



UNIVERSIDAD DEL BÍO-BÍO

SISTEMA DE GESTIÓN DE SIMULACIONES DE EFICIENCIA ENERGÉTICA A
TRAVÉS DE ENERGYPLUS, PARA EL LABORATORIO DE ESTUDIOS URBANOS
UBB.

PROYECTO DE TÍTULO PRESENTADO EN CONFORMIDAD A LOS REQUISITOS PARA
OBTENER EL TÍTULO DE INGENIERO CIVIL EN INFORMÁTICA

Alumno: Yoham Stuar González Cárdenas
Profesor Guía: Pedro Gerónimo Campos Soto

Julio de 2016

Resumen

Este proyecto se presenta para dar conformidad a los requisitos exigidos por la Universidad de Bío-Bío en el proceso de titulación para a la carrera de Ingeniería Civil en Informática.

El proyecto titulado “Sistema de gestión de simulaciones de eficiencia energética a través de EnergyPlus, para el Laboratorio de Estudios Urbanos (LEU) UBB” analiza distintas estrategias para gestionar peticiones de simulaciones de consumo energético en EnergyPlus (E+) generadas por trabajos e investigaciones externas, haciendo un uso eficiente de los recursos tecnológicos con los que cuenta el clúster del Laboratorio de Estudios Urbanos. Además contempla el análisis, diseño y construcción de una aplicación distribuida hecha en lenguaje Java para esos fines.

La principal contribución que este proyecto constituye es una solución concreta y medible a un problema real del Laboratorio de Estudios Urbanos. Así mismo, brinda una conexión entre otros trabajos desarrollados para el LEU y sus instalaciones.

Una vez finalizado el proceso de pruebas, se demuestra que, dadas las características del clúster LEU, sus configuraciones y la arquitectura del sistema desarrollado durante este proyecto, la mejor manera de distribuir la carga de simulación es 50-25-25, significando que para reducir el tiempo total que tarda el clúster en responder a las solicitudes de simulación es conveniente que el clúster resuelva un 50% de las simulaciones, mientras que cada nodo se encargue de un 25% de ellas.

Índice General

1. Generalidades	1
1.1 Introducción	1
1.2 Origen del tema y problemática	2
1.3 Objetivos del proyecto	2
Objetivo general	2
Objetivos específicos	3
1.4 Justificación	3
1.5 Descripción del ambiente	4
Servidor	4
Nodos	4
Computador de desarrollo	5
1.6 Límites	5
1.7 Trabajos similares	6
2 Marco teórico	7
2.1 Eficiencia energética	7
2.2 Sistemas de simulación energética	8
2.3 EnergyPlus	9
2.4 Otros sistemas de simulación	11
BLAST	11
ECOTECH	11
Energy Express	12
HAP	12
TAS	13
TRACE	13
2.5 Mosix	14
2.6 Socket de Java	15
2.7 TCP / IP y comunicaciones UDP / IP	16
La comunicación por datagramas	16
Comunicación por flujos (streams)	16
2.8 Otros conceptos clave	18
Programación concurrente	18
Procesos e hilos	19
Creación de hilos en Java	20
Una simulación ejemplo de E+	21
Simulación paramétrica	21

3	Clúster	22
3.1	El concepto de Clúster	22
3.2	Clasificación de Clúster computacionales	23
	Clúster de Alto Rendimiento	24
	Clúster de Alta Disponibilidad	25
	Clúster de Alta Eficiencia	25
3.3	Componentes de un clúster	26
	Nodos	26
	Almacenamiento	27
	Sistema Operativo	27
	Conexiones de Red.....	28
	Middleware	29
4	Definición del sistema	31
4.1	Propuesta	31
4.2	Ambiente de Ingeniería de Software	31
	Metodología.....	31
	Herramientas de apoyo	31
4.3	Siglas y abreviaciones	32
5	Análisis	33
5.1	Alternativas estudiadas	33
	Alternativa 1: MOSIX.....	33
	Alternativa 2: Aplicación cliente-servidor	35
	Conclusión de alternativas.....	35
5.2	Casos de uso	36
	Actores	36
	Diagramas	36
	Casos de uso resumidos.....	36
6	Diseño y construcción	39
6.1	Diseño arquitectónico	39
6.2	Diagramas de clases	39
6.3	Diseño interfaz de Usuario	45
7	Pruebas	47
7.1	Plan de pruebas	47
7.2	Objetivos del plan de pruebas	47
7.3	Elementos de pruebas	47
7.4	Especificación de las pruebas	47

7.5	Resultado de pruebas	48
8	Conclusiones y trabajos futuros.....	50
	Trabajo futuro	50
	Bibliografía	51

Índice de ilustraciones

Ilustración 1 - Arquitectura clúster LEU	5
Ilustración 2 - Diagrama de Casos de Uso	36
Ilustración 4 - Arquitectura cliente-servidor	39
Ilustración 5 - Clase Cliente	41
Ilustración 6 - Clase Servidor	42
Ilustración 7 - Clase HiloDeCliente	42
Ilustración 8 - Clase Calculos	43
Ilustración 9 - Clase SimpleZip	43
Ilustración 10 - Clases Proceso y ThreadManager	44
Ilustración 11 - Clase Nodo	44
Ilustración 12 - Interfaz Gráfica	45

Índice de tablas

Tabla 1 - Redes usadas en clúster	29
Tabla 2 - Caso de Uso: Conectarse al Clúster LEU	37
Tabla 3 - Caso de Uso: Suministrar archivos.....	37
Tabla 4 - Caso de Uso: Proveer parámetros.....	37
Tabla 5 - Caso de Uso: Recibir resultados.....	38
Tabla 6 - Valores del campo "Terrain"	46

1. Generalidades

1.1 Introducción

Las herramientas de simulación energética permiten establecer el consumo de energía de edificaciones a partir de modelos arquitectónicos detallados, tomando en cuenta información sobre equipamientos incluidos (como calefactores o aire acondicionado), clima, etc. De esta forma, es posible modificar los modelos buscando un funcionamiento más eficiente de las edificaciones, de forma previa a su construcción.

Algunos de los objetivos que persigue la simulación energética son la investigación de nuevas estrategias para la construcción en la obra analizada, la evaluación de opciones de diseño más convenientes, la verificación de que se cumpla con la normativa vigente y el análisis económico que determine el impacto de las medidas de conservación.

¿Y cómo se hace? Cabe aclarar primero que la simulación energética de edificios es un análisis que se recomienda comenzar en la etapa de anteproyecto, realizado por un experto en la materia. Los planos del edificio, a través de un modelo en 3D, son cargados tan minuciosamente como sea posible en un software (homologado por el USGB¹) junto a datos del manejo y flujo de energía. Con la colaboración de distintos algoritmos, junto a esta información, el sistema estudiará el comportamiento de diferentes variables. Estos cálculos facilitan el control del accionar futuro, considerando, por ejemplo, variables del clima, habitabilidad, los materiales que lo compondrán, temperatura, iluminación ideal, el aprovechamiento de la luz y ventilación natural, entre otros.

En este contexto surge EnergyPlus, un software de simulación, cuyo desarrollo es financiado por el DoE² de USA, usado en el clúster del Laboratorio de Estudios Urbanos de la Universidad. Los distintos equipos del clúster del LEU y su configuración se encuentran descritos en la quinta sección de este capítulo.

¹ U.S. Green Building Council, es una organización sin ánimo de lucro que promueve la sostenibilidad en el diseño, construcción y funcionamiento de los edificios en EE.UU.

² DoE United States Department of the Energy es el gabinete del gobierno de los Estados Unidos responsable de la política energética y de la seguridad nuclear

En este documento se exponen en detalle los distintos aspectos que forman parte del presente proyecto, que está dirigido a hacer más eficiente el proceso de simulación utilizando EnergyPlus en las instalaciones del LEU.

1.2 Origen del tema y problemática

En el Laboratorio de Estudios Urbanos (LEU) se realizan estudios de distribución espacial de fenómenos en el territorio, análisis de efectos de riesgos en el territorio, estudios de administración municipal y tecnologías de información y comunicaciones aplicadas a la ciudad, como sistemas de información geográfica y realidad virtual aplicada. En el último tiempo, además, se ha agregado una línea de investigación relativa al uso eficiente de energía en viviendas. Debido a esto, se encuentran en desarrollo algunas aplicaciones de software que requieren hacer uso intensivo de simulaciones de gasto energético.

Así, este proyecto surge como un apoyo a dicha línea de investigación, buscando dotar al LEU de un sistema que permita gestionar adecuadamente las peticiones de simulaciones que provengan de distintas aplicaciones y poderlas ejecutar de manera automática evitando el registro manual, que es lo que se hace actualmente. Permitiendo hacer un uso eficiente de los recursos computacionales existentes.

El problema detectado es que, a pesar de la infraestructura tecnológica del laboratorio, en que se cuenta con un clúster de computadores, se realizan ejecuciones de E+ una a la vez, siendo un desperdicio de recursos considerando que una ejecución puede tardar desde de unos diez segundos la más sencilla hasta unos 30 minutos las ejecuciones más complejas. Además, en muchas ocasiones es necesario realizar múltiples ejecuciones comparando diferentes variantes de un mismo modelo, con la consiguiente demora en la obtención de resultados del actual esquema.

A partir de la situación antes descrita, en este proyecto se busca mejorar el uso de la capacidad de cómputo instalada en el laboratorio, en base al cumplimiento de los siguientes objetivos.

1.3 Objetivos del proyecto

Objetivo general

El objetivo de este proyecto es desarrollar una plataforma que permita gestionar diversas peticiones de simulaciones de consumo energético en E+ generadas por otras aplicaciones,

haciendo un uso eficiente de los recursos tecnológicos con los que cuenta el clúster del LEU, lo que se traduce en menores tiempos de ejecución.

Objetivos específicos

- Establecer diferentes configuraciones factibles para la ejecución de simulaciones de E+ en el clúster LEU.
- Determinar la mejor configuración para reducir el tiempo de ejecuciones de E+.
- Análisis y diseño de los distintos módulos de un sistema de gestión de peticiones de simulación:
 - Módulo encargado distribución: balancea la carga de simulaciones entre los nodos del clúster y el servidor según la técnica estudiada.
 - Módulo Java-E+: comunicar la aplicación con el motor de cálculos de E+.
 - Módulo Otras Aplicaciones: conectarse a otras aplicaciones que requieran peticiones de E+
- Validar el sistema de gestión de simulaciones integrándose con al menos una de las herramientas desarrolladas previamente que requieren de simulación.

1.4 Justificación

El desarrollo de este proyecto es la instancia final para aplicar todos los conocimientos brindados tanto por la universidad como por la formación personal en diversas áreas involucradas directamente con el proyecto, por mencionar algunas, ingeniería y arquitectura de software, administración adecuada de recursos, la capacidad de comprender la problemática en una organización, formulación, evaluación e implementación de proyectos informáticos. Incluyendo otros conocimientos más avanzados, por ejemplo, administración de servidores Linux, sistemas distribuidos, paralelismo y concurrencia.

Una vez finalizado, el proyecto proveerá una solución concreta a un problema real y medible para el Laboratorio de Estudios Urbanos. De la misma manera, brinda una conexión entre otros trabajos desarrollados para el LEU y sus instalaciones, entre ellos un sistema para la generación de paquetes de eficiencia energética, que requiere de simulaciones de alternativa de diseño, desarrollada por Manuel Vásquez. De esta forma, el resultado del presente proyecto será una plataforma de gestión para las distintas solicitudes de simulación provenientes de otras aplicaciones preexistentes escritas en Java, PHP y Python.

1.5 Descripción del ambiente

A continuación se describen las características importantes para el desarrollo del proyecto.

El clúster LEU está formado por 5 equipos, el nodo principal (o servidor) y otros 4 equipos (nodos).

Servidor

Procesador: 2 Intel Xeon E5620 2.4Ghz, 12Mb de caché

Cantidad de Núcleos: 4 en cada procesador, con un total de 8 cores

Memoria RAM: 4 Módulos de 4GB cada uno, con un total de 16GB

Tarjeta de Video: Tarjeta Video NVIDIA QUADRO FERMI 600 LP 1GB, BLK

Tarjeta Video NVIDIA QUADRO NVS300 512MB, PCIE16X

Tarjeta de red: Integrada Gigabit Ethernet

Nodos

Cantidad de equipos iguales: 4

Procesador: AMD Phenom II x6 1075T 3GHZ, 9Mb de caché

Cantidad de Núcleos: 6

Memoria RAM: 8GB

Tarjeta de Video: Tarjeta Video NVIDIA QUADRO FERMI 2000/1GB

Tarjeta de red: Integrada Gigabit Ethernet

Todos los equipos cuentan con Debian Jessie, los nodos con la versión testing mientras que el servidor con la versión estable. Ver ilustración 1.

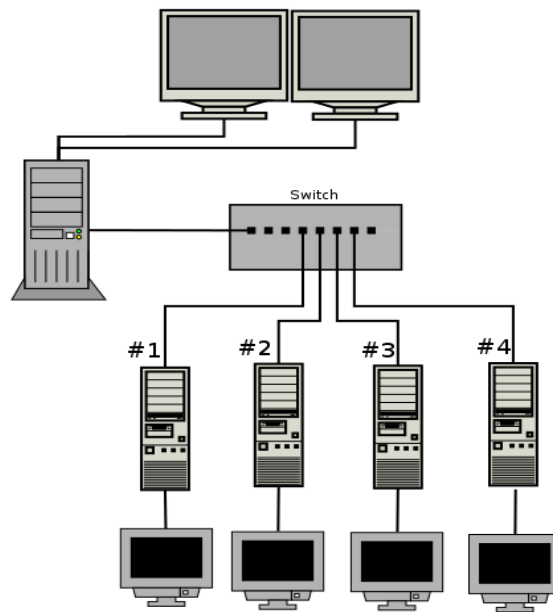


Ilustración 1 - Arquitectura clúster LEU

Computador de desarrollo

Este proyecto fue desarrollado en un laptop de las siguientes características:

HP Envy 15

Intel core i7

12 GB de RAM

Debian Jessie (64 bits)

Otro aspecto relevante es mencionar quiénes son los posibles usuarios del sistema. La aplicación desarrollada durante este proyecto puede ser utilizada por profesionales del área del Diseño y Construcción del LEU para realizar simulaciones de E+, por sus conocimientos en lo que a software de simulación energética se refiere.

1.6 Límites

Este proyecto contempla el desarrollo de una aplicación cliente-servidor escrita en el lenguaje de programación orientado a objetos Java que permita ejecutar simulaciones usando el motor de cálculo de EnergyPlus. También está considerada la interacción con otros sistemas

desarrollados usando PHP y/o Python. Del mismo modo, considera una breve investigación sobre los temas clave, por ejemplo, clúster y sistemas de simulación energética.

Este proyecto no atiende al análisis de la integridad de los archivos de entrada ni de salida de las ejecuciones.

1.7 Trabajos similares

El trabajo de Vásquez M [1], consistió principalmente en el desarrollo de un sistema software que permitiera la automatización al proceso de creación de paquetes de mejoramiento energético. Un paquete de mejoramiento es un conjunto de alternativas clasificadas y ordenadas por algún criterio, por ejemplo, económico o de eficiencia, cuyo objetivo es entregar una mejora de eficiencia energética a una vivienda en estudio.

A pesar de que ambos trabajos están relacionados con el mejor uso de la energía, la principal diferencia entre estos ellos es que Vásquez orientó su proyecto exclusivamente a la creación de paquetes para mejorar la eficiencia energética, mientras que en este proyecto se trabajará en la gestión de simulaciones.

En trabajos anteriores no se contaba con la tecnología ni los procedimientos necesarios para automatizar las simulaciones, por lo que se optó por el registro manual de toda la información necesaria para ejecutarla. El laboratorio conserva esta mecánica de trabajo, este trabajo permitirá que las simulaciones sean ejecutadas y gestionadas de manera automatizada.

El enfoque del trabajo realizado por Silva J y Venegas R [2] estuvo puesto siempre en preparar el mini clúster del LEU para el renderizado de imágenes usando tanto CPU como GPU. Por otro lado, este proyecto busca las configuraciones que faciliten el trabajo con EnergyPlus, lo que quiere decir que el enfoque anterior del clúster puede que no resulte 100% conveniente. Aunque, sin duda alguna, el trabajo de Silva y Venegas constituyó una pieza importante en el desarrollo de este proyecto al proveer una base de configuración y no comenzar desde cero el proceso de adaptación de los equipos.

2 Marco teórico

2.1 Eficiencia energética

Según la Agencia Chilena de Eficiencia Energética (ACEE) en su sitio web³ *“El uso eficiente de la energía es reducir la cantidad de energía eléctrica y de combustibles que utilizamos, pero conservando la calidad y el acceso a bienes y servicios”*. Muchas veces esta disminución es relacionada con mejoras tecnológicas en los aparatos o en los espacios habitables. Sin embargo, esto no es siempre cierto, debido a que la disminución en el consumo energético puede provenir también de un mejor manejo de recursos, incluso de cambios en el comportamiento.

Se diferencia además del ahorro energético, que podría implicar la disminución o incluso la abstinencia del desarrollo de ciertas actividades con el objetivo de eludir el consumo energético. Por citar un ejemplo, en el caso de no dejar las luces encendidas al dejar la habitación, estamos ante un ahorro de energía. Por otro lado, si reemplazamos las ampollitas incandescentes de la habitación por unas de bajo costo energético, es eficiencia energética dado que nos proporciona los mismos niveles de servicio usando menos energía.

Tampoco se debe confundir la eficiencia energética con la Energía Renovable (ER), que se refiere la energía que se obtiene de fuentes naturales virtualmente inagotables, ya sea por la inmensa cantidad de energía que contienen o por su capacidad de regenerarse por medios naturales, tales como el sol o el viento. En resumen, la ER es un tipo de fuente de energía, mientras que la eficiencia energética es un análisis de todo el sistema, que podrá presentar como medidas de reducción de consumo de energía, el uso de ER.

Al igual que con otras definiciones, la frontera entre el uso eficiente de la energía y la conservación de la energía puede ser difusa, pero ambos son importantes en términos ambientales y económicos. Este es especialmente el caso cuando las acciones se dirigen al ahorro de combustibles fósiles. [3]

³ <http://www.acee.cl/> - Sitio de la Asociación Chilena de EE

2.2 Sistemas de simulación energética

La historia de los sistemas de simulación energética puede ser relatada desde distintas ópticas; por ejemplo, según su uso (análisis, entrenamiento, investigación), tipos de modelo de simulación, lenguajes de programación o entornos y grupos de interés (manufactura, militares, transportación, etc.)

En los EE. UU., el desarrollo de software a gran escala vinculado con sistemas HVAC⁴ vio sus inicios durante los 60's, en el momento que proyectos del gobierno estadounidense necesitaron medir el ambiente térmico en los refugios antiatómicos mediante simulaciones horarias del proceso de transmisión de calor y humedad entre sus ocupantes humanos y sus murallas bajo condiciones de ventilación limitadas. Fueron las industrias del gas y la electricidad de la época quienes iniciaron las simulaciones basadas en cálculos horarios en general. Todo esto condujo a la formación del Grupo de Trabajo en Requerimientos Energéticos de la ASHRAE⁵ para desarrollar una simulación horaria completa del desempeño energético de edificios, también originó los trabajos de los APEC (Procedimientos Automatizados para Consultores de Ingeniería) para los cálculos en la carga en refrigeración. [4]

Aprovechando las capacidades de procesamiento y representación gráfica de los computadores de la época, emergió el cálculo energético sobre edificios. Actualmente se cuenta con una extensa variedad de software de simulación energética (tanto libre como comercial). El Departamento de Energía de los Estados Unidos mantiene desde 1996 un directorio⁶ de aproximadamente un centenar de herramientas computacionales, además de desarrollar algunas gratuitas, como Energy-Plus (originalmente denominado DOE), utilizado como motor de cálculo. En el directorio se encuentran descripciones y referencias de cada sistema, y varios están validados por las normas ISO 13790:2008-09⁷ y ASHRAE 90.1⁸. Algunos centros universitarios han elaborado programas de uso liberado, como ESP-r de la University of Strathclyde desde 1977, Trnsys desarrollado por la U. de Wisconsin desde 1978, Archisum de la U. Politécnica de Catalunya desde 1992 o Casanova de la U. Siegen desde

⁴ Heating, Ventilating and Air Conditioning, se refiere al concepto de climatización

⁵ La American Society of Heating, Refrigerating and Air-Conditioning Engineers, se dedica a la tecnología sustentable para la construcción de ambientes. Se concentra en la construcción de sistemas, eficiencia energética, calidad del aire interior, refrigeración y sustentabilidad.

⁶ <http://www.buildingenergysoftwaretools.com/> aquí puede ser consultado el directorio.

⁷ ISO. (2008). ISO 13790:2008-09: Esta norma brinda métodos de cálculo para la evaluación del consumo anual de energía para la calefacción y la refrigeración de un edificio, sea residencial o no.

⁸ ASHRAE. (2007). Standard 90.1. Esta norma proporciona los requerimientos mínimos para el diseño energéticamente eficiente de edificios con excepción de edificios residenciales de poca altura.

el 2000. Aunque la mayoría no mantiene una actualización regular o distribución amplia. Otros programas han surgido por legislaciones o certificaciones específicas, como LIDER [5] en España, CCTE [6] en Chile, CEPE⁹ en California y el PHPP [7] para el estándar CasaPasiva de Alemania. También desarrolladores de productos, instaladores o investigadores independientes han elaborado algunas herramientas. [8]

En un sistema de simulación energética los planos del edificio, son cargados a través de un modelo en 3D (ESP-r es un ejemplo), tan minuciosamente como sea posible en un software junto a datos del manejo y flujo de energía.

Varios sistemas se pueden relacionar entre sí, permitiendo realizar la modelación con un programa, luego el cálculo en otro, y la revisión de datos por separado. Promoviendo la transferencia de archivos y la utilización encadenada de programas.

Ciertamente el grueso de la labor fundamental en el núcleo computacional de las simulaciones se realizó hace más de 25 años, sin embargo, la simulación de edificios está en constante evolución y mejora. Los avances en la tecnología de la información han acelerado la adopción de herramientas de simulación debido a la rápida disminución en los costos de hardware y los avances entorno al desarrollo de herramientas de software. Todos esos desarrollos han contribuido a la proliferación y reconocimiento de la simulación como disciplina clave en el diseño de edificios y el proceso de operación de estos.

En este contexto aparece EnergyPlus, el sistema de simulación energética usado en el Laboratorio de Estudios Urbanos de la Universidad del Bío-Bío. A continuación se provee una breve descripción del software y sus principales características.

2.3 EnergyPlus

EnergyPlus es un programa de simulación de energía de todo el edificio, que diversos profesionales como ingenieros, arquitectos e investigadores utilizan para modelar tanto el consumo de energía --calefacción, refrigeración, ventilación, iluminación, cargas de proceso y no reguladas-- como el uso del agua en los edificios.

⁹ Certified Energy Plans Examiner (CEPE) Esta certificación indica que un individuo entiende el estándar energético para edificios de 2013 de la comisión de energía de California, USA.

EnergyPlus es un programa basado en consola que lee sus datos de entrada y escribe los resultados (salida) por medio de archivos de texto. Además utiliza la filosofía Garbage in, Garbage out (Entra basura, sale basura es una posible traducción) que se refiere al hecho de que la calidad de la salida depende de la calidad de la entrada de datos. Así, por ejemplo si un programa es alimentado con información incorrecta es poco probable que sus salidas sean útiles o siquiera informativas. [9] [10]

EnergyPlus está escrito en C++, corre como aplicación de consola tanto en Windows, Mac y Linux. Para correr una simulación estándar hacen falta como mínimo los siguientes archivos:

- EnergyPlus.exe (el archivo ejecutable, para Windows)
- Energy+.ini (descrito más abajo)
- Energy+.idd (el archivo de diccionario de entrada de datos *input data dictionary*)
- in.idf (el archivo de entrada)
- in.epw – opcional (el archivo de información del clima)

El archivo Energy+.ini es leído por el ejecutable. Es muy importante para la ejecución de la simulación, específicamente porque contiene un puntero hacia dónde será encontrado el archivo Energy+.idd.

INI

Este es el archivo de inicialización de EnergyPlus. Es un archivo de entrada en formato ASCII que permite al usuario especificar la ruta hacia el directorio que contiene al archivo Energy+.idd. No es necesario editar este archivo, salvo el caso que se mueva el archivo EnergyPlus.exe.

IDD

El input data dictionary (IDD) es un archivo ASCII (texto) que contiene una lista de todos los posible objetos de EnergyPlus y una especificación de la información que cada uno de esos objetos necesita.

IDF

El input data file (IDF) es un archivo ASCII que contiene la información que describe el edificio y el sistema HVAC que será simulado.

EPW

El EnergyPlus weather file es un archivo ASCII que contiene la información del clima horaria o sub-horaria requerida por el programa de simulación.

2.4 Otros sistemas de simulación

A la par de EnergyPlus, durante las últimas décadas se han desarrollado un sinnúmero de aplicaciones para el cálculo de simulaciones energéticas.

BLAST

www.bso.uiuc.edu/BLAST

El sistema Building Loads Analysis and System Thermodynamics (BLAST) predice el consumo, rendimiento y costo de los edificios. Contiene tres subprogramas principales: Space Loads Prediction, Air System Simulation, y Central Plant. El primero, calcula las cargas espaciales horarias dadas por datos meteorológicos y la construcción del edificio y detalles operativos utilizando un radiante, convección, y balance de calor conductivo para todas las superficies y un balance de calor del aire de la habitación. Ello incluye las cargas de transmisión, la carga solar, las ganancias de calor internas, las cargas de infiltración, y la estrategia de control de la temperatura utiliza para mantener la temperatura del espacio. BLAST se puede utilizar para investigar el desempeño energético de opciones de diseño tanto de edificios nuevos o modificados¹⁰ de casi cualquier tipo y tamaño.

ECOTECT

www.ecotect.com

Ecotect es una herramienta de diseño y de análisis arquitectónico altamente visual que vincula un modelador 3D completo con una amplia variedad de funciones de análisis de rendimiento cubriendo térmica, la energía, la iluminación, el sombreado, la acústica y los aspectos de costes. Mientras que sus capacidades de modelado y análisis pueden manejar la geometría de cualquier tamaño y complejidad, su principal ventaja es un enfoque en retroalimentación en las primeras etapas del proceso de diseño del edificio.

¹⁰ El texto original dice "retrofit buildings". Retrofitting es el proceso de modificar algo luego de su manufactura. En el caso puntual de los edificios, significa cambiar sus sistemas y estructura tras su construcción y ocupación iniciales.

Además del informe estándar basados en gráfico y en tablas, los resultados del análisis pueden ser mapeadas sobre superficies de edificios o se mostrados directamente dentro de los espacios. Esto incluye la visualización de los resultados del análisis espacial y volumétrico, incluidos los datos 3D CFD¹¹ importados. Animación de tiempo real se proporcionan junto con el trazado de acústica interactiva y rayos solares que se actualiza en tiempo real con los cambios en la geometría del edificio y las propiedades de los materiales.

Energy Express

www.ee.hearne.com.au

Energy Express es una herramienta de diseño, creado por la CSIRO¹², para estimar el consumo de energía y el coste en etapas de diseño. La interfaz de usuario permite una rápida y precisa creación y manipulación de modelos. Energy Express incluye un modelo de transferencia de calor multi-zona dinámico acoplado a un modelo HVAC integrado de manera que temperaturas de la zona se ven afectados por cualquier deficiencia HVAC.

Energy Express para arquitectos proporciona gráfico entrada de la geometría y la edición, visualización de informes múltiples, comparación de diseños alternativos y resultados, simplificado modelo de HVAC, y ayuda en línea detallada. Energy Express para Ingenieros proporciona estas capacidades junto con la estimación de la carga máxima, y detallado modelo de climatización, la edición gráfica de la manipulación de sistema de aire y planos térmicos de las instalaciones.

HAP

www.commercial.carrier.com

El Hourly Analysis Program (HAP) proporciona dos herramientas en un solo paquete: dimensionado de sistemas HVAC comerciales y la simulación del rendimiento energético del edificio por hora para estimar los costos y uso anuales de de la energía. Los datos de entrada y los resultados de los cálculos de diseño del sistema pueden ser utilizados directamente en los estudios de energía. HAP está diseñado para el ingeniero en ejercicio, facilita el eficiente trabajo del día a día de la estimación de cargas, el diseño de sistemas y evaluación del desempeño energético. Informes tabulares y de salida gráfica proporcionar tanto los

¹¹ Computational Fluid Dynamic, resuelve y analiza problemas sobre el flujo de sustancias.

¹² Commonwealth Scientific and Industrial Research Organisation (CSIRO) es la agencia del gobierno federal para la investigación científica en Australia con sede en Canberra y numerosos lugares alrededor de Australia

resúmenes e información detallada de sobre la construcción, el sistema y el rendimiento del equipo.

HAP es adecuado para una amplia gama de nuevos diseños y aplicaciones de modernización. Se proporcionan amplias funciones para la configuración y el control sistemas HVAC air-side y equipos terminales. Carga parcial los modelos de rendimiento se proporcionan para unidades DX divididas, DX unidades envasadas, bombas de calor, enfriadores y torres de refrigeración. Bucles hidráulicos se pueden simular con configuraciones primarias, secundarias y primarias solamente, usando una velocidad constante o bombas de velocidad variable.

TAS

www.edsl.net

TAS es una suite de productos de software, que simulan el rendimiento térmico dinámico de los edificios y sus sistemas. El módulo principal es el TAS Building Designer, que lleva a cabo la simulación dinámica del edificio con el flujo de aire natural y forzada integrada. Tiene una entrada de geometría basado en gráficos en 3D que incluye un enlace CAD. Tas Systems es un sistema HVAC/control del simulador, que puede ser directamente acoplado junto con el simulador de edificio. Realiza flujos de aire automático y dimensionado de plantas y la energía total demandada. El tercer módulo, TAS Ambiens, es un paquete robusto y fácil de usar de 2D CFD que produce una sección transversal de la variación microclima en un espacio.

TAS combina la simulación térmica dinámica de la estructura del edificio con ventilación natural cálculos que incluyen un control avanzado las funciones de apertura de abertura y la capacidad de simular sistemas de modo mixto complejos. El software tiene procedimientos de medida de planta de calentamiento y enfriamiento, que incluyen arranque óptimo. TAS cuenta con más 20 años de uso comercial en el Reino Unido y en todo el mundo.

TRACE

www.tranecds.com

TRACE se divide en cuatro fases de cálculo distintas: diseño, sistemas, equipos y economía. Durante la fase de diseño del programa primero calcula las ganancias de calor edificio para la conducción a través de las superficies de la construcción, así como las ganancias de calor de la gente, las luces, y electrodomésticos y el impacto de la ventilación y la infiltración. Por último, el programa de los tamaños de todas las bobinas y de tratamiento de aire sobre la base de estas cargas máximas. Durante la fase del sistema, la respuesta dinámica del edificio se simula para un año de 8760 horas (o reducido) mediante la combinación de perfiles de

carga de las habitaciones con las características del sistema de la zona de operaciones seleccionadas para predecir la carga impuesta sobre el equipo. La fase equipo utiliza las cargas horarias de la bobina fase de sistemas para determinar la forma en la refrigeración, calefacción, y equipos de movimiento de aire consumirá energía. La fase económica combina entradas económicas suministradas por el usuario con el uso de la energía de la fase equipo para calcular cada alternativa costo de servicios públicos, costo de instalación, coste de mantenimiento y vida coste del ciclo. [11]

2.5 Mosix

Mosix es un sistema de gestión que proporciona una imagen de única de sistema (SSI).

SSI

En computación distribuida, una Imagen Única de Sistema (Single System Image) es una propiedad de un sistema que oculta la naturaleza heterogénea y distribuida de los recursos, y los presenta a los usuarios y a las aplicaciones como un recurso computacional unificado y sencillo. SSI significa que el usuario tiene una visión global de los recursos disponibles independientemente del nodo al que están físicamente asociados esos recursos. Además, SSI puede asegurar que un sistema continuará funcionando después de algún fallo (alta disponibilidad) así como asegurar que el sistema se cargue uniformemente y se provea de multiprocesamiento común (gestión y planificación de recursos).

Las metas más importantes que se persiguen en el diseño de un SSI son, principalmente, la completa transparencia en la gestión de recursos, escalabilidad, y la capacidad de soportar aplicaciones de usuario.

Algunos ejemplos de SSI son:

- Genesis
- Kerrighed
- Mosix/OpenMosix
- TruCluster

MOSIX soporta tanto procesos concurrentes interactivos como trabajos por lotes. Incorpora el descubrimiento de recursos automático y la distribución dinámica de la carga de trabajo por

la migración preventiva de procesos. MOSIX se implementa como una capa de software que permite que las aplicaciones se ejecuten en nodos remotos como si se ejecutaran localmente.

Los usuarios pueden iniciar aplicaciones (secuenciales y paralelas) en un nodo, mientras que MOSIX busca de forma automática y transparentemente las ejecuta en otros nodos. No hay necesidad de modificar las aplicaciones, copiar archivos, iniciar sesión o asignar los procesos a los nodos remotos - todo se hace de forma automática. La asignación de aplicaciones a los nodos es supervisada por un amplio conjunto de algoritmos en línea que supervisan el estado de los recursos y tratan de mejorar el rendimiento global de la asignación dinámica de recursos, por ejemplo, balanceo de carga.

Una característica única de MOSIX es que opera en el nivel de proceso, a diferencia de los sistemas que operan en el nivel de trabajo (job-level). Esto significa que MOSIX se adapta y redistribuye la carga de trabajo cuando el número de procesos de trabajo y / o sus demandas cambian. Esto es especialmente útil para trabajos paralelos.

La última versión de MOSIX puede administrar clústeres y multi-cluster en nubes privadas. La gestión flexible permite a los propietarios de los diferentes grupos compartir sus recursos computacionales, al mismo tiempo conservar la autonomía de desconectar sus racimos en cualquier momento, sin interrumpir los programas que ya están corriendo.

MOSIX puede funcionar en entornos no virtualizados o Virtual Machine (VM). Un entorno no virtualizado requiere parchear el kernel de Linux y proporciona un mejor rendimiento, mientras que una máquina virtual puede ejecutarse sobre Linux o Windows sin modificar.

MOSIX es apropiado para ejecutar aplicaciones distribuidas y concurrentes de baja a moderada cantidad de E / S, tales como aplicaciones HPC. Es especialmente adecuado para: utilización eficiente de los recursos de todo el clúster; la ejecución de aplicaciones con requisitos de recursos impredecibles o tiempos de ejecución; ejecutar (y conservar) procesos largos y la combinación de nodos de diferentes velocidades. [22]

2.6 Socket de Java

Un socket es un punto final de un enlace de comunicación bidireccional entre dos programas que se ejecutan en la red. La clase *Socket* se utilizan para representar la conexión entre un programa cliente y un programa servidor. El paquete *java.net* proporciona dos clases -Socket

y ServerSocket- que implementan el lado del cliente de la conexión y el lado del servidor de la conexión, respectivamente. [13]

2.7 TCP / IP y comunicaciones UDP / IP

Hay dos protocolos de comunicación que se pueden utilizar para la programación de sockets: la comunicación por datagramas y la comunicación por flujos (streams).

La comunicación por datagramas

El protocolo de comunicación de datagramas, conocido como UDP (User Datagram Protocol), es un protocolo sin conexión, lo que significa que cada vez que se envían datagramas, también hay que enviar el descriptor de socket local y la dirección de un conector de recepción. Como se puede apreciar, los datos adicionales se deben enviar cada vez que se realiza una comunicación.

Comunicación por flujos (streams)

El protocolo de comunicación por flujos se conoce como TCP (transfer control protocol). A diferencia de UDP, TCP es un protocolo orientado a la conexión. Con el fin de hacer la comunicación a través del protocolo TCP, una conexión primero se debe establecer entre el par de sockets. Mientras que uno de los sockets espera una solicitud de conexión (servidor), y el otro pide una conexión (cliente). Una vez que dos sockets se han conectado, pueden ser utilizados para transmitir datos en ambas (o en cualquiera de las) direcciones.

Ahora se proveen las diferencias entre estos dos protocolos, para entender por qué escoger UDP o TCP.

En UDP cada vez que se envía un datagrama, se tiene que enviar el descriptor local y la dirección de un conector del socket receptor junto con él. Dado que TCP es un protocolo orientado a la conexión, por otro lado, una conexión debe ser establecida antes de que la comunicación entre el par de sockets comience. Así, hay un tiempo de setup de conexión en TCP.

En UDP, hay un límite de tamaño de 64 kilobytes del datagrama se puede enviar a un lugar determinado, mientras que en TCP no hay límite. Una vez que se establece una conexión, el par de sockets se comportan como flujos: todos los datos disponibles se leen inmediatamente en el mismo orden en que se reciben.

UDP es un protocolo no fiable - no hay ninguna garantía de que los datagramas que han sido enviados serán recibidos en el mismo orden por socket receptor. Por otro lado, TCP es un protocolo fiable; se garantiza que los paquetes que envíe serán recibidos en el orden en el que fueron enviados.

En resumen, TCP es útil para la implementación de servicios de red - como el inicio de sesión remoto (rlogin, telnet) y transferencia de archivos (FTP) - que requieren datos de longitud indefinida que se transferirán. [25]

Para los requerimientos de este proyecto, se utilizará el protocolo TCP porque se necesita un traspaso rápido y confiable de archivos. A continuación se muestra de manera general cómo se escribe una aplicación cliente-servidor usando TCP en Java. Primero el lado del servidor.

```
imports necesarios (como los paquetes java.io y java.net, por ejemplo)
clase Servidor{
int PUERTO = 1234 (representa el puerto por el cual se esperarán conexiones)
ServerSocket ss (Objeto de la clase ServerSocket necesario para la comunicación con
el lado del cliente)
Intentar(){ (crear un socket puede lanzar varias excepciones)
Mientras (siempre){ (opcional si se quiere que el servidor escuche todas las
peticiones que reciba mientras el programa esté en ejecución)
    ss = new ServerSocket(PUERTO) (instancia de ServerSocket)
    Socket conexión = ss.accept() (espera la conexión de un cliente)
    Creación de flujos de entrada y salida
    Toda la lógica que se requiera del servidor va aqui
}fin mientras
}fin intentar
}fin clase
```

Mientras que del lado del cliente se tiene lo que sigue.

```
imports necesarios (al igual que el servidor)
clase Cliente{
intentar{
int PUERTO = 1234 (representa el puerto por el cual se esperarán conexiones)
string HOST = "192.168.1.101" (representa la dirección IP de la máquina servidor, en
el caso de un enlace local "localhost" será suficiente)
Socket conexion = new Socket(PUERTO, HOST)
Creacion de flujos de entrada y salida
Logica de cliente (que usualmente es enviar datos y archivos)
    }fin bloque intentar
}fin clase
```

En este proyecto se presenta un cliente y varios servidores (el servidor LEU y sus nodos). El cliente es cualquier aplicación php, Python o web, que requiera simulaciones E+ y el servidor es el clúster LEU completamente. El computador principal y sus nodos son clientes y servidores a la vez, ya que se envían y reciben archivos en ambas direcciones.

Unas sencillas recomendaciones que abarcan la mayoría de los casos en los que se produce una excepción al crear un socket:

- Asegurarse de iniciar siempre la aplicación servidor primero.
- Asegurarse que el servidor acepta conexiones. (método accept).
- Asegurarse que ambos lados de la conexión estén utilizando el mismo número de puerto.

2.8 Otros conceptos clave

En esta sección se explican otros conceptos importantes para el desarrollo del proyecto.

Programación concurrente

En informática, la idea tras el desarrollo concurrente es lograr la ejecución solapada de varias tareas de manera que se realicen de manera paralela y al mismo tiempo. [15]

Un ejemplo podría ser un procesador de texto, que constantemente se encuentra revisando la ortografía del archivo que estamos editando, que nos permite al mismo tiempo imprimir otro

archivo, y muchas otras tareas. En este caso, se ejecuta un único programa, el procesador de texto, con su respectivo proceso, sin embargo, cada una de las funciones que posee (ortografía, imprimir, etc) posee un hilo de ejecución distinto que corre de manera concurrente y asíncrona.

Procesos e hilos

En la programación concurrente, existen dos unidades de ejecución: los procesos y los hilos. Un computador tiene muchos hilos y procesos activos. Incluso si tuviera un solo núcleo de ejecución. En este caso, el tiempo de procesador se comparte a través de procesos e hilos haciendo uso de una funcionalidad de Sistema Operativo llamada *time slicing*.

Cada vez es más común encontrar computadores con varios procesadores o con varios núcleos de ejecución. Esto mejora enormemente la capacidad de un sistema de ejecutar procesos e hilos concurrentemente. Sin embargo también es posible la concurrencia en sistemas más sencillos.

Un proceso, es ejecutar un programa en un procesador, lo que significa que es una unidad de ejecución que requiere recursos definidos como tiempo de procesador y memoria reservada. Usualmente, un programa cuenta con un único proceso, a menos que el programador codifique algo distinto.

Un ejemplo más claro es un programa c/c++ que use *fork()*¹³ donde dos procesos se ayudan para resolver el mismo problema sincronizándose y comunicándose, haciendo uso de semáforos y monitores en el primer caso, y segmentos de memoria compartida para el último.

Los hilos (a veces llamados procesos ligeros o hebras) al igual los procesos proveen un ambiente de ejecución, pero crear un hilo nuevo requiere menos recursos que crear un proceso nuevo.

Los hilos existen dentro de los procesos –cada proceso tiene al menos uno. Los hilos comparten los recursos del proceso, incluyendo memoria y archivos abiertos. Esto provee una eficiente y a la vez problemática comunicación.

¹³ La llamada a sistema *fork* se usa para crear nuevos procesos. Pertenece al estándar POSIX.

En Java, la ejecución de múltiples hilos es una de las características esenciales de la plataforma. Cada aplicación cuenta con al menos un hilo (o muchos, si es que se cuentan los hilos “system” que se encargan del manejo de memoria y señales). Pero, desde el punto de vista del programador, se comienza con sólo un hilo, llamado el *main thread*. Este hilo, es capaz de crear hilos adicionales como se ve más adelante. [15]

Creación de hilos en Java

En Java, existen dos maneras de ejecutar un hilo:

1. Implementar la interfaz Runnable

La interfaz Runnable define un único método (run) que contiene el código que se desea ejecute el hilo. El constructor de la clase Thread recibe un objeto Runnable como en el siguiente ejemplo.

```
public class HelloRunnable implements Runnable {

    public void run() {
        System.out.println("Hello from a thread!");
    }

    public static void main(String args[]) {
        (new Thread(new HelloRunnable())).start();
    }
}
```

2. Heredar de la clase Thread

La clase Thread implemente Runnable, aunque su método run no hace nada. Una aplicación que herede Thread debe proveer una implementación de run, como se aprecia en el siguiente trozo de código.

```
public class HelloThread extends Thread {

    public void run() {
        System.out.println("Hello from a thread!");
    }

    public static void main(String args[]) {
        (new HelloThread()).start();
    }
}
```

}

Ambos ejemplos pueden ser encontrados en el sitio de Oracle en [16].

Para la aplicación desarrollada en este proyecto se prefiere el primer método, ya que si en un futuro se desea alterar el esquema de herencias del sistema (añadiendo nuevas funcionalidades, por ejemplo) no haga falta reescribir todo y sea más fácil de mantener.

Una simulación ejemplo de E+

Para llevar a cabo una simulación E+ (como se menciona en el apartado EnergyPlus) hacen falta dos archivos: El archivo de entrada (.idf) que provee una descripción de la construcción del edificio y el archivo de clima (.epw) que describe el clima de la zona donde se encuentra ubicado el edificio.

En su versión para sistemas Windows, E+ cuenta con una interfaz de usuario sencilla que permite explorar el disco para buscar los archivos necesarios y controlar otros parámetros de la simulación, pero a causa de que estamos usando Debian (Linux), y lo que interesa es hacer muchas simulaciones de manera concurrente (no sólo una) esta opción no está disponible ni es útil.

Para realizar una simulación E+ bajo Linux basta usar uno de estos dos comandos:

```
energyplus -w archivo/de/clima/a/usar.epw archivodeEntrada.idf
```

O bien,

```
runenergyplus archivo/de/clima/a/usar.epw archivodeEntrada.idf
```

Al intentar realizar una simulación hay que recordar que ambos archivos deben encontrarse en carpetas con permiso de escritura, porque E+ va escribir sus reportes de salida en una carpeta llamada "Output" y la ubicará donde se encuentran estos archivos (particularmente donde está el IDF).

Para la aplicación se usará el segundo comando.

Simulación paramétrica

Para terminar este capítulo conviene aclarar el concepto de simulación paramétrica. Es ir editando el archivo de entrada y por cada edición generar una nueva simulación. Es decir, si vamos a editar el archivo de entrada con 30 combinaciones de valores distintos, se producirán en total 30 simulaciones distintas.

3 Clúster

3.1 El concepto de Clúster

Puede parecer sencillo referirse a este concepto, sin embargo nos enfrentamos a un problema, ni los expertos ni la gente que implementa clúster logra proveer una definición exacta que se ajuste a todos los casos.

Una posible definición puede ser:

“Un conjunto de máquinas unidas por una red de comunicación trabajando por un servicio conjunto. Según el servicio puede ser dar alta disponibilidad, alto rendimiento, etc...” [12]

Pero esta explicación es muy vaga. Digamos, por ejemplo, que tenemos dos consolas de videojuegos conectadas para jugar en red. ¿Es esto un clúster?

Con el objetivo de exponer la diversidad que existe en la materia se incluyen algunas definiciones de autoridades en el tema.

Un clúster consiste en un conjunto de máquinas y un servidor de clúster dedicado, para realizar los relativamente infrecuentes accesos a los recursos de otros procesos, se accede al servidor de clúster de cada grupo [13]

Un clúster es la variación de bajo precio de un multiprocesador masivamente paralelo (miles de procesadores, memoria distribuida, red de baja latencia), con las siguientes diferencias: cada nodo es una máquina quizás sin algo del hardware (monitor, teclado, mouse, etc.), el nodo podría ser SMP. Los nodos se conectan por una red de bajo precio como Ethernet o ATM aunque en clúster comerciales se pueden usar tecnologías de red propias. El interfaz de red no está muy acoplado al bus I/O. Todos los nodos tienen disco local. Cada nodo tiene un sistema operativo UNIX con una capa de software para soportar todas las características del clúster. [14]

Es una clase de arquitectura de computador paralelo que se basa en unir máquinas independientes cooperativas integradas por medio de redes de interconexión para proveer un sistema coordinado, capaz de procesar una carga. [15]

Un clúster debiera entregar una combinación de las siguientes propiedades [16]:

- Alto Rendimiento
- Alta Disponibilidad
- Balanceo de carga
- Escalabilidad

3.2 Clasificación de Clúster computacionales

Así como no hay certeza en el concepto, es difícil también dar en el clavo en lo que se refiere a su clasificación. A continuación se intenta dar claridad en este tema.

Según el Dr. Rajkumar Buyya en [17], hay seis criterios distintos para clasificar un clúster.

Propiedad de nodo (Node Ownership)

Clúster dedicado (Dedicated Clusters)

Clúster no dedicado (Non-Dedicated Clusters)

Hardware del Nodo

Clúster de PC's

Clúster de Workstations

Clúster de SMP's

Sistema operativo del nodo

Linux

Solaris

NT

AIX

SCO/Compaq

Digital VMS

HP-UX

Microsoft Wolfpack

Configuración del Nodo

Clúster Homogéneo

Todos los nodos tendrán arquitecturas similares y correrán el mismo SO

Clúster Heterogéneo

Todos los nodos tendrán diferentes arquitecturas y diferente SO

Niveles de Clustering

Group Clusters (#nodos: 2-99)

Departmental Clusters (#nodos: 10 a 100)

Organizational Clusters (#nodos: muchos 100s)

National Metacomputers (Wan/basados en internet)

International Metacomputers (Basados en internet, #nodos 1000 a varios millones)

Application target

High Performance Clusters (Clústers de Alto Rendimiento)

Grandes aplicaciones exigentes

High Availability Clusters (Clústers de Alta Disponibilidad)

Aplicaciones críticas para misión (Mission Critical Applications)

High Throughput Clusters (Clústers de Alta Eficiencia)

En esta última clasificación se centra el desarrollo de este capítulo.

Clúster de Alto Rendimiento

Es un conjunto de computadores diseñado para dar altas prestaciones en capacidad de cálculo, ejecutar tareas que requieren de gran capacidad computacional, grandes cantidades de memoria, o ambos a la vez. El cumplir estas tareas puede comprometer los recursos del clúster por largos periodos de tiempo. Los motivos para utilizar un clúster de alto rendimiento son el tamaño del problema por resolver y el precio de la máquina necesaria para resolverlo.

Para garantizar esta capacidad de cálculo, los problemas necesitan ser paralelizados, ya que el método con el que los clúster agilizan el procesamiento es dividir el problema en problemas más pequeños y calcularlos en los nodos, por lo tanto, si el problema no cumple con esta característica, no puede utilizarse el clúster para su cálculo.

Para resolver un problema paralelizado se debe hacer uso de bibliotecas especiales.

Clúster de Alta Disponibilidad

El principal objetivo de este tipo de clúster es el de proveer disponibilidad y confiabilidad. Intenta que los servicios que brindan se encuentren utilizables el mayor tiempo posible. Hace uso de software que detecta y se recupera de fallos, mientras que su hardware se evita tener un único punto de fallos¹⁴.

Es posible clasificarlos en:

Alta disponibilidad de infraestructura: Si llega a fallar el hardware en algún nodo del clúster, su software es capaz de levantar sus servicios en otras máquinas y una vez recuperada del fallo se le devuelven sus servicios. Esta característica garantiza la alta disponibilidad en los servicios que provee el clúster.

Alta disponibilidad de aplicación: Si se produce un fallo del hardware o de las aplicaciones de alguno de los nodos del clúster, el software de alta disponibilidad es capaz de arrancar automáticamente los servicios que han fallado en cualquiera de las otras máquinas y cuando la máquina que ha fallado se recupera, los servicios son nuevamente migrados a la máquina original. Esta capacidad de recuperación automática de servicios nos garantiza la integridad de la información, ya que no hay pérdida de datos. [16]

No hay que confundir un clúster de alto rendimiento con un clúster de alta disponibilidad. El primero es una configuración de equipos diseñado para proporcionar capacidades de cálculo mayores que la que proporcionan los equipos individuales, mientras que el segundo está diseñado para garantizar el funcionamiento ininterrumpido de ciertas aplicaciones.

Clúster de Alta Eficiencia

Su objetivo es ejecutar la mayor cantidad de tareas en el menor tiempo posible. Existe independencia de datos entre las tareas individuales. El retardo entre los nodos del clúster no es considerado un gran problema.

¹⁴ Punto único de fallos: es un componente de un sistema que tras un fallo en su funcionamiento ocasiona un fallo global en el sistema completo, dejándolo inoperante.

Este tipo de clúster es también se pueden separar en Clúster Comerciales (Alta disponibilidad, Alta eficiencia) y Clúster Científicos (Alto rendimiento).

Aunque existen diferencias en los requerimientos de las aplicaciones, muchas de las características de las arquitecturas de hardware y software en los clúster son las mismas. Más aún, un clúster de determinado tipo, puede también presentar características de los otros.

El objetivo que persigue el clúster LEU-UBB es optimizar el tiempo de ejecución de aplicaciones, por lo que para este proyecto, el clúster cae en la categoría de Clúster de Alta Eficiencia

3.3 Componentes de un clúster

Varios elementos de software y hardware forman parte de un clúster, se destacan:

- Nodos
- Almacenamiento
- Sistemas Operativos
- Conexiones de Red
- Middleware
- Protocolos de Comunicación y servicios
- Aplicaciones
- Ambientes de Programación Paralela

Nodos

Pueden ser simples computadores, sistemas multi-procesador o estaciones de trabajo (workstations). Generalmente, un nodo es un punto de intersección o de unión de varios elementos que confluyen en el mismo lugar. Sin embargo, nodo puede referirse a conceptos diferentes, aquí se concepto que se maneja en redes: cada una de las máquinas es un nodo, y si la red es Internet, cada servidor constituye también un nodo.

El clúster puede estar conformado por nodos dedicados o por nodos no dedicados. La diferencia entre estos conceptos radica en que un nodo dedicado es usado en tareas estrictamente relacionadas con el clúster, por lo que no cuenta con teclado, mouse ni monitos. En cambio, los nodos no dedicados cuentan con estos dispositivos y el clúster lo usa sólo cuando no existen ciclos de reloj que no están siendo utilizados.

Cabe aclarar que a la hora de diseñar un clúster conviene que los nodos tengan características similares, es decir, deben guardar cierta similitud de arquitectura y sistemas operativos, ya que si se conforma un clúster con nodos totalmente heterogéneos (existe una diferencia grande entre capacidad de procesadores, memoria, disco duro) será ineficiente debido a que el Middleware delegará o asignará todos los procesos al nodo de mayor capacidad de cómputo y solo distribuirá cuando este se encuentre saturado de procesos, por eso es recomendable construir un grupo de computadores lo más similares posible.

Almacenamiento

El almacenamiento puede consistir en una NAS¹⁵, una SAN¹⁶, o almacenamiento interno en el servidor. El protocolo más comúnmente utilizado es NFS (Network File System), sistema de ficheros compartido entre servidor y los nodos. Sin embargo existen sistemas de ficheros específicos para clusters como Lustre (CFS) y PVFS2.

Tecnologías en el soporte del almacenamiento en discos duros:

- IDE o ATA: velocidades de 33, 66, 100, 133 y 166 MB/s
- SATA: velocidades de 150, 300 y 600 MB/s
- SCSI: velocidades de 160, 320, 640 MB/s. Proporciona altos rendimientos.
- SAS: aún a SATA-II y SCSI. Velocidades de 300 y 600 MB/s
- Unidades de cinta (DLT) son utilizadas para copias de seguridad por su bajo coste.

Sistema Operativo

¹⁵ NAS (Network Attached Storage) es un dispositivo específico dedicado al almacenamiento a través de red (normalmente TCP/IP) que hace uso de un sistema operativo optimizado para dar acceso a través de protocolos CIFS, NFS, FTP o TFTP.

¹⁶ DAS (Direct Attached Storage) consiste en conectar unidades externas de almacenamiento SCSI o una SAN (Storage Area Network) a través de un canal de fibra.

Un sistema operativo para clúster debe ser multiproceso y multiusuario. Otras características deseables son la facilidad de uso y acceso. Un sistema operativo es un programa o conjunto de programas de computadora destinado a permitir una gestión eficaz de sus recursos.

Algunos de los que podemos encontrar son:

- GNU/Linux
 - ABC GNU/Linux2
 - OpenMosix
 - Rocks3
 - Kerrighed
 - Cóndor
 - Sun Grid Engine
- Unix
 - Solaris
 - HP-UX
 - Aix
- Windows
 - NT
 - 2000 Server
 - 2003 Server
 - 2008 Server
- Mac OS X
 - Xgrid
- Solaris
- FreeBSD

Conexiones de Red

Los nodos de un cluster pueden conectarse mediante una simple red Ethernet con placas comunes (adaptadores de red o NICs), o utilizarse tecnologías especiales de alta velocidad como Fast Ethernet, Gigabit Ethernet, Myrinet, InfiniBand, SCI, etc.

Se provee en la siguiente tabla, un resumen de las principales características de las redes más utilizadas para clúster.

Red	Características	Latencia (en microsegundos)	Ancho de Banda (en Gbit/s)
Ethernet	-Coste relativamente bajo -Red de administración por excelencia	-30 a 100 dependiendo del protocolo utilizado	-1 (Gigabit Ethernet) -10 (10 Gigabit Ethernet)
Myrinet	-Alto rendimiento -Procesamiento por chip integrado	-3	-10
InfiniBand	-Escalable, eficiente y brinda altos anchos de banda	-10	-2 a 96 (dependiendo del uso de canales)
SCI	-Puede brindar transparencia al usuario -Adecuada para clúster pequeño o mediano	-1,3	-0,6

Tabla 1 - Redes usadas en clúster

Middleware

La palabra "Middleware" se ha utilizado para describir una amplia variedad de productos de software. Esto puede causar un problema para entender exactamente que es el middleware. En la literatura, hay muchas definiciones diferentes de middleware que varían desde capa de software entre el sistema operativo (y / o la red) y aplicación [18] para "pegar" dos aplicaciones [19]. También ha sido descrito como una importante herramienta de integración [20] que permite la conexión modular para el software distribuido [21].

Una posible definición para middleware puede ser: el middleware es el software que ayuda a una aplicación a interactuar o comunicarse con otras aplicaciones, redes, hardware y/o sistemas operativos. Este software ayuda a los programadores, relevándolos de complejas conexiones necesarias en sistemas distribuidos. También provee herramientas para mejorar la calidad de servicio, la seguridad, el paso de mensajes, servicio de archivos y directorios, etc., que pueden ser invisibles para el usuario. Esta definición describe a la mayor parte de los productos de middleware disponibles actualmente.

Otras de las funciones o prestaciones del middleware se pueden destacar el balanceo de carga y la escalabilidad.

El balanceo de carga es optimizar el funcionamiento de un sistema usando la migración de procesos para transferir los procesos de un saturado a otro que esté ocioso. Escalabilidad es poder detectar nodos nuevos conectados al sistema y utilizarlos.

4 Definición del sistema

4.1 Propuesta

Lo que se propone en este Proyecto de Título es investigar e implementar una manera de minimizar los tiempos de ejecución de las simulaciones de E+ realizadas en el Laboratorio de Estudios Urbanos, aprovechar los recursos tecnológicos con los que se dispone y unir las peticiones de simulación gestionando de manera eficiente mediante una aplicación cliente escrita en Java.

Como solución se propone el desarrollo de una aplicación Java que permita gestionar simulaciones independientes de E+ de manera automática, reduzca el tiempo que tardan estas ejecuciones en el LEU y se conecte a otras aplicaciones que soliciten ejecuciones de simulación y se alimenten del resultado de dichas simulaciones.

4.2 Ambiente de Ingeniería de Software

Metodología

Considerando el tamaño del proyecto y las condiciones en las que se desarrolla, esto es, requerimientos que pueden cambiar y detección de nuevas necesidades, se propone utilizar la metodología Iterativa/incremental. Su sencillo ciclo de vida permitirá mantener el control del proyecto.

Herramientas de apoyo

Entre las herramientas y/o software necesarias para este proyecto se destacan:

- NetBeans IDE en su versión 8.1 necesario para codificar la aplicación en el lenguaje Java

- EnergyPlus en su versión 8.4, motor de cálculo. Es la aplicación encargada de las simulaciones basado en las construcciones de los edificios y el clima que los rodea.

4.3 Siglas y abreviaciones

E+: EnergyPlus, motor de cálculo para simulaciones energéticas

JVM: Java Virtual Machine

UML: Unified Modeling Language

5 Análisis

5.1 Alternativas estudiadas

Durante del desarrollo de este proyecto se manejan dos alternativas, por un lado el uso del middleware MOSIX que realiza migración de procesos (distribuir los procesos de un nodo saturado a otro que esté desocupado). Mientras que la otra alternativa consiste en el desarrollo de una nueva aplicación cliente-servidor escrita en Java que asegure el correcto parametrizado de archivos y su ejecución en paralelo.

Alternativa 1: MOSIX

La primera alternativa que se estudió para este proyecto se trata de un middleware que se encargue del balanceo de carga y que distribuya los procesos entre las distintas máquinas del mini clúster: MOSIX. Esta alternativa ya se encuentra implementada en el Laboratorio. A continuación se describen las principales características de MOSIX.

Mosix es un sistema de manejo de clúster Linux que provee SSI. Además de proveer manejo de multi-clúster tipo nube. Incluye descubrimiento proactivo de recursos y distribución de carga de trabajo para una asignación adecuada de proceso a través de los nodos disponibles.

Mosix está implementado como una capa de software que provee a sus usuarios y aplicaciones un ambiente de ejecución Linux sin modificar. Como resultado, los usuarios no necesitan modificar sus aplicaciones o registrarte en los nodos remotos.

A pesar de lo conveniente que puedan sonar todos estos antecedentes, MOSIX no es una herramienta útil para el desarrollo de este proyecto. Extraído directamente desde la “Administrator’s, User’s and Programmer’s Guides and Manuals” en el Capítulo 9: “What can MOSIX do for me”:

“Some programs are not suitable for MOSIX, including programs that use shared-memory or threads (threads use shared-memory) and some system/administration-programs.”

Algunos programas no son apropiados para MOSIX, incluyendo programas que usan memoria compartida o hilos (los hilos usan memoria compartida) y algunos programas de sistema y administración.

En su sitio de preguntas frecuentes [24], los desarrolladores de MOSIX declaran:

¿Por qué la memoria compartida no está soportada?

Porque es imposible cambiar el contenido de la memoria en un nodo y esperar que el mismo cambio se reflejará inmediatamente en la memoria de los otros nodos (con el que se comparte memoria).

¿Puedo ejecutar aplicaciones con hilos?

No, las aplicaciones con hilos son creadas con la llamada al sistema "CLONE_VM" que utiliza memoria compartida, motivo por el cual no pueden ejecutarse bajo MOSIX.

Como el objetivo principal de este proyecto es el desarrollo de un sistema gestor de simulaciones, que permita reducir los tiempos de simulaciones paramétricas (múltiples simulaciones editando el archivo de entrada) es pertinente la construcción de una aplicación multihilos. El enfoque utilizado para crear estas simulaciones es hacer uso de la clase Process. Como se detalla a continuación:

```
try {  
    Process p = Runtime.getRuntime().exec(comando);  
} catch (IOException e) {  
    System.out.println(e.getMessage());  
}
```

El método exec de la clase Process ejecuta el comando que recibe en forma de string, creando un nuevo proceso. Sin permitir controlar la cantidad de hilos a utilizar.

Otra versión de este método recibe otros parámetros, como un arreglo de strings que sirven para establecer un conjunto de variables de entorno y un objeto File para establecer dónde ejecutar el comando dado.

En consecuencia, como el desarrollo de la aplicación es obligadamente mutihilo, y además no hay control sobre cuantos hilos se ejecutan por comando, no conviene usar Mosix en este proyecto. Sin dejar de lado que al momento de intentar utilizar E+ desde Mosix, nos lanzará error debido a que E+ utiliza justamente memoria compartida, característica con la que Mosix no es compatible.

Alternativa 2: Aplicación cliente-servidor

Otra opción es el desarrollo de una aplicación en Java que se encargue de parametrizar ejecuciones de E+ y ejecutarlas en paralelo.

Analizar la relación costo/beneficio de esta alternativa no resulta tan complicado. Porque se mantiene completo control sobre el funcionamiento de la aplicación, al saber cómo se comportan cada uno de sus componentes. Sin dejar de lado que se trata de un sistema de tamaño reducido. Además, no hace falta invertir tiempo en dominar nuevas tecnologías.

Tiempo que se invirtió, por ejemplo, en aprender el correcto uso de MOSIX y documentarse correctamente.

Esta aplicación debe ser capaz de:

- ser operada remotamente
- recibir solicitudes no solo desde aplicaciones Java (sino también python/php/etc)
- realizar ejecuciones parametrizadas de E+
 - correr todas las ejecuciones paramétricas simultáneamente.
- recibir archivos IDF y EPW
- retornar los archivos que resultaron de los cálculos previos
- mantener un registro de las ejecuciones que ha realizado.

Conclusión de alternativas

Por las razones expuestas anteriormente se escoge el desarrollo de una aplicación ajustada a las necesidades de este proyecto, cuyo diseño y construcción serán exploradas en las siguientes secciones de este capítulo y el siguiente.

5.2 Casos de uso

Actores

Sólo se observa un actor, y en este caso es el usuario del software, quien puede ser un académico, un profesional, o cualquier otro usuario con conocimientos mínimos sobre el contexto de las simulaciones con E+.

Diagramas

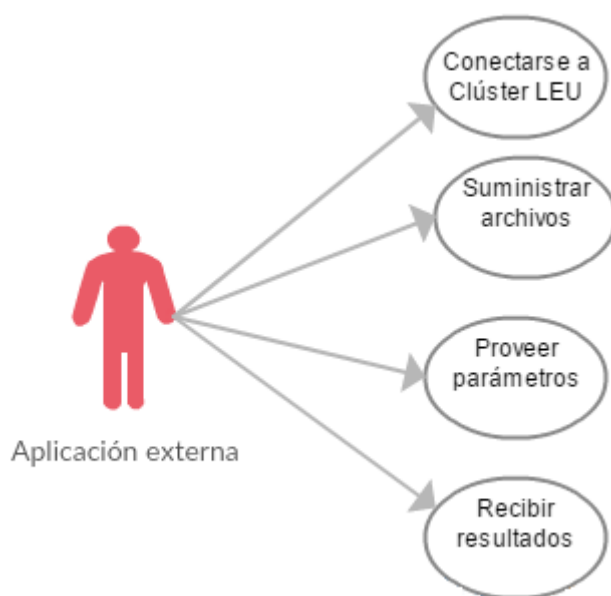


Ilustración 2 - Diagrama de Casos de Uso

Las interacciones del sistema están limitadas al envío y recepción tanto de archivos como de parámetros necesarios para atender las solicitudes de simulación entre el sistema y una aplicación externa.

Casos de uso resumidos

Caso de uso	Conectarse al Clúster LEU
Resumen	Este caso de uso permite una aplicación externa realizar una conexión de red mediante sockets TCP con el Clúster del Laboratorio de Estudios Urbanos UBB.
Precondiciones	N.A.

Postcondiciones	El sistema ya conoce a qué cliente va a atender.
------------------------	--

Tabla 2 - Caso de Uso: Conectarse al Clúster LEU

Caso de uso	Suministrar archivos
Resumen	Este caso de uso permite a una aplicación externa enviar los archivos necesarios para una simulación de E+ (archivo de entrada y archivo de clima) mediante sockets TCP.
Precondiciones	Caso de uso "Conectarse al Clúster LEU"
Postcondiciones	El sistema ya sabe sobre qué archivos trabajar.

Tabla 3 - Caso de Uso: Suministrar archivos

Caso de uso	Proveer parámetros
Resumen	Este caso de uso permite a una aplicación externa indicarle al sistema qué parámetros han de ser probados mediante un flujo UTF.
Precondiciones	Caso de uso "Suministrar archivos"
Postcondiciones	Sistema listo para simulación paramétrica

Tabla 4 – Caso de Uso: Proveer parámetros

Caso de uso	Recibir resultados
Resumen	Este caso de uso permite a una aplicación externa recibir todos los resultados de simulación generados por E+ en el clúster LEU.
Precondiciones	Caso de uso "Proveer parámetros"
Postcondiciones	Cliente atendido con sus respectivos

	archivos de resultados recibidos.
--	-----------------------------------

Tabla 5 - Caso de Uso: Recibir resultados

6 Diseño y construcción

6.1 Diseño arquitectónico

Para el desarrollo de este proyecto se trabaja con el modelo de diseño Cliente-Servidor. La arquitectura cliente-servidor es un modelo de aplicación distribuida en el que las tareas se reparten entre los proveedores de recursos o servicios, llamados servidores, y los demandantes, llamados clientes. Un cliente realiza peticiones a otro programa, el servidor, quien le da respuesta. Esta idea también se puede aplicar a programas que se ejecutan sobre una sola computadora, aunque es más ventajosa en un sistema operativo multiusuario distribuido a través de una red de computadoras. Lo que hace necesario el uso de sockets.

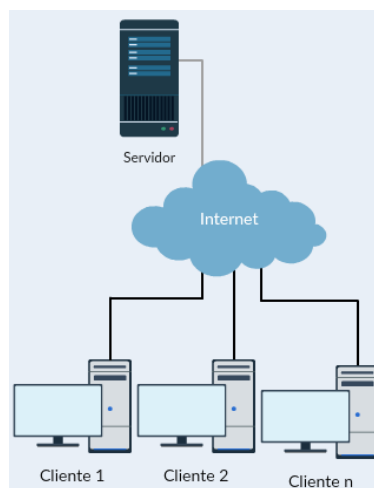


Ilustración 3 - Arquitectura cliente-servidor

6.2 Diagramas de clases

A continuación se presenta el diagrama de clases que describe la implementación de este proyecto.

Para el desarrollo de este proyecto se consideran tres perfiles; cliente, servidor y nodo, que se especifican en seguida.

Cliente: encargado de realizar las peticiones de simulación al clúster, proporcionando los archivos base, parámetros y valores necesarios.

Servidor: quien recibe tanto las peticiones como los archivos, los distribuye en el clúster, prepara archivos y transmite el resultado de todo el trabajo de simulación al cliente.

Nodo: visto como unidad, representa al resto de unidades de cómputo del clúster cuya función es realizar simulaciones en conjunto con el servidor y otros nodos. Además envía los resultados de sus simulaciones al servidor.

Desde la perspectiva del cliente se presenta el siguiente diagrama.

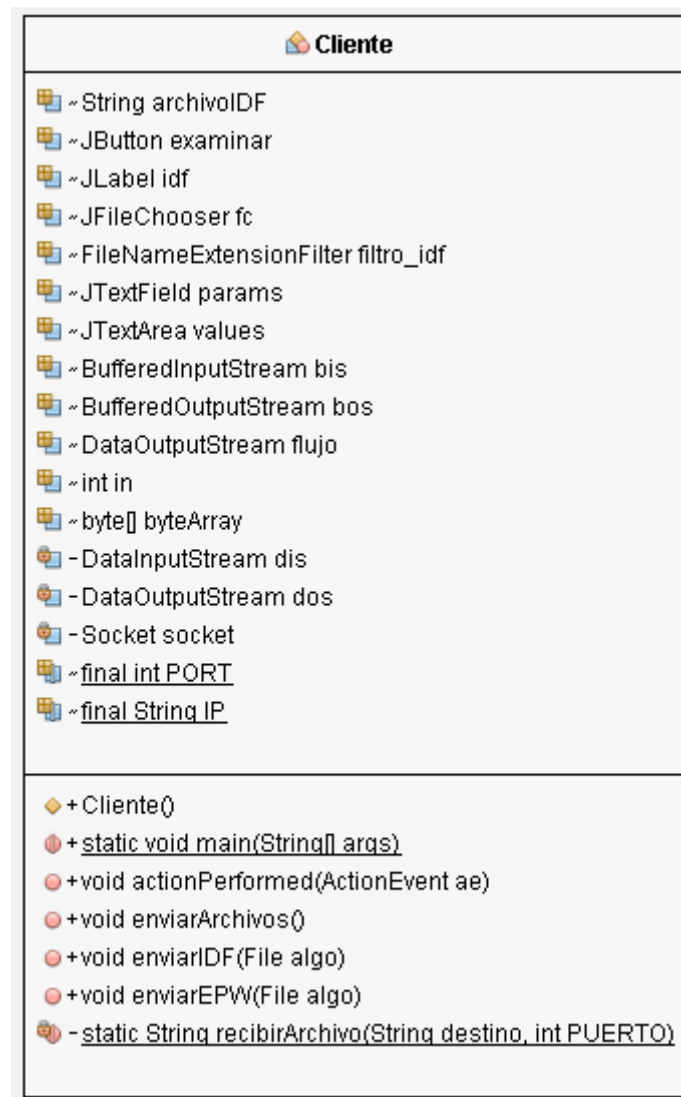


Ilustración 4 - Clase Cliente

Esta clase se encarga de transmitir tanto los archivos necesarios para una simulación como los valores y los parámetros indispensables para su correcta realización, además de abrir una conexión socket para la recepción de todas las simulaciones hechas por el clúster.

Del lado del servidor se presenta el siguiente diagrama

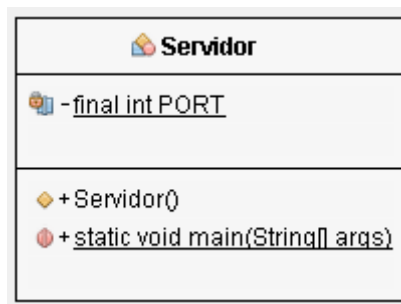


Ilustración 5 - Clase Servidor

La clase Servidor se usa para que por cada cliente que requiera una conexión al clúster, se cree una nueva instancia de HiloDeCliente, que es donde realmente se realiza todo el procesamiento de archivos.

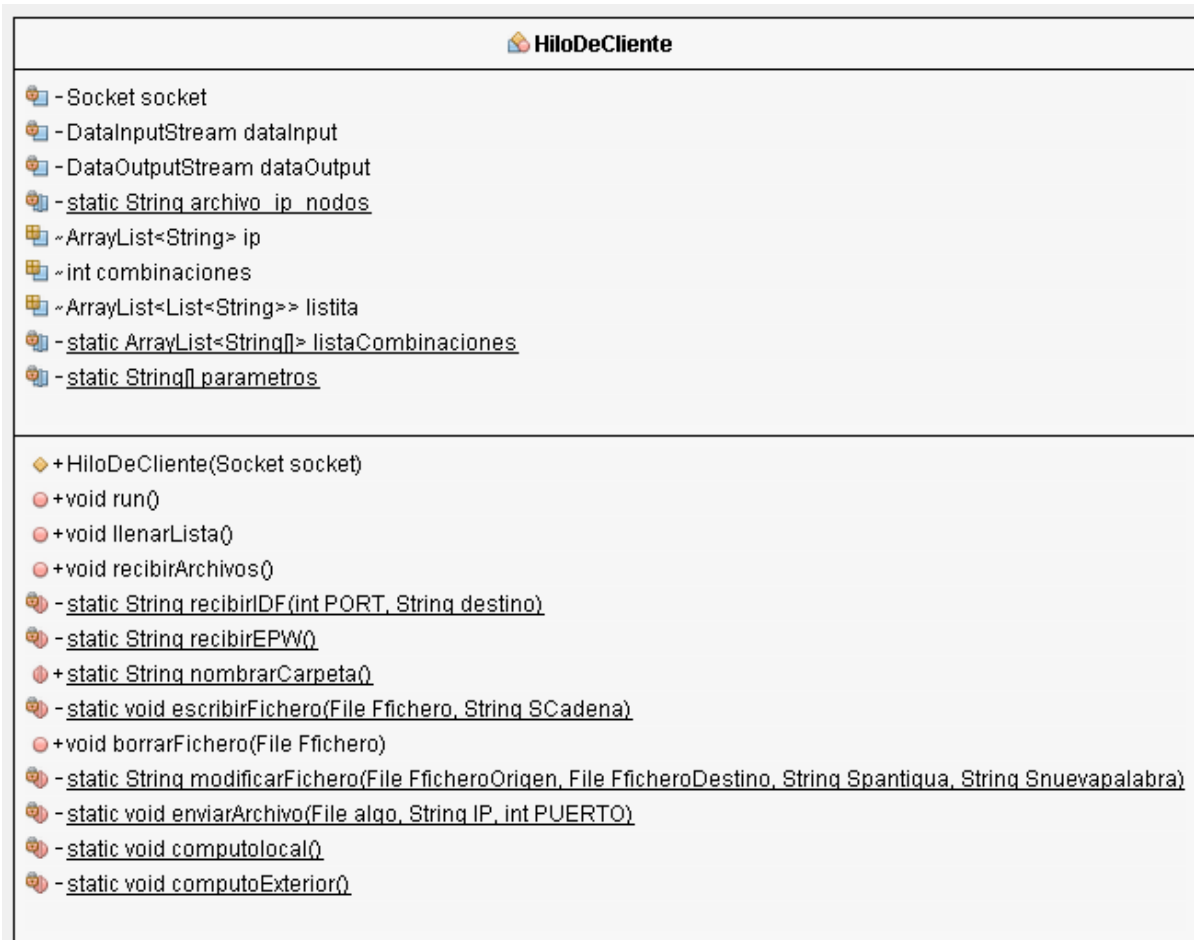


Ilustración 6 - Clase HiloDeCliente

Esta clase es la responsable de la transmisión de información entre cliente-servidor y servidor-nodos. Recibe/Envía archivos y además realiza parte del trabajo de simulaciones a nivel local.

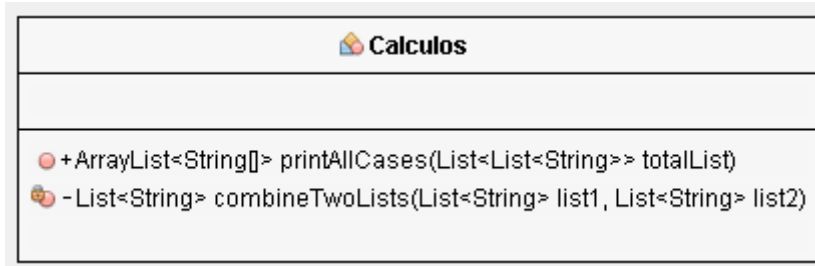


Ilustración 7 - Clase Calculos

Esta clase es utilizada para crear el espacio muestral compuesto por todas las combinaciones entre los distintos valores de los parámetros que envió el cliente. Por ejemplo, si los parámetros son P1 = 'parametro1' y P2 = 'parametro2' y sus valores son P1 = {"blanco", "negro"} y P2 = {"alto", "medio", "bajo"}. El espacio muestral formado por las combinaciones sería:

- Combinación 1: {blanco, alto}
- Combinación 2: {blanco, medio}
- Combinación 3: {blanco, bajo}
- Combinación 4: {negro, alto}
- Combinación 5: {negro, medio}
- Combinación 6: {negro, bajo}

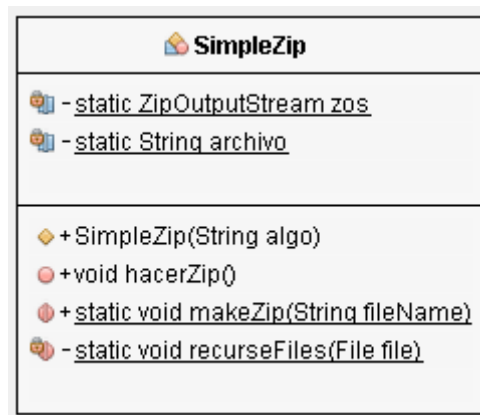


Ilustración 8 - Clase SimpleZip

Esta clase se encarga de tomar la carpeta que contiene todos los resultados de las simulaciones del clúster y comprimirlas en un único archivo zip a fin de facilitar su envío al cliente.

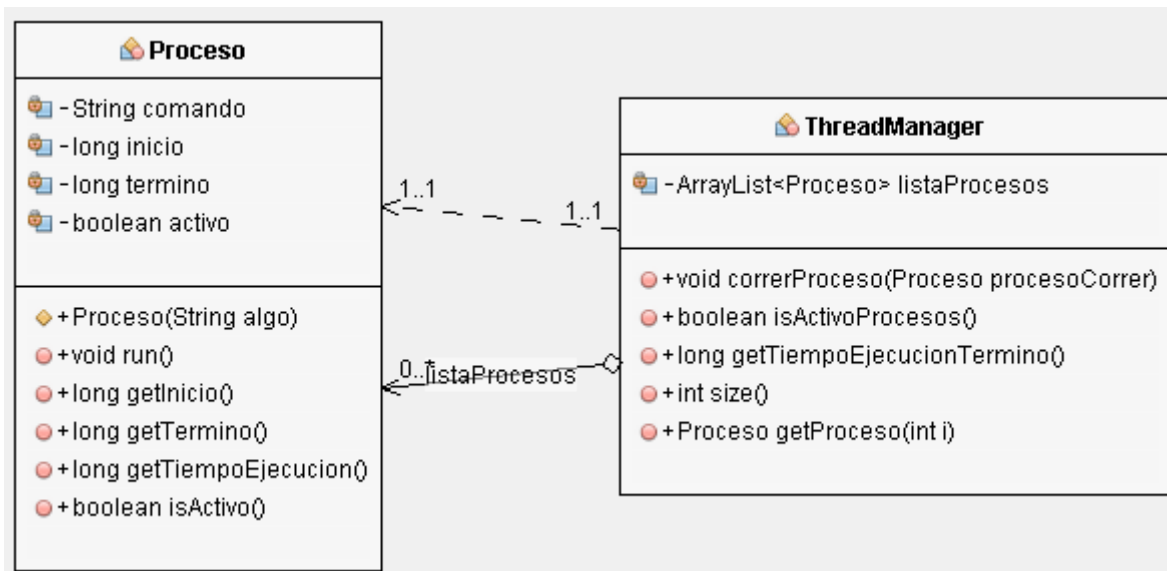


Ilustración 9 - Clases Proceso y ThreadManager

Estas clases se encargan de administrar las ejecuciones de cada proceso de simulación y también permite monitorear la actividad del grupo de simulaciones por equipo. Principalmente para obtener los tiempos que tomó el grupo de simulaciones.

Y, por último, el perfil de Nodo, que la única diferencia que tiene con el de Servidor, es que no cuenta con Servidor y HiloDeCliente sino con una única clase Nodo que es una simple modificación de HiloDeCliente, generando el siguiente diagrama.

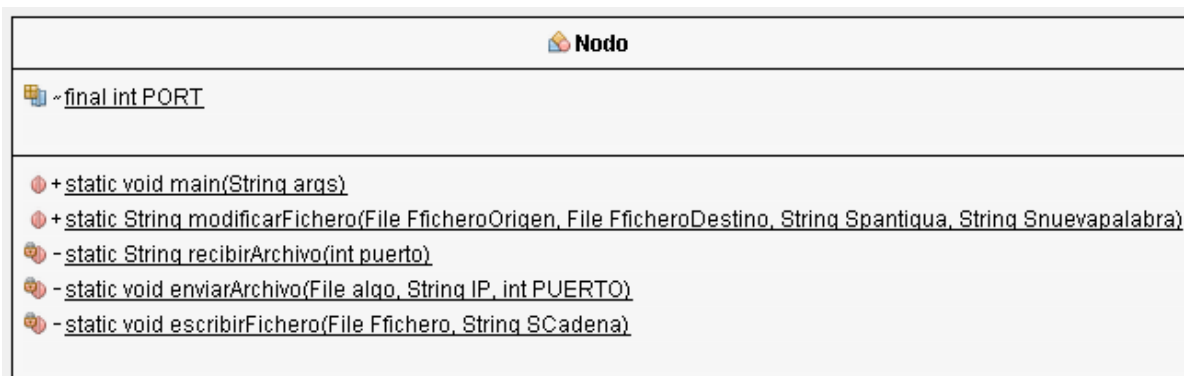


Ilustración 10 - Clase Nodo

Cabe mencionar que el perfil de Nodo también se hace uso de las clases Proceso, Thread Manager y SimpleZip porque sus funcionalidades también son requeridas en este nivel. Además, describen el mismo diagrama expuesto con anterioridad.

6.3 Diseño interfaz de Usuario

El objetivo de este proyecto es interactuar con otras aplicaciones, no con personas. Pero con fines académicos y de estudio, se diseñó una pequeña interfaz para simular la presencia de otra aplicación. El diseño de esa interfaz, se describe a continuación.

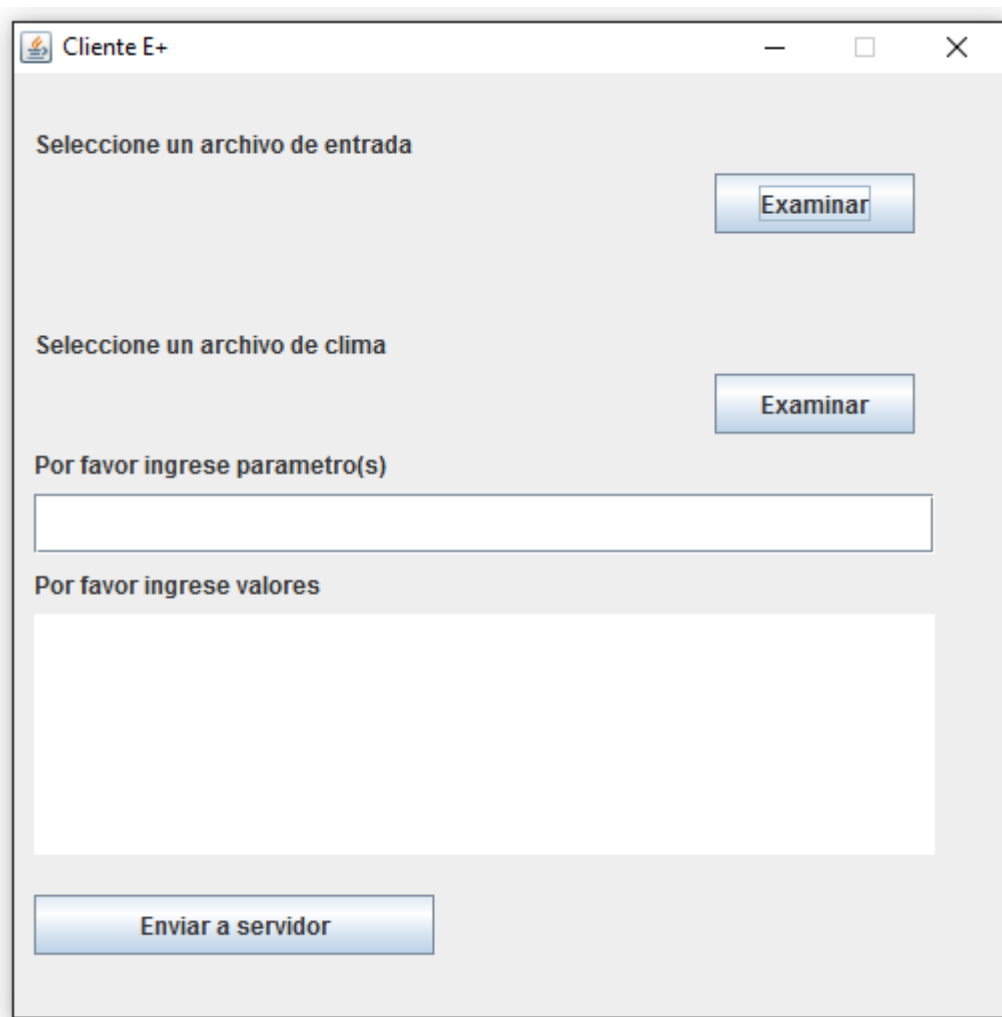


Ilustración 11 – Interfaz Gráfica

De arriba hacia abajo, los dos primeros botones sirven para buscar los archivos necesarios para la simulación (archivo .epw y archivo .idf). Que abrirán un cuadro selector de archivos. La primera caja de texto, corresponde a las marcas que deben ser puestas con anterioridad en el archivo de entrada, que el servidor buscará y editará por el contenido de la segunda caja de texto. Un ejemplo del texto que pudiera contener este campo podría ser “%marca1”. Al momento de editar el archivo de entrada, se recomienda anteponer un carácter especial como “%”, “\$”, o “#” para asegurarse de que la cadena a buscar se encuentre sólo una vez en

el archivo y evitar errores en el proceso de simulación. Para incluir más de una marca, basta separarlas con un espacio en blanco, quedando nuestro texto así:
%marca1 %marca2 %marca3.

La segunda caja de texto contiene los valores por los que serán reemplazadas las marcas definidas con anterioridad. Aquí es necesario un conocimiento más detallado del archivo de entrada, del simulador EnergyPlus y del dominio del mundo del diseño y construcción de edificios en general. Aquí también las opciones son muchas, pero está acotado a los distintos valores que puede tomar cierta variable del archivo de entrada. Por ejemplo, si tomamos el campo "Terrain"¹⁷ del objeto "Building"¹⁸ de un archivo de entrada, debemos reemplazarlo únicamente por los valores legales que este puede tomar (tomados de [26]):

VALOR TIPO TERRENO	DESCRIPCIÓN DEL TERRENO
Country	Plano, campo abierto
Suburbs	Áspero, campo arbolado, Suburbios
City	Ciudad, afueras de la ciudad o centro de grandes ciudades
Ocean	Océano, pantanos planos.
Urban	Urbano, industrial y forestal

Tabla 6 - Valores del campo "Terrain"

Supóngase que nuestra %marca1 apunta al campo Terrain de nuestro idf. Y que se desea que para las simulaciones la marca tome todos estos valores anteriores, entonces la entrada de la segunda caja debe ser:

Country Suburbs City Ocean Urban

Cada cadena se encuentra separada por un espacio en blanco.

Para añadir valores de otras marcas basta añadir en una nueva línea los valores correspondientes.

También la cantidad de marcas y la cantidad de listas valores debe ser la misma. Es imposible que existan marcas sin valores y viceversa, de todas formas, este caso se encuentra validado.

Respecto de la interfaz del perfil servidor y nodo todas sus interacciones son por la salida estándar (línea de comandos).

¹⁷ El terreno del sitio afecta la forma en la que el viento golpea el edificio - al igual que la altura del edificio. Además, el método de conducción externo normalmente tiene sus propios parámetros para el cálculo.

¹⁸ El objeto "Building" describe los parámetros que se usan durante la simulación del edificio.

7 Pruebas

En este capítulo se describen las pruebas a las que fue sometida la aplicación de este proyecto, con el fin de proveer información objetiva sobre la calidad del producto. Estas pruebas fueron realizadas una vez finalizada la implementación.

7.1 Plan de pruebas

Dada la naturaleza de este proyecto, se realizan solo pruebas de carga, que pertenecen a la categoría de pruebas de rendimiento.

7.2 Objetivos del plan de pruebas

Este plan persigue observar el comportamiento de la aplicación bajo distintos escenarios de distribución de carga y encontrar el que nivele de mejor manera el uso de los recursos tecnológicos.

7.3 Elementos de pruebas

Módulo conexión servidor-nodos: se encarga de que se realice transferencia (envío/recepción) de información a través de los distintos nodos, el servidor y el cliente.

Módulo conexión Java-E+: se encarga de generar un ambiente adecuado para la ejecución de simulaciones EnergyPlus desde una aplicación Java.

7.4 Especificación de las pruebas

Como se menciona anteriormente, en este apartado se detalla las pruebas de carga a las que ha sido sometida la aplicación. Desde este enfoque, la variable de interés es el tiempo (medido en milisegundos y minutos) que se tarda el clúster en responder a 100 simulaciones de prueba, dadas distintas configuraciones de carga. Estas pruebas fueron realizadas con dos nodos. Así, la carga de simulación es distribuida entre el servidor y otros dos equipos del clúster.

Las pruebas se realizaron sobre el archivo 5ZoneAirCooled.idf que usa el archivo de clima USA_IL_Chicago-OHare.Intl.AP.725300_TMY3.epw, ambos provistos en la documentación oficial de E+ con las siguientes marcas y valores:

Lista de marcas

%TERRAIN %SOLDIST %STARTVERPOS

Listas de valores

Suburbs Country City Ocean Urban

FullExterior MinimalShadowing FullInteriorAndExterior FullExteriorWithReflections

FullInteriorAndExteriorWithReflections

UpperLeftCorner LowerLeftCorner UpperRightCorner LowerRightCorner

Se probaron tres escenarios:

- Escenario 1: Servidor recibe un 50% de carga, mientras que cada nodo recibe un 25%.
- Escenario 2: Servidor recibe un 60% de carga, mientras que cada nodo recibe un 20%.
- Escenario 3: Servidor recibe un 70% de carga, mientras que cada nodo recibe un 15%.

7.5 Resultado de pruebas

A continuación se sintetizan los resultados obtenidos del proceso de pruebas de desempeño a la aplicación construida durante el desarrollo de este proyecto. El detalle de los datos puede encontrarse en el Apéndice 1.

	SERVIDOR	NODO
Escenario 1		
Tiempo total promedio (en minutos)	2.16514867	2.144516667
Escenario 2		
Tiempo total promedio (en minutos)	2.70437333	1.705206667
Escenario 3		
Tiempo total promedio (en minutos)	3.13866667	1.263966667

Tabla 7 - Resumen Pruebas Clúster LEU

Esta tabla abrevia los resultados obtenidos tras haber realizado el proceso de pruebas al sistema de gestión de peticiones en el clúster LEU.

Los datos corresponden al promedio de 5 muestras por cada escenario probado, donde se evaluó cuánto tiempo (en minutos) tardaron el servidor y los nodos en responder a 100 solicitudes de simulación de E+.

Así, por ejemplo, se obtuvo que cuando la carga se distribuye 50% en el servidor y 25% en cada nodo, en promedio, el servidor tardó 2.16514867 minutos en atender su parte de la carga, mientras que cada nodo ocupó 2.144516667 minutos en hacerse cargo de su 25%.

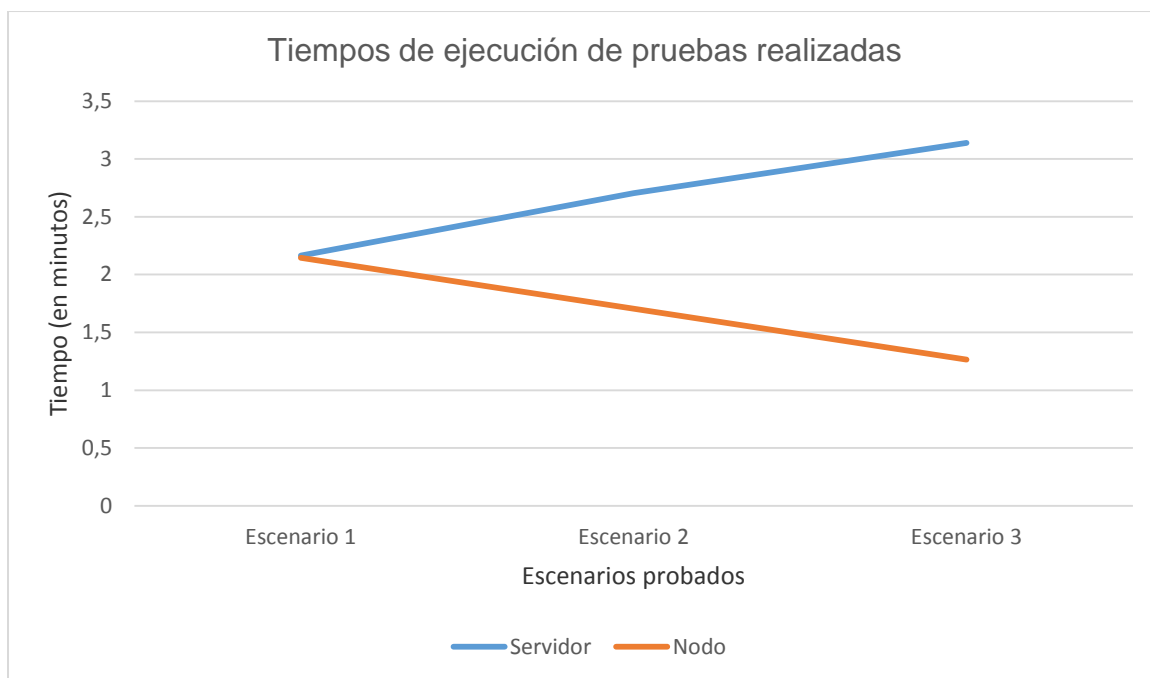


Ilustración 12- Tiempos de ejecución pruebas realizadas

En este gráfico queda representado el tiempo que tarda el clúster LEU en procesar 100 simulaciones bajo los distintos escenarios de prueba anteriormente descritos.

En el eje horizontal se encuentran los tres escenarios probados (en medida porcentual), mientras que en el eje vertical, se muestra el tiempo que tardó cada equipo (en minutos) en responder a la carga de trabajo asignada en cada caso.

Se infiere que la mejor distribución de cargas es 50-25-25 debido a que se presenta una menor diferencia entre los tiempos de simulación entre el servidor y el nodo.

8 Conclusiones y trabajos futuros

Sin duda alguna la eficiencia energética es un tema importante que nos atañe a todos como sociedad. Este proyecto presenta desde un enfoque simple una manera de resolver un problema real del LEU. Gracias a la implementación de una aplicación orientada a la conexión entre un varios clientes y un servidor.

El desarrollo de una aplicación cliente-servidor a pesar de no ser el enfoque más actual es una mejora sencilla e importante a la opción de tener que editar los archivos de entrada manualmente uno tras otro para poder realizar simulaciones paramétricas.

Se demostró que la mejor configuración de distribución de cargas es 50-25-25, donde el servidor asume un 50% de la carga de trabajo mientras que, por otro lado, cada nodo se responsabiliza de un 25% del trabajo a realizar, lo que constituye la mejor diferencia en los tiempos de ejecución de las simulaciones.

El presente proyecto, cumple a plenitud cada uno de sus objetivos. Y es una alternativa fiable y viable para el Laboratorio de Estudios Urbanos.

Trabajo futuro

Las alternativas de mejora a este proyecto son muchas y muy variadas, por mencionar dos, se encuentra el uso de soporte a muchos usuarios concurrentes y el correr aplicaciones Java en un ambiente que permita la migración de hilos a través de los distintos equipos de un clúster. [27]. Cualquiera de los dos significaría un aumento enorme en el desempeño de cualquier proyecto Java.

Por otro lado también existe la opción de desarrollar una aplicación con más funciones y un acabado gráfico mejorado y bajo un ambiente que soporte la migración segura y transparente de hilos Java.

Bibliografía

- [1] M. J. Vásquez Cruz, «Sistema de apoyo a la generación de paquetes para mejoras de eficiencia energética en viviendas de Concepción, Chile,» Concepción, 2015.
- [2] J. Silva Ortiz y R. A. Venegas Muñoz, Cluster de alto rendimiento para renderizado gráfico utilizando CPU y GPU., Concepción, 2012.
- [3] M. Diesendorf, Greenhouse Solutions with Sustainable Energy, 2007.
- [4] T. Kusuda, «Early history and future prospects of Building system,» [En línea]. Available: http://www.ibpsa.org/proceedings/BS1999/BS99_P-01.pdf. [Último acceso: 2016].
- [5] «LIDER,» [En línea]. Available: <http://www.codigotecnico.org/index.php/menu-programa-lider>. [Último acceso: 2016].
- [6] «Minvu,» [En línea]. Available: http://www.minvu.cl/aopensite_20070402125030.aspx. [Último acceso: 2016].
- [7] «Passive House Institute,» [En línea]. Available: http://passivehouse.com/04_php/04_php.htm. [Último acceso: 2016].
- [8] G.-A. R., G. A., B. W., B. A. y M. C., Características relevantes de la simulación energética de viviendas unifamiliares., 2014.
- [9] EnergyPlus - Getting Started, «EnergyPlus,» 2016. [En línea]. Available: <http://nrel.github.io/EnergyPlus/GettingStarted/GettingStarted/>.
- [10] EnergyPlus - InputOutputReference, «EnergyPlus,» 2016. [En línea]. Available: <http://nrel.github.io/EnergyPlus/InputOutputReference/01a-InputOutputReference/>.
- [11] D. B. Crawley, J. W. Hand, M. Kummert y B. T. Griffith., «Contrasting the Capabilities of Building Energy Performance Simulation Programs,» 2005.
- [12] M. C. i. Co`it, El manual para el clustering con openMosix, 2002.
- [13] P. B. G. G. G. Abraham Silberschatz, Operating System Concepts, 2008.
- [14] Z. X. Kai Hwang, Scalable Parallel Computing, 1998.
- [15] D. J. B. D. S. J. E. D. U. A. R. Thomas Sterling, BOWULF: A PARALLEL WORKSTATION FOR SCIENTIFIC COMPUTATION.
- [16] «Wikipedia - Cluster AD,» [En línea]. Available: https://es.wikipedia.org/wiki/CI%C3%BAster_de_alta_disponibilidad. [Último acceso: 2016].
- [17] R. Buyya, High Performance Cluster Computing: Architectures and Systems, Melbourne, Australia, 1999.
- [18] P. Bernstein, «Middleware,» 1996.
- [19] D. S. ,. M. S. M. P. Y. J. X. S. e. a. G. Wieerhold, «OntoAgents – a Project in the DARPA DAML Program,» 2000.
- [20] J. Charles, «Middleware Moves to the Forefront,» 1999.
- [21] D. S. e. a. M. Astley, «Customizable Middleware for Modular Distributed Software,» 2001.
- [22] «MOSIX,» 2016. [En línea]. Available: http://www.mosix.cs.huji.ac.il/pub/MOSIX_wp.pdf.

- [23] «Oracle,» 2016. [En línea]. Available:
<https://docs.oracle.com/javase/tutorial/networking/sockets>.
- [24] MOSIX, «MOSIX Frequently Asked Questions - Flat listing,» [En línea]. Available:
http://www.mosix.cs.huji.ac.il/faq/output/faq_flat.html. [Último acceso: 2016].
- [25] «JavaWorld,» 2016. [En línea]. Available:
<http://www.javaworld.com/article/2077322/core-java/core-java-sockets-programming-in-java-a-tutorial.html>.
- [26] «Big Ladder,» [En línea]. Available: <http://bigladdersoftware.com/epx/docs/8-0/input-output-reference/page-006.html>. [Último acceso: 2016].
- [27] W. Zhu, Cho-Li Wang y F. C. M. Lau, Lightweight Transparent Java Thread Migration for Distributed JVM, Hong Kong, 2003.
- [28] 2015. [En línea]. Available: <http://www.acee.cl/eficiencia-energetica/que-es>.
- [29] «Wikipedia,» [En línea]. Available:
[https://es.wikipedia.org/wiki/CI%C3%BAster_\(inform%C3%A1tica\)](https://es.wikipedia.org/wiki/CI%C3%BAster_(inform%C3%A1tica)). [Último acceso: 2016].
- [30] Oracle, «Oracle,» [En línea]. Available:
<https://docs.oracle.com/javase/7/docs/api/java/lang/Thread.html>. [Último acceso: 2016].
- [31] R. K. K. Toni A. Bishop, «A SURVEY of MIDDLEWARE,» 2003.

Apéndice 1 – Detalle pruebas

PRUEBA 1 (SERVER 50% N1 25%, N2 25%)

Lado Server

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	124901	2.08168333	127105	2.11841667	116735	1.94558333
2	126096	2.1016	123234	2.0539	122150	2.03583333
3	128831	2.14718333	119559	1.99265	123454	2.05756667
4	131039	2.18398333	122046	2.0341	124030	2.06716667
5	131267	2.18778333	122514	2.0419	124345	2.07241667
6	131759	2.19598333	124958	2.08263333	126205	2.10341667
7	136117	2.26861667	118781	1.97968333	126656	2.11093333
8	130282	2.17136667	123972	2.0662	125617	2.09361667
9	126758	2.11263333	122086	2.03476667	125592	2.0932
10	133678	2.22796667	126449	2.10748333	125856	2.0976
11	137370	2.2895	126868	2.11446667	127315	2.12191667
12	132877	2.21461667	127072	2.11786667	125544	2.0924
13	128609	2.14348333	127183	2.11971667	126647	2.11078333
14	128990	2.14983333	122860	2.04766667	126783	2.11305
15	136806	2.2801	127636	2.12726667	126798	2.1133
16	130608	2.1768	124035	2.06725	126198	2.1033
17	133608	2.2268	128395	2.13991667	127209	2.12015
18	135762	2.2627	122496	2.0416	127007	2.11678333
19	136724	2.27873333	124322	2.07203333	123586	2.05976667
20	136329	2.27215	129414	2.1569	127844	2.13073333
21	131130	2.1855	123734	2.06223333	125968	2.09946667
22	125234	2.08723333	127352	2.12253333	118929	1.98215
23	136215	2.27025	122929	2.04881667	126310	2.10516667
24	135625	2.26041667	118995	1.98325	126009	2.10015
25	131689	2.19481667	117059	1.95098333	116943	1.94905
26	132020	2.20033333	118477	1.97461667	126128	2.10213333
27	134008	2.23346667	124739	2.07898333	126191	2.10318333
28	132830	2.21383333	127277	2.12128333	118175	1.96958333
29	131321	2.18868333	124546	2.07576667	126203	2.10338333
30	134579	2.24298333	128597	2.14328333	123691	2.06151667
31	133682	2.22803333	126348	2.1058	120174	2.0029
32	129903	2.16505	126863	2.11438333	127018	2.11696667
33	136083	2.26805	127985	2.13308333	126907	2.11511667
34	137630	2.29383333	128449	2.14081667	126771	2.11285
35	132158	2.20263333	126188	2.10313333	126623	2.11038333
36	135017	2.25028333	129347	2.15578333	126619	2.11031667
37	134688	2.2448	128834	2.14723333	127329	2.12215
38	134608	2.24346667	126466	2.10776667	128605	2.14341667

39	135802	2.26336667	129395	2.15658333	126749	2.11248333
40	137110	2.28516667	125185	2.08641667	127778	2.12963333
41	124916	2.08193333	121452	2.0242	121277	2.02128333
42	136074	2.2679	128795	2.14658333	124047	2.06745
43	135923	2.26538333	128672	2.14453333	122534	2.04223333
44	128811	2.14685	117605	1.96008333	125572	2.09286667
45	131642	2.19403333	127316	2.12193333	125552	2.09253333
46	132561	2.20935	128440	2.14066667	125547	2.09245
47	135587	2.25978333	121056	2.0176	126922	2.11536667
48	127083	2.11805	126159	2.10265	126731	2.11218333
49	134124	2.2354	124971	2.08285	125651	2.09418333
50	134135	2.23558333	122376	2.0396	127056	2.1176
Tiempo total por grupo	137668	2.29446667	129444	2.1574	128634	2.1439
En promedio, una simulación tardó (en minutos)		2.21019967		2.08219733		2.08519333

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	126701	2.11168333	121983	2.03305
2	125090	2.08483333	126158	2.10263333
3	121658	2.02763333	123302	2.05503333
4	118489	1.97481667	121586	2.02643333
5	125724	2.0954	126820	2.11366667
6	120781	2.01301667	123647	2.06078333
7	125417	2.09028333	126251	2.10418333
8	121247	2.02078333	122808	2.0468
9	116292	1.9382	118327	1.97211667
10	121058	2.01763333	125163	2.08605
11	121980	2.033	126069	2.10115
12	126975	2.11625	127508	2.12513333
13	126395	2.10658333	127413	2.12355
14	127750	2.12916667	123502	2.05836667
15	126698	2.11163333	127650	2.1275
16	120067	2.00111667	127788	2.1298
17	127770	2.1295	124370	2.07283333
18	127865	2.13108333	124824	2.0804
19	127024	2.11706667	129269	2.15448333
20	127199	2.11998333	125243	2.08738333
21	119685	1.99475	126246	2.1041

22	122010	2.0335	127174	2.11956667
23	125667	2.09445	126581	2.10968333
24	125677	2.09461667	122009	2.03348333
25	125663	2.09438333	126085	2.10141667
26	126459	2.10765	126050	2.10083333
27	121563	2.02605	124356	2.0726
28	124396	2.07326667	126472	2.10786667
29	125418	2.0903	118647	1.97745
30	122454	2.0409	127254	2.1209
31	126995	2.11658333	127178	2.11963333
32	121273	2.02121667	126452	2.10753333
33	119839	1.99731667	127617	2.12695
34	120876	2.0146	127449	2.12415
35	124260	2.071	122711	2.04518333
36	127112	2.11853333	127557	2.12595
37	124103	2.06838333	123847	2.06411667
38	122735	2.04558333	127538	2.12563333
39	126980	2.11633333	126293	2.10488333
40	128010	2.1335	126162	2.1027
41	116322	1.9387	125861	2.09768333
42	122328	2.0388	126120	2.102
43	126121	2.10201667	119422	1.99036667
44	125184	2.0864	124518	2.0753
45	122188	2.03646667	123860	2.06433333
46	117243	1.95405	123820	2.06366667
47	125814	2.0969	126867	2.11445
48	126659	2.11098333	127172	2.11953333
49	120571	2.00951667	126181	2.10301667
50	122232	2.0372	126079	2.10131667
Tiempo total por grupo	128045	2.13408333	129276	2.1546
En promedio, una simulación tardó (en minutos)		2.06267233		2.087753

Lado Nodo

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	128068	2.134466667	128225	2.137083333	126426	2.1071
2	128841	2.14735	128038	2.133966667	123838	2.063966667
3	125725	2.095416667	127101	2.11835	123741	2.06235
4	127190	2.119833333	126325	2.105416667	125280	2.088
5	129868	2.164466667	128186	2.136433333	126036	2.1006
6	128812	2.146866667	127855	2.130916667	127308	2.1218

7	129759	2.16265	127266	2.1211	126304	2.105066667
8	129895	2.164916667	128851	2.147516667	127848	2.1308
9	129333	2.15555	129315	2.15525	128095	2.134916667
10	130157	2.169283333	128261	2.137683333	127945	2.132416667
11	127013	2.116883333	127993	2.133216667	121472	2.024533333
12	127236	2.1206	127674	2.1279	123759	2.06265
13	129268	2.154466667	120994	2.016566667	124245	2.07075
14	125741	2.095683333	126159	2.10265	127247	2.120783333
15	119833	1.997216667	127446	2.1241	127056	2.1176
16	126867	2.11445	121935	2.03225	127569	2.12615
17	126538	2.108966667	127025	2.117083333	122368	2.039466667
18	122606	2.043433333	127631	2.127183333	127034	2.117233333
19	125728	2.095466667	121528	2.025466667	122681	2.044683333
20	129649	2.160816667	127255	2.120916667	125578	2.092966667
21	129571	2.159516667	126741	2.11235	127358	2.122633333
22	129716	2.161933333	125666	2.094433333	127962	2.1327
23	127953	2.13255	127345	2.122416667	126512	2.108533333
24	129846	2.1641	127456	2.124266667	126547	2.109116667
25	129089	2.151483333	121461	2.02435	126768	2.1128
Tiempo total por grupo	130528	2.175466667	129328	2.155466667	128126	2.135433333
En promedio, una simulación tardó (en minutos)		2.129534667		2.109154667		2.097984667

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	122413	2.040216667	124167	2.06945
2	122733	2.04555	125050	2.084166667
3	118398	1.9733	124620	2.077
4	123014	2.050233333	123409	2.056816667
5	123975	2.06625	127358	2.122633333
6	124246	2.070766667	127962	2.1327
7	121020	2.017	126512	2.108533333
8	124167	2.06945	126547	2.109116667
9	125050	2.084166667	128038	2.133966667
10	124620	2.077	127101	2.11835
11	123409	2.056816667	126325	2.105416667
12	121322	2.022033333	128186	2.136433333

13	123681	2.06135	125725	2.095416667
14	122087	2.034783333	127190	2.119833333
15	122372	2.039533333	129868	2.164466667
16	122059	2.034316667	128812	2.146866667
17	121763	2.029383333	127101	2.11835
18	123688	2.061466667	127993	2.133216667
19	121361	2.022683333	128038	2.133966667
20	123981	2.06635	127993	2.133216667
21	122454	2.0409	127674	2.1279
22	123507	2.05845	127456	2.124266667
23	124034	2.067233333	125725	2.095416667
24	120395	2.006583333	129333	2.15555
25	124046	2.067433333	125728	2.095466667
Tiempo total por grupo	125068	2.084466667	130305	2.17175
En promedio, una simulación tardó (en minutos)		2.04653		2.115940667

PRUEBA 2 (SERVER 60% N1 20%, N2 20%)

Lado Server

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	167701	2.79501667	152436	2.5406	155380	2.58966667
2	153352	2.55586667	156473	2.60788333	148998	2.4833
3	167178	2.7863	147681	2.46135	156369	2.60615
4	160072	2.66786667	151377	2.52295	156461	2.60768333
5	152799	2.54665	156517	2.60861667	156598	2.60996667
6	167507	2.79178333	146500	2.44166667	156934	2.61556667
7	166102	2.76836667	157112	2.61853333	144027	2.40045
8	161261	2.68768333	153656	2.56093333	156225	2.60375
9	162707	2.71178333	147447	2.45745	143645	2.39408333
10	166180	2.76966667	148959	2.48265	157552	2.62586667
11	163913	2.73188333	157914	2.6319	157469	2.62448333
12	165524	2.75873333	155130	2.5855	147321	2.45535
13	167136	2.7856	154962	2.5827	157086	2.6181
14	165782	2.76303333	155189	2.58648333	157599	2.62665
15	160708	2.67846667	149589	2.49315	152866	2.54776667
16	157288	2.62146667	155741	2.59568333	157410	2.6235
17	163300	2.72166667	150112	2.50186667	157629	2.62715
18	162220	2.70366667	158326	2.63876667	151588	2.52646667

19	157099	2.61831667	158536	2.64226667	150304	2.50506667
20	167917	2.79861667	158081	2.63468333	154107	2.56845
21	156675	2.61125	156061	2.60101667	152730	2.5455
22	167771	2.79618333	157598	2.62663333	156843	2.61405
23	167430	2.7905	148718	2.47863333	157108	2.61846667
24	152924	2.54873333	156170	2.60283333	148310	2.47183333
25	162121	2.70201667	157152	2.6192	155809	2.59681667
26	163659	2.72765	153621	2.56035	152686	2.54476667
27	167245	2.78741667	156877	2.61461667	156728	2.61213333
28	165600	2.76	157129	2.61881667	155467	2.59111667
29	166155	2.76925	155324	2.58873333	156336	2.6056
30	166724	2.77873333	157580	2.62633333	157863	2.63105
31	167160	2.786	158283	2.63805	154904	2.58173333
32	166111	2.76851667	155843	2.59738333	154256	2.57093333
33	160600	2.67666667	156031	2.60051667	157172	2.61953333
34	166152	2.7692	156868	2.61446667	156697	2.61161667
35	164008	2.73346667	149848	2.49746667	157368	2.6228
36	166818	2.7803	156212	2.60353333	157219	2.62031667
37	166093	2.76821667	157597	2.62661667	155701	2.59501667
38	165523	2.75871667	158443	2.64071667	157883	2.63138333
39	165567	2.75945	157888	2.63146667	157356	2.6226
40	163423	2.72371667	157756	2.62926667	148516	2.47526667
41	150628	2.51046667	143174	2.38623333	155481	2.59135
42	161786	2.69643333	153753	2.56255	156823	2.61371667
43	166787	2.77978333	156423	2.60705	146964	2.4494
44	167158	2.78596667	151216	2.52026667	153681	2.56135
45	167419	2.79031667	152944	2.54906667	154988	2.58313333
46	167471	2.79118333	147063	2.45105	156594	2.6099
47	167810	2.79683333	157579	2.62631667	156576	2.6096
48	153416	2.55693333	147290	2.45483333	143103	2.38505
49	160532	2.67553333	157223	2.62038333	156214	2.60356667
50	163267	2.72111667	155853	2.59755	157067	2.61778333
51	162697	2.71161667	158099	2.63498333	156850	2.61416667
52	168094	2.80156667	151162	2.51936667	155933	2.59888333
53	164215	2.73691667	147021	2.45035	156052	2.60086667
54	167815	2.79691667	156398	2.60663333	157171	2.61951667
55	167128	2.78546667	158444	2.64073333	156039	2.60065
56	166138	2.76896667	158858	2.64763333	157441	2.62401667
57	167786	2.79643333	153775	2.56291667	150452	2.50753333
58	165465	2.75775	158146	2.63576667	158053	2.63421667
59	165122	2.75203333	156286	2.60476667	157786	2.62976667
60	167138	2.78563333	152809	2.54681667	157462	2.62436667
Tiempo total por grupo	168128	2.80213333	158912	2.64853333	158233	2.63721667

En promedio, una simulación tardó (en minutos)		2.73093806		2.57395917		2.57868056
--	--	------------	--	------------	--	------------

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	154136	2.56893333	157633	2.627216667
2	150365	2.50608333	155467	2.591116667
3	156004	2.60006667	167136	2.7856
4	156381	2.60635	160072	2.667866667
5	150492	2.5082	157328	2.622133333
6	151239	2.52065	156498	2.6083
7	157223	2.62038333	157424	2.623733333
8	146899	2.44831667	153621	2.56035
9	156364	2.60606667	152730	2.5455
10	157452	2.6242	157152	2.6192
11	157818	2.6303	155741	2.595683333
12	156280	2.60466667	148310	2.471833333
13	147681	2.46135	154783	2.579716667
14	157317	2.62195	167178	2.7863
15	154860	2.581	158081	2.634683333
16	155563	2.59271667	157580	2.626333333
17	157747	2.62911667	157410	2.6235
18	157545	2.62575	152686	2.544766667
19	155530	2.59216667	156728	2.612133333
20	154105	2.56841667	154617	2.57695
21	155113	2.58521667	161261	2.687683333
22	153597	2.55995	158536	2.642266667
23	145512	2.4252	153352	2.555866667
24	156332	2.60553333	157901	2.631683333
25	152188	2.53646667	167701	2.795016667
26	156615	2.61025	157629	2.62715
27	144315	2.40525	156061	2.601016667
28	153006	2.5501	155324	2.588733333
29	154258	2.57096667	150112	2.501866667
30	157223	2.62038333	160708	2.678466667
31	157486	2.62476667	148718	2.478633333
32	157176	2.6196	157108	2.618466667
33	154300	2.57166667	151739	2.528983333
34	157510	2.62516667	166102	2.768366667
35	155767	2.59611667	153544	2.559066667
36	156093	2.60155	163913	2.731883333

37	150924	2.5154	157598	2.626633333
38	157859	2.63098333	151588	2.526466667
39	158238	2.6373	152799	2.54665
40	154425	2.57375	167507	2.791783333
41	157714	2.62856667	158326	2.638766667
42	156308	2.60513333	155809	2.596816667
43	156555	2.60925	165524	2.758733333
44	153680	2.56133333	156170	2.602833333
45	156548	2.60913333	157129	2.618816667
46	146934	2.4489	150304	2.505066667
47	156963	2.61605	156843	2.61405
48	157328	2.62213333	150997	2.516616667
49	145000	2.41666667	154107	2.56845
50	157901	2.63168333	165782	2.763033333
51	157424	2.62373333	162707	2.711783333
52	157633	2.62721667	156877	2.614616667
53	156498	2.6083	158062	2.634366667
54	151739	2.52898333	157945	2.632416667
55	150997	2.51661667	156336	2.6056
56	157945	2.63241667	166180	2.769666667
57	154617	2.57695	157863	2.63105
58	158062	2.63436667	146934	2.4489
59	154783	2.57971667	156963	2.61605
60	153544	2.55906667	145000	2.416666667
Tiempo total por grupo	158294	2.63823333	167745	2.79575
En promedio, una simulación tardó (en minutos)		2.57697528		2.617565

Lado Nodo

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	101166	1.6861	98004	1.6334	100217	1.670283333
2	102365	1.706083333	100380	1.673	98937	1.64895
3	100396	1.673266667	100849	1.680816667	100681	1.678016667
4	97241	1.620683333	97775	1.629583333	100539	1.67565
5	102504	1.7084	97736	1.628933333	98901	1.64835
6	102010	1.700166667	98501	1.641683333	100099	1.668316667
7	102052	1.700866667	100351	1.672516667	99893	1.664883333
8	98869	1.647816667	100831	1.680516667	97226	1.620433333
9	102449	1.707483333	98296	1.638266667	100165	1.669416667
10	103081	1.718016667	99672	1.6612	100678	1.677966667

11	99573	1.65955	100741	1.679016667	95262	1.5877
12	102865	1.714416667	101316	1.6886	100503	1.67505
13	98754	1.6459	94651	1.577516667	100685	1.678083333
14	102218	1.703633333	101146	1.685766667	98542	1.642366667
15	102750	1.7125	100787	1.679783333	100000	1.666666667
16	102493	1.708216667	98622	1.6437	97443	1.62405
17	98995	1.649916667	100713	1.67855	99756	1.6626
18	101657	1.694283333	101607	1.69345	101982	1.6997
19	99690	1.6615	102721	1.712016667	100297	1.671616667
20	102378	1.7063	101384	1.689733333	100877	1.681283333
Tiempo total por grupo	103090	1.718166667	102741	1.71235	102052	1.700866667
En promedio, una simulación tardó (en minutos)		1.686255		1.6634025		1.660569167

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	99061	1.651016667	96733	1.612216667
2	101897	1.698283333	97385	1.623083333
3	101588	1.693133333	99212	1.653533333
4	98900	1.648333333	100179	1.66965
5	99523	1.658716667	97643	1.627383333
6	99510	1.6585	98784	1.6464
7	99702	1.6617	99771	1.66285
8	101454	1.6909	100098	1.6683
9	101725	1.695416667	99515	1.658583333
10	102302	1.705033333	100832	1.680533333
11	99775	1.662916667	99727	1.662116667
12	101795	1.696583333	97115	1.618583333
13	100149	1.66915	97519	1.625316667
14	101860	1.697666667	100312	1.671866667
15	97504	1.625066667	100082	1.668033333
16	102463	1.707716667	91853	1.530883333
17	99396	1.6566	99364	1.656066667
18	101137	1.685616667	100145	1.669083333
19	101605	1.693416667	101145	1.68575
20	102200	1.703333333	99874	1.664566667
Tiempo total por grupo	102472	1.707866667	101207	1.686783333

En promedio, una simulación tardó (en minutos)		1.677955		1.64774
--	--	----------	--	---------

PRUEBA 3 (SERVER 70% N1 15%, N2 15%)

Lado Servidor

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	182343	3.03905	178172	2.96953333	184684	3.07806667
2	178465	2.97441667	183059	3.05098333	177640	2.96066667
3	177189	2.95315	180347	3.00578333	184609	3.07681667
4	181595	3.02658333	179378	2.98963333	184186	3.06976667
5	178414	2.97356667	179330	2.98883333	186159	3.10265
6	179109	2.98515	182894	3.04823333	188377	3.13961667
7	183763	3.06271667	182205	3.03675	185652	3.0942
8	179761	2.99601667	178429	2.97381667	185518	3.09196667
9	184405	3.07341667	181912	3.03186667	185809	3.09681667
10	168902	2.81503333	182918	3.04863333	189807	3.16345
11	184017	3.06695	184739	3.07898333	187901	3.13168333
12	183844	3.06406667	184918	3.08196667	188016	3.1336
13	167073	2.78455	183950	3.06583333	184028	3.06713333
14	185127	3.08545	185214	3.0869	186612	3.1102
15	180896	3.01493333	184690	3.07816667	188736	3.1456
16	181132	3.01886667	185035	3.08391667	182083	3.03471667
17	178368	2.9728	185060	3.08433333	189709	3.16181667
18	181082	3.01803333	185301	3.08835	189694	3.16156667
19	182374	3.03956667	183639	3.06065	188624	3.14373333
20	184394	3.07323333	185159	3.08598333	187274	3.12123333
21	184181	3.06968333	178782	2.9797	185238	3.0873
22	184372	3.07286667	182790	3.0465	185555	3.09258333
23	165686	2.76143333	180882	3.0147	186633	3.11055
24	164395	2.73991667	182009	3.03348333	185237	3.08728333
25	179811	2.99685	181853	3.03088333	189090	3.1515
26	164836	2.74726667	182021	3.03368333	189625	3.16041667
27	173301	2.88835	183149	3.05248333	189069	3.15115
28	178702	2.97836667	182585	3.04308333	186823	3.11371667
29	174792	2.9132	182757	3.04595	183103	3.05171667
30	181196	3.01993333	184958	3.08263333	190024	3.16706667
31	181818	3.0303	184999	3.08331667	188275	3.13791667
32	183976	3.06626667	185052	3.0842	189905	3.16508333
33	180693	3.01155	180970	3.01616667	184621	3.07701667

34	185647	3.09411667	184976	3.08293333	190419	3.17365
35	168500	2.80833333	182697	3.04495	189053	3.15088333
36	180825	3.01375	185014	3.08356667	189100	3.15166667
37	180808	3.01346667	182973	3.04955	190330	3.17216667
38	182808	3.0468	184581	3.07635	190002	3.1667
39	185848	3.09746667	184618	3.07696667	190023	3.16705
40	184438	3.07396667	185432	3.09053333	190527	3.17545
41	175776	2.9296	181146	3.0191	184142	3.06903333
42	178614	2.9769	183667	3.06111667	187033	3.11721667
43	180183	3.00305	180200	3.00333333	188057	3.13428333
44	180684	3.0114	182523	3.04205	185986	3.09976667
45	183759	3.06265	185121	3.08535	186854	3.11423333
46	179310	2.9885	179314	2.98856667	182150	3.03583333
47	178099	2.96831667	177211	2.95351667	185823	3.09705
48	179072	2.98453333	183006	3.0501	185762	3.09603333
49	180209	3.00348333	181109	3.01848333	188488	3.14146667
50	180762	3.0127	185001	3.08335	188253	3.13755
51	180816	3.0136	184857	3.08095	188304	3.1384
52	180937	3.01561667	184794	3.0799	188231	3.13718333
53	180158	3.00263333	184882	3.08136667	190626	3.1771
54	182475	3.04125	184894	3.08156667	188376	3.1396
55	182134	3.03556667	185179	3.08631667	188982	3.1497
56	181003	3.01671667	183683	3.06138333	186112	3.10186667
57	179686	2.99476667	181126	3.01876667	185595	3.09325
58	182329	3.03881667	184920	3.082	190793	3.17988333
59	186339	3.10565	185270	3.08783333	188322	3.1387
60	185409	3.09015	184828	3.08046667	190257	3.17095
61	178512	2.9752	176555	2.94258333	181473	3.02455
62	180179	3.00298333	184743	3.07905	182073	3.03455
63	176679	2.94465	182839	3.04731667	189737	3.16228333
64	165893	2.76488333	180935	3.01558333	188448	3.1408
65	165000	2.75	180678	3.0113	186779	3.11298333
66	177983	2.96638333	182185	3.03641667	186988	3.11646667
67	180419	3.00698333	185137	3.08561667	184579	3.07631667
68	183436	3.05726667	182127	3.03545	189091	3.15151667
69	174525	2.90875	178270	2.97116667	174918	2.9153
70	184691	3.07818333	182774	3.04623333	189375	3.15625
Tiempo total por grupo	186586	3.10976667	185500	3.09166667	190940	3.18233333
En promedio, una simulación tardó (en minutos)		2.98903738		3.0472431		3.11651833

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	174006	2.9001	181595	3.02658333
2	181707	3.02845	185762	3.09603333
3	179935	2.99891667	182790	3.0465
4	180623	3.01038333	186854	3.11423333
5	179454	2.9909	178782	2.9797
6	176455	2.94091667	184958	3.08263333
7	183697	3.06161667	187033	3.11721667
8	176285	2.93808333	188057	3.13428333
9	178496	2.97493333	188231	3.13718333
10	184835	3.08058333	185185	3.08641667
11	181973	3.03288333	182757	3.04595
12	183370	3.05616667	185528	3.09213333
13	183576	3.0596	168902	2.81503333
14	180149	3.00248333	182343	3.03905
15	184568	3.07613333	190527	3.17545
16	185370	3.0895	183763	3.06271667
17	182983	3.04971667	184649	3.07748333
18	185521	3.09201667	180896	3.01493333
19	184193	3.06988333	183901	3.06501667
20	184770	3.0795	184870	3.08116667
21	178057	2.96761667	184017	3.06695
22	180787	3.01311667	185014	3.08356667
23	181109	3.01848333	184696	3.07826667
24	184697	3.07828333	180882	3.0147
25	180955	3.01591667	178465	2.97441667
26	180096	3.0016	182150	3.03583333
27	178671	2.97785	183844	3.06406667
28	179337	2.98895	183347	3.05578333
29	183032	3.05053333	190626	3.1771
30	183982	3.06636667	188376	3.1396
31	184720	3.07866667	181082	3.01803333
32	184977	3.08295	190023	3.16705
33	179040	2.984	184405	3.07341667
34	184575	3.07625	182973	3.04955
35	184813	3.08021667	183639	3.06065
36	183122	3.05203333	185052	3.0842
37	184653	3.07755	185986	3.09976667
38	184576	3.07626667	182572	3.04286667
39	185274	3.0879	183149	3.05248333
40	185014	3.08356667	178368	2.9728
41	184437	3.07395	177189	2.95315

42	184632	3.0772	178414	2.97356667
43	183045	3.05075	188253	3.13755
44	184592	3.07653333	180970	3.01616667
45	183180	3.053	184538	3.07563333
46	184343	3.07238333	184999	3.08331667
47	183781	3.06301667	185159	3.08598333
48	182476	3.04126667	182697	3.04495
49	182397	3.03995	179923	2.99871667
50	184361	3.07268333	184976	3.08293333
51	184741	3.07901667	185945	3.09908333
52	184835	3.08058333	167073	2.78455
53	183998	3.06663333	181132	3.01886667
54	183614	3.06023333	185823	3.09705
55	184401	3.07335	184142	3.06903333
56	184652	3.07753333	186112	3.10186667
57	185185	3.08641667	190002	3.1667
58	185945	3.09908333	179761	2.99601667
59	185528	3.09213333	188304	3.1384
60	185441	3.09068333	179109	2.98515
61	181028	3.01713333	181028	3.01713333
62	184696	3.07826667	188982	3.1497
63	183901	3.06501667	188488	3.14146667
64	184649	3.07748333	181853	3.03088333
65	180486	3.0081	185441	3.09068333
66	179923	2.99871667	182009	3.03348333
67	183347	3.05578333	180486	3.0081
68	184538	3.07563333	182585	3.04308333
69	182572	3.04286667	182021	3.03368333
70	184870	3.08116667	185127	3.08545
Tiempo total por grupo	186648	3.1108	191926	3.19876667
En promedio, una simulación tardó (en minutos)		3.04739214		3.0591881

Lado Nodo

MUESTRA	NUMERO 1		NUMERO 2		NUMERO 3	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	74900	1.248333333	74403	1.24005	73763	1.229383333
2	74464	1.241066667	74803	1.246716667	73947	1.23245
3	73519	1.225316667	74099	1.234983333	74182	1.236366667
4	74198	1.236633333	75404	1.256733333	75560	1.259333333
5	74784	1.2464	75157	1.252616667	70946	1.182433333

6	75471	1.25785	74433	1.24055	74548	1.242466667
7	74694	1.2449	72961	1.216016667	71685	1.19475
8	75502	1.258366667	74485	1.241416667	74739	1.24565
9	75655	1.260916667	74851	1.247516667	75186	1.2531
10	74638	1.243966667	75899	1.264983333	74795	1.246583333
11	69422	1.157033333	74614	1.243566667	71714	1.195233333
12	75622	1.260366667	73064	1.217733333	72523	1.208716667
13	75266	1.254433333	71399	1.189983333	74131	1.235516667
14	72840	1.214	74018	1.233633333	75057	1.25095
15	74727	1.24545	70902	1.1817	73013	1.216883333
Tiempo total por grupo	75658	1.260966667	75904	1.265066667	75562	1.259366667
En promedio, una simulación tardó (en minutos)		1.239668889		1.23388		1.228654444

MUESTRA	NUMERO 4		NUMERO 5	
Proceso #	Duración (en milisegundos)	Duración (en minutos)	Duración (en milisegundos)	Duración (en minutos)
1	75309	1.25515	70901	1.181683333
2	74374	1.239566667	75612	1.2602
3	71048	1.184133333	74017	1.233616667
4	72204	1.2034	74794	1.246566667
5	73967	1.232783333	69426	1.1571
6	71058	1.1843	70946	1.182433333
7	73617	1.22695	75186	1.2531
8	73878	1.2313	71399	1.189983333
9	76398	1.2733	69286	1.154766667
10	75048	1.2508	73763	1.229383333
11	66639	1.11065	73947	1.23245
12	73759	1.229316667	75266	1.254433333
13	72720	1.212	73064	1.217733333
14	69286	1.154766667	74900	1.248333333
15	72967	1.216116667	74803	1.246716667
Tiempo total por grupo	76412	1.273533333	75654	1.2609
En promedio, una simulación tardó (en minutos)		1.213635556		1.219233333

