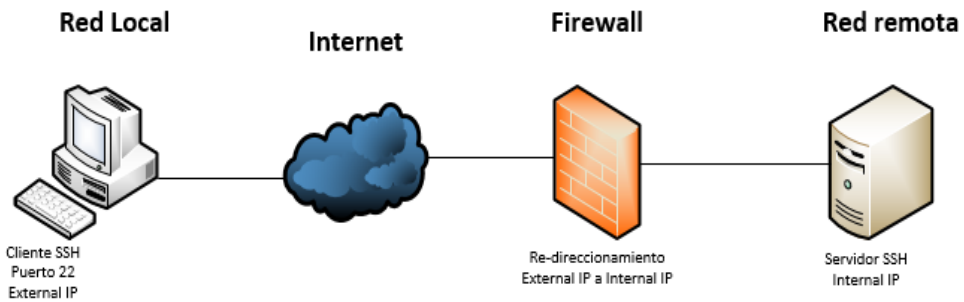


Figura 5 Túnel SSH

[1] WIFI: Wireless Fidelity (Fidelidad inalámbrica) Tecnología de comunicación inalámbrica.

Según Smaldone para manipular esta información existen técnicas de cifrado que aseguran la confidencialidad, integridad y autentificación (Smaldone, 2004). Entre estas técnicas tenemos:

- Cifrado simétrico: Consiste en la utilización de una clave pública para el envío de mensajes la misma debe ser conocida por cliente y el servidor.
- Cifrado Asimétrico: Usa claves públicas y privadas para la transmisión de datos. Cuenta con dos extremos, en un extremo se conoce la clave privada, los mensajes son enviados y se cifran en base a la clave pública para luego ser descifrado por el otro extremo.

- **Protocolo SFTP (SSH File Transfer Protocol)**

Es un protocolo de comunicación, que permite la manipulación de archivos para copiarlos o transferirlos desde el computador local al remoto mediante una interfaz de usuario (Figura 6). Para su implementación, utiliza el puerto 22 de TCP.

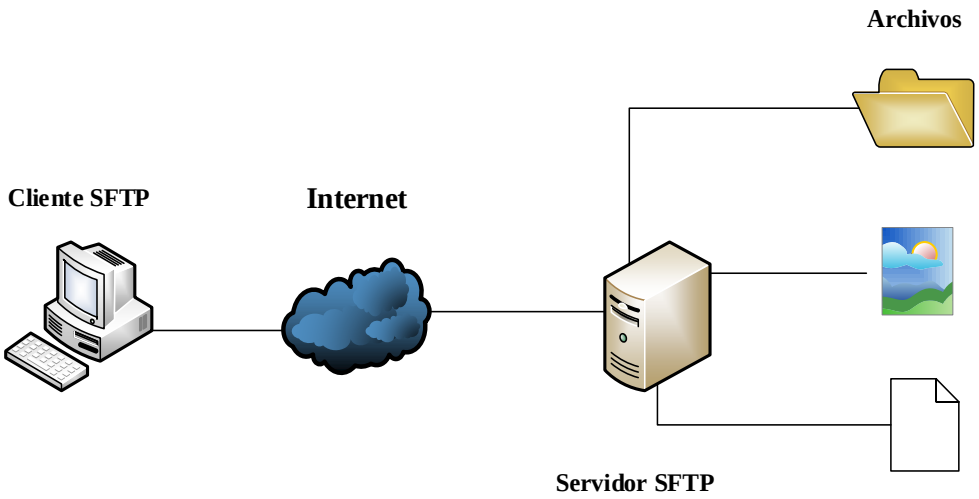


Figura 6 Diagrama SFTP

2.5.4 Otros

- **Terminal**

Es una interfaz de líneas de comandos o consola, permite la manipulación del computador mediante el uso comandos. El usuario ejecuta órdenes digitando los comandos, mientras que el computador se encarga de ejecutarlos y mostrar un resultado en pantalla. Históricamente este método existe desde hace varios años; se destacaba por mostrar resultados inmediatos. Con el pasar de los tiempos éstas evolucionaron hasta convertirse en interfaces gráficas. (Universidad de las Ciencias Informáticas, 2013).

2.6 Herramientas y softwares

2.6.1 Sistemas operativos

- **Red Hat Enterprise LINUX (RHEL)**

Perteneciente a la compañía RED HAT ENTERPRISE es una distribución comercial de GNU/LINUX, conocida por ser la primera en ser desarrollada. Es un sistema operativo enfocado primordialmente para la administración de clústeres y centro de datos debido a su alto control y manejo de información (Cobbaut, 2015).

- **Debian**

Es un sistema operativo de licencia libre, se caracteriza por trabajar sobre el núcleo de Linux basado en el proyecto GNU con más de 50000 paquetes de software, actualmente se encuentra en la versión 8.8 y cuenta con más de 9 millones de desarrolladores que aportan a la mejora de su código. (Debian news, 2017).

- **Ubuntu**

Sistema operativo basado en GNU/Linux distribuido como software libre, posee su propio entorno de escritorio conocido como Unity. Ubuntu es compatible con hardware de computadores o servidores de 32 y 64 bits. Actualmente se encuentra en la versión 17.01, se caracteriza por la facilidad de uso orientado hacia el usuario. Ubuntu cuenta con su propio repositorio para instalación de aplicaciones. Posee altas políticas y permisos convirtiéndolo en un software seguro (Ubuntu, 2017).

2.6.2 Aplicaciones para HPC

- **Moab**

La Suite clúster de MOAB es un paquete para la administración de carga de trabajo, se especializa en el monitoreo, gestión y reportes de un sistema clúster. Acelera los procesos unificando los recursos tanto físicos como lógicos, optimizando las técnicas de análisis para garantizar resultados eficientes.

- **Torque**

Torque es un administrador de recursos para el control y gestión de tareas sobre los diferentes nodos disponibles en el clúster.

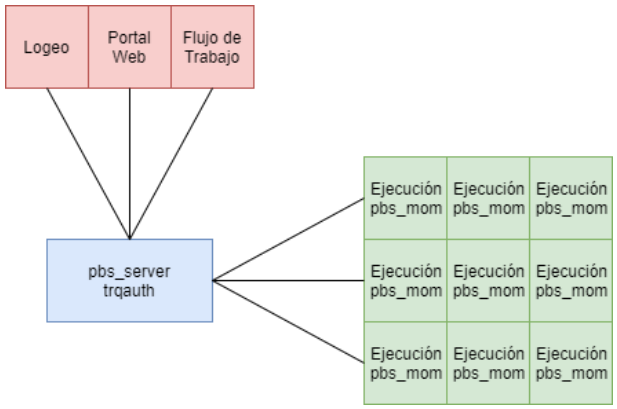


Figura 7 Diagrama de Topología de Torque

Fuente: (Haymore, 2014)

Pbs_server: Torque Batch Server [2] aplicativo que se corre sobre el nodo principal de un clúster. Pbs_server provee de:

- Configuración de recursos, políticas y colas.
- Scripts y estados del trabajo que se almacenan en el pbs_server.
- Registros e historiales para controlar tiempos de trabajo.

[2] Torque Batch Server conocido como Servidor Lote de Torque

Pbs_mom: Torque Batch Execution mini-Server [3]aplicativo que corre en los nodos segundarios del clúster. El nodo principal se conecta a cada nodo segundario para distribuir las cargas de trabajo. Realizando acciones como:

- Monitorea y limita el uso de los recursos.
- Notifica al pbs_server cuando termine de realizar un trabajo.
- Interactúa con el nodo principal para realizar la misma tarea en varios nodos.

Trqauth: Torque Autorization Daemon [4]programa que se ejecuta sobre cada nodo del clúster y permite la intercomunicación usuario-máquina.

El flujo de trabajo de Torque se enfoca utilizando los nodos principales y los nodos segundarios (Figura 8). Las tareas son enviadas desde el nodo principal hacia los nodos segundarios dónde son procesadas basándose en los parámetros especificados en el script de ejecución.

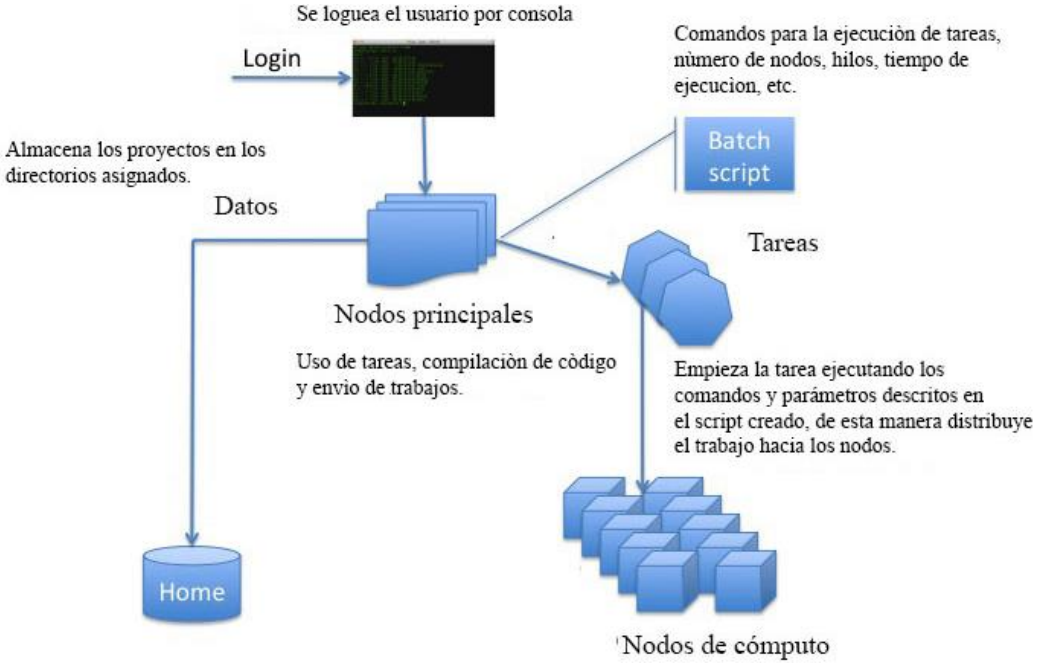


Figura 8 Flujo de Torque

Fuente: Adaptado (Universidad de Chicago, 2017).

[3] Torque Batch Execution mini-Server denominado Mini servidor de ejecución de Lote de Torque
[4] Torque Autorization Daemon denominado como proceso o demonio de autorización de Torque

- **Beast v1.8.4**

Bayesian Evolutionary Analysis Sampling Trees (BEAST) es una aplicación multiplataforma utilizada para análisis filogenéticos bayesianos de secuencias moleculares utilizando la cadena de Markov Monte Carlo (MCMC). Es un mecanismo empleado para la generación de árboles de probabilidades, utilizado para probar hipótesis evolutivas en una única topología (Drummound, Ho, Rawlence, & Rambaut , 2007).

- **BEAUti**

Bayesian Evolutionary Analysis Utility (BEAti) Es un programa de uso exclusivo de BEAST. Presenta varios paneles de configuración los cuales contienen una variedad de parámetros que pueden ser utilizados posteriormente para la configuración de archivos XML (Drummound, Ho, Rawlence, & Rambaut , 2007).

- **Beagle**

Es una biblioteca para BEAST de alto rendimiento sirve para equilibrar procesos y realizar cálculos avanzados de procedencia Bayesiana o filogenética. Su principal característica es el uso de procesadores en paralelo, permitiendo que los procesos se ejecuten de manera óptima (BEAST, 2016).

"Beagle permite realizar cálculos sobre plataformas de hardware GPU y CPU multicore" (The National Center for Biotechnology Information, 2011). Proporciona métodos y algoritmos para evaluar las probabilidades.

2.6.3 Herramientas de acceso al HPC

- **Forticlient**

FortiClient es una oferta de seguridad unificada diseñada para PC, portátiles, tablets[5] y dispositivos móviles. Las características incluyen SSL y IPsec VPN, antivirus, antimalware, filtrado web, firewall de aplicaciones, evaluación de vulnerabilidades y más.

[5] Tablet: Dispositivo electrónico, es un tipo de computador portátil.

FortiClient está totalmente integrado con varias aplicaciones de su propia distribución para administración, monitoreo y registro. FortiClient Enterprise Management Server (EMS) simplifica el despliegue, registro, administración y supervisión de extremo a extremo. (Forticlient, 2017).

- **Putty**

Es una herramienta de distribución libre, utiliza el protocolo SSH para realizar la conexión entre un cliente y un servidor (Eng Hee, 2010). Putty se caracteriza por permitir conexiones remotas de diferentes tipos, estas son:

- **Raw**

Permite la conexión entre un usuario y un host remoto. Es un protocolo de comunicación inseguro que permite enviar o recibir objetos. Utilizado en sistemas donde el acceso a un servidor es libre y no necesita seguridad, permitiendo que cualquier persona con acceso a la red puede ingresar al mismo, (CODECALL, 2011).

- **Telnet**

Protocolo que se utiliza para conectar aplicaciones en internet y terminales utiliza TCP. “El propósito del protocolo TELNET es proporcionar un sistema de comunicaciones general, bidireccional y orientado a bytes.” (Universidad Carlos III de Madrid, 2002).

- **Rlogin**

Conocido como Remote Login Protocol[6] su uso es similar al de TELNET a diferencia que este utiliza UDP. Permite el acceso remoto creando una sesión accesible sólo a los usuarios que se encuentren en una lista de hosts (Rhost). Esta lista describe cuales son los roles de cada usuario. Rlogin es utilizado en inicio de sesiones simples donde no se necesitan muchos controles sobre la interacción cliente/host. (TechTarget, 2017).

[6] Remote Login Protocol o Protocolo de inicio de sesión

- **Security Shell (SSH)**

Es un protocolo que facilita la comunicación entre dos sistemas haciendo uso de una arquitectura cliente/servidor. Conecta usuarios desde un cliente local hacia un host remoto independientemente del sistema operativo que ambos utilicen.

- **MOBAXTERM**

Paquete de herramientas para conexión remota, entre un cliente y un host remoto. Es una aplicación para uso exclusivo de sistemas operativos Windows[7], Mobaxterm proporciona muchas funciones que se adaptan a los programadores, administradores de TI y a todos los usuarios que necesitan gestionar sus trabajos remotos.

Está basado en el protocolo SSH y realiza la misma función que PUTTY, pueden añadirse múltiples complementos que mejoran su eficiencia y permiten al usuario realizar diferentes tareas. Proporciona herramientas como:

- **VNC**

Denominado Computación Virtual en Red, es un protocolo simple que permite acceder de manera remota a interfaces gráficas de usuario. Como se aprecia en la (Figura 9) el protocolo permite a un servidor actualizar el framebuffer[8] que se muestra en un visor, es potencialmente aplicable a todos los sistemas operativos. (Universidad de Cambridge, 2012)

[7] Windows: sistema operativo licenciado de la empresa Microsoft.
[8] Framebuffer: Espacio de memoria, ésta se reserva para mantener imágenes ráster de forma temporal conocidas como frames, para luego ser enviadas a un dispositivo o monitor (Alegsa, 2010).

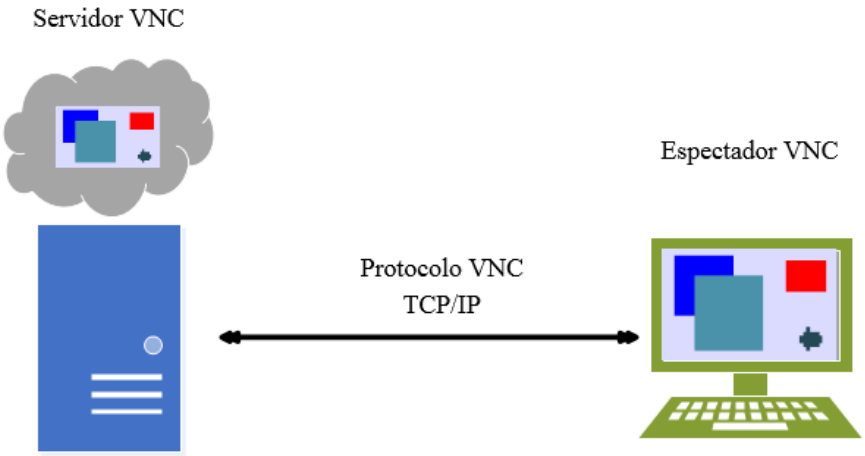


Figura 9 Protocolo VNC

Fuente: Traducido (Universidad de Cambridge, 2012).

- **RDP**

Conocido como Protocolo de Escritorio Remoto, permite acceder a un host remoto o a un host en una red local, consiste de un entorno cliente-servidor. Este protocolo es propio de los sistemas operativos Windows. Trabaja con interfaz de usuario y permite acceder a la consola de dichos dispositivos (Figura 10). Además, permite que toda información compartida entre el host y el cliente local esté comprimida ofreciendo un mejor rendimiento (Rose, 2008).

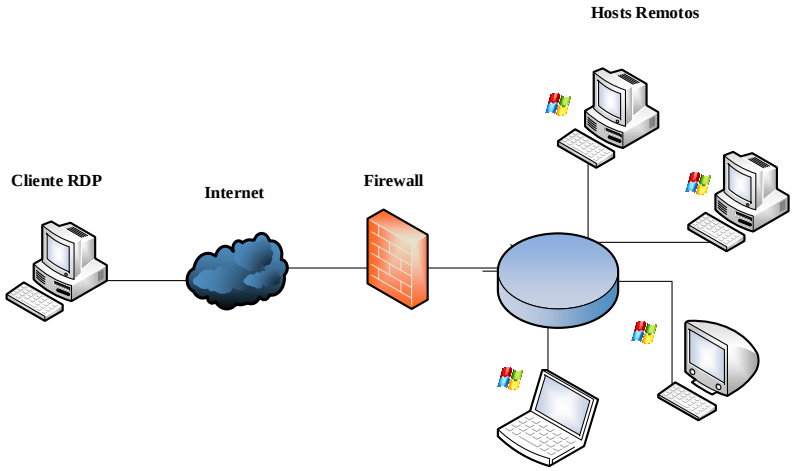


Figura 10 Protocolo RDP

- **X11**

X11 es un sistema de ventanas destinado a proporcionar un entorno GUI en los sistemas operativos Unix. Este servidor se basa y se compila con el fin de ejecutarse de forma nativa sobre una plataforma Windows. También puede ser utilizado para renderizar escritorios remotos (MobaXterm, 2017).

- **MOSH**

Conocida como Shell móvil, según Brockmeter la aplicación remota de terminal se caracteriza por permitir roaming[9], soporta conectividad intermitente, y proporciona eco local inteligente y edición de línea de pulsaciones de usuario" (Brockmeter, 2012). Independientemente de donde se encuentren los usuarios y que red utilicen para acceder, esta tecnología permite que la conexión se mantenga constante A diferencia de otras herramientas, MOSH utiliza UDP lo que le permite tener sesiones de terminal activas. Es utilizado como una variante del protocolo SSH. Es más sensible y robusto principalmente en conexiones Wifi y móviles.

- **WINSCP**

Es una herramienta de software libre. Es un cliente SFTP para Windows que emplea el protocolo SSH. Es útil en la transferencia de archivos desde un servidor remoto a un cliente local sean éstos de igual o diferentes sistemas informáticos, se caracteriza por poseer una interfaz gráfica facilitando el uso a los usuarios, (WinSCP, 2017).

WINSCP permite la manipulación de los archivos y directorios, haciendo con ellos operaciones básicas como son la transferencia, eliminación y creación, modificación de permisos y privilegios.

[9] Roaming: Mantiene una conexión inclusive cuando el usuario cambia de IP o de zona de cobertura.

2.6.4 Herramientas de desarrollo

2.6.4.1 Gestor de Base de datos

- **MYSQL**

MySQL en el gestor de base de datos de código abierto más popular del mundo. Con su excelente rendimiento, fiabilidad y facilidad de uso, MySQL se ha convertido en la elección líder para aplicaciones basadas en web. Oracle impulsa la innovación de MySQL, ofreciendo nuevas capacidades para la próxima generación de aplicaciones web, móviles e incrustadas (MySQl, 2017).

Entre alguna de las características de MySQL tenemos:

- Amplio uso del lenguaje SQL.
- Disponible para la mayoría de plataformas y sistemas operativos.
- Excelente administrador tanto en modo consola como modo gráfico.
- Base de datos transaccionales incluyendo el uso de índices primarios y foráneos.
- Conectividad segura.
- Permite la replicación.

- **POSTGRESQL**

PostgreSQL es un gestor de base de datos relaciones de código abierto, con aproximadamente 16 años de desarrollo, es compatible con la mayoría de los sistemas operativos (PostgreSQL, 2017).

Algunas de sus características son:

- Es una base de datos transaccional/relacional.
- Amplia documentación.
- Permite respaldos.
- Soporta variedad de tipos de datos como son: enteros, cadena de bits, etc.
- Es un sistema multiplataforma.

- Su código libre puede ser modificado por los usuarios.
- Fácil manejo y administración.

PostgreSQL puede ser utilizado, modificado y distribuido por cualquier persona para cualquier propósito, ya sea privado, comercial o académico (The PostgreSQL Development Group, 2017).

- **ORACLE**

Oracle es un gestor de base de datos, su propósito es de almacenar y recuperar información. Es una base de datos relacional utilizada en la mayoría de empresas y negocios alrededor del mundo (TechTarget, 2017).

Se destaca por tener una arquitectura cliente/servidor y poseer un veloz motor de ejecución, es un software con licenciamiento de la empresa Oracle Corporation una de las compañías más grandes de desarrollo de software alrededor del mundo (I.E.S. San Vicente, 2017).

Entre algunas características se tiene las siguientes (Oracle, 2017):

- Su base de datos posee una estructura lógica y otra física que están separadas permitiendo que el almacenamiento se pueda gestionar sin afectar su rendimiento y funcionamiento.
- Puede ser instalado en cualquier plataforma y es soportado por todos los sistemas operativos.
- Ofrece disponibilidad, integridad y fiabilidad de la información.

2.6.4.2 Lenguajes de programación

- **PHP**

PHP (Hypertext Preprocessor) es un lenguaje muy popular, en la actualidad es mayormente utilizado en el desarrollo web debido a que es de código abierto y además se adapta a HTML para dar formatos de contenido. PHP se distingue de otros lenguajes de programación porque se ejecuta en la parte del servidor evitando así que su código sea accesible desde la parte exterior.(PHP, 2017).

PHP es configurado de manera que las peticiones que lleguen al servidor sean ejecutadas y devuelvan una respuesta al cliente; incluso puede ser especificado para que éste ejecute comandos en HTML u otros lenguajes de programación. PHP es multiplataforma, permite la programación orientada a objetos, además de poseer una amplia documentación.

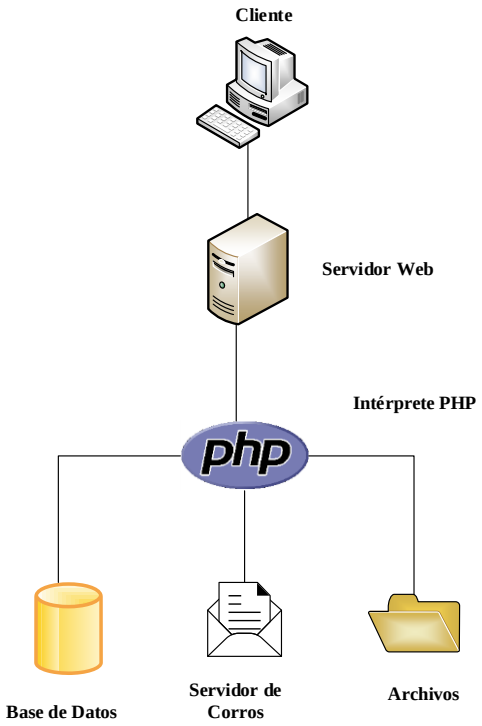


Figura 11 Diagrama PHP

Fuente: (Webucator, 2017)

- **Java**

Java es un lenguaje de programación orientado a objetos. Según Pavón este lenguaje puede ser ejecutado en varios sistemas operativos gracias a su fácil distribución, ofrece seguridad y un alto soporte por parte de la comunidad de usuarios. (Pavón, 2010).

Entre algunas de las características que JAVA ofrece están:

- Encapsulamiento.
- Gestión de las excepciones
- Soporte para procesos multihilo.

Como cualquiera de los lenguajes de programación, JAVA posee una estructura propia, paradigmas de programación y reglas de sintaxis. Es un lenguaje derivado del lenguaje C que a diferencia utiliza paquetes donde se encuentra el código, objetos y dentro de estas las clases, métodos y variables. (IBM, 2017)

2.7 Herramientas para construcción de sitios web

- **HTTP**

El protocolo de Transferencia de Hipertexto es uno de los más utilizados en la internet. Su principal propósito es la transferencia de archivos entre ellos los de formato HTML entre un cliente y un servidor web (CCM, 2008).

Figura 12. Muestra la arquitectura HTTP, en ella intervienen tres entidades el cliente, el servidor web y el almacén de datos que utilizan el protocolo TCP/IP como medio de acceso. Esta arquitectura funciona de la siguiente manera:

- El cliente crea una solicitud
- El servidor procesa la solicitud creada y luego regresa con la respuesta HTTP.

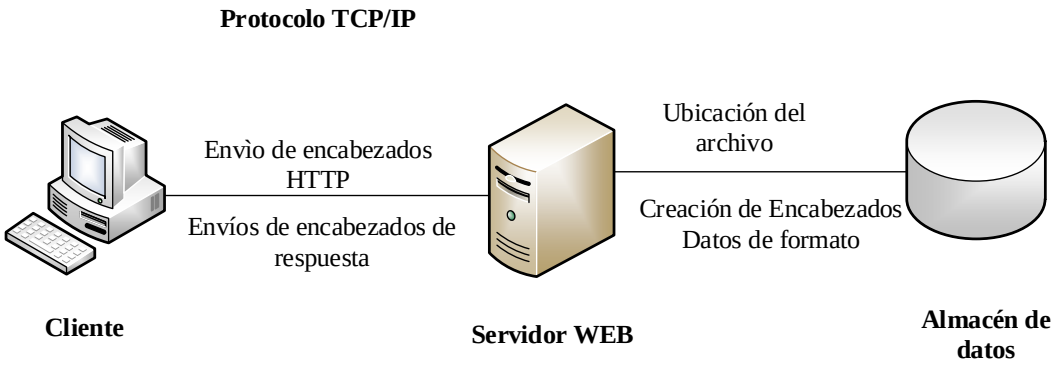


Figura 12 Arquitectura HTTP

Fuente: (CCM, 2008).

- **HTML5**

Lenguaje de marcado de hipertexto (HTML) es utilizado en internet para la especificación de contenido de las páginas web. Entre sus características puede ser escrito en cualquier editor de texto y es puramente código ASCII[10]. Según la Universidad del Pais Vasco "HTML es un lenguaje sencillo y eficiente. Aunque no puede competir con los procesadores de texto en capacidades de formato, es universal, es hipertexto e hipermedia, es muy accesible, sus ficheros ocupan poco espacio en disco" (Universidad del Pais Vasco, 2017).

- **CSS**

Las hojas de estilo en cascada (Cascading Style Sheets) es utilizada para dar estilos a documentos HTML y XML generalmente separando el código de la presentación para mantenerlos ordenados. Es un mecanismo que describe cómo debe mostrarse el documento en pantalla y ser visualizado por el usuario. Esto ofrece a los desarrolladores orden y control total sobre el estilo de sus documentos (World Wide Web Consortium , 2017).

- **JavaScript**

Al igual que muchos otros lenguajes de programación este se encarga de controlar el comportamiento de las páginas web haciéndolas más dinámicas en lo que respecta al formato (DesarrolladorWeb, 2017), permite una mejor interacción con el usuario. Se caracteriza por ser un lenguaje integrador debido a que no solo se encuentra en páginas web sino también en códigos de sistemas operativos.

- **Cpanel**

Cpanel es un panel que permite gestionar cuentas o dominios de alojamiento, proporciona una excelente interfaz gráfica. Ésta a su vez posee una estructura dividida en tres diferentes niveles permitiendo así a los administradores tener más control sobre las diversas funciones del sitio web y la administración (Green, 2015). En esencia, Cpanel permite:

[10] ASCII: Sistemas de codificación de los caracteres alfanuméricos

- Crear y administrar cuentas de correo electrónico
- Administrar configuración de seguridad
- Configurar dominios, subdominios, dominios estacionados y redirecciones.
- Gestionar archivos, carpetas y controlar el uso del espacio en disco.
- Acceso a bases de datos y seguimiento del rendimiento de su sitio web

- **Composer**

Composer es una herramienta para la gestión de dependencias en PHP. Permite declarar las bibliotecas de las que depende un proyecto. Composer no es un gestor de paquetes, pero los administra por proyecto, instalándolos en un directorio (GetComposer, 2017). Se caracteriza por ser software libre y puede ser ejecutado bajo cualquier sistema operativo, también:

- Permite declarar las bibliotecas de las que hace dependencia.
- Averigua las versiones de los paquetes que pueden ser instalados y actualizados.

- **Archivos XML (eXtensible Markup Language)**

El XML es planteado como un lenguaje estándar para el intercambio de información para diferentes programas de forma segura, libre y fiable Es utilizado en múltiples lenguajes de programación y hasta en la creación de páginas web; esto se debe gracias a la organización y manejo de los datos.

2.8 Patrón de Software y Frameworks

- **Patrón Modelo-Vista-Controlador (MVC)**

El denominado Modelo-Vista-Controlador es una arquitectura de desarrollo de software, que tiene como objetivo principal separar la interfaz de usuario, la aplicación y la lógica del negocio en tres distintos componentes que presentan relación entre ellos. (Universidad de Alicante, 2016).

El modelo-vista-controlador se presenta como una arquitectura madura mostrando su validez en todo tipo de aplicaciones sobre diferentes lenguajes en varias plataformas (Universidad de Alicante, 2016). Su complejidad se involucra en las dimensiones del proyecto a desarrollar siendo mayormente utilizado donde se manejan mayor cantidad de datos e información (Figura 13).

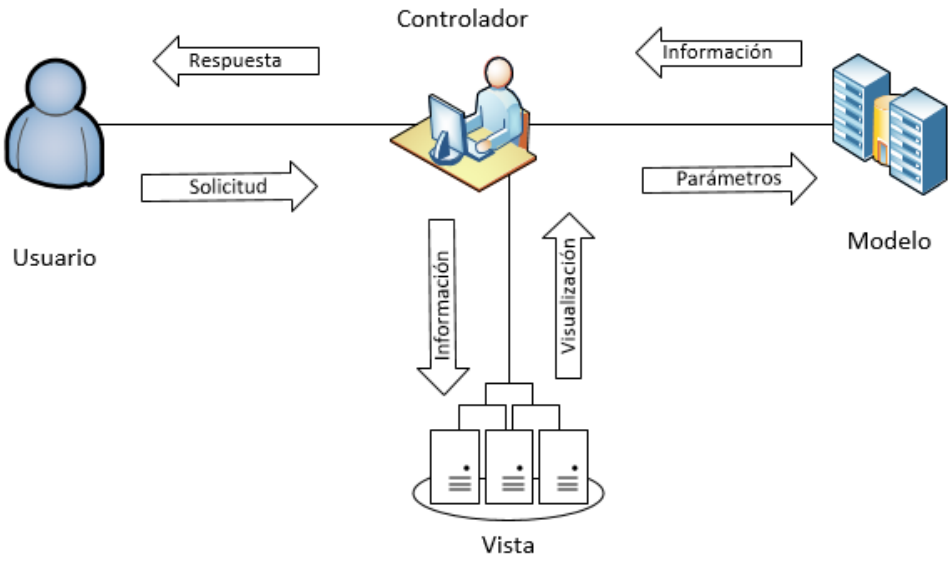


Figura 13 Patrón Modelo-Vista-Controlador

Fuente: (Universidad de Alicante, 2016)

Está conformado por tres elementos principales:

- Modelo: Controla los datos, la lógica del negocio y la persistencia del sistema a desarrollar, a través de mecanismos de actualización y accesos a la información.
- Vista: Se trata de la interfaz gráfica, permite la interacción usuario-sistema y muestra la información pertinente.
- Controlador: Es un intermediario entre el modelo y la vista, se encarga de mantener la comunicación entre ellas, permite el control del flujo de los datos e información adaptándolo según los requerimientos del usuario.

2.9 Frameworks de desarrollo de software

- **Yii Framework**

Yii es un framework[11] PHP de alto rendimiento y basado en componentes para desarrollar rápidamente aplicaciones web. El nombre Yii significa simple y evolutivo (Yii, 2017). Yii implementa el modelo de arquitectura modelo-vista-controlador. El directorio de modelos contiene todas las clases de modelo, el directorio de visas contiene todas las secuencias de comandos de vista y el directorio de controladores contiene todas las clases de controlador (Yii, 2017). En la (figura 14) se muestra la estructura estática de una aplicación:

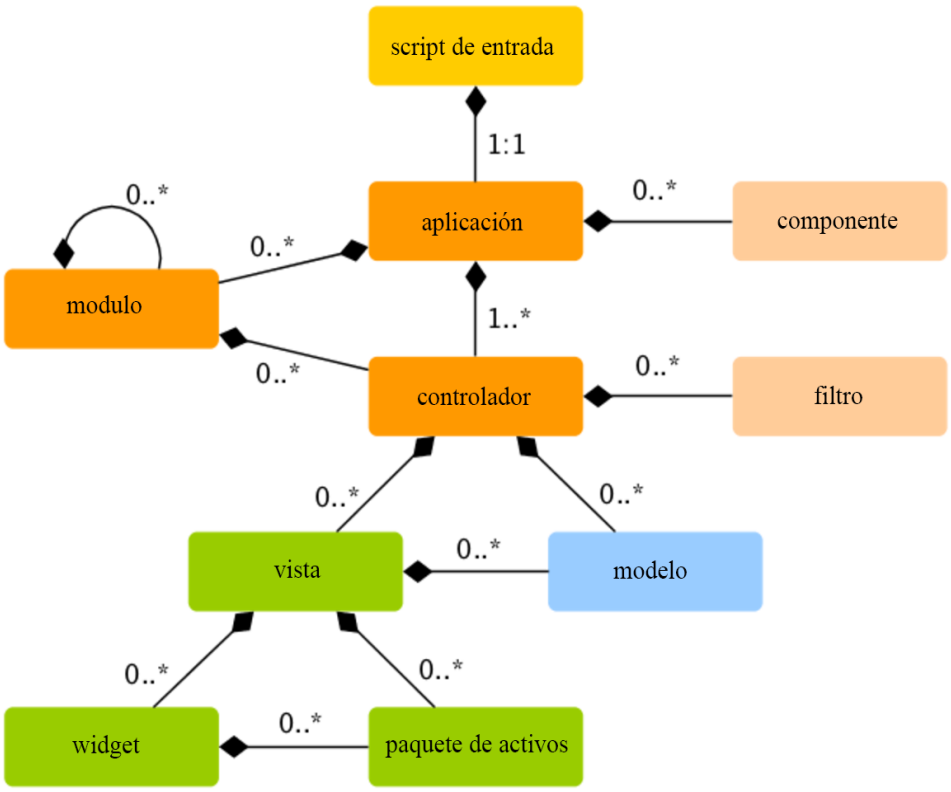


Figura 14 Diagrama de la estructura de Yii

Fuente: Traducido (Yii, 2017).

[11] Framework: Marco de Trabajo

Cada aplicación tiene un script de entrada, toma una petición entrante y crea una instancia de aplicación para manejarla. La aplicación resuelve la solicitud a los elementos del modelo-vista-controlador.

Los widgets se utilizan en las vistas para ayudar a crear elementos de interfaz de usuarios complejos y dinámicos. Figura 15. Muestra como una aplicación maneja una solicitud (Yii, 2017).

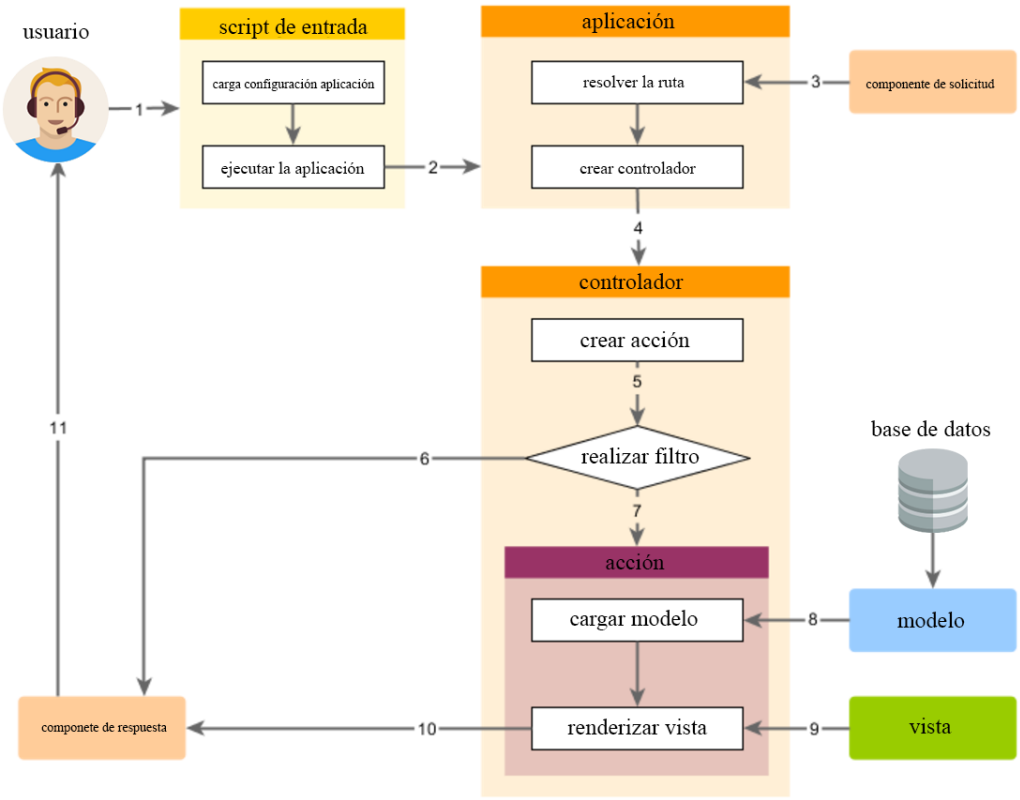


Figura 15 Diagrama de manejo de solicitudes

Fuente: Traducido (Yii, 2017).

- Un usuario crea una petición al script de entrada.
- El script de entrada carga la configuración de la aplicación y crea una instancia de la aplicación para manejar la petición.
- La aplicación resuelve la petición de la ruta solicitada con la ayuda del componente de solicitud.
- La aplicación crea una instancia de controlador para gestionar la solicitud.
- El controlador crea una instancia de acción y realiza los filtros para la acción.

- Si algún filtro falla, la acción se cancela.
- Si todos los filtros pasan, la acción se ejecuta.
- La acción carga un modelo de datos, posiblemente desde una base de datos.
- La acción convierte una vista, proporcionándola con el modelo de datos.
- El resultado renderizado se devuelve al componente de aplicación de respuesta.
- El componente de respuesta envía el resultado renderizado al navegador del usuario.

- **Laravel**

Laravel es un framework de desarrollo para servicios y aplicaciones web. Se caracteriza por ser de código abierto. Su programación se basa en código PHP. Además, permite crear aplicaciones de manera fácil y segura, se enfoca en el patrón de diseño modelo-vista-controlador permite administrar, controlar e interactuar con los datos y los usuarios. Posee un código modular permitiéndole actualizarse y ser escalable (Platzi, 2015).

- **Codeignider**

Codeignider es un marco de desarrollo de aplicaciones y servicios web, es un conjunto de herramientas que proporcionan una gran variedad de bibliotecas. Se caracteriza por tener una interfaz sencilla y poseer una excelente estructura lógica. Permite centrarse creativamente en el proyecto al minimizar la cantidad de código necesario para la creación de tareas. (Codigniter, 2017). Codeignider está bajo la licencia del Instituto Tecnológico de Massachusetts (MIT ó Massachusetts Institute of Technology), utiliza el enfoque Modelo-Vista-Controlador.

- **WordPress**

WordPress es un CMS ó sistema gestor de contenidos el cual se enfoca en la creación de sitios web, contiene una alta gama de características, es fácil y atractivo al uso. WordPress es distribuido conforme a la licencia estándar y para su funcionamiento necesita poseer un servidor que soporte PHP y MySQL (WordPress, 2017).

WordPress fue diseñado para ser instalado en un servidor web. Cuenta con algunas de las siguientes características:

- Núcleo Flexible
- Puede elegir tener el conjunto de archivos de WordPress, que es la trastienda de lo que muestra su bitácora, en el mismo directorio de éste o en un directorio diferente.
- Administración de usuarios
- Cada usuario registrado puede definir un perfil, con detalles como su dirección de correo electrónico, cuentas de mensajería instantánea, etc., si ellos desean hacerlo público.
- Fácil instalación y actualización.
- Generación dinámica de páginas web.

2.10 Metodología de desarrollo de software

2.10.1 Metodología de desarrollo SCRUM

Es una metodología ágil para el desarrollo de software. Se enfoca en entregables y actividades. Parte del trabajo en equipo y acciones cooperativas aplicando mecanismos de control como son inspecciones y revisiones frecuentes. Scrum parte de la definición de los requerimientos iniciales como base fundamental, es excelente para proyectos en los cuales los requisitos cambian, fomentando el seguimiento y control (Peralta, 2003).

El ciclo de vida de Scrum es incremental e iterativo, además se adapta a cualquier tipo de desarrollo donde cada uno de los entregables es muy importante dentro de un proyecto. En esta metodología se presentan los Sprint[12] o iteraciones que tienen una duración de aproximadamente un mes donde el supervisor y la parte administrativa valora el progreso del proyecto. El uso de la metodología Scrum considera cinco fases, estas a su vez están delimitadas por tiempos para no alargar el desarrollo de forma (Figura 16).

[12] Sprint: cada una de las iteraciones que se producen en la metodología SCRUM

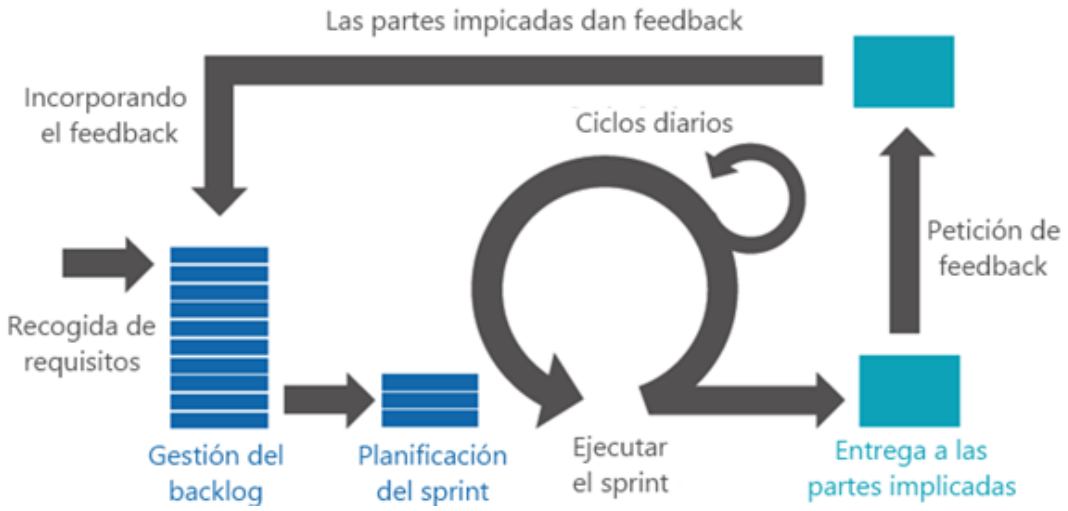


Figura 16 Modelo SCRUM

Fuente: (Scrum con TFS(Team Foundation Service), 2013)

2.10.1.1 Roles

Como en todas las metodologías, Scrum presenta diferentes roles donde todas las responsabilidades y tareas del desarrollo se reparten entre ellos.

- Dueño del producto (Product Owner)

Es el que representa al cliente y los interesados que han solicitado la resolución del problema o el producto en sí. Es quien entrega los requisitos y gestiona su prioridad en el desarrollo de los mismos.

- Líder del proyecto (Scrum Master)

Es el responsable de todo lo relacionado con el proceso de Scrum, eso abarca la metodología y el cumplimiento de la documentación a ser entregada, se asegura que se lleven a cabo todos los procesos y que el equipo cumpla con los entregables.

- Equipo (Team)

Es el grupo de todas las personas involucradas en el desarrollo del producto, no existe el rol de líder entre ellos ya que siempre organizar sus tareas y responsabilidades, son los encargados de llevar el análisis, diseño, desarrollo y pruebas.

2.10.1.2 Fases de SCRUM

Son varios las fases que componen a esta metodología, entre ellos se tiene:

- Recogida de requisitos
- Gestión del backlog (Lista de tareas)
- Planificación de la iteración
- Ejecución de la iteración
- Inspección y adaptación

Recogida de requisitos

Es la primera fase de la metodología SCRUM, se centra en la especificación de los requerimientos tantos funcionales como no funcionales para concretar el desarrollo. En esta fase se presentan las denominadas historias de usuario, las cuales definen las peticiones del Scrum master y el Product Owner, las cuales han sido tomadas en cada encuentro o reunión.

Éstas historias de usuario no son más que requisitos cortos, concisos y de fácil redacción que especifican la naturaleza de un requerimiento en lenguaje natural para luego ser priorizadas dependiendo su necesidad.

Gestión del Backlog

Se procede a la jerarquización de las tareas, cada una de ellas tendrá un tiempo de desarrollo y una prioridad en el mismo. En esta fase de descartan los requerimientos innecesarios y solo se toman los que tengan mayor relevancia, se realiza un listado y se les da un código identificador.

Planificación de la iteración (Sprint)

Cada sprint es conocido como una unidad de trabajo o iteración, en el cual se congregan varias tareas delimitadas por un periodo de tiempo para su desarrollo. En la planificación del sprint se presentan dos etapas fundamentales:

- **Selección de requisitos**

En esta parte interactúa tanto el equipo como el cliente, el equipo se encarga de realizar preguntas al cliente, mientras éste se encarga de solventarlas. Se realiza la selección de los requerimientos más necesarios para proceder con su desarrollo.

- **Planificación de la iteración**

Se crea una lista de los requerimientos a desarrollar, se realiza la estimación del esfuerzo y del tiempo de desarrollo. Se le asigna a cada miembro del equipo de desarrollo una actividad en concreto a ser elaborada.

Ejecución de la iteración

Cada uno de los sprint o unidades de trabajo tiene un tiempo fijo de cuatro semanas máximo, donde el principal objetivo es dar solución a cada uno de los requerimientos propuestos para dicha iteración. Se debe realizar el análisis de cada requerimiento indicando como fue solucionado y llevado a cabo por el equipo de desarrollo.

Inspección y adaptación

Última etapa del proceso de Scrum, se reúnen todos los interesados y el equipo del proyecto, luego de esto se pretende realizar un debate sobre los cambios realizados y los requerimientos desarrollados. Esto se presenta en dos sub-fases:

- **Revisión de la iteración**

El equipo presenta a los interesados todos y cada uno de los requerimientos completados hasta la fecha, de manera que se incrementen y se adapten a todo en proyecto. Luego el cliente revisa los avances y da su aprobación o rechazo para posterior mejora.

- **Retrospectiva de la iteración**

El equipo de trabajo se reúne para calificar su forma de trabajo y verificar como han sido cumplidas las responsabilidades de cada miembro, así se procede a solventar problemas de desarrollo y de cooperación entre el equipo.

CAPITULO III
ANÁLISIS, DISEÑO E IMPLANTACIÓN DE LA SOLUCIÓN

Las aplicaciones con procesamiento en paralelo demandan un alto número de recursos de hardware y software. Sin embargo, además de cumplir con las especificaciones técnicas, también entra en juego el tiempo de traspaso de mensajes y la velocidad de procesamiento, los cuales dependen directamente del equipo informático y las conexiones de las redes de datos.

La universidad de las Fuerzas Armadas ESPE cuenta con un clúster de alto rendimiento HPC, que no posee ninguna aplicación, lo cual inhabilita el uso a los investigadores. Se plantea la instalación de aplicaciones en paralelo para ayudar a los proyectos de investigación de la universidad.

Se creó un sistema para la gestión de usuarios para administrar el acceso al HPC, se creó un micrositio informativo sobre utilidades y recursos disponibles utilizando la metodología SCRUM la cual se enfoca directamente a los entregables y los procesos, en conjunto con los implicados del proyecto realizando revisiones periódicas hasta completar el desarrollo.

Se define cuáles son los implicados en el desarrollo, según el capítulo dos sobre los roles del desarrollo Scrum, son todos los interesados en el proyecto, son juez y parte para el cumplimiento del mismo (Tabla 1).

Tabla 1

Equipo de trabajo

Rol	Encargado	Área
Product Owner	Ingeniero Santiago Salvador	UIICS - Encargado del HPC
SCRUM Master	Ingeniero Diego Marcillo	Tutor del trabajo de titulación
Team	Freddy Vera	Desarrollador Documentador
	Luis Rocha	Desarrollador Tester Documentador

2.11 Micrositio de HPC

El desarrollo del Micrositio es el primer módulo de la primera iteración según la metodología Scrum descrita en el capítulo dos, el micrositio se enfoca en mostrar un apartado informativo sobre el HPC y su uso, entre ellos información de hardware, software, proyectos actuales, personal responsable, entre otros.

2.11.1 Planificación

Para la creación del micrositio se elaboró la planificación de las fases, ayudando a estimar tiempos de desarrollo y evitar inconvenientes. Las fases dependen de la disponibilidad de los recursos y de los interesados:

- **Análisis y Especificación de los Requisitos**

  Especificación de requisitos.
  Definición del micrositio.
  Elección del Gestor de contenido.
  Definir roles de usuarios.

- **Diseño**

  Crear un mapa de navegación.
  Diseñar el micrositio.
  Creación de los contenidos.
  Creación de los perfiles de usuarios.

- **Implantación**

  Elaboración del micrositio.

En la primera reunión con el Product Owner, en colaboración con el Scrum Master se concretó la definición de los requerimientos que se implementaron en el micrositio informativo del HPC.

2.11.2 Análisis y Especificación de los Requisitos

Según Scrum el siguiente paso es la recolección de requerimientos, este proceso se detalla en el capítulo dos, corresponde a la primera fase de la metodología.

2.11.2.1 Especificación de requerimientos

Conforme se concretaron las reuniones con los interesados para el desarrollo se procede a la elicitación de requerimientos (Tabla 2), como se menciona en la primera fase de la metodología Scrum se forma una pila o lista dónde se clasifican y priorizan acorde a su dificultad, tiempo de desarrollo y número de recursos, para luego otorgarles un código de identificación en el desarrollo.

Tabla 2

Especificación de requerimientos del Micrositio

Código	Requerimientos Funcionales
MS01	El micrositio deberá contar con el ingreso y el control de roles para la administración de los usuarios.
MS02	El micrositio deberá contener organización en los contenidos, cada apartado estar en una página web diferente.
MS03	El micrositio deberá ser cómodo y fácil manejo para los usuarios.
MS04	El micrositio deberá usar un gestor de contenido de fácil implementación y uso.
MS05	El gestor utilizado deberá permitir realizar copias de seguridad de la información por posibles fallas y caídas del sistema.
MS06	El micrositio deberá contar con HTML para la especificación de contenido y de CSS para la especificación del formato.
MS07	El micrositio deberá mostrar información de la infraestructura tanto física como lógica del HPC.
MS08	El micrositio deberá mantener los colores que muestren la identidad de la ESPE.
MS09	El micrositio deberá contar con un apartado de soporte o ayuda al usuario.
MS10	El micrositio deberá tener un apartado de noticias actuales.
MS11	El micrositio deberá mostrar la información del estado de los proyectos actuales que se están ejecutando en el HPC.
MS12	El micrositio deberá contar con un apartado de acceso permitiendo los usuarios entrar al sistema de gestión del HPC Rumiñahui.
Código	**Requerimientos no Funcionales**

Continúa

NMS01	El micrositio deberá ser compatible para cualquier navegador web.
NMS02	El micrositio deberá estar disponible siempre.

Nota: Código utilizado es <<MS>> Requisito Funcional Micrositio; <<NMS>> Requisito No Funcional Micrositio.

2.11.2.2 Definición de los requerimientos del Micrositio

Al terminar de concretar los requerimientos funcionales y no funcionales del micrositio, se procede a la elaboración. A continuación, se detallan los requisitos en función del tiempo de desarrollo:

MS01: Los usuarios podrán acceder al sistema, sólo los cuales hayan sido registrados por el administrador, cada uno tendrá su perfil personalizado y su contraseña de acceso único que deberán solicitar por medio de un formulario, una vez autorizado podrán acceder al panel de administración del micrositio.

MS02: El Micrositio contará con varias ventanas y pestañas organizadas por contenidos; al ser solo informativo aceptará cualquier formato en texto e imágenes

MS03: El Micrositio es diseñado para acceder y utilizar de manera fácil, dependiendo del rol del usuario, ellos tendrán acceso a ciertas características que estén disponibles, contará con colores y fuentes agradables.

MS04: A pedido del Product Owner el Ingeniero Santiago Salvador, el gestor utilizado para la elaboración del micrositio es WordPress v4.5.2, cuenta con gestión de usuarios y roles.

MS05: WordPress se caracteriza por ser un gestor que maneja sus propios backups[13], gracias a esto puede crear copias de seguridad del Micrositio, permitiendo regresar a estados anteriores en presencia de alguna caída o fallo en el sistema

.

[13] Backups: Respaldos de seguridad.

MS06: Al utilizar un gestor de contenido se crean automáticamente las páginas web utilizando HTML, para el formato existen variedades de plugins los cuales fueron implementados, también se utiliza CCS para mantener al micrositio en orden y con una excelente presentación.

MS07: El ingeniero Santiago Salvador entregó información referente al HPC Rumiñahui la misma que fue organizada en cada página del micrositio.

MS08: Este micrositio estará bajo el uso de la Universidad de las Fuerza Armadas ESPE, contará con los colores verde, rojo y negro para asegurar su identidad de pertenecía a dicha institución.

MS09: El micrositio mostrará un apartado para soporte a los usuarios, contará con la información necesaria en caso de ayuda o tutorías, además se presenta una lista de todos los contactos encargados de la administración del micrositio.

MS10: Para las noticias actuales, el micrositio contará con plugins para la actualización de noticias con información referente al uso del HPC, el usuario encargado solo debe actualizar las fuentes de las noticias y estas se mostrarán en una ventana y en pequeñas secciones ubicadas en el micrositio.

MS11: El Micrositio se mantendrá actualizado con la información más reciente del HPC para ello se recibió un documento en el formato Word con los proyectos y programas actuales gestionados por el Ingeniero Santiago Salvador, los mismo que serán instalados en el HPC Rumiñahui.

MS12: Se realiza la creación de una página en el Micrositio el cual cuenta con los enlaces que podrán re direccionar a los usuarios al sistema de gestión del HPC Rumiñahui, el usuario simplemente tiene que ingresar sus credenciales para acceder.

NMS01: Al utilizar un gestor de contenido, se garantiza que el micrositio sea accesible desde cualquier dispositivo y navegador web.

MNS02: A pedido del ingeniero Santiago Salvador la página es gestionada por el Cpanel de la institución con lo cual no se garantiza disponibilidad ya que aquellos servidores están fuera de la administración de los desarrolladores. El gestor escogido garantiza que el micrositio sea accesible y escalable.

2.11.2.3 Analizar recursos disponibles.

El ambiente de pruebas entregado por el Ing. Santiago Salvador cuenta con un Cpanel 64.0.24. Además de contener con una lista de softwares incluidos (Tabla 3), de los cuales se tiene acceso total a cada una de sus características.

Tabla 3

Recursos Software Disponibles

Nombre de recurso	Versión	Descripción
MySQL	5.1	Gestor de base de datos.
PHP	5.6.30	Ambiente de programación.
phpMyAdmin	4.6.6	Administrador del ambiente de programación.
Apache	2.2	Servidor web de código abierto, permite la creación de páginas web

Los softwares descritos han sido probados, testeados y actualizados para verificar su funcionalidad y compatibilidad con los gestores de contenido y frameworks utilizados en el desarrollo.

2.11.2.4 Elección del Gestor de contenido.

Se concreta por petición del Product Owner la creación del micrositio con el Gestor de contenidos WordPress por ser robusto, seguro y por su corta curva de aprendizaje, desarrollo y acople a cualquier proyecto.

2.11.3 Diseño

El micrositio está distribuido como se describe en la (Figura 17) donde se especifica el contenido de cada página, se utiliza el diseño de distribución en cascada para mostrar la navegación del micrositio con y sus interrelaciones.

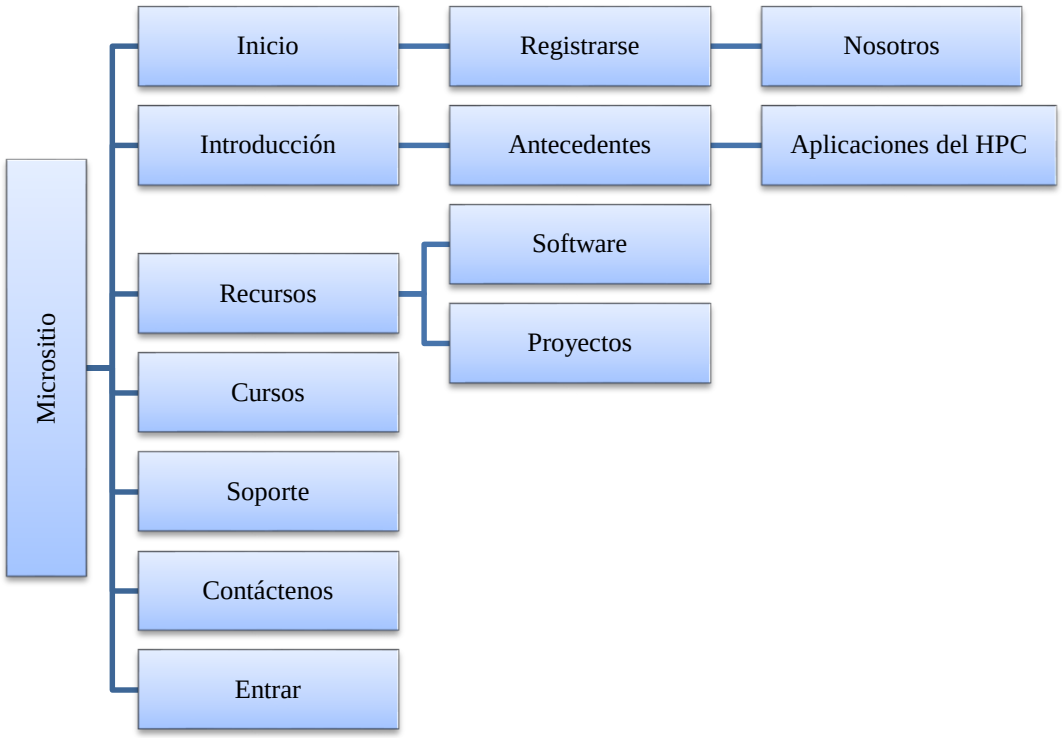


Figura 17 Mapa de navegación del Micrositio

2.12 Sistema de Gestión de Investigadores para el uso del HPC

El desarrollo del sistema se presenta en la iteración 2 aplicando la metodología Scrum, se basa en un modelo Workflow, el mismo que permitirá hacer control de las peticiones y respuestas del usuario para el uso del HPC.

2.12.1 Planificación

Se describen las tareas a realizar durante el proceso de desarrollo del sistema. El proyecto se divide como se muestra a continuación:

- **Análisis y Especificación de los Requisitos**
  Estudio previo y entrevistas con el cliente.
  Especificación de los requerimientos.
  Definición del sistema de gestión de usuarios.
  Roles de sistema.
  Casos de Uso.

- **Adaptación a nuevos entornos y aprendizaje**
  Aprendizaje Herramienta de Desarrollo Yii 2.0.
  Aprendizaje del Framework Modelo – Vista – Controlador.
  Aprendizaje de la tecnología Composer.
- **Diseño**
  Obtención de la Arquitectura de la aplicación.
  Definición de la base de datos.
  Diagrama de clases.
  Diagrama de secuencia del sistema.
  Diagrama de componentes.

Concretado la planificación para la creación del sistema se definen los requisitos necesarios para que pueda funcionar.

2.12.2 Análisis y Especificación de los requerimientos

Los requerimientos son tomados y enlistados (Tabla 4) como se menciona en la metodología Scrum, de esta forma, se proceden a ser priorizados para posteriormente ser desarrollados.

Tabla 4

Especificación de requerimientos del Sistema

Código	Requisitos Funcionales
SG01	El sistema debe permitir el ingreso de nuevos usuarios.
SG02	El sistema debe manejar tres roles específicos: Administrador, director, Investigador.
SG03	El sistema mostrará reportes de las solicitudes y las respuestas.
SG04	El sistema enviará al correo la confirmación del usuario para su ingreso.
SG05	El sistema permitirá visualizar el estado de las solicitudes enviadas por el investigador.
SG06	El sistema deberá mostrar un calendario de las horas y fechas de uso.
SG07	El sistema tendrá un apartado para la visualización de videos tutoriales.
SG08	El sistema enviará correos a los usuarios para informar el estado de sus solicitudes.

Continúa

SG09	El sistema deberá permitir ingresar nuevos programas que no se encuentren instalados, con el link de la página oficial del programa.
SG10	El administrador podrá dar de baja a los usuarios, así como aceptar o declinar solicitudes.
SG11	El usuario podrá revisar su perfil personal.
SG12	El administrador podrá ingresar nuevos departamentos.
SG13	El sistema debe tener administración de proyectos.
Código	**Requisitos no funcionales**
NSG01	El sistema podrá ser visualizado en cualquier buscador web.
NSG02	El sistema deberá contar con disponibilidad, accesibilidad, escalabilidad.

Nota: Código utilizado es <<SG>> Requisito Funcional Sistema de Gestión; <<NSG>> Requisito No Funcional Sistema de Gestión

2.12.2.1 Definición del sistema de gestión de usuarios

Concretados los requerimientos del cliente, como se menciona en la metodología Scrum se procede a crear el sistema. Los requerimientos son detallados a continuación:

SG01: El sistema permitirá el registro de usuarios completando un formulario de datos personales, posterior se enviará un correo electrónico a la dirección que haya especificado. El administrador valora su petición y le concede el acceso con el rol de acuerdo con el tipo de usuario.

SG02: El sistema va a contar con tres roles como se describen en la Tabla 5.

Tabla 5

Roles del sistema de gestión

Rol	Descripción
Administrador	• Control total de los módulos del sistema. • Asignación de roles. • Aceptar o rechazar solicitudes revisados por el director. • Visualizar estado de solicitudes.
Director	• Aceptar o rechazar solicitudes del investigados. • Visualizar estado de solicitudes.

Continúa

Investigador	• Crear solicitudes para uso del HPC. • Visualizar estado de solicitudes.

SG03: El sistema permitirá a los usuarios, generar un reporte de la solicitud realizada, mostrando el resumen que contendrá el estado, los softwares utilizados y los recursos requeridos.

SG04: El sistema permitirá al usuario evidenciar su registro mediante un correo electrónico de confirmación.

SG05: El sistema permitirá evidenciar el estado de las solicitudes del Investigador con diferentes colores:

- Rojo muestra que la solicitud fue rechazada.
- Verde muestra que la solicitud fue aprobada.
- Azul muestra que la solicitud ha sido enviada.
- Sin color muestra que el proceso de la solicitud ha terminado.

SG06: El sistema permitirá a los usuarios ver en un calendario el horario asignado por proyecto con las horas diarias y totales que abarca toda la ejecución de sus modelos o simulaciones.

SG07: El sistema permitirá al usuario acceder a videos tutoriales sobre el uso y el empleo de herramientas necesarias para conectarse y usar el HPC Rumiñahui.

SG08: El sistema permitirá al usuario informarse sobre el estado actual de su solicitud mediante una notificación vía correo electrónico.

SG09: El sistema contará con un módulo de administración de programas disponibles en el HPC Rumiñahui. Para ello se deberá llenar un formulario de información del programa, incluyendo el link de descarga, su versionamiento, licencia, etc. Los usuarios con el rol de investigadores podrán requerir los programas en las solicitudes y serán guardadas con el estado de “no instalado”.

SG10: El administrador del sistema tendrá control total, como parte de sus funciones podrá aceptar o rechazar solicitudes de los investigadores y en base a ello enviar una respuesta a dicha solicitud. También puede eliminar usuarios del sistema.

SG11: Cada usuario tendrá un perfil propio que podrá ser personalizado a su conveniencia dependiendo del rol asignado,

SG12: El sistema contará con el módulo de administración de los departamentos, el administrador será el único que pueda acceder a la gestión de dicho módulo, los demás roles solo podrán visualizarlos y utilizarlos en la creación de las solicitudes.

SG13: El sistema contará con la administración de los proyectos que los investigadores hayan creado una vez sean aceptados por el director y el administrador.

NSG01: El sistema será realizado en HTML5, CSS3 y Bootstrap, se garantiza que el sistema será visualizado desde cualquier navegador web.

NSG02: El sistema estará alojado en el servidor propio de la universidad, por tal motivo no se garantiza la disponibilidad, con relación a la accesibilidad los usuarios podrán acceder desde cualquier dispositivo y también será escalable si se presentan requerimientos futuros.

- **Diagramas de casos de uso**

El uso de diagramas UML permite a los lectores entender mejor la solución al requerimiento planteado, a pesar de que la metodología Scrum no contribuye al uso de estas herramientas se las utilizan para mejorar la documentación apoyando a la ingeniería de requerimientos.

Figura 18 De forma general cada usuario tiene acceso a registrarse para acceder al sistema con su rol asignado previamente por el administrador.

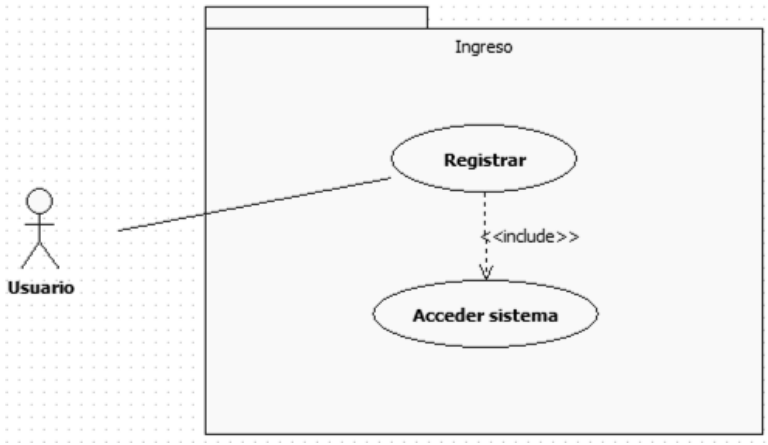


Figura 18 Ingreso de usuarios al sistema

El usuario registrado tendrá acceso a su perfil donde dependiendo del rol, sea este administrador, director o investigador podrá hacer uso de las diferentes características del sistema. Estos roles cumplen funciones en específico como se observa en la (Figura 19).

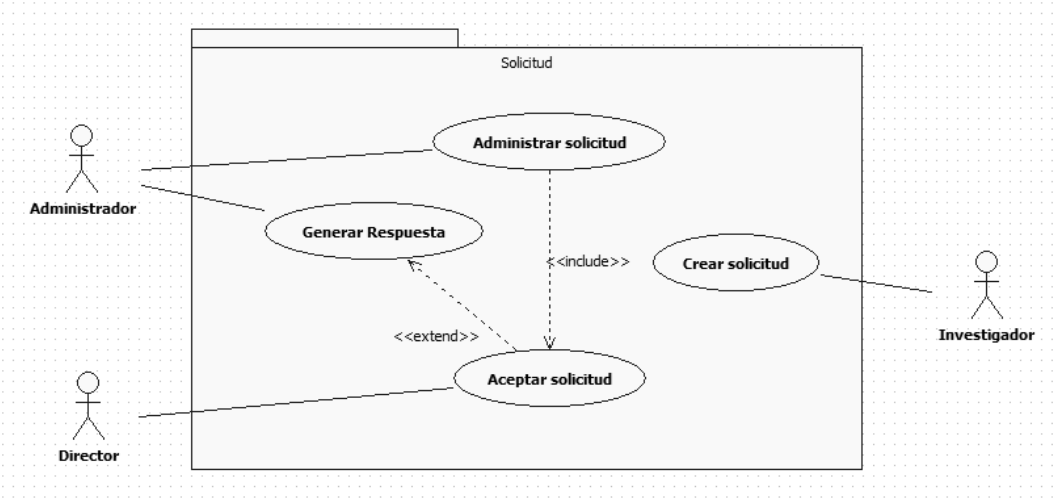


Figura 19 Solicitudes y respuestas

Terminado el proceso de registro e ingreso al sistema cada usuario tiene su rol asignado, el investigador es el rol con menos privilegios, éste se encarga de realizar las solicitudes dando inicio al flujo de trabajo. El director es el segundo rol al mando, su principal función es revisar las solicitudes de los investigadores para ser rechazado

o aceptada, luego pasa al administrador quien da la aprobación final y retorna una respuesta al investigador.

- **Especificación de casos de uso**

Los casos de uso descritos en la (Figura 17) y la (Figura 18) son especificados en las siguientes tablas:

Tabla 6

Especificación caso de uso - Registrar

ID	CU-01
Descripción	Permite registrar a los usuarios al sistema.
Precondición	Ninguna
Postcondición	Autorizar el acceso al sistema aceptando el mensaje de confirmación desde el correo electrónico.
Flujo normal	1. El usuario ingresa sus datos personales. 2. Se registran los datos ingresados en el sistema. 3. Se envía un correo electrónico para confirmar el registro y activar el acceso.
Flujo alterno	Si el usuario ya está registrado el sistema proporciona la opción de Iniciar sesión.
Excepciones	1. Si se intenta registrar un usuario que ya exista, el sistema le informará. 2. Si el usuario olvida ingresar algún campo, el sistema le notificará en la pantalla.
Notas	Se requieren los siguientes datos para que sean ingresados: • Cédula • Nombres y Apellidos • Email • Teléfono • Usuario • Institución • Fecha

Tabla 7

Descripción caso de uso - Acceder al sistema

ID	CU-02
Descripción	Permite a los usuarios acceder al sistema con el rol que se haya sido asignado.
Precondición	Estar registrado en el sistema.
Postcondición	Ninguna
Flujo normal	
	1. El usuario ingresa sus credenciales (usuario, contraseña). 2. Ingresa al sistema.
Flujo alterno	
	El usuario puede cerrar sesión en cualquier momento.
Excepciones	Si al ingresar sus credenciales una o ambas son incorrectas se le informará en pantalla.
Nota	

Tabla 8

Descripción caso de uso - Crear solicitud

ID	CU-03
Descripción	Permite a los usuarios crear solicitudes para hacer uso del HPC.
Precondición	Haber ingresado al sistema.
Postcondición	Ninguna
Flujo normal	
	1. El usuario debe llenar un formulario que se despliega en la pantalla. 2. El usuario envía el formulario para su revisión por parte del director y el administrador del sistema. 3. El usuario recibe la petición junto con la respuesta del administrador otorgándole los permisos necesarios para que haga uso del HPC.
Flujo alterno	
	Ninguna
Excepciones	Ninguna
Nota	La solicitud debe ser llenada por su totalidad para que pueda ser revisada, dado el caso ésta puede ser modificada por el director o el administrador realizando ajustes acordes a las necesidades del investigador.

Tabla 9

Descripción caso de uso - Administrar solicitud

ID	CU-04
Descripción	Permite al administrador crear, modificar o eliminar una solicitud creada por un investigador.
Precondición	Estar registrado en el sistema con el rol de administrador.
Postcondición	Ninguna
Flujo normal	
	1. El usuario puede realizar cualquier acción (crear, modificar, eliminar) sobre una solicitud, en el caso que desee modificar, tiene la opción de crear la respuesta para el investigador. 2. Realizada cualquier acción se les informará a los implicados mediante el correo electrónico.
Flujo alterno	
	El usuario puede cerrar sesión en cualquier momento. En caso de que una solicitud sea eliminada o cancelada, ésta no se borrará del sistema solo cambiará su estado.
Excepciones	
Nota	Toda acción realizada por el administrador será reflejada en el estado de la solicitud, de esta forma los demás usuarios podrán verla.

Tabla 10

Descripción caso de uso - Aceptar solicitud

ID	CU-05
Descripción	Permite al administrador y al director aceptar o rechazar una solicitud realizada por un investigador.
Precondición	Estar registrado en el sistema como director o administrador
Postcondición	Ninguna
Flujo normal	
	1. El usuario revisa las solicitudes recibidas y activas. 2. El usuario valora estas solicitudes y puede cambiar a su estado como rechazado.
Flujo alterno	
	Si el usuario cambia al estado rechazada la solicitud se le notificará al investigador. En caso que la solicitud sea aprobada, será notificado con un correo electrónico y se le enviará la respuesta por parte del administrador.
Excepciones	
Nota	

Tabla 11

Descripción caso de uso - Generar respuesta

ID	CU-06
Descripción	Permite al administrador generar una respuesta en función a la solicitud creada por el investigador.
Precondición	Estar registrado en el sistema como administrador.
Postcondición	Ninguna
Flujo normal	
	1. El usuario ingresa sus credenciales (usuario, contraseña). 2. Ingresa como administrador. 3. Selecciona una de las solicitudes que desee. 4. Selecciona editar y puede empezar a completar la respuesta. 5. El usuario guarda la respuesta y se envía al investigador.
Flujo alterno	
	El usuario puede cerrar sesión en cualquier momento. El usuario puede realizar la respuesta en cualquier momento.
Excepciones	Ninguna
Nota	Ninguna

2.12.3 Diseño

En esta etapa se definirá el modelo de base de datos de forma general del sistema (Figura 20), el cual se enfoca en ser guía de referencia para la implementación de dichos módulos. Cada uno de estos módulos se realizarán en la segunda iteración.

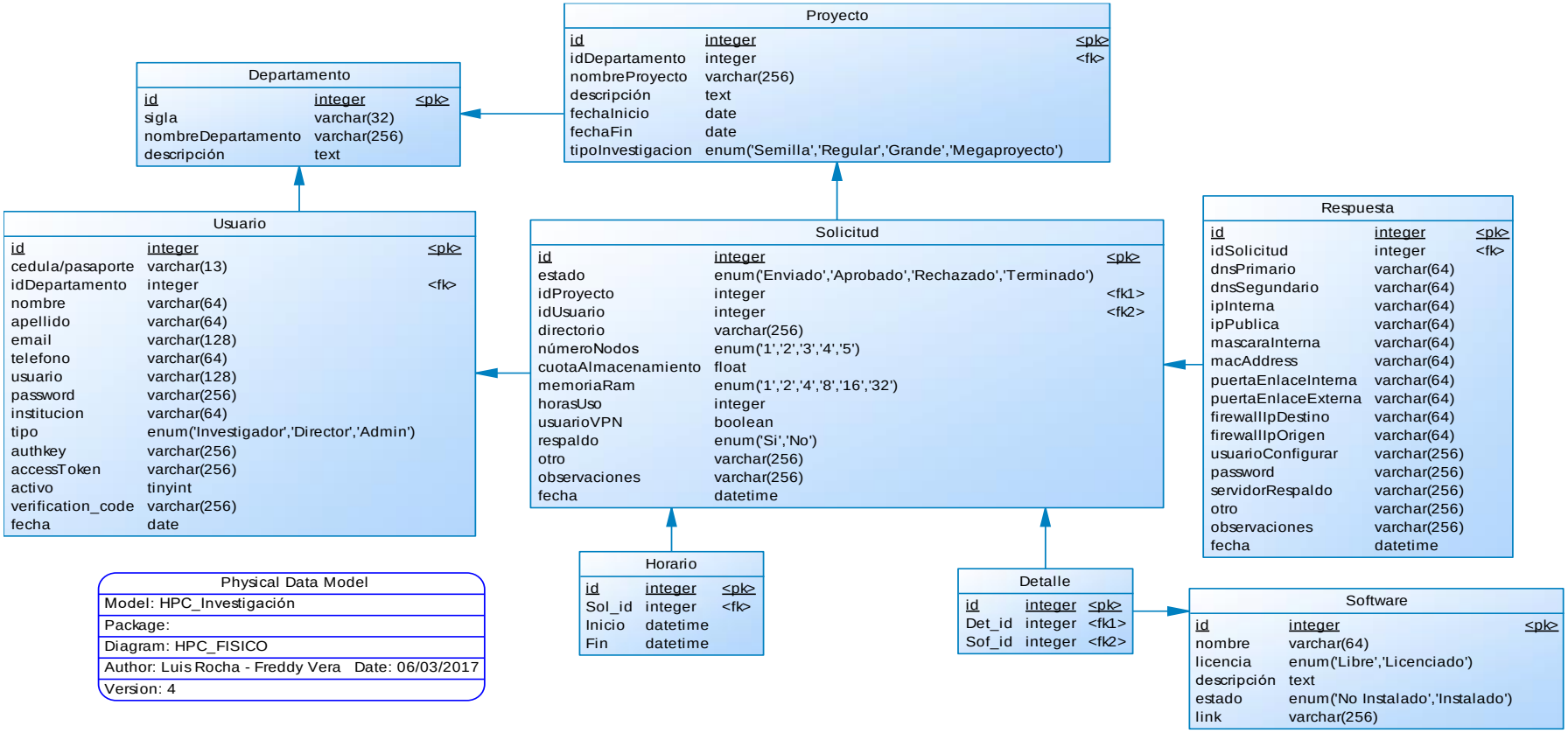


Figura 20 Modelo Físico de la Base de Datos

- **Diagrama de Clases del sistema**

  Para la representación de la funcionalidad del sistema de gestión se elaborará un diagrama de clases, las cuales muestran las relaciones entre ellas y todos métodos a ser implementados en cada módulo (Figura 21).

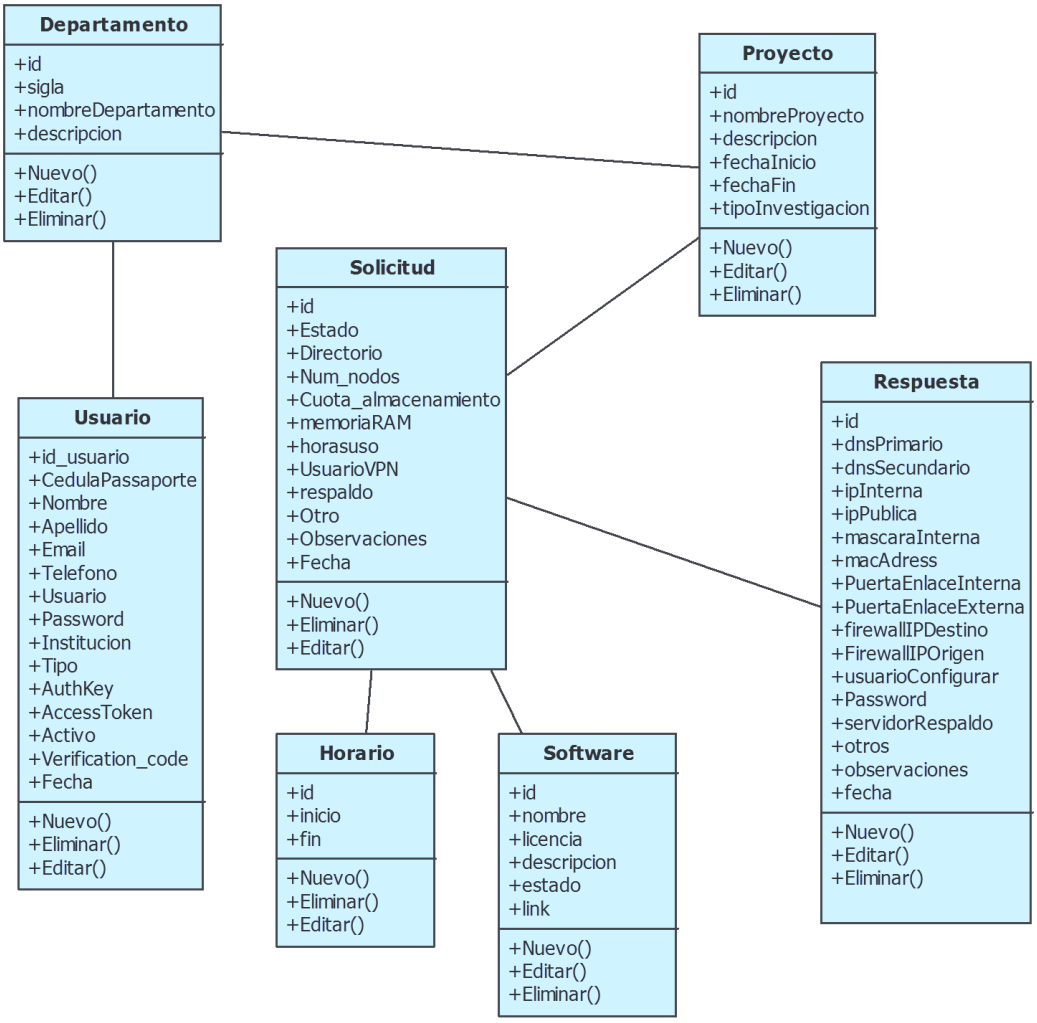


Figura 21 Diagrama de Clases del sistema WEB

El diagrama presenta siete clases las cuales permiten un correcto flujo de trabajo para el sistema web. Cada una de ellas con sus métodos y variables para su control por el administrador.

- **Diagrama de Componentes del sistema**

El sistema web de gestión de Investigadores será instalado sobre la plataforma Cpanel previamente entregada por el Product Owner (Figura 22). El sistema será creado en el localhost y luego será transferido a la plataforma suministrada para su producción.

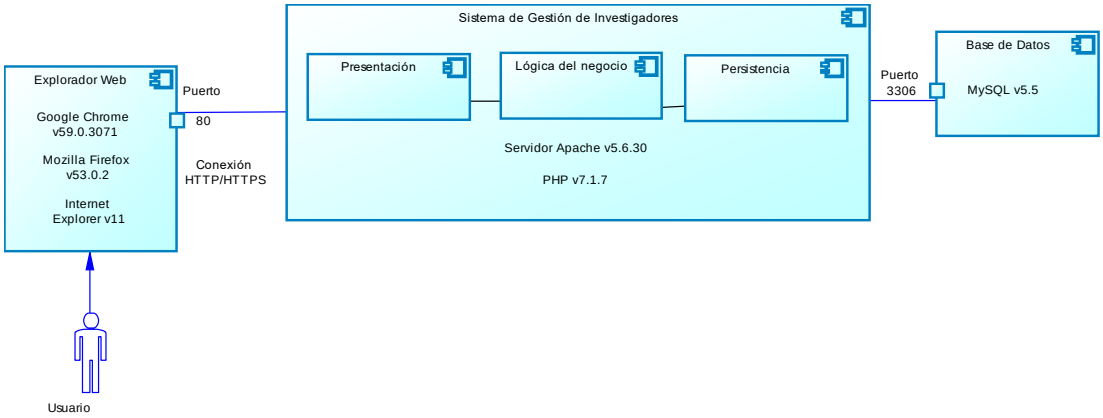


Figura 22 Diagrama de Componentes

En la parte central se encuentra el modelo-vista-controlador el cual es el patrón de diseño utilizado para la creación del sistema, además se presenta el servidor apache y el lenguaje utilizado en este caso PHP.

En la parte derecha se encuentra el gestor de base de datos utilizado, en este caso es MySQL y en la parte izquierda los diferentes navegadores de internet con los cuales el usuario tendrá acceso al sistema. También se muestran los diferentes protocolos y puertos utilizados.

- **Diagrama de secuencia del sistema**

El sistema de gestión sigue un flujo de trabajo Figura 23, se presenta la interacción de todos los actores para su funcionamiento, permitiendo que una solicitud pueda ser aprobada o rechazada.

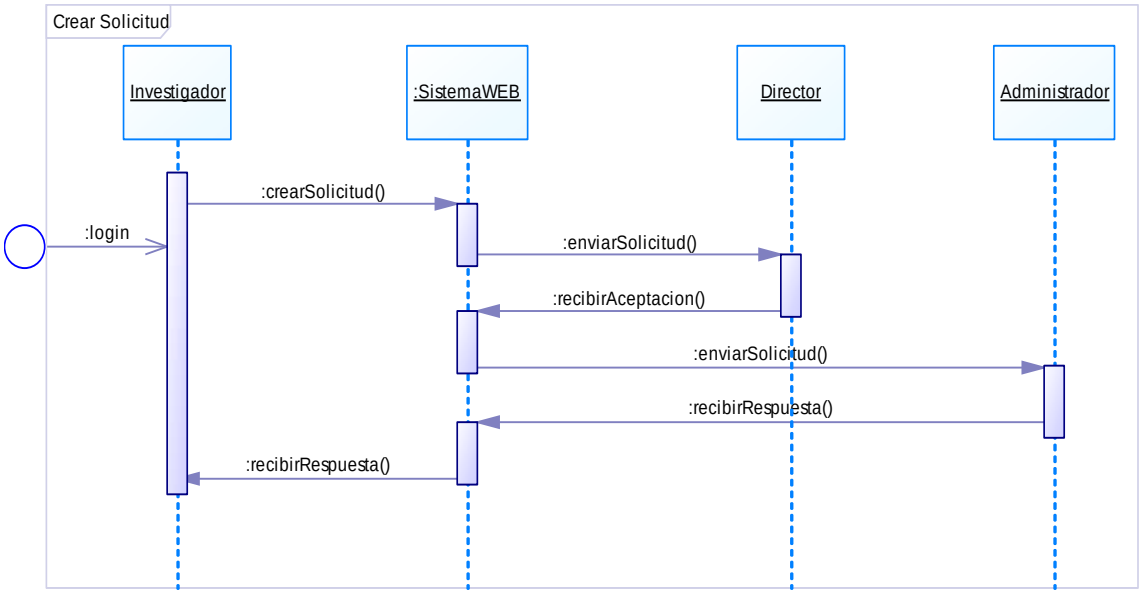


Figura 23 Diagrama de secuencia - Crear solicitud

Se observa la interacción de todos los actores con el sistema como un flujo de trabajo, cuando el investigador crea una solicitud, ésta debe ser revisada y valorada por el director y el administrador, el mismo que dará respuesta a dicha solicitud dependiendo sus requerimientos. La solicitud será enviada al investigador con la especificación de los recursos a los que tendrá acceso y podrá utilizar cuando ingrese al HPC.

2.13 Aplicación en el HPC

El desarrollo de esta tercera iteración consta de la instalación de una aplicación en la arquitectura HPC para que pueda ser probada y testeada por los investigares. La aplicación debe ser apta para un sistema paralelizable y que pueda sea soportada por los actuales recursos que tiene el HPC Rumiñahui.

2.13.1 Planificación

Para la elección de la aplicación se procede a crear la planificación de las fases a considerar en esta iteración:

- **Análisis y Especificación de los Requisitos**

  Investigar aplicaciones paralelizables.

  Enlistar aplicaciones que cumplan con las especificaciones técnicas disponibles en la infraestructura del HPC.

  Seleccionar una aplicación para su instalación.

- **Pruebas**

  Pruebas de concepto de la aplicación

  Comparación entre tiempos de respuesta.

- **Especificación de requerimientos**

Según se propone en el capítulo 2 por la metodología Scrum se procede a realizar la especificación de requerimientos para la instalación de la aplicación en el HPC Rumiñahui Tabla 12.

Tabla 12

Especificación de requerimientos de la aplicación

Código	Requisitos Funcionales
AP01	La aplicación debe ser de código abierto.
AP02	La aplicación debe pertenecer al área de bioinformática.
AP03	La aplicación debe poder ejecutarse en un sistema sin interfaz de usuario (GUI) o en modo terminal.
Código	**Requisitos no funcionales**
NAP01	La aplicación debe ser compatible con el sistema operativo RedHat.

Nota: Código utilizado es <<AP>> Requisito Funcional Aplicación; <<NSG>> Requisito No Funcional Aplicación

2.13.2 Análisis

AP01: Al ser la primera aplicación esta debe ser de código abierto, de esta manera puede ser descargada desde su repositorio, instalada y probada sin ningún tipo de inconveniente. Se enlistaron diferentes softwares disponibles que cumplían dichas características Tabla 13.

Tabla 13

Softwares disponibles

Software	Licencia	Descripción
BEAST	NO	Análisis filogenético
MATLAB	SI	Procesamiento digital de imágenes y señales
QIIME	SI	Análisis de Meta genómica
MZmine	NO	Análisis de metabólica
ChromEvol	NO	Análisis de evolución cromosómica
GARLI	NO	Análisis filogenéticos
RaXML	NO	Análisis filogenéticos
MrBayes	NO	Análisis filogenéticos
IGTP	NO	Análisis filogenéticos a partir de árboles genéticos
RapidMiner	SI	Análisis y minería de datos
R	SI	Lenguaje de programación para análisis estadístico.
PAML	SI	Análisis de selección positiva
PhyML	SI	Análisis filogenéticos
Galaxy	SI	Plataforma para el uso de datos de NGS (entre otros)
Gromacs	SI	Dinámica molecular y análisis estructural
Pymol	SI	visualización molecular
NAMD	SI	Dinámica molecular y análisis estructural
VMD	SI	visualización molecular, análisis estructural, interface NAMD, Gromacs
Grace	SI	Herramienta de gráficas para datos científicos en 2D
Schrodinger	NO	Solución de modelación para biología

De los softwares mencionados se seleccionó BEAST v1.8.0 programa bioinformático para el análisis filogenético, y el que más se ajusta a los requerimientos del Ingeniero Francisco Flores.

AP02: Actualmente en la Universidad de las Fuerzas Armadas ESPE, cuenta con requerimientos de proyectos orientados en el área de bioinformática, por tal motivo es necesario que la aplicación instalada cumpla dicho pedido realizando cálculos y análisis relacionados en el área de biotecnología.

AP03: Actualmente el clúster HPC Rumiñahui cuenta con el Sistema Operativo RedHat para servidores en modo consola, dado esta inconveniente, se deben tomar aplicaciones que solo trabajen en ambientes orientados en modo consola o terminal. Adicionalmente se debe instalar la biblioteca BEAGLE según el pedido del Ingeniero Francisco Flores, dicha herramienta favorece acelerando el procesamiento de datos en forma paralela.

NAP01: Es necesario que la aplicación trabaje bajo GNU/Linux, ya se va a trabajar sobre el sistema operativo RedHat.

2.13.3 Arquitectura del HPC (Situación actual)

A partir del año 2016, la Universidad de Fuerzas Armadas cuenta con una infraestructura de supercomputación dedicada para el uso de la Investigación y Docencia. Cuenta con las siguientes características:

- 730 núcleos
- 58.560 hilos
- 16,52 Teraflops/s (16 billones de operaciones aritméticas)
- 50 terabytes de almacenamiento unificado.

Arquitectura se basa en un sistema operativo abierto (código abierto) con hardware dedicado.

En la situación actual de los equipos que dispone la universidad están divididos en cuatro secciones (Figura 24):

1. Rack servidores de prueba.
2. Rack computo de alto rendimiento (HPC).
3. Rack de procesamiento, almacenamiento y respaldo.
4. Rack de Networking[14].

[14] Networking: integración de sistemas informáticos.

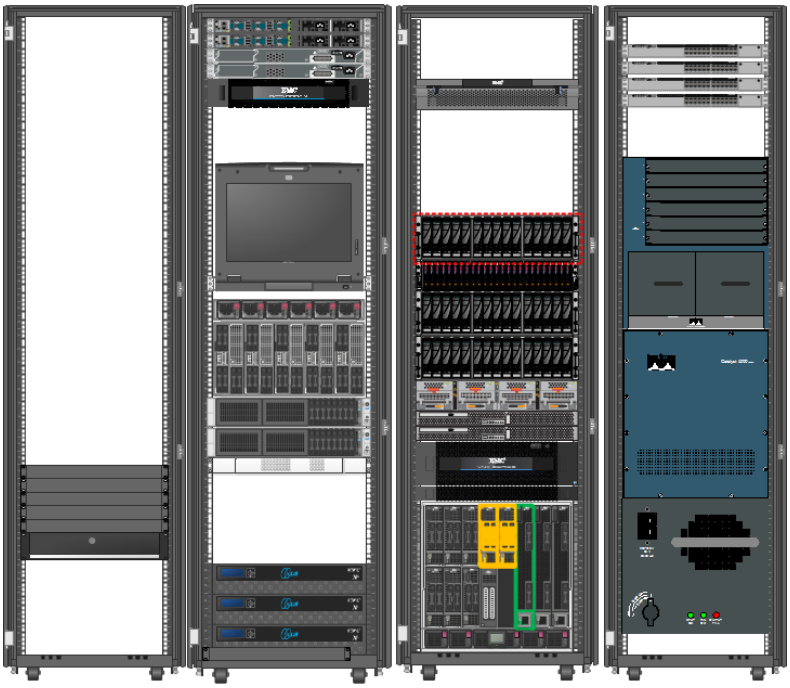


Figura 24 Distribución de Equipos

Fuente: (ESPE-UTICS, 2015).

Para el presente proyecto de investigación nos enfocaremos en el Rack de computo de alto rendimiento (Figura 25), a continuación, se detallan los componentes físicos disponibles.

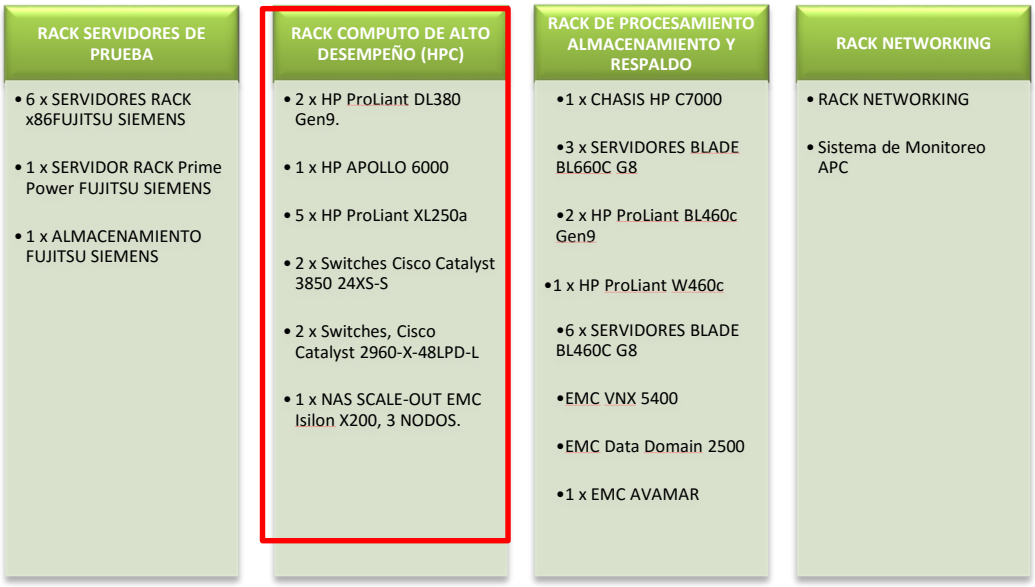


Figura 25 Detalle de componentes físicos

Fuente: (ESPE-UTICS, 2015).

De la distribución de los equipos se tiene dos nodos principales los cuales son los encargados de la administración, los cuales se conectan a cinco nodos de cómputo por medio de la red de clúster compartiendo el sistema de almacenamiento (Figura 26).

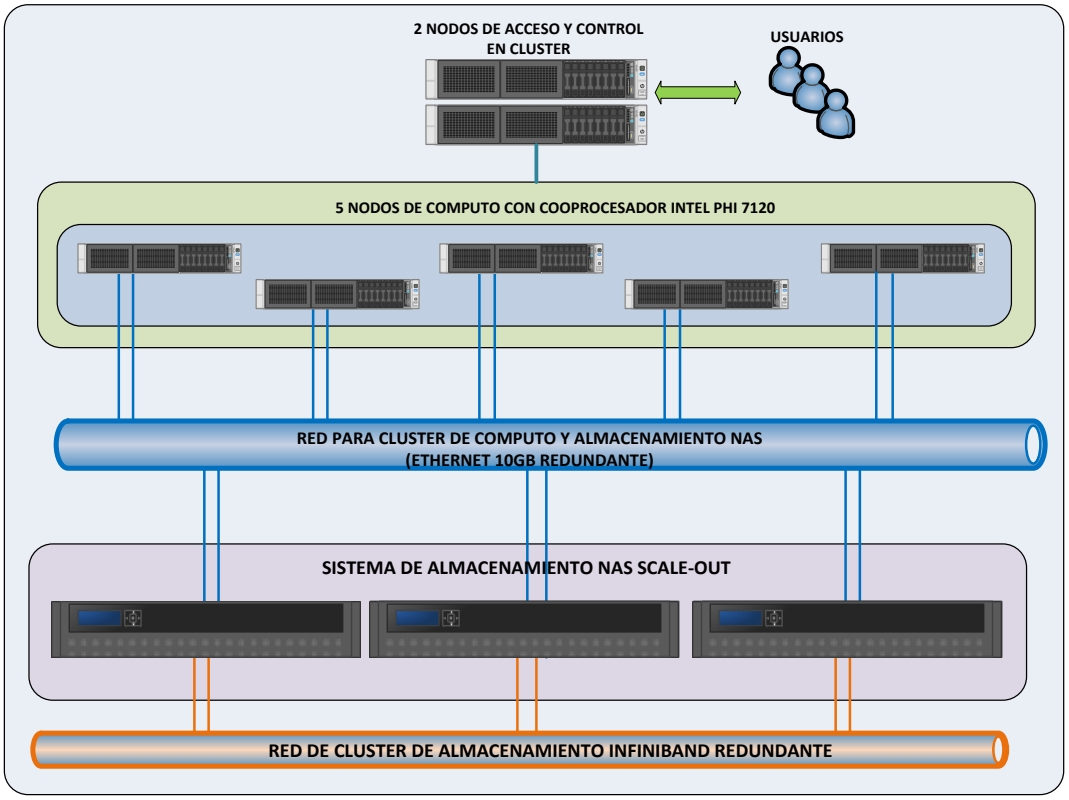


Figura 26 Infraestructura del clúster HPC

Fuente: (ESPE-UTICS, 2015).

De todos los equipos antes mencionados solo se trabaja con los 5 nodos de cómputo con coprocesador Intel y nodos principales de acceso y control del clúster. En la Tabla 14 se muestran las especificaciones técnicas de los nodos mencionados entre ellos su procesador, memoria y aceleración.

Tabla 14

Características técnicas

Equipo	Cantidad	Características
ProLiant DL380 Gen9	2	Procesador: • Intel Xeon E5-2600 • Dos procesadores con 12 núcleos cada uno. • 2,30 GHz Memoria: • 132 GB memoria RAM DDR4

ProLiant XL250a Gen9	5	Procesador:
		• Intel serie E5-2600 v3
		• Dos procesadores con 12 núcleos cada uno.
		• 2.30GHz
		Memoria:
		• 16 slot DDR4
		• 1 TB de memoria RAM
		• 2133 MHz
		Aceleración:
		• 2 coprocesadores Xeon Intel Phi 7120p
		  ○ 61 núcleos
		  ○ 1,33 GHz
		  ○ 16 GB de memoria
		  ○ Arquitectura 64-bits

La infraestructura del HPC también incluye un paquete básico de software para su funcionamiento (Tabla 15).

Tabla 15

Softwares iniciales

Software	Versión	Descripción
Red Hat Server	6.7	Sistema Operativo
Moab Lite	2.1.1	Administrador de clúster
Torque	2.1.1	Administrador de tareas de procesamiento en el clúster
OpenMPI	2.1.1	Biblioteca para el uso de programación paralela
Yum	3.2.20	Paquete de gestión de descargas para Redhat.

Los softwares enunciados son de código abierto y están instalados en los nodos principales y en los nodos de cómputo secundarios que componen el clúster.

2.13.4 Arquitectura del HPC (Situación propuesta)

Según el estudio realizado y teniendo instalada las aplicaciones en el HPC Rumiñahui, se definen los métodos de acceso y el diseño del diagrama lógico (Figura 27).

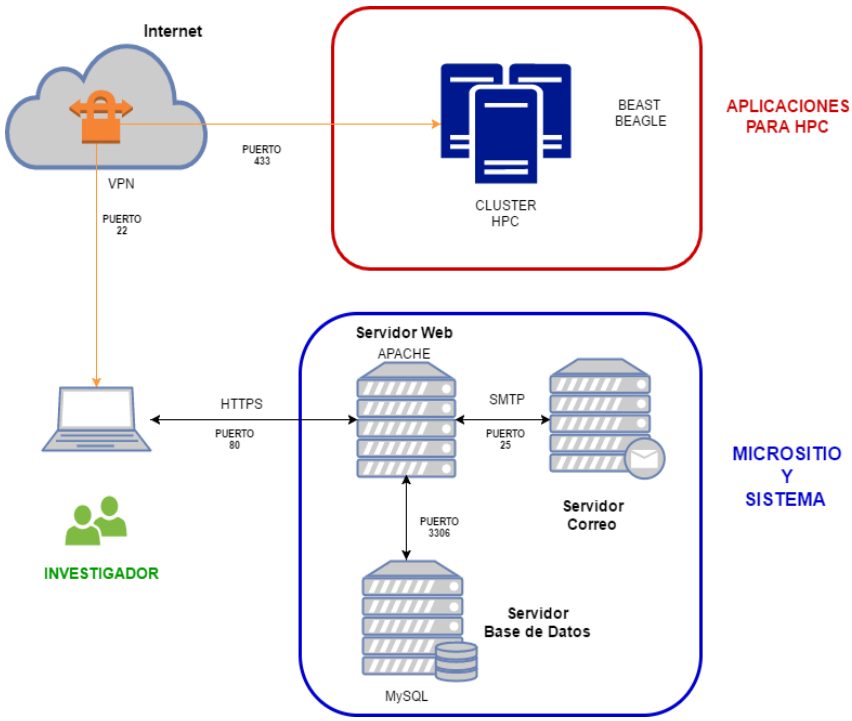


Figura 27 Diagrama propuesto

La propuesta desarrollada se divide en tres partes principales:

La primera parte en color rojo cuenta con la infraestructura HPC, es decir el rack de cómputo de alto rendimiento, dónde se encuentran instaladas las aplicaciones, bibliotecas y complementos necesarias para su funcionamiento y el uso de los investigadores (Tabla 16).

Tabla 16

Aplicaciones instaladas

Software	Versión	Arquitectura (bits)
BEAST	1.8.0	x86, x64
Beagle	2.1	x86, x64
JAVA	1.8	x86, x64
JAVA (SE) Runtime	1.8	x86, x64
Compilador GCC	4.4.7	x86, x64

Continúa

Autoconf	2.64	x86, x64
Intel SDK for OpenCL	6.0	x86, x64
Opencl Runtime	14.1	x86, x64

Todos los softwares mencionados son instalados en el HPC para su funcionamiento, la mayoría de ellos son bibliotecas y programas que son complementos de otros.

La segunda en color azul se enfoca en el micrositio y el sistema de gestión, éstos serán controlados por un servidor propio y administrado por la unidad de Tecnologías de la Información y Comunicación. Esta parte se conforma por:

- Un servidor Web
- Un servidor de correos
- Un servidor de Base de Datos

La tercera parte es externa, el usuario utilizando una conexión a internet podrá acceder al micrositio y sistema web por medio del protocolo HTTPS y a su vez tendrá la facilidad de acceder al HPC Rumiñahui por medio de una VPN la cual será entregada por el administrador del clúster.

El flujo de trabajo propuesto (Figura 28), tras crear una tarea y haciendo uso de un script de ejecución, esta es sometida por medio de la herramienta Torque para el procesamiento en paralelo.

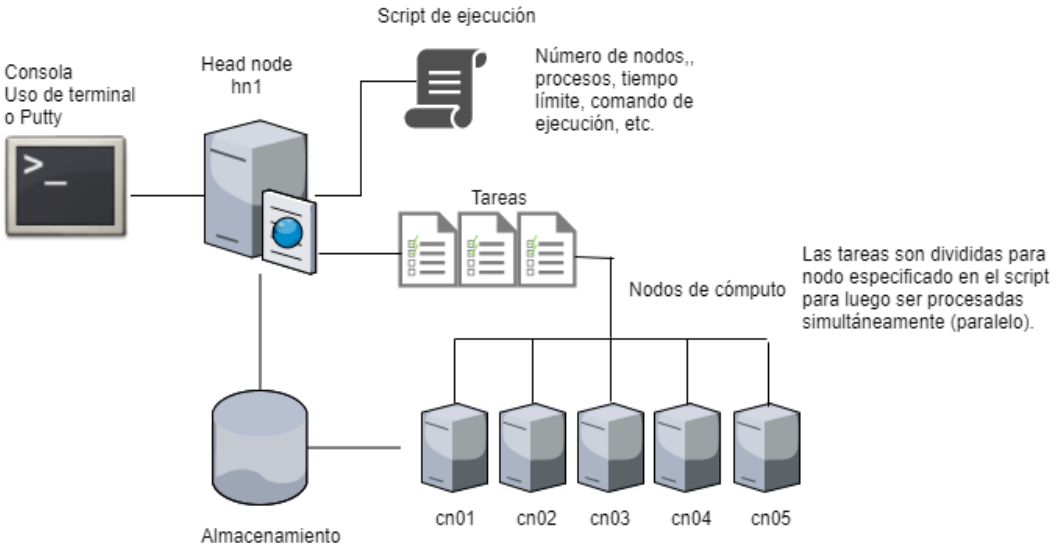


Figura 28 Flujo de trabajo propuesto

El proceso consiste en que el usuario se conecta haciendo uso de una terminal o modo consola mediante una conexión VPN, acto seguido se ingresa en el nodo principal (hn1) con el usuario y contraseña, posterior a esto puede hacer uso del clúster. Para ejecutar una o varias tareas es necesario que el usuario cree un script[15] y defina el número de nodos, procesos, memoria que valla a utilizar para la ejecución del mismo.

La tarea se envía a los nodos especificados con los demás parámetros descritos en el script de ejecución, cuando termina el proceso los resultados son archivados en la unidad de almacenamiento del clúster.

[15] Script: texto con comandos que cumple ciertas características y parámetros de ejecución dentro de un ambiente de programación.

CAPITULO IV
PRUEBAS DE PUNTOS DE REFERENCIA Y RESULTADOS

4.1 Escenarios de pruebas

Para la experimentación se realizaron 3 escenarios de prueba para establecer los puntos de referencia de un archivo de formato XML que nos entregó el Ing. Francisco Flores, investigador en el área de biotecnología.

El archivo XML contiene información para el análisis filogenético del virus de la peste porcina. Contiene un alineamiento múltiple de las secuencias del gen de la glicoproteína E2 de 20 aislados en el Ecuador y múltiples aislados a nivel mundial que sirven como referencia. EL archivo también contiene información sobre diferentes parámetros que el programa BEAST debe calcular para generar árboles filogenéticos durante 10 millones de cadenas de Montecarlo Markov.

Entre los parámetros a calcular están los libres del modelo de evolución Kishino Hasegwa Yano, el modelo de reloj molecular estricto y el modelo demográfico de tamaño constante. Cada secuencia está acompañada del año en que la cepa a la cual pertenece fue aislada, lo cual permitirá generar un árbol ultra métrico calibrado.

El primer escenario se utiliza el archivo strict2.xml en la aplicación Beast en una computadora tradicional sin muchos recursos de hardware, en el segundo escenario se utiliza el archivo strict2.xml en la aplicación Beast instalada en el supercomputador HPC Rumiñahui, y en el tercer escenario se utiliza el archivo strcit2.xml en la aplicación Beast instalada en el supercomputador HPC Rumiñahui utilizando el procesamiento en paralelo.

4.2 Escenario 1: Beast instalado en un computador (Laptop)

En el escenario 1, se procede a realizar la ejecución del archivo strict2.xml utilizando Beast El computador posee las siguientes características:

Marca: HP 630

- Hardware
  - Procesador I3-370 - Arquitectura 64 bits
  - Memoria RAM 3Gbs
  - Disco duro-SATA Hitachi 500Gbs

- Software
  - Java 1.8
  - Beast 1.8.0
  - Beagle 2.1

4.2.1 Caso de prueba N1

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8.

Recopilación de Datos

Durante la ejecución del programa Beast se obtuvo el total de la memoria RAM utilizada (Figura 29), se obtiene el resultado final al completar ejecución de la aplicación (Figura 30).

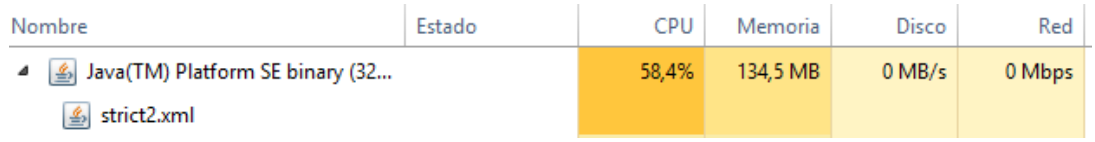


Figura 29 Memoria utilizada Prueba N1

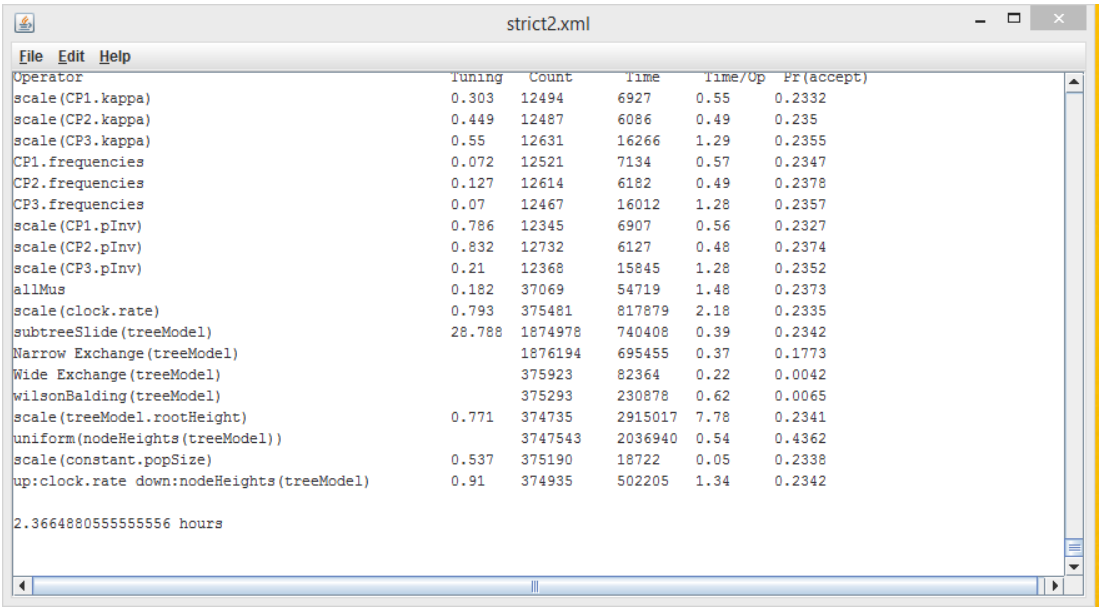


Figura 30 Ejecución Prueba N1

Tabla 17 muestra los datos obtenidos durante la prueba N1 en el escenario 1.

Tabla 17

Datos caso de prueba N1

Tiempo	Memoria RAM	Procesadores	Velocidad
2.36	134.5	2	2.4 GHz

4.2.2 Caso de prueba N2

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 y además Beagle 2.1.

Recopilación de Datos

Durante la ejecución del programa se visualiza el consumo de memoria RAM (Figura 31). También se observa el resultado total de la ejecución con el tiempo de procesamiento (Figura 32).

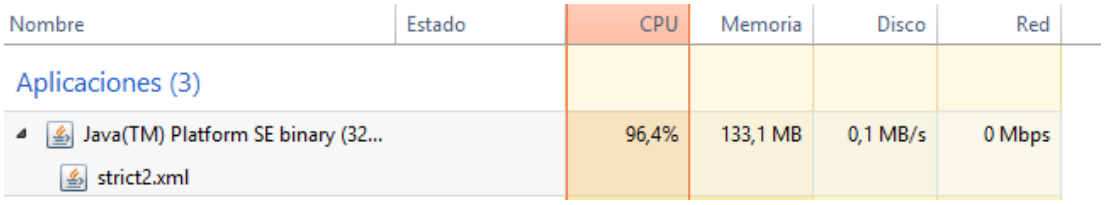


Figura 31 Memoria utilizada Prueba N2

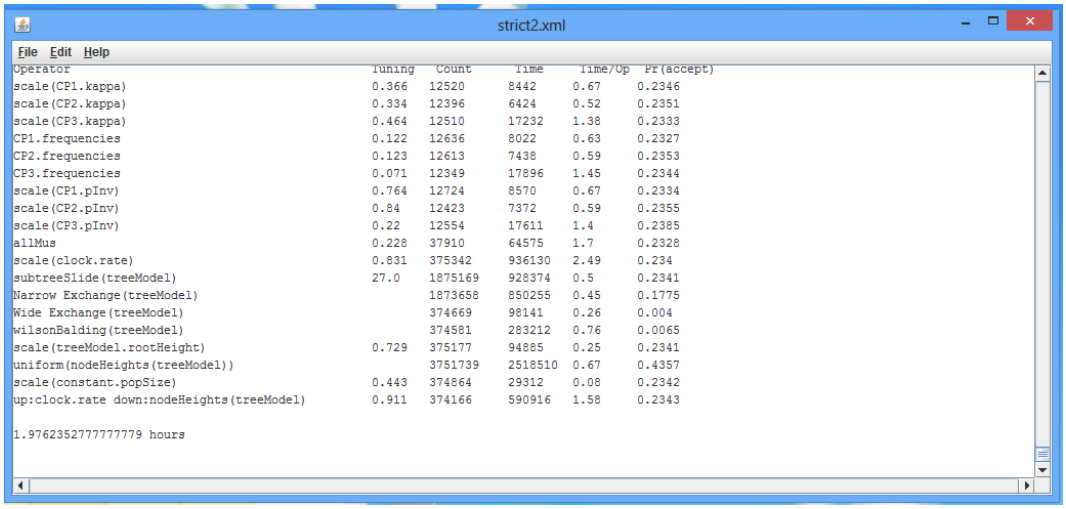


Figura 32 Ejecución Prueba N2

Tabla 18 muestra los datos obtenidos durante la prueba N2 en el escenario 1.

Tabla 18

Datos caso de prueba N2

Tiempo	Memoria RAM	Procesadores	Velocidad
1.97	133.1 MB	2	2.4 GHz

4.2.3 Caso de prueba N3

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 además de la librería Beagle 2.1 y el parámetro de procesamiento Beagle_SSE.

Recopilación de Datos

Durante la ejecución del programa Beast se obtuvo el consumo de memoria RAM (Figura 33), al terminar la ejecución se obtuvo el tiempo total de procesamiento (Figura 34).

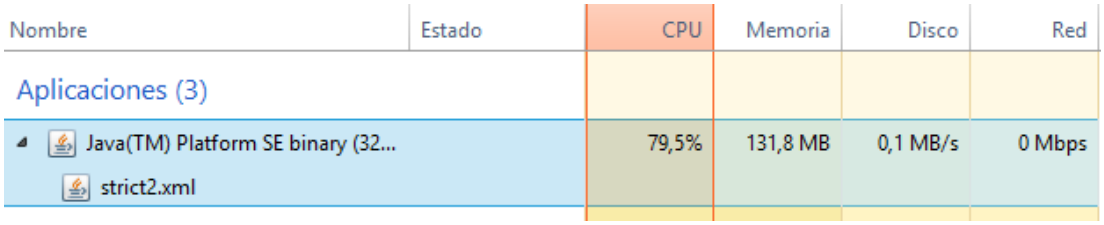


Figura 33 Memoria utilizada prueba N3

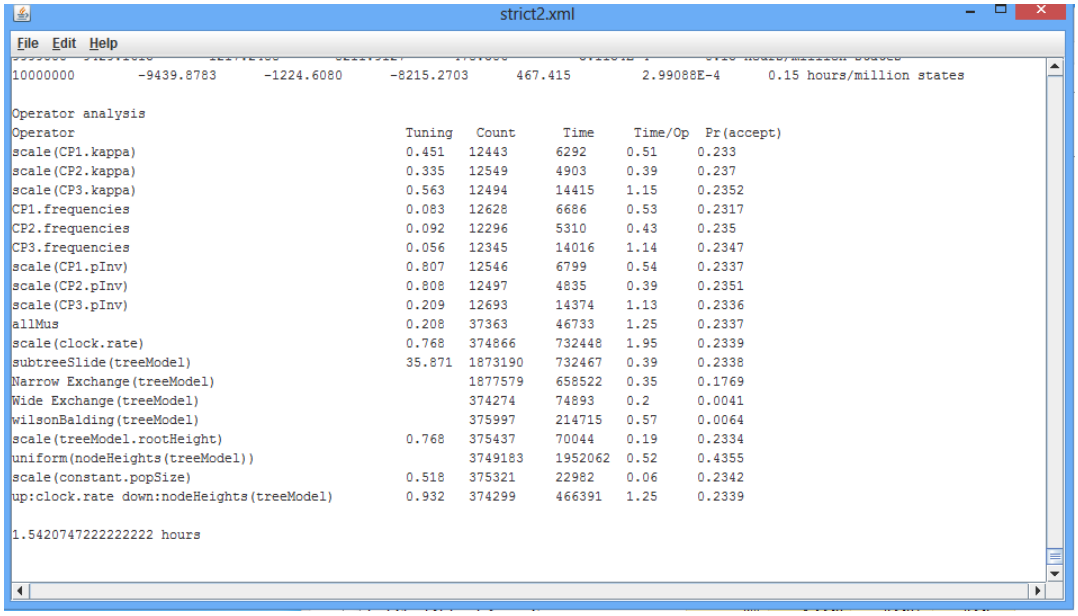


Figura 34 Ejecución prueba N3

Tabla 19 muestra los datos obtenidos durante la prueba N3 en el escenario 1.

Tabla 19

Datos caso de prueba N3

Tiempo	Memoria RAM	Procesadores	Velocidad
1.54	134.5 MB	2	2.4 GHz

4.2.4 Caso de prueba N4

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 además de la librería Beagle 2.1 y el parámetro de procesamiento Beagle_cpu.

Recopilación de Datos

Figura 35. Muestra el consumo de memoria RAM durante la ejecución del software Beast y la biblioteca Beagle.

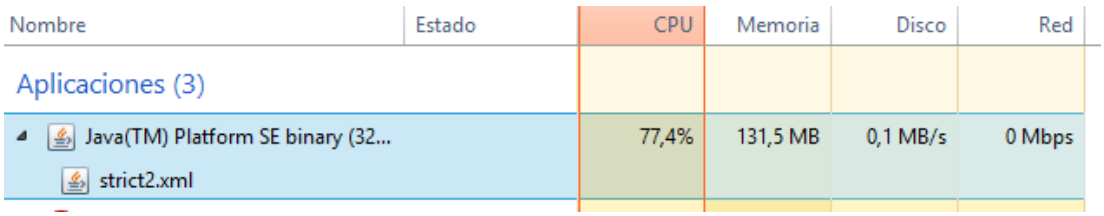


Figura 35 Memoria utilizada prueba N4

Figura 36. Muestra el tiempo total de la ejecución de la prueba.

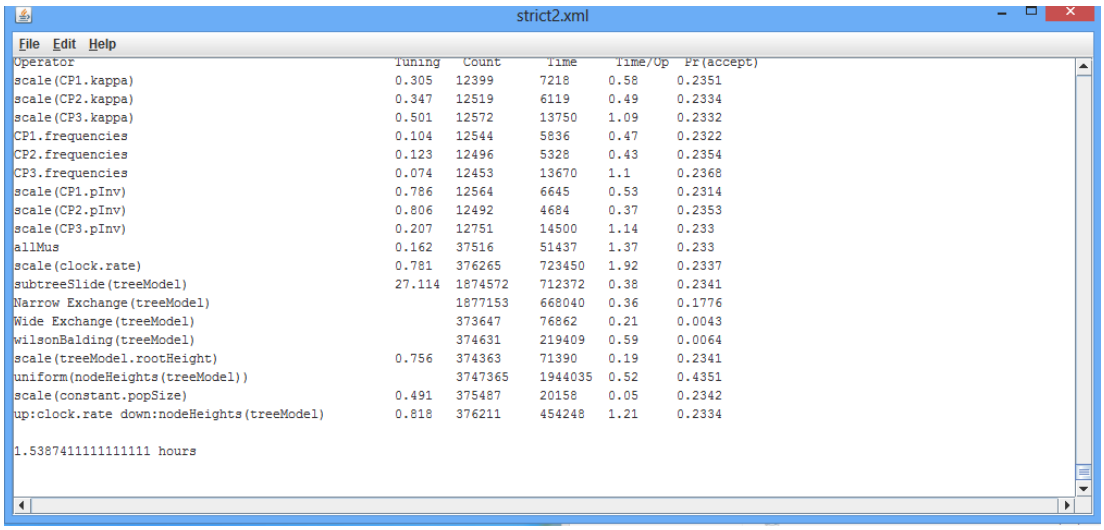


Figura 36 Ejecución prueba N4

Tabla 20 muestra los datos obtenidos durante la prueba N4 en el escenario 1

Tabla 20

Datos caso de prueba N4

Tiempo	Memoria RAM	Procesadores	Velocidad
1.53	131.5 MB	2	2.4 GHz

Tabla 21. Recopilación de todos los datos de las pruebas de ejecución de la aplicación BEAST utilizando las bibliotecas y componentes de Beagle.

Tabla 21

Recopilación de datos Escenario 1

Escenario: Laptop				
Tester: Freddy Vera		**Fecha: 13/07/2017**		
		Software: Beastv1.8.0		
Parámetros	**Pruebas**			
	N1	**N2**	**N3**	**N4**
Tiempo ejecución (horas)	2.36	1.93	1.54	1.538
Tiempo por proceso (horas/proceso*millón)	0.22	0.20	0.15	0.14
RAM utilizada (Mb)	134.5	133.5	134.5	131.5
Bibliotecas utilizadas	**Pruebas**			
	N1	**N2**	**N3**	**N4**
Java	x	x	x	x
Beagle		x	x	x
Beagle_SEE			x	
Beagle_cpu				x

4.3 Escenario 2: Beast instalado en el nodo principal Rumiñahui

En el escenario de prueba N2, se realiza la ejecución del archivo strict2.xml en el supercomputador Rumiñahui que posee las siguientes características:

Supercomputador:

Marca: Hp ProLiant DL350 Gen9

- Hardware
  - Intel Xeon E5-2670 Arquitectura X86_64 (64 Bits)
  - 125 GB Memoria RAM
  - Disco Solido EMC Isilon X200 50 TB
- Software
  - Red Hat Linux Server release 6.7
  - Java 1.8
  - Beast 1.8
  - Beagle 2.1

4.3.1 Caso de prueba N5

En este caso de prueba se ejecuta el software Beast v1.8.0 bajo la biblioteca nativa de Java v1.8.

El comando usado: java -jar lib/beast.jar /app/strict2.xml

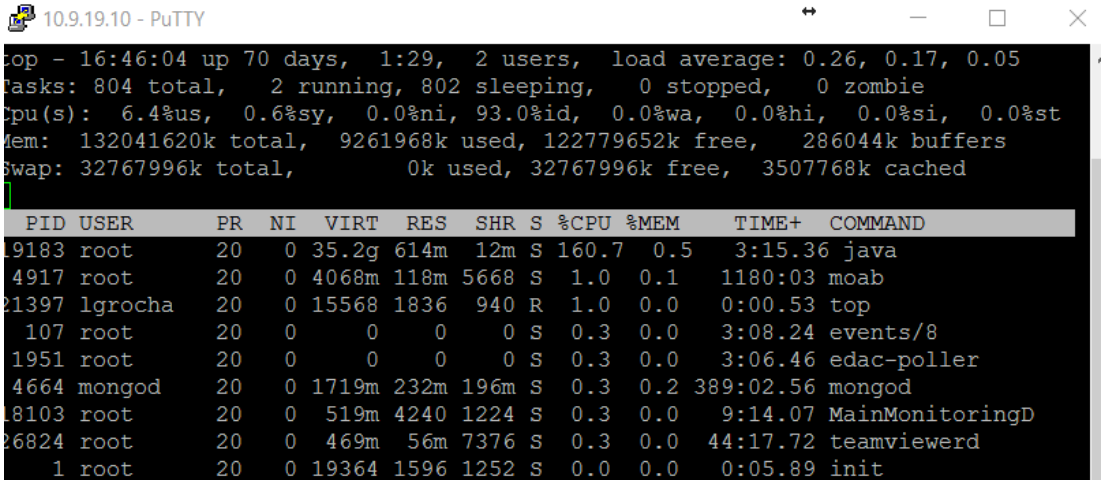


Figura 37 Memoria utilizada prueba N5

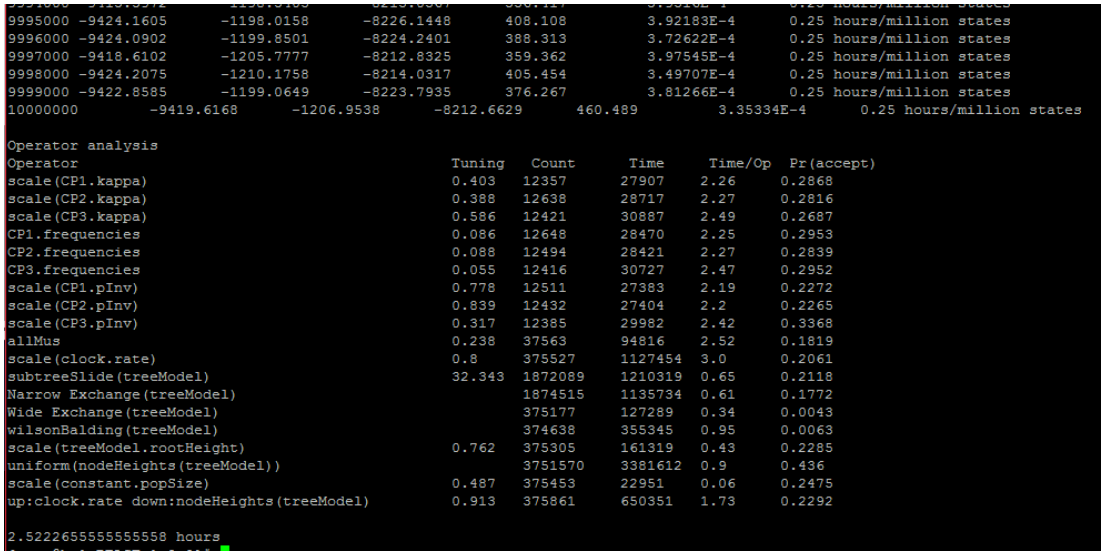


Figura 38 Ejecución prueba N5

En la Tabla 22 se muestran los datos recopilados del escenario 2 en su prueba N5.

Tabla 22

Datos caso de prueba N5

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
2.52	614 MB	0.25 h/p*millón	Intel Xeon E5-2670	2.30 GHz

4.3.2 Caso de prueba N6

En este caso de prueba se ejecuta el software Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 y la biblioteca Beagle 2.1.

El comando usado: java -jar lib/beast.jar -beagle /app/strict2.xml

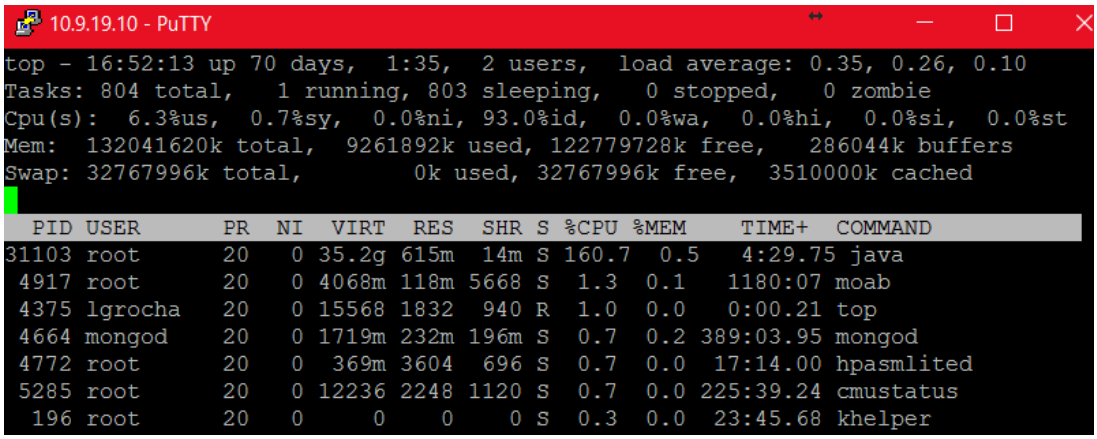


Figura 39 Memoria utilizada prueba N6

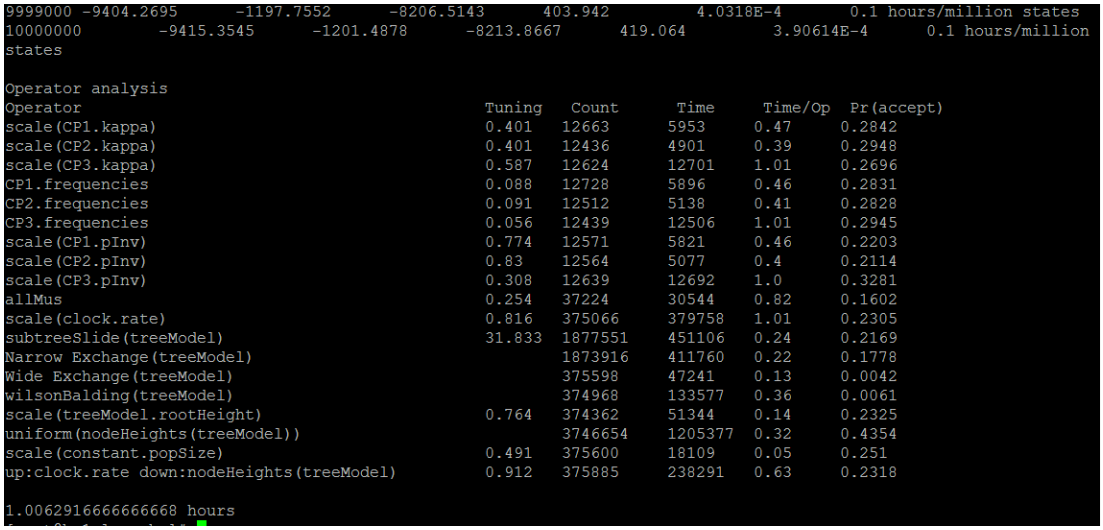


Figura 40 Ejecución prueba N6

En la Tabla 23 se muestra la recopilación de los datos generados en el escenario 2 en la prueba N6.

Tabla 23

Datos caso de prueba N6

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
1.00	615 MB	0.1 h/p*millón	Intel Xeon E5-2670	2.30 GHz

4.3.3 Caso de prueba N7

En este caso de prueba se ejecuta el software Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 y la biblioteca Beagle con el parámetro Beagle_SSE.

El comando usado: java -jar lib/beast.jar -beagle -beagle_SSE /app/strict2.xml

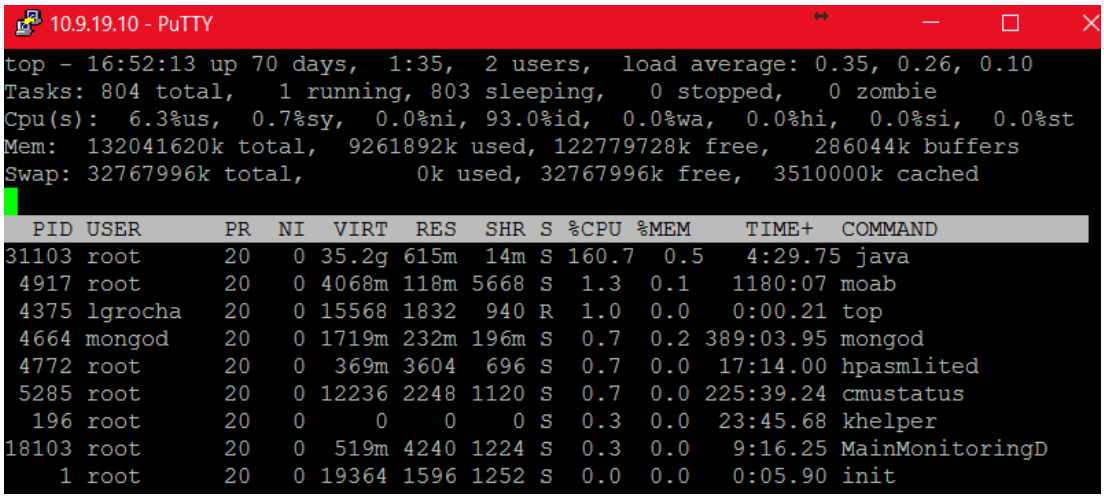


Figura 41 Memoria utilizada prueba N7

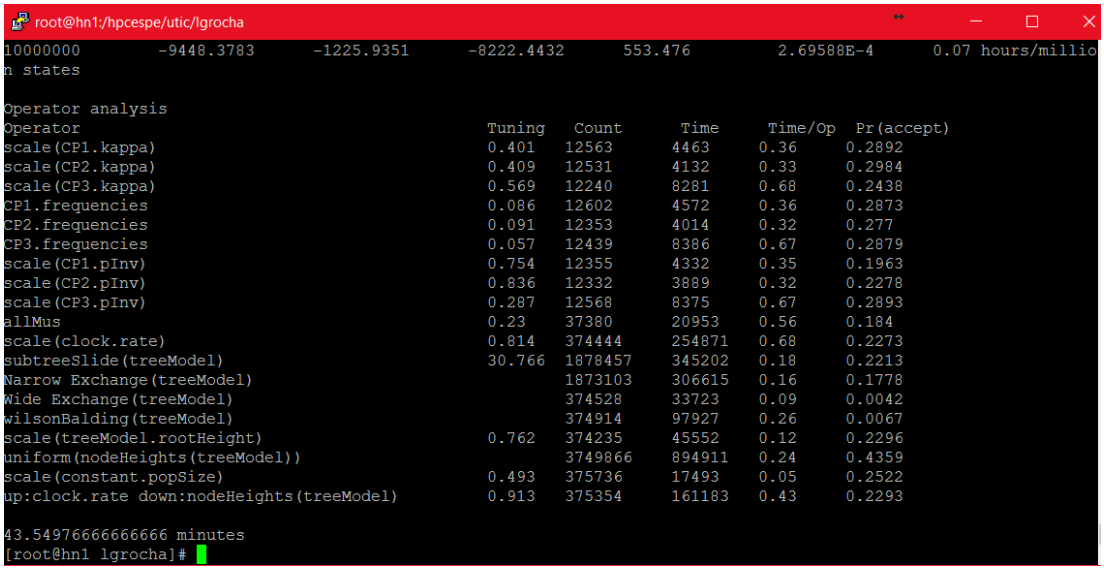


Figura 42 Ejecución prueba N7

En la Tabla 24 se muestra la recopilación de los datos generados en el escenario 2 en la prueba N7.

Tabla 24

Datos caso de prueba N7

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.72	615 MB	0.0.7 h/p*millón	Intel Xeon E5-2670	2.30 GHz

4.3.4 Caso de prueba N8

En este caso de prueba se ejecuta el software Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 y la biblioteca Beagle con el parámetro Beagle_CPU.

El comando usado: java -jar lib/beast.jar -beagle -beagle_cpu /app/strict2.xml

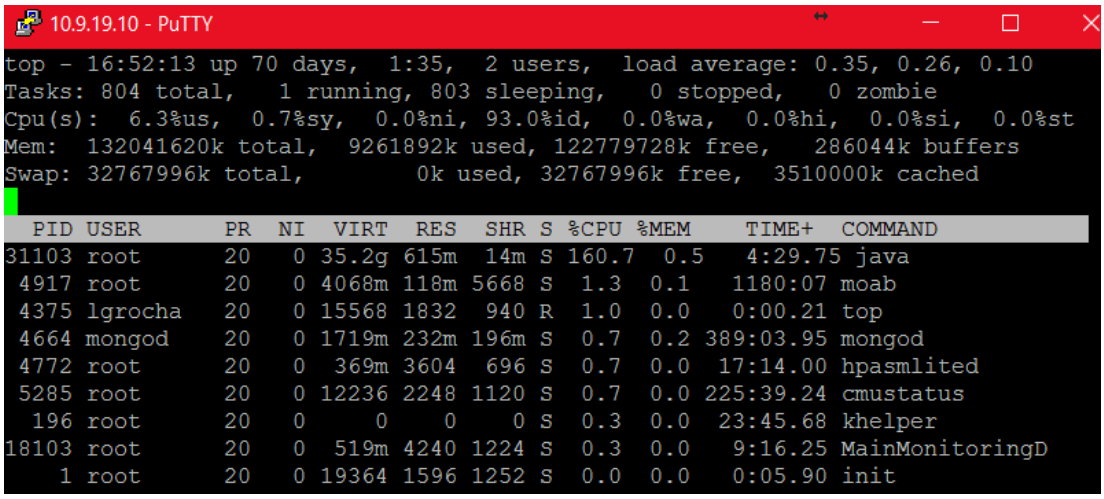


Figura 43 Memoria utilizada prueba N8

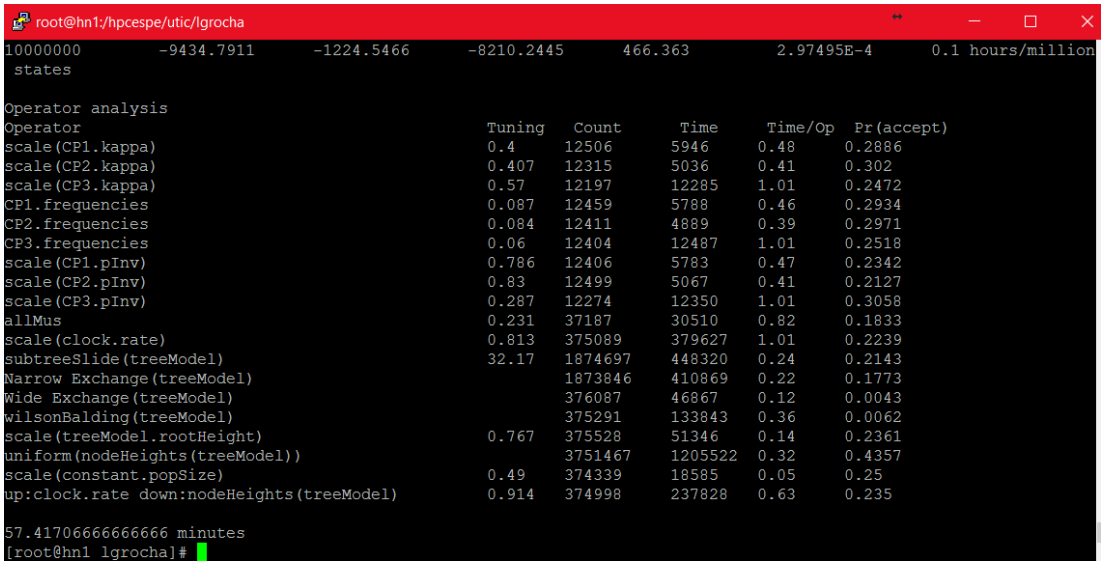


Figura 44 Ejecución prueba N8

En la Tabla 25 se muestra la recopilación de los datos obtenidos en el escenario 2 prueba N8.

Tabla 25

Datos caso de prueba N8

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.96	615 MB	0.0.9 h/p*millón	Intel Xeon E5-2670	2.30 GHz

En la Tabla 26 se muestra la recopilación de los datos obtenidos de todas las pruebas evaluadas en el escenario 2.

Tabla 26

Recopilación de datos Escenario 2

Escenario: Supercomputadora Rumiñahui				
Tester: Luis Rocha	**Fecha: 13/07/2017**			
	Software: Beastv1.8.0			
Parámetros	**Pruebas**			
	N5	**N6**	**N7**	**N8**
Tiempo ejecución (horas)	2.52	1.00	0.72	0.96
Tiempo por proceso (h/p*millón)	0.25	0.10	0.072	0.09
RAM utilizada	614	615	615	615
Bibliotecas utilizadas	**Pruebas**			
	N5	**N6**	**N7**	**N8**
Java	x	x	x	x
Beagle		x	x	x
Beagle_SSE			x	
Beagle_cpu				x

4.4 Escenario 3: Beast instalado la supercomputadora Rumiñahui

En el escenario N3, se procede a realizar la ejecución del archivo strict2.xml con el uso de la biblioteca nativa de Beast con la biblioteca de Beagle instalado en el clúster de procesamiento paralelo HPC con el sistema de colas Torque y Moab.

4.4.1 Caso de prueba N9

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 en el clúster con el sistema de colas Torque y Moab.

El script usado:

```
#PBS -N PRUEBA_N9
#PBS -S /bin/sh
#PBS -l mem=8912mb, nodes=cn01, walltime=1:30:00
#PBS -bea
#PBS -M luigrocha@gmail.com
cd /app/opt/BEASTv1.8.0/lib
java -jar beast.jar -overwrite /app/strict2.xml
```

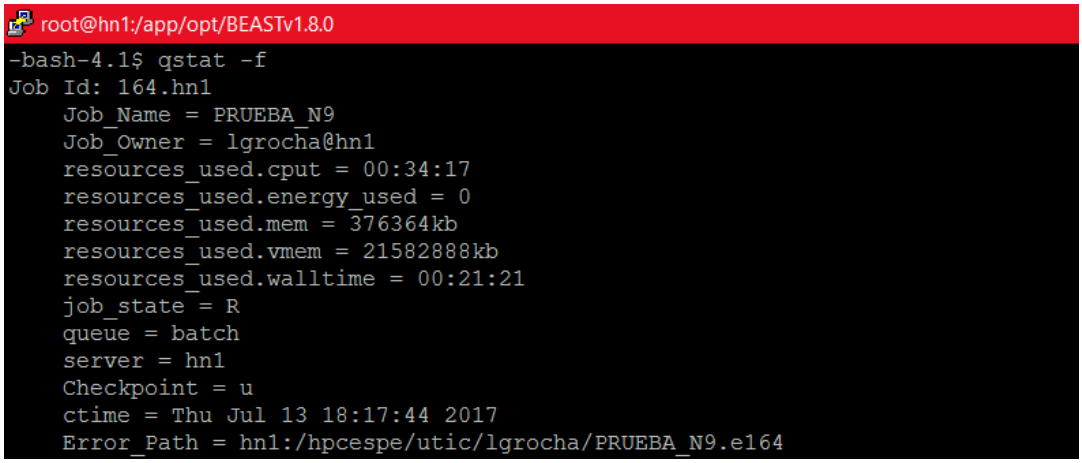


Figura 45 Memoria utilizada prueba N9

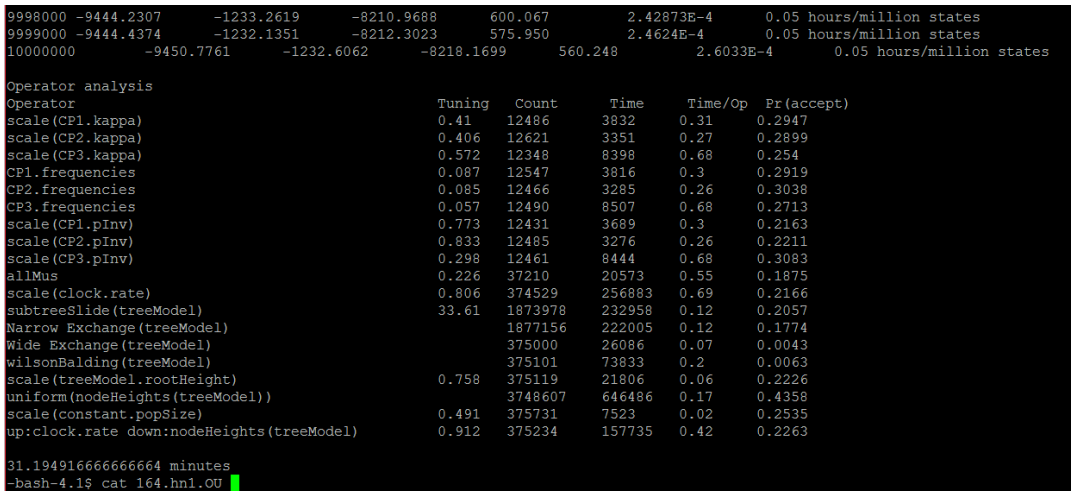


Figura 46 Ejecución prueba N9

En la Tabla 25 se muestra la recopilación de los datos obtenidos en el escenario 2 prueba N8.

Tabla 27

Datos caso de prueba N9

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.305	376.36 MB	0.03 h/p*millón	Intel Xeon Phi Coprocessors	1.33 GHz

4.4.2 Caso de prueba N10

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 con la biblioteca Beagle en el clúster con el sistema de colas Torque y Moab.

El script usado:

```
#PBS -N PRUEBA_N10
#PBS -S /bin/sh
#PBS -l mem=8912mb, nodes=cn01, walltime=1:30:00
#PBS -bea
#PBS -M luigrocha@gmail.com
cd /app/opt/BEASTv1.8.0/lib
java -jar beast.jar -overwrite -beagle /app/strict2.xml
```

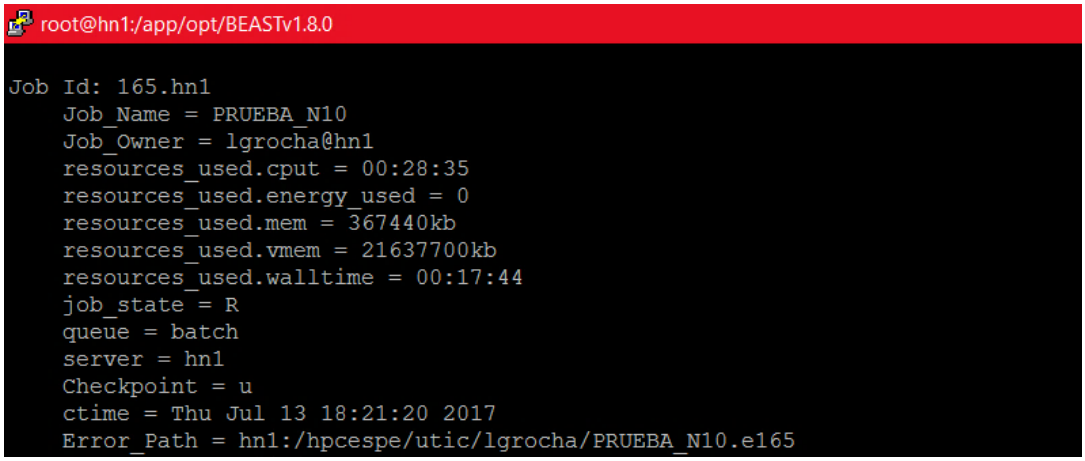


Figura 47 Memoria utilizada prueba N10

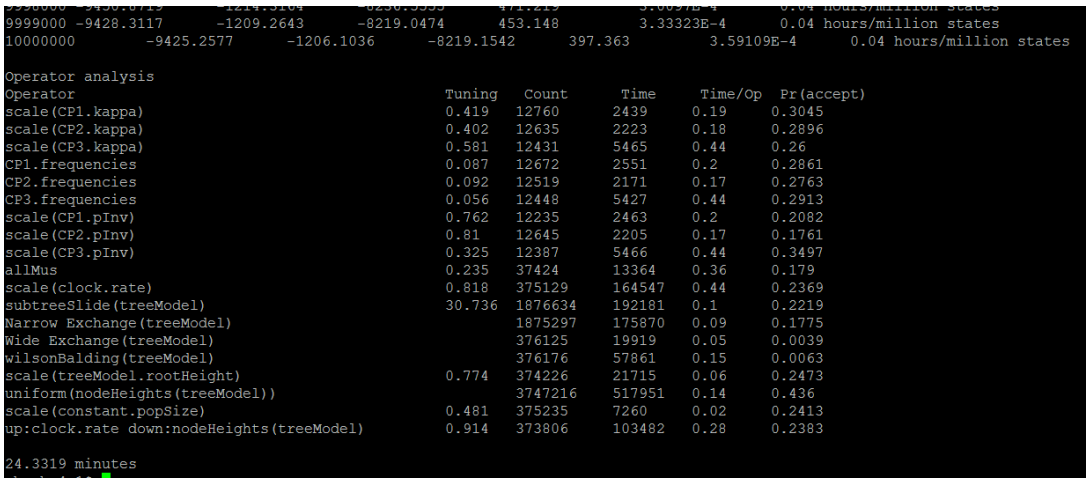


Figura 48 Ejecución prueba N10

Tabla 28. Muestra la recopilación de los datos generados al ejecutar la prueba 10 en el escenario 3.

Tabla 28

Datos caso de prueba N10

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.40	358.82 MB	0.04 h/p*millón	Intel Xeon Phi Coprocessors	1.33 GHz

4.4.3 Caso de prueba N11

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 con la librería Beagle y el parámetro Beagle_SEE en el CLUSTER con el sistema de coles Torque y Moab.

El script usado:

```
#PBS -N PRUEBA_N10
#PBS -S /bin/sh
#PBS -l mem=8912mb, nodes=cn01, walltime=1:30:00
#PBS -bea
#PBS -M luigrocha@gmail.com
cd /app/opt/BEASTv1.8.0/lib
java -jar beast.jar -overwrite -beagle -beagle_SEE /app/strict2.xml
```

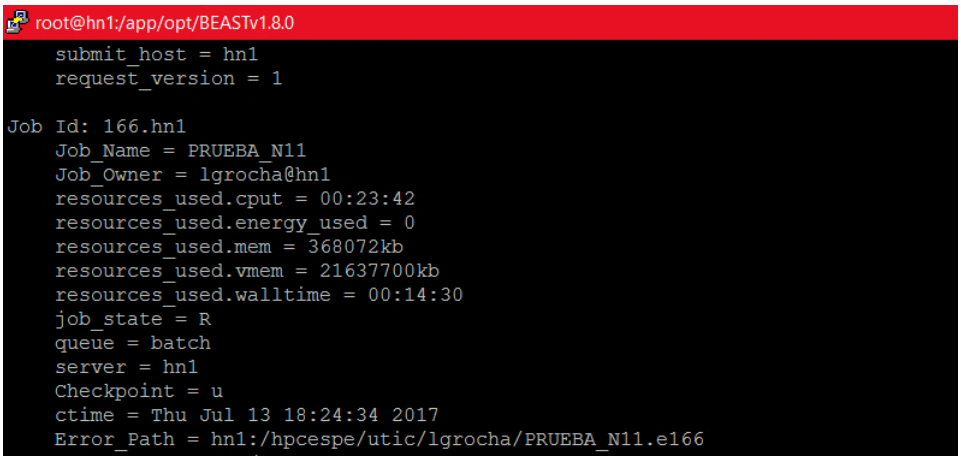


Figura 49 Memoria utilizada prueba N11

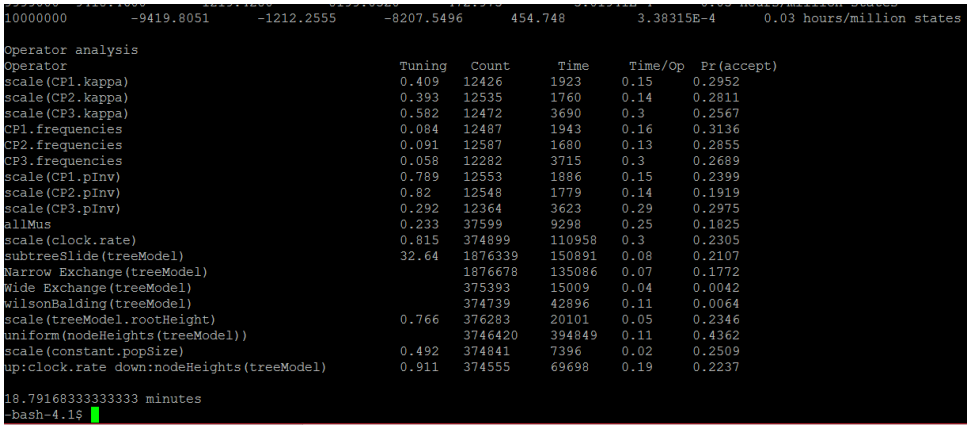


Figura 50 Ejecución prueba N11

Tabla 29. Muestra la recopilación de los datos generados al ejecutar la prueba 11 en el escenario 3.

Tabla 29

Ejecución prueba N11

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.31	368.07 MB	0.03 h/p*millón	Intel Xeon Phi Cooprocessors	1.33 GHz

4.4.4 Caso de prueba N12

En este caso de prueba se realiza la ejecución del Beast v1.8.0 bajo la biblioteca nativa de Java v1.8 en el CLUSTER con el sistema de colas Torque y Moab.

El script usado:

```
#PBS -N PRUEBA_N12
#PBS -S /bin/sh
#PBS -l mem=8912mb, nodes=cn01, walltime=1:30:00
#PBS -bea
#PBS -M luigrocha@gmail.com
cd /app/opt/BEASTv1.8.0/lib
java -jar beast.jar -overwrite -beagle -beagle_cpu /app/strict2.xml
```

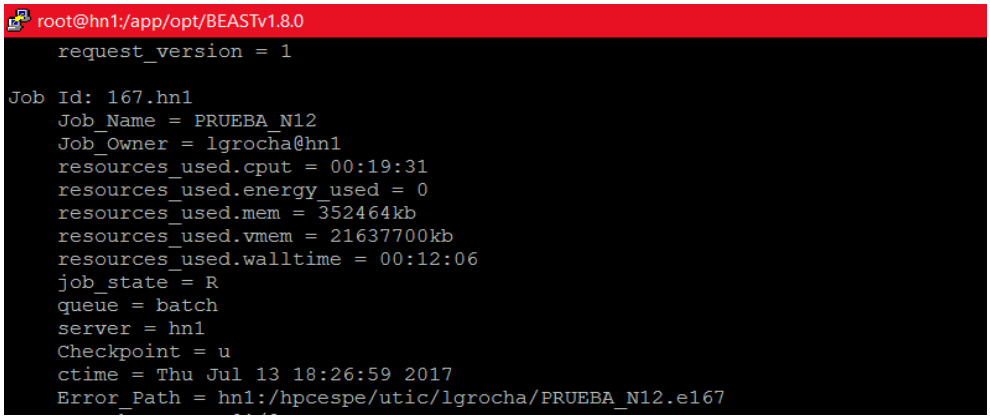


Figura 51 Memoria utilizada prueba N12

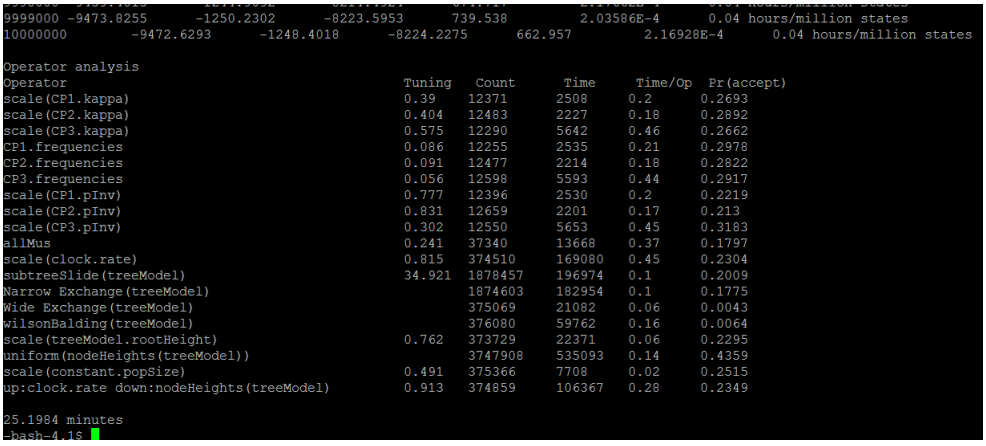


Figura 52 Ejecución prueba N12

Tabla 30. Muestra la recopilación de los datos obtenidos al ejecutar la pruena 12 en el escenario 3.

Tabla 30

Datos caso de prueba N12

Tiempo (horas)	Memoria RAM	Proceso (velocidad)	Procesador	Velocidad
0.42	352.46 MB	0.04 h/p*millón	Intel Xeon Phi Coprocessors	1.33 GHz

En la figura 53 se observa la ejecución de las pruebas N9, N10, N11 y N12 de forma paralela en el supercomputador Rumiñahui.

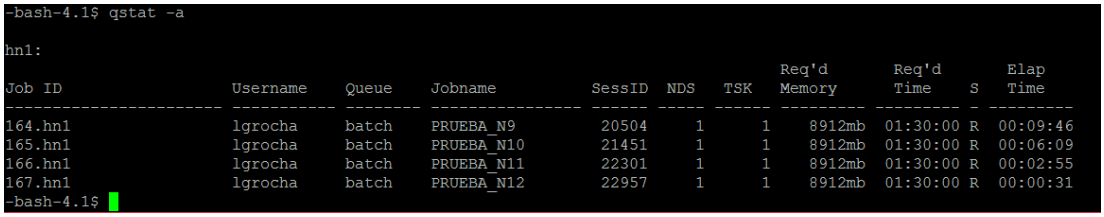


Figura 53 Ejecución pruebas Escenario 3

Tabla 31

Recopilación de datos Escenario 3

Escenario: Supercomputadora Rumiñahui (Procesamiento paralelo)				
Tester: Luis Rocha		**Fecha: 13/07/2017**		
		Software: Beastv1.8.0		
Parámetros	**Pruebas**			
	N9	**N10**	**N11**	**N12**
Tiempo ejecución (horas)	0.51	0.40	0.31	0.42
Tiempo por proceso (horas/proceso*millón)	0.05	0.04	0.03	0.04
Memoria RAM utilizada. (MB)	376.36	367.44	368.07	352.464
Bibliotecas utilizadas	**Pruebas**			
	N9	**N10**	**N11**	**N12**
Java	x	x	x	x
Beagle		x	x	x
Beagle_SEE			x	
Beagle_cpu				x

4.5 Análisis de resultados

4.5.1 Discusión

Realizadas las pruebas, se procede al análisis de los datos obtenidos del software BEAST v1.8.0 luego de ser ejecutado en diferentes arquitecturas de hardware utilizando las bibliotecas de Beagle v2.1 y sus componentes adicionales.

4.5.2 Descripción de las pruebas realizadas

Cada una de las pruebas ejecutaba el script strict2.xml haciendo uso de BEAST y Beagle, las cuales muestran variaciones en los resultados. Para ello se evaluó en base a los siguientes parámetros:

Memoria RAM utilizado: La memoria utilizada por el programa BEAST al momento de ejecutarse, siendo este medido en Megabytes (MB).

Tiempo de ejecución: El tiempo total en horas (h) de ejecución desde que se inicia el script hasta la culminación del mismo.

Velocidad de procesamiento: La velocidad que tarda cada uno de los procesos en BEAST, esta métrica está dada en hora sobre proceso por millón (h/p*m).

4.5.3 Análisis del escenario 1

En el primer escenario se utilizó un computador tradicional Se realizaron 4 pruebas utilizando Beast y los parámetros de la biblioteca Beagle para dar ejecución al script strict2.xml (Tabla 32).

Tabla 32

Resultados Escenario 1

Prueba	Tiempo	Memoria RAM	Cores	Velocidad procesador	Velocidad	Parámetros Beagle
N1	2.36	134.5	2	2.4Ghz	0.22	ninguno
N2	1.93	133.1	2	2.4Ghz	0.20	-beagle
N3	1.54	134.5	2	2.4Ghz	0.15	beagle_SSE
N4	1.529	131.5	2	2.4Ghz	0.14	-beagle_cpu

El script strict2.xml en su ejecución realiza 1000000 de procesos la velocidad de los mismos incrementa haciendo uso de la biblioteca Beagle con el parámetro CPU ya que gestiona de mejor manera los recursos del computador.

Figura 54. Muestra la variación de memoria RAM utilizada por cada prueba. Donde le eje de las X define los parámetros de Beagle y el eje de las Y el consumo de memoria RAM en megabytes (MB).

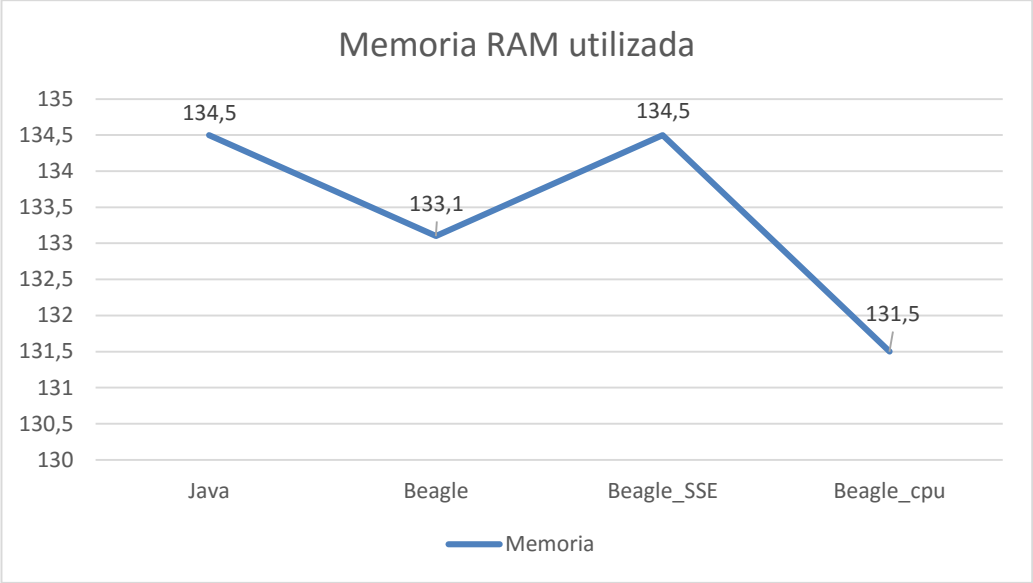


Figura 54 Memoria RAM Escenario 1

Figura 55. Muestra el tiempo transcurrido en horas desde la ejecución del script hasta su culminación. Donde el eje izquierdo muestra los parámetros de Beagle utilizados y el eje inferior muestra el tiempo en horas.

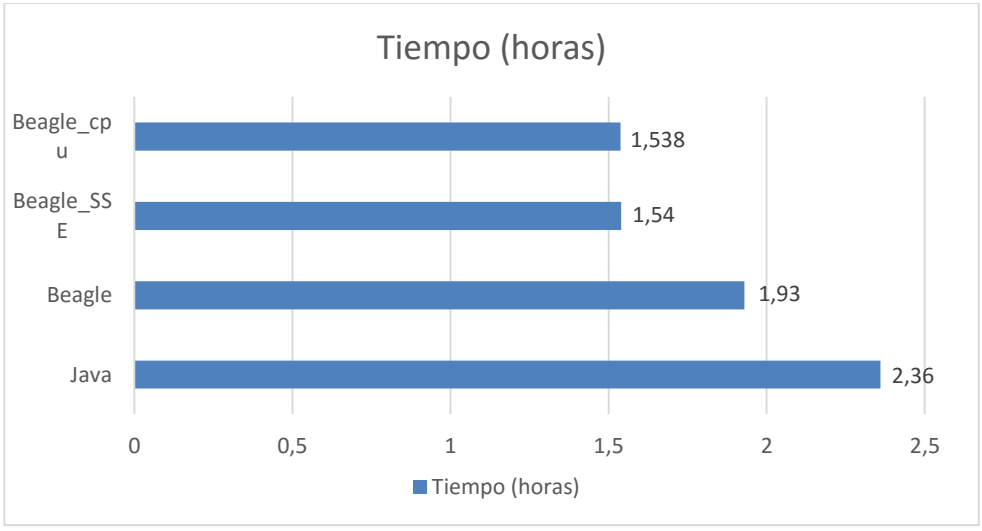


Figura 55 Tiempo de Ejecución Escenario 1

Figura 56. Muestra el tiempo que tarda cada uno de los procesos en realizarse en este caso 1000000 de procesos. Como se observa el eje de las x muestra los parámetros de Beagle y el eje de las Y el tiempo por proceso.

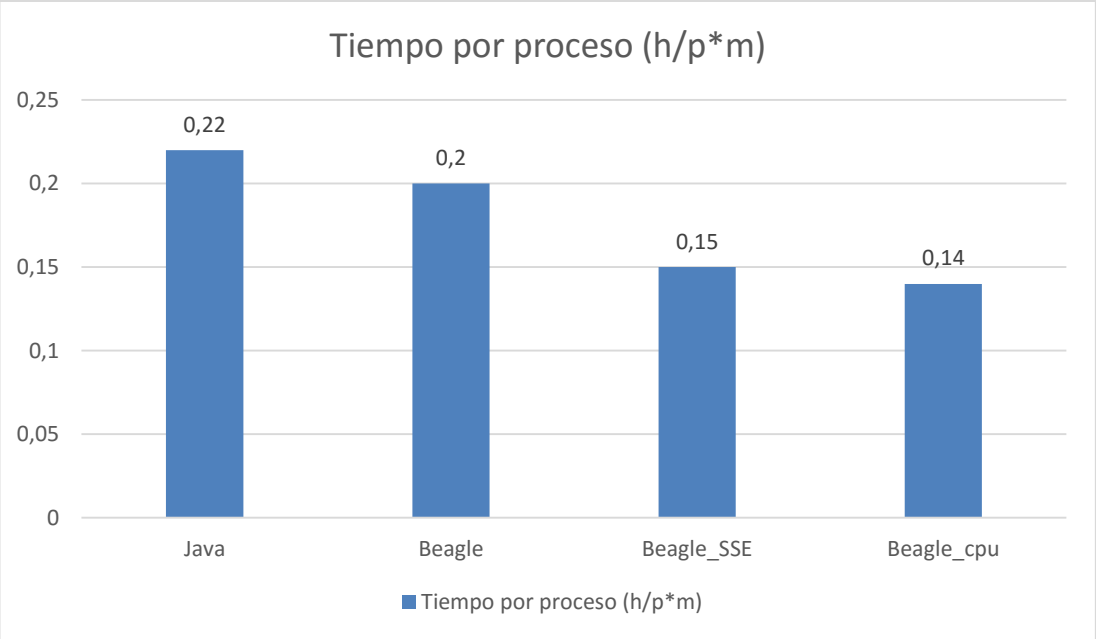


Figura 56 Tiempo por proceso Escenario 1

Según los datos obtenidos en el escenario N1 se deduce que la mejor alternativa utilizada es Beast y Beagle utilizando el parámetro Beagle_cpu, disminuyendo la velocidad de ejecución y el consumo de memoria RAM del equipo.

4.5.4 Análisis del escenario 2

Para el escenario 2 se utilizó la infraestructura del HPC, para esto se tomó al nodo principal como sujeto de pruebas donde se realizó la ejecución del script strict2.xml utilizando el software Beast, la biblioteca Beagle y sus componentes.

Tabla 32. Muestra los diferentes resultados encontrados al realizar las 4 pruebas en el escenario 2.

Tabla 33

Resultados Escenario 2

Prueba	Tiempo	Memoria RAM	Cores	Velocidad procesador	Velocidad	Parámetros Beagle
N5	2.52	614	24	2.3Ghz	0.25	ninguno
N6	1.00	615	24	2.3Ghz	0.10	-beagle
N7	0.72	615	24	2.3Ghz	0.072	-beagle_SSE
N8	0.96	615	24	2.3Ghz	0.09	-beagle_cpu

Figura 57. Muestra la variación de memoria RAM utilizada al ejecutar el software Beast y Beagle con sus diferentes parámetros sobre el nodo principal del HPC.

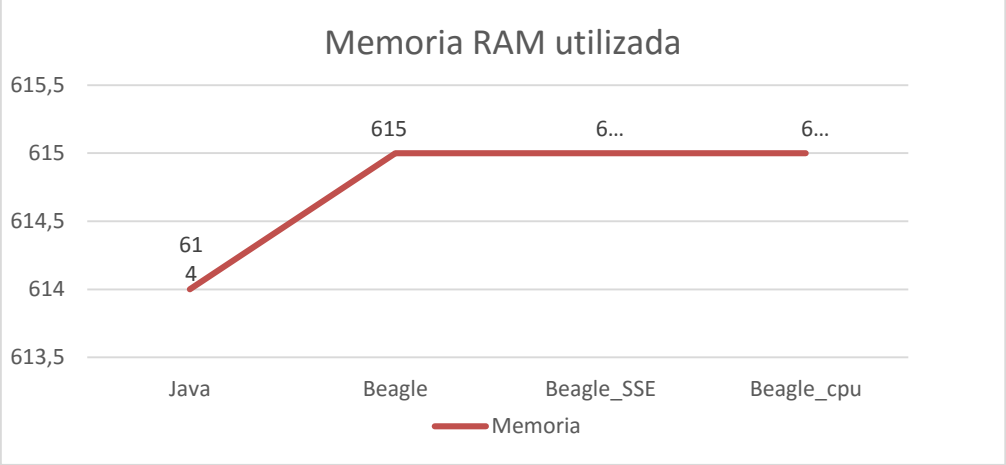


Figura 57 Memoria RAM utilizada Escenario 2

El eje X representa los complementos de Beagle, el eje Y la memoria RAM en megabytes (MB). Se aprecia un aumento de la memoria RAM al usar la biblioteca Beagle, los tiempos de ejecución son bajos compensado la memoria utilizada

Figura 58. Muestra el tiempo de ejecución al ejecutar el software Beast y su biblioteca Beagle utilizando el script strict2.xml.

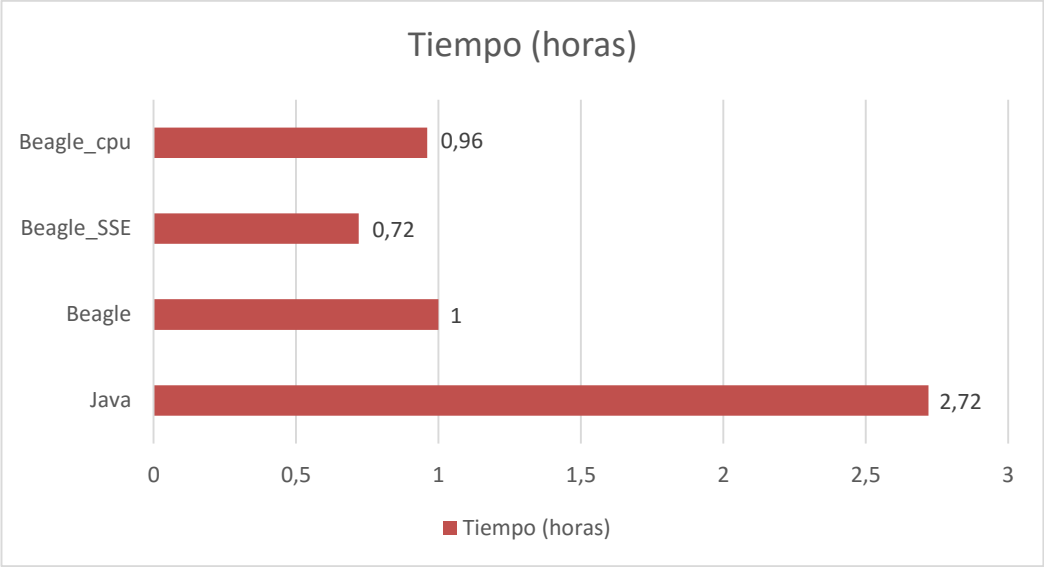


Figura 58 Tiempo de ejecución escenario 2

En el eje Y se presentan los complementos de Beagle y en el eje X se presenta el tiempo en horas. Se aprecia que el tiempo disminuye considerablemente al momento de utilizar Beagle, gestionando mejor los procesos y aumentando la velocidad de procesamiento. Al analizar los resultados del escenario se comprueba que la mejor alternativa es Beast utilizando la biblioteca Beagle y el complemento Beagle_SSE que tardan menos tiempo en procesar la información y presentar un resultado.

Figura 59. Muestra el tiempo que tarda en crearse cada uno de los procesos del strict2.xml, dependiendo de la biblioteca utilizada y los componentes de Beagle.

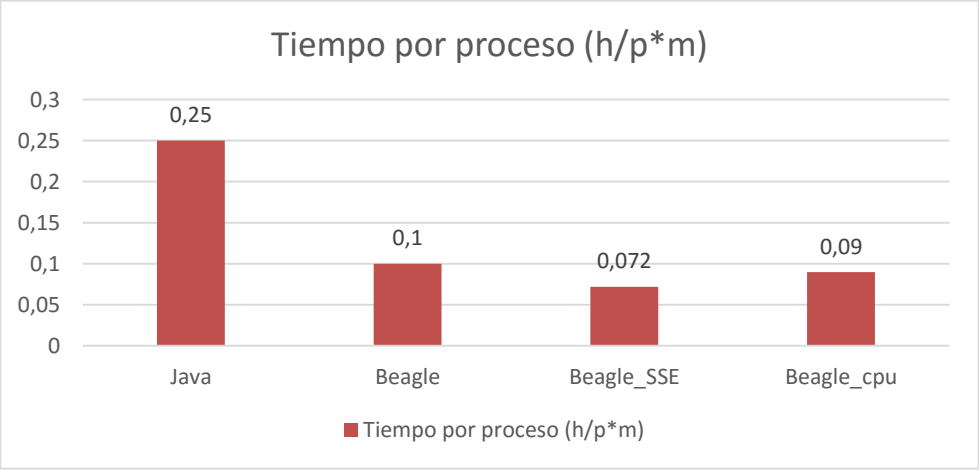


Figura 59 Tiempo por proceso escenario 2

El eje de las X muestra los complementos utilizados de Beagle y el eje de las Y la variación de (hora/proceso*millón). El mejor tiempo de ejecución se presenta al utilizar el software Beast y la biblioteca Beagle con el complemento Beagle_SSE.

4.5.5 Análisis del escenario 3

En este escenario se utilizó toda la infraestructura del HPC y el programa Torque el cual se encarga de la gestión de las colas y procesamiento de tareas de los nodos de cómputos en el clúster.

En este escenario se utiliza un script de ejecución de extensión Shell en el cual se encuentran especificados los recursos a utilizar para el procesamiento en el HPC. Seleccionados dichos parámetros se envía en trabajo a encolamiento para su ejecución.

Entre los parámetros admitidos en el script de ejecución se tiene:

- Nombre de la tarea a encolar.
- Número de nodos a utilizar.
- Número de procesos por nodo.
- Tiempo máximo de ejecución del script.
- Memoria RAM mínima.
- Memoria RAM máxima.
- Definir ubicación de salida de resultados.

Tabla 34. Muestra los resultados obtenidos al ejecutar Beast y Beagle en los nodos de procesamiento del HPC utilizando Torque para el trabajo en paralelo.

Tabla 34

Resultados Escenario 3

Prueba	Tiempo	Memoria RAM	Cores	Velocidad del procesador	Velocidad	Parámetros Beagle
N9	0.51	376.36	61	1.33GHz	0.05	ninguno
N10	0.40	367.44	61	1.33GHz	0.04	-beagle
N11	0.30	368.07	61	1.33GHz	0.03	-beagle_SSE
N12	0.42	352.46	61	1.33GHz	0.04	-beagle_cpu

Figura 60. Muestra la memoria RAM utilizada luego de ejecutar Beast y Beagle en el clúster por medio del programa Torque.

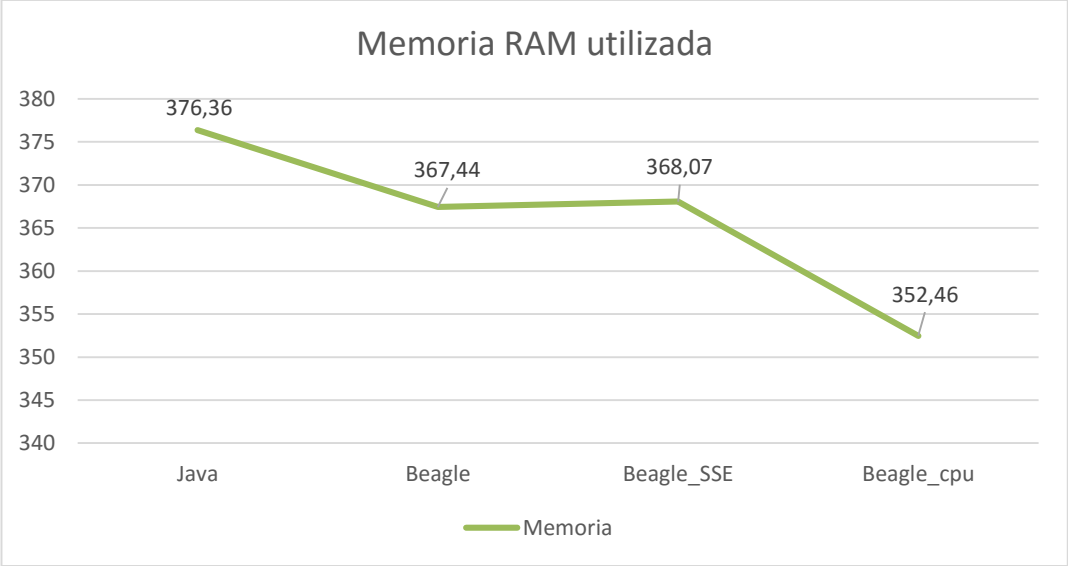


Figura 60 Memoria RAM utilizada escenario 3

En el eje de la X se presentan los complementos de Beagle utilizados, en el eje de las Y la memoria RAM utilizada. Se aprecia que la memoria RAM disminuye al utilizar el Beagle_cpu, este gestiona mejor los recursos del clúster al realizar tareas en paralelo.

Figura 61. Muestra el tiempo total de ejecución de la tarea desde el momento que empieza a ejecutarse en el clúster utilizando el programa Torque

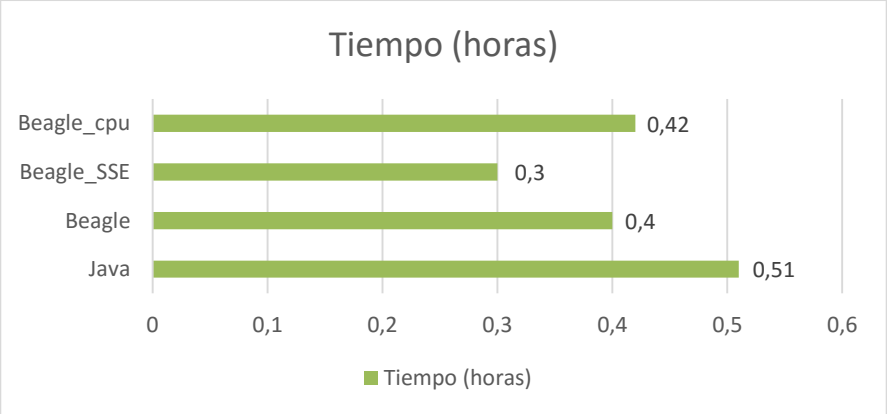


Figura 61 Tiempo de ejecución Escenario 3

En el eje Y se presentan los complementos de Beagle y en el eje X se presenta el tiempo en horas. Se aprecia la disminución de los tiempos en relación con los escenarios propuestos. El complemento de Beagle_SSE es el mejor optimizando los recursos del clúster en el procesamiento en paralelo con el menor tiempo de ejecución.

Figura 62. Muestra los tiempos por proceso generados por cada complemento de Beagle al ejecutarlos en el clúster HPC y el programa Torque para el procesamiento en paralelo.

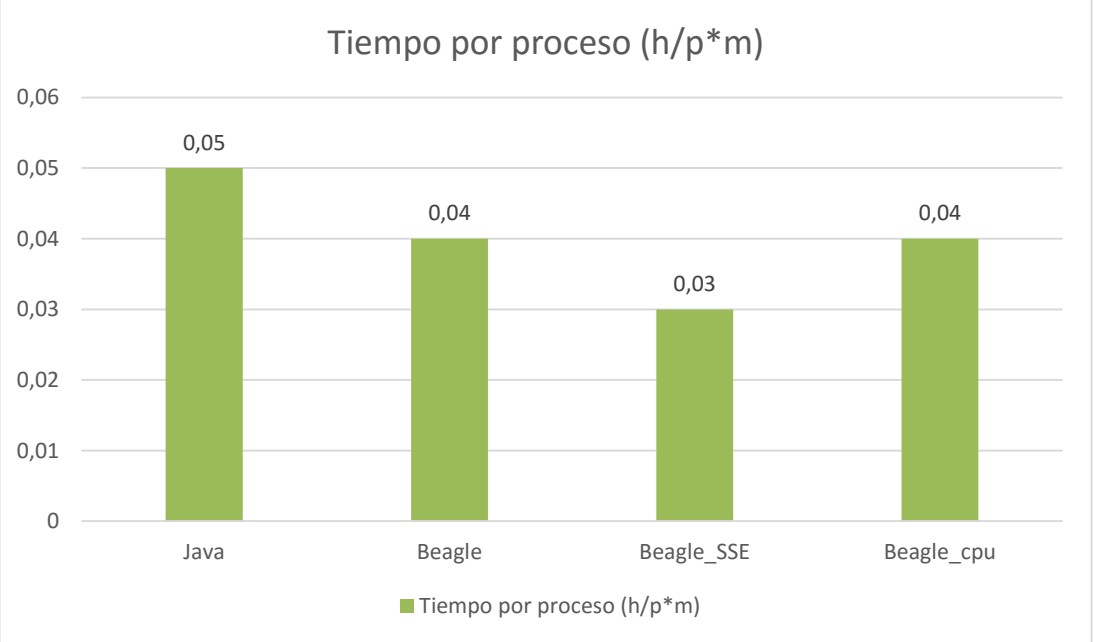


Figura 62 Tiempo por proceso Escenario 3

El eje de las X muestra los complementos utilizados de Beagle y el eje de las Y la variación de (hora/proceso*millón). Se aprecia una reducción de los tiempos por proceso inclusive sin el uso de Beagle. Estos resultados se obtuvieron al ejecutar Beast y Beagle en los nodos de procesamiento del clúster. El complemento Beagle_SSE ofrece el mejor tiempo por proceso.

4.6 Análisis comparativo

Con todas las pruebas de ejecución de Beast y Beagle se procede a realizar la comparativa por escenario. Como se muestra en la (Figura 63) se tomaron los tiempos de ejecución totales como factor principal mostrando las siguientes variaciones:

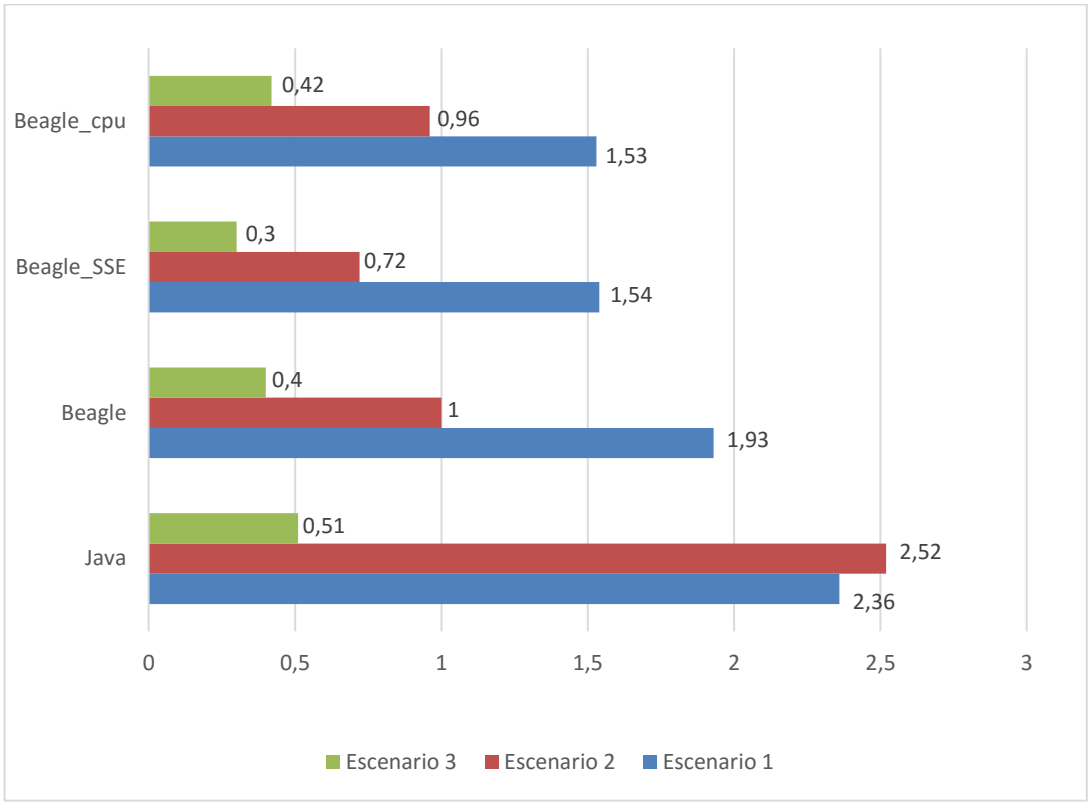


Figura 63 Comparativa entre tiempos totales por escenario

La diferencia en los resultados de cada escenario depende de la arquitectura donde se ejecutó el programa Beast y los complementos de Beagle. Se elige el escenario 3 al ofrecer el mejor tiempo de respuesta, aportando una solución más rápida a los requerimientos de los investigadores

En memoria RAM utilizada se realiza la comparación de los escenarios como se observa en la (Figura 64), el escenario 3 consume una media de 366,08 megabytes (MB) al hacer la ejecución del programa Beast y Beagle con procesamiento en paralelo.

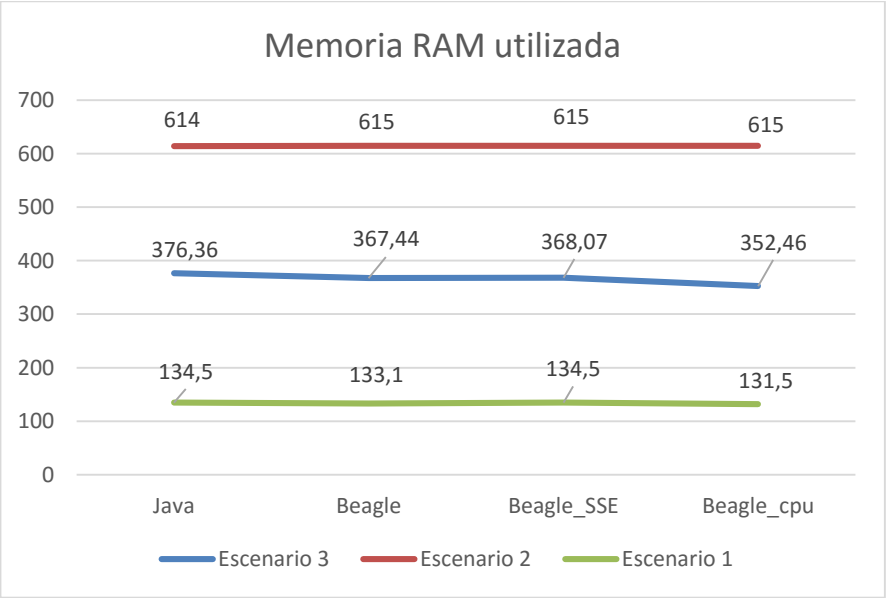


Figura 64 Comparativa entre memoria RAM por escenario

CAPITULO V
CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURO

5.1 Conclusiones

Se escoge Beast y su biblioteca Beagle por ser aplicaciones paralelizables de alto rendimiento que cumplen con los requerimientos en el área de bioinformática logrando proveer a los investigadores una herramienta innovadora para el análisis de sus modelos matemáticos.

Se desarrolla un micrositio informativo y un sistema de gestión de solicitudes que ayudará a la administración de los investigadores para el uso del supercomputador Rumiñahui utilizando la metodología de desarrollo ágil Scrum.

La ejecución de simulaciones y modelos matemáticos utilizando el software BEAST en la supercomputadora disminuyó el tiempo de procesamiento, evidenciando una optimización en un 513.3% en comparación a una arquitectura convencional sin HPC.

Se desarrollaron manuales de políticas y puesta en producción que están en línea para fácil acceso tanto para administradores como investigadores para ayudarlos a involucrarse e innovar en los proyectos de investigación utilizando el supercomputador Rumiñahui.

5.2 Líneas de trabajos Futuros

Como resultado de la investigación realizada se plantean las siguientes líneas de trabajo futuro divididas en tres etapas:

Corto plazo:

- Instalación de otros softwares a petición de las necesidades de los investigadores enviadas a la Unidad de Gestión de la Investigación (UGI), los cuales deben ser de código libre y que puedan ser instalados y ejecutados bajo una arquitectura HPC.

Mediano plazo:

Preparación del HPC en hardware y software para el primer concurso de programación paralela en la Universidad de las Fuerzas Armadas ESPE, donde podrán participarán concursantes internos y externos a la institución.

Largo plazo:

- Adaptación a la versión 2.0 del sistema de Gestión de Investigadores en conjunto con la consola de comandos del clúster, en el cual los envíos de trabajos se hacen directamente desde el sistema sin la necesidad de ingresar al HPC. En este caso el investigador una vez registrado, solo sube el archivo XML y elige el programa con el que lo desee ejecutar.
- Instalación de la herramienta para control y administración del clúster MOAB HPC Suite, actualmente el sistema se encuentra utilizando su versión gratuita Lite. Esto ayudará al control de los recursos y optimización de cargas de trabajos en los nodos de cómputo.

REFERENCIAS BIBLIOGRAFICAS

Aguilar , J., & Leiss, E. (2004). Introducción a la Programación Paralela. *1*.

Alegsa, L. (2010, 12 05). *Alegsa*. Obtenido de http://www.alegsa.com.ar/Dic/framebuffer.php

Alvarez, D., Jorquera, C., Sepúlveda, G., & Zamora, C. (2014, 07 25). *Redes Privadas Virtuales VPN*. Obtenido de http://profesores.elo.utfsm.cl/~agv/elo322/1s14/projects/reports/G20/Redes%20Privadas%20Virtuales%20(VPN).pdf

Arellano, A., & Fernádez, J. (2014, 06). *Repositorio ESPE*. Obtenido de http://repositorio.espe.edu.ec/bitstream/21000/8882/1/T-ESPE-048109.pdf

ATHEMES. (2017). Obtenido de https://athemes.com/theme/sydney/

BEAST. (2016). Obtenido de http://beast.bio.ed.ac.uk/

BEAST. (2016). Obtenido de http://beast.bio.ed.ac.uk/BEAGLE

Brockmeter, J. (2012, 04 10). *"Into the Mosh Pit: A Mobile Shell Replacement for SSH"*. Obtenido de https://www.linux.com/learn/mosh-pit-mobile-shell-replacement-ssh

CCM. (2008). *Protocolo HTTP*. Obtenido de http://es.ccm.net/contents/264-el-protocolo-http

Centro Nacional de Cáculo Científico. (2004, 04 15). *MrBayers*. Obtenido de http://www.cecalc.ula.ve/BIOINFO/servicios/herr2/MrBayes/index.php

Cobbaut, P. (2015, 05 24). *Linux Fundamentals*. Obtenido de http://linux-training.be/linuxfun.pdf

CODECALL. (2011, 10 20). Obtenido de http://forum.codecall.net/topic/66411-puttys-raw-tcp-connection/

Codigniter. (2017). *Codeigniter*. Obtenido de https://www.codeigniter.com/user_guide/general/welcome.html

Cueto Díaz. (2004). *Estudio e Implementación de un Cluster Clase.* Pontificia Universidad Católica de Valparaiso (PUCV).Facultad de Ingeniería. Escuela de Ingeniería Eléctrica (EIE), Chile.

Dávila Neuman, G. (2006). *El razonamiento inductivo y deductivo dentro del proceso investigativo en ciencias experimentales y.* Caracas: Laurus.

Debian news. (2017, 05 06). *All about Debian.* Obtenido de https://www.debian.org/intro/about

DesarrolladorWeb. (2017). *Javascript*. Obtenido de https://desarrolloweb.com/javascript/

Drummound, A., Ho, S., Rawlence, N., & Rambaut , A. (2007, 07 06). *Molecular Evolution.* Obtenido de http://www.molecularevolution.org/molevolfiles/beast/BEAST14_MANUAL-7-6-07.pdf

Eng Hee, Y. (2010). *Putty a free SSH client*. Obtenido de http://www.nus.edu.sg/comcen/news/HPC/articles/Putty.pdf

ESPE-UTICS. (2015). Infraestructura Actual en el data Center de Investigación Espe. Quito.

Evomics. (2017). *Garli*. Obtenido de Evolution and Genomics: http://evomics.org/resources/software/molecular-evolution-software/garli/

Forticlient. (2017). Obtenido de https://www.fortinet.com/support-and-training/support/product-downloads.html

Forticlient Administration Guide. (2017, 05 12). Obtenido de http://docs.fortinet.com/uploaded/files/3562/FortiClient-5.4.3-Administration-Guide.pdf

GetComposer. (2017). *Composer Documentation.* Obtenido de https://getcomposer.org/doc/00-intro.md

Green, S. (2015, 05 15). *Me Domains*. Obtenido de https://domain.me/all-you-need-to-know-about-cpanel/

Haymore, B. (2014, 08). *Linux Cluster Institutte.* Obtenido de http://www.linuxclustersinstitute.org/workshops/archive/20th/files/11-MoabTorque/Moab-Torque.pdf

I.E.S. San Vicente. (2017, 03). *Oracle* . Obtenido de https://iessanvicente.com/colaboraciones/oracle.pdf

IBM. (2017). *Introducción a la programación en JAVA*. Obtenido de https://www.ibm.com/developerworks/ssa/java/tutorials/j-introtojava1/index.html

MobaXterm. (2017). Obtenido de http://mobaxterm.mobatek.net/documentation.html#1_1

Morrison, R. (2003). Cluster Computing. *GNU*.

MrBayers. (2017). *MrBayers: Bayesian Inference of Phylogeny*. Obtenido de http://mrbayes.sourceforge.net/

Murillo, J. (2008). *PostgradoUNE.* Obtenido de http://www.postgradoune.edu.pe/documentos/Experimental.pdf

MySQl. (2017). *About MySQL*. Obtenido de https://www.mysql.com/about/

Navas, A. A. (2006). Clusters.

Oracle. (2017). *Oracle documentation.* Obtenido de https://docs.oracle.com/cd/E19593-01/E22994/gizfh.html

Pavón, J. (2010). *Introducción al lenguaje JAVA.*

Peralta, A. (2003). Metodología Scrum. Uruguay: Universidad ORT Uruguay.

PHP. (2017). *PHP Documentation.* Obtenido de http://php.net/manual/es/intro-whatis.php

Platzi. (2015). *Laravel Framework.* Obtenido de https://platzi.com/blog/laravel-framework-php/

PostgreSQL. (2017). *All about PostgreSQL.* Obtenido de https://www.postgresql.org/about/

REGO. (2006). *Balanceamento de carga em clusters de alto desempenho:uma extensão para a LAM/MPI.* Universidade Estadual de Maringá.

Reino, A. (2000, 01 26). Introducción a XML en castellano. México.

Romo. (2003). *"Procesamiento Paralelo"* . Obtenido de Universidad Internacional del Ecuador: http://www.internacional.edu.ec/academica/informatica/creatividad/uide-bits/uide-bits-05-

Rose, M. (2008, 11 04). *TechTarget.* Obtenido de http://searchenterprisedesktop.techtarget.com/definition/Remote-Desktop-Protocol-RDP

Sabino, C. (2008, 02). *El proceso de Investigación.* Obtenido de https://metodoinvestigacion.files.wordpress.com/2008/02/el-proceso-de-investigacion_carlos-sabino.pdf

Scrum con TFS(Team Foundation Service). (2013, 01 23). Obtenido de http://fernandoescolar.github.io/2013/01/23/scrum-con-tfs/

Smaldone, J. (2004, 01 20). *Introducción a Secure Shell*. Obtenido de http://es.tldp.org/Tutoriales/doc-ssh-intro/introduccion_ssh-0.2.pdf

TechTarget. (2017). *Oracle*. Obtenido de http://searchoracle.techtarget.com/definition/Oracle

TechTarget. (2017). *Rlogin (Remote Login)*. Obtenido de http://searchnetworking.techtarget.com/definition/rlogin

The National Center for Biotechnology Information. (2011, 10 01). Obtenido de https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3243739/

The PostgreSQL Development Group. (2017). *PostgreSQL Documentation.* Obtenido de https://www.postgresql.org/files/documentation/pdf/9.0/postgresql-9.0-A4.pdf

Ubuntu. (2017). *The Story of Ubuntu*. Obtenido de https://www.ubuntu.com/about/about-ubuntu

Universidad Carlos III de Madrid. (2002). *Protocolo Telnet*. Obtenido de http://www.it.uc3m.es/~teldat/TeldatC/castellano/protocolos/Dm738v10_Protocolo_TELNET.PDF

Universidad de Alicante. (2016). *Modelo-Vista-Controlador*. Obtenido de https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html

Universidad de Cambridge. (2012). Obtenido de http://www.hep.phy.cam.ac.uk/vnc_docs/howitworks.html

Universidad de Chicago. (2017). *Beagle*. Obtenido de The CI supercomputer for medical simulations: http://beagle.ci.uchicago.edu/using-beagle/

Universidad de las Ciencias Informáticas. (2013, 04 04). Obtenido de https://computernewage.com/2013/04/04/primeros-pasos-con-la-terminal-de-linux/

Universidad del Pais Vasco. (2017). *Lenguaje HTML*. Obtenido de http://www.vc.ehu.es/jiwotvim/ISOFT2010-2011/Teoria/BloqueIV/HTML.pdf

UTIC, E. (2016). Computación de Alto Rendimiento.

Webucator. (2017). Obtenido de https://www.webucator.com/tutorial/learnphp/php-basics/how-php-works-reading.cfm

WinSCP. (2017). *WinSCP Documentation*. Obtenido de https://winscp.net/eng/docs/lang:es

WordPress. (2017). *WordPress Documentation*. Obtenido de https://wordpress.org/about/gpl/

World Wide Web Consortium . (2017). *Guia breve de CSS*. Obtenido de http://www.w3c.es/Divulgacion/GuiasBreves/HojasEstilo

Yii. (2017, 06 14). *The definitive guide of Yii*. Obtenido de http://www.yiiframework.com/doc-2.0/guide-intro-yii.html