UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA DE SISTEMAS

Tesis previa a la obtención del título de: INGENIERA DE SISTEMAS E INGENIERO DE SISTEMAS

TEMA:

ANÁLISIS, DISEÑO, DESARROLLO E INTEGRACIÓN DEL MÓDULO DE ADMINISTRACIÓN PARA EL SISTEMA UPS SCHEDULE PARA LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO

AUTORES:

CRISTINA ELIZABETH ALULEMA ORTIZ

CARLOS MIGUEL CÁRDENAS RIOFRIO

DIRECTOR:

TUFIÑO CÁRDENAS RODRIGO EFRAÍN

Quito, junio de 2014

DECLARATORIA DE RESPONSABILIDAD Y AUTORIZACIÓN DE USO DEL TRABAJO DE TITULACIÓN

Nosotros, autorizamos a la Universidad Politécnica Salesiana la publicación total o parcial de este trabajo de grado y su reproducción sin fines de lucro.

Además, declaramos que los conceptos y análisis desarrollados y las conclusiones del presente trabajo son de exclusiva responsabilidad de los autores.

Cristina Elizabeth Alulema Ortiz
CI: 1717558769

Carlos Miguel Cárdenas Riofrio
CI: 1717438624

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

RESUMEN

El proceso de modernización y reforma educativa que se está implementando en el país, obliga a los centros de formación universitarios específicamente a la Universidad Politécnica Salesiana, a implementar alternativas administrativas que optimicen el uso y administración de espacios físicos, y de esta manera optimizar en los próximos periodos lectivos el uso de aulas, laboratorios, audiovisuales, etc. en las carreras que se imparten en la universidad de forma automática, mediante la aplicación de parámetros específicos que viabilicen la definición de estructuras interactivas.

En base a la problemática se plantea el desarrollo de un sistema capaz de facilitar los procesos rutinarios presentes cada periodo lectivo como revisión de archivos Excel por cada carrera, creación manual de distributivos y horarios, entre otros. “UPS Schedule” fue dividido en varios módulos por su complejidad; en base a esta condición el proyecto se enfoca en el Módulo Administrativo como parte del conjunto de alternativas que conforman el sistema, que viabilizan la consecución de los objetivos de la investigación.

La elaboración del Módulo Administrativo está enfocado en tecnologías que permitan la interacción con el usuario de forma fácil, ágil e intuitiva. De esta forma, los principales beneficiarios serán los representantes que constituyen la universidad como son alumnos, docentes y personal administrativo de la Sede Quito, campus Sur.

ABSTRACT

The process of modernization and educational reform being implemented in the country, requires university training centers specifically to the Salesian University, to implement management alternatives that optimize the use and administration of physical spaces, and thus optimize the next class periods the use of classrooms, laboratories, audio, etc.. In careers that are taught in college automatically by applying specific parameters that enable the definition of interactive structures.

Based on the problematic development of a system to facilitate routine processes present each semester as review of Excel files for each race, manual creation of distributive and schedules, among others arises. "UPS Schedule" was divided into several modules by their complexity; based on this condition, the project focuses on the Administrative Module as part of the set of alternatives that make up the system, that make possible the attainment of the objectives of the research.

The development of the Administrative Module is focused on technologies that allow interaction with the user in an easy, fast and intuitive. Thus, the main beneficiaries will be the representatives that are like college students, faculty and staff from Headquarters Quito, South campus.

INTRODUCCIÓN

El trabajo de titulación forma parte del sistema UPS Schedule, un sistema para generar horarios y administrar aulas de la Universidad Politécnica Salesiana Sede Quito, los módulos que se desarrollarán permitirá modificar la asignación de recursos y podrá mantener un histórico de los horarios que fueron asignados en periodos anteriores, designación de aulas a todas las carreras pertenecientes al campus.

Actualmente la Universidad Politécnica Salesiana, posee aulas subutilizadas y carreras con déficit de aulas, en este marco resulta imperativo establecer una administración organizada de las estructuras físicas de cada uno de los campus que conforman la sede Quito, aspectos que son considerados en el sistema mejorado para la asignación de espacios físicos.

En los capítulos siguientes se describen de manera sucinta las siguientes temáticas: Análisis y diseño del sistema, construcción, pruebas, integración entre el resto de módulos que conforman el sistema UPS Schedule y finalmente como resultado de la investigación se formulan conclusiones y recomendaciones que puedan ser utilizados para futuras investigaciones.

Planteamiento del problema

Actualmente la Universidad Politécnica Salesiana, no tiene un sistema que permita administrar las aulas de los campus, la asignación de espacios físicos de acuerdo a las carreras y proyectos que se desarrollan, ni tampoco de un sistema que valide que aulas se puedan reasignar a una carrera o proyecto y evitar el inconveniente de tener materias sin aulas por falta de coordinación. En este marco, la Universidad Politécnica Salesiana Sede Quito, tiene la necesidad de implantar un nuevo sistema que sea intuitivo y amigable, y que esté acorde a las necesidades del usuario, facilitando de esta manera el uso y administración.

Con el módulo de administración lo que se propone es dar el control al usuario administrador sobre todos los recursos físicos que posee la universidad como crear, actualizar, asignar y permitir crear las reglas de negocio que se van a aplicar en los

diferentes módulos del sistema, así como también generar nuevos periodos donde se considere toda la información necesaria para el inicio del mismo.

Objetivo general

Obtener un módulo funcional para la administración de campus, administración de espacios físicos y administración de parámetros que trabaje de forma integrada para el sistema UPS Schedule de la Universidad Politécnica Salesiana Sede Quito.

Objetivos específicos:

- Analizar el proceso de administración de campus, espacios físicos y parámetros.
- Definir los requerimientos específicos del módulo administración de campus, espacios físicos y parámetros.
- Establecer interfaces de comunicación con los otros módulos del sistema.
- Diseñar el módulo en base a la arquitectura base del sistema.
- Programar el módulo utilizando herramientas de software libre.
- Probar el funcionamiento del módulo y la correcta interacción con el resto del sistema.
- Integrar el módulo al sistema UPS Schedule.

Justificación del proyecto

Ante la problemática que la universidad presenta, el módulo de administración aportará al sistema el manejo de infraestructura de la sede como también los parámetros o reglas que otros módulos utilizarán. Adicional será el encargado de generar los nuevos periodos lectivos valiéndose de la información apropiada.

El propósito del módulo de administración es el de optimizar tiempo, esfuerzo y recursos. Para ello, se crearán cuatro opciones: tipo de estructura, estructura, parámetros y periodo.

De esta manera, se coadyuvará en la resolución de la problemática prevalente: la falta de asignación de recursos, la carencia de sistemas especializados en administración y organización de los campus. Situación que se verá reflejado en los procesos del Departamento Técnico de Administración, en beneficio de los principales representantes que constituyen la universidad como son alumnos, docentes y personal administrativo.

Alcance del proyecto

UPS Schedule

Se va a desarrollar una solución informática que brinde soporte al proceso de administración de campus, espacios físicos, parámetros y generación de horarios para minimizar los problemas que el proceso actual tiene.

El proyecto está distribuido en siete módulos, donde cada módulo será desarrollado con la mentalidad de facilitar las tareas al usuario. El documento de titulación abarcará únicamente el Módulo Administrativo.
Todos los grupos de trabajo se deben basar en ciertos lineamientos generales para el desarrollo del sistema los cuales se detallan a continuación:

- El sistema es una aplicación web que se desarrollará en base al documento de requerimientos que se encuentra en el portal de Sourceforge del sistema. http://sourceforge.net/projects/upsschedule/files/docs/
- Las interfaces de usuario serán lo más intuitivas y amigables posibles.
- Se establecerá un estándar para la nomenclatura y etiquetado de clases, métodos, variables, tablas, campos. Esté será adoptado por todos los grupos de trabajo.
- Se utilizará una arquitectura basada en un modelo MVC para el desarrollo del sistema.

Módulo Administrativo

Este módulo está dirigido para un usuario con perfil de administrador ya que en éste realizará la carga inicial de información que será utilizada en los otros módulos, como principal función tiene la administración de los tipos de estructuras creación y asignación de estructuras, generará de forma automática nuevos periodos en base a información precedente.

Como aporte para el resto de grupos se creará una opción de parámetros donde podrán especificar valores estáticos que se usarán durante la ejecución del sistema.

Tipo de estructura

Aquí se definirán los tipos de espacios físicos que existen dentro de la Universidad Politécnica Salesiana Sede Quito, permitiendo la creación, modificación y eliminación de campus, bloques, aulas y laboratorios.

Estructura

Esta opción se encargará de presentar como se encuentran organizados y distribuidos los espacios físicos existentes dentro de la Universidad Politécnica Salesiana Sede Quito.
Permitirá crear, modificar y eliminar objetos de los tipos de estructura definidos y asignarlos a donde corresponden obteniendo una estructura escalonada en forma de árbol.

El usuario podrá imprimir y exportar toda la información generada a tipos de archivos Word, Excel y PDF.

Parámetros

En esta opción se definirán las reglas como tal del sistema, se especificarán los parámetros que serán constantes para todos los módulos, como por ejemplo el tiempo de sesión que tendrá cada uno de los usuarios que ingrese.

Los valores que aquí se definan dependen de las necesidades que tengan cada uno de los grupos durante el desarrollo.

Periodo

Se encargará de generar automáticamente un nuevo periodo en base a la información existente del periodo actual. Una vez que se proceda a crear el nuevo periodo el sistema solicitará la confirmación por parte del usuario administrador para iniciar el proceso, puesto que creado el periodo no se podrá reversar.

De igual forma presenta los periodos que han sido creados y despliega el periodo en el que se encuentra actualmente.

CAPÍTULO 1

ANÁLISIS Y DISEÑO

El propósito de este capítulo es describir de forma detallada la solución que se va a implementar, los requerimientos de software, y se muestra el análisis de viabilidad del proyecto.

1.1 Análisis de viabilidad

Para el análisis se consideró tres aspectos importantes: viabilidad técnica, económica y operacional.

1.1.1 Viabilidad técnica.

El sistema será implementado en ambiente web, ofreciendo páginas dinámicas y un acceso a la aplicación desde cualquier dispositivo y desde cualquier lugar.

Se ha utilizado software libre y con la plataforma GNU/Linux, para un correcto funcionamiento.

Las tecnologías usadas serán:

- Motor de base de datos Postgres
- X-Prime
- Primefaces

Se dispone del hardware necesario en la Universidad Politécnica Salesiana Sede Quito, para implantar el sistema permitiendo un funcionamiento correcto.

Características:
1 Gabinete para servidores Blades, con los siguientes componentes:

- 1 Enclosure Blade c3000, con conexión eléctrica bifásica
- Switches de Lan 1GB E2 para Enclosure c3000
- Sistema de ventiladores (4 instaladas) y fuentes redundantes (2 instaladas)
- 1 Módulo On Board administración

Servidores para aplicaciones:

Servidor BL460C con el siguiente detalle:

- 1 Procesador Intel Xeon QuadCore, 2.33 GHz 1x4MB cache
- 4 GB RAM
- 1 Smart Array RAID 1
- Discos de 146GB 10k SAS HDD

Los servidores actualmente se los está configurando para que tengan salida a Internet.

El hardware utilizado por los estudiantes son:

Computador portátil Sony VAIO

- Procesador: Intel CORE i3
- Memoria: 6 GB
- Disco: 500 GB
- Sistema: Windows 8
- Arquitectura: 64 bits
- Funciones: desarrollo, pruebas y base de datos

Computador portátil HP Pavilion dv4

- Procesador: AMD Dual - Core
- Memoria: 4 GB
- Disco: 500 GB
- Sistema: Windows 7
- Arquitectura: 64 bits
- Funciones: desarrollo, pruebas y base de datos

Por lo tanto técnicamente el software utilizado es viable por cuanto permite un funcionamiento eficaz y eficiente del sistema acorde a los requerimientos de la investigación.

1.1.2 Viabilidad económica.

Los gastos del proyecto son mínimos, ya que la Universidad Politécnica Salesiana dispone del hardware necesario para el correcto funcionamiento del sistema.

Los gastos del desarrollador son los consumibles (hojas, tinta de impresión, etc.) ya que el desarrollo del sistema se lo hará durante un año y utilizando los siguientes recursos:

Tabla 1. Detalle de la viabilidad económica

PRESUPUESTO: Administración de campus, espacios físicos y parámetros para el sistema UPS Schedule para la Universidad Politécnica Salesiana Sede Quito**.**

Ítem	Rubro		Unidad	Cantidad (N°)	Costo mensual ($)	Costo total ($)
1	**Recurso humano**					
	Analistas- Programador(Autores tesis)		Personas	2	640.00	1,280.00
					Total recurso humano ($)	**1,280.00**
2	**Recurso Material**			**Cantidad (N°)**	**Costo unitario ($)**	**Costo total ($)**
	Equipos cómputo	PC Escritorio		1	450.00	450.00
		Portátil		2	850.00	1,700.00
		Impresora		1	75.00	75.00
	Materiales de oficina	Resma papel Bond		2	3.00	6.00
		Copias		200	0.03	6.00
		Solicitudes		5	0.50	2.50
		Matricula y derecho		2	635.00	1270.00
					Total Recurso Material ($)	**3,509.50**
3	**Otros Servicios**			**Cantidad (N°)**	**Precio unitario ($)**	**Total ($)**
	Luz eléctrica			--	25.00	25.00
	Teléfono			--	20.00	20.00
	Agua			--	15.00	15.00
	Internet			--	23.00	23.00
					Total otros servicios($)	**83.00**
					Total presupuesto	**4,872.50**

Nota: El presupuesto total establecido para la investigación, serán asumidos por los tesistas.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

1.1.3 Viabilidad operacional.

Se puede obtener una gran ventaja en la parte operacional ya que en la actualidad la UPS no cuenta con un sistema automatizado para generar horarios y asignar espacios físicos, optimizando el tiempo invertido en los procesos manuales por parte de los

usuarios, el mejoramiento del manejo y control de los recursos para la obtención de un sistema eficiente.

1.2 Análisis de requerimientos

Se va a describir en forma detallada los requerimientos de software del módulo Administrativo para la Universidad Politécnica Salesiana Sede Quito.

En los siguientes apartados se podrá conocer detalladamente la funcionalidad y las operaciones que realizara el Módulo Administrativo en base a las necesidades que actualmente se presentan.

1.2.1 Propósito.

El propósito de este documento es recolectar de forma detallada las necesidades del personal administrativo de la universidad, base para el desarrollo del módulo administrativo.

Definir las interfaces necesarias y su funcionalidad, las operaciones y reglas fundamentales para la elaboración de nuevos periodos lectivos.

1.2.2 Alcance.

Este módulo será creado para la administración del sistema UPS Schedule, específicamente crear un nuevo periodo, administrar una estructura, administrar los parámetros del sistema, administrar el tipo de estructura y generar reportes de la estructura de cada campus.

1.2.3 Definiciones y abreviaturas.

- **UPS**

  Abreviatura de Universidad Politécnica Salesiana.

- **Parámetros**

  Son reglas que son constantes para todo el sistema que caracterizan al sistema UPS Schedule.

- **Estructura**

  Hace referencia a los espacios físicos de la UPS como la sede, campus, bloques, aulas con sus respectivas descripciones.

- **Periodo**

  Es un número que hace referencia al semestre cursado y puede tener diferentes estados entre estos están activo, inactivo.

- **Horario**

  Son las horas de clases asignadas a un docente o a un espacio físico.

- **Distributivo**

  Son las materias asignadas a cada docente al iniciar el nuevo periodo académico.

- **Tipo de estructura**

  Son los diferentes tipos de estructuras o espacios físicos que pueden existir.

- **Tipo de aula**

  Son todos los diferentes tipos de aulas que puede tener la UPS, entre estas pueden estar regulares, CECASIS, laboratorio.

- **Estado de horario**

  Son todos los estados que puede tener un horario entre estos pueden estar cruce, correcto y otros.

1.2.4 Descripción general.

- **Perspectivas del producto**

Este módulo es dependiente del módulo gestión de usuarios, seguridad y auditoria que pertenecen al sistema UPS Schedule. Requiere una base de datos y un servidor de aplicaciones.

- **Funciones del producto**

Los usuarios acceden a través de un proceso de identificación mostrando las siguientes funciones.

- ✓ **Crear periodo**

Mediante esta función el administrador del sistema podrá crear un nuevo periodo duplicando los datos del periodo anterior activo, copiando los datos distributivos y horarios pertenecientes al periodo anterior.

Este proceso se realiza mediante consultas a la base de datos que proporciona como resultado el periodo anterior activo con su id y se duplica los datos con un nuevo id y un nuevo número de periodo, mediante el id proporcionado anteriormente se hace otra consulta para que muestre los distributivos pertenecientes a este periodo, los datos obtenidos se los duplica, guardando los id_distributivo obtenidos de la consulta y los nuevos generados en un array para luego utilizarlos en la consulta para duplicar los datos de los horarios, para estos se hace consultas de todos los distributivos del periodo anterior y con los datos obtenidos se procede a duplicarlos.

Cabe mencionar que este proceso puede tener un tiempo estimado de dos a cuatro minutos.

- ✓ **Gestión estructura**

El administrador podrá crear, modificar o eliminar una estructura. La estructura es un espacio físico que puede ser una sede, campus, bloque, aula.

- ✓ **Gestión de parámetros**

Mediante esta funcionalidad se definirá las reglas como tal al sistema. Se especificara los parámetros que serán constantes para todo el sistema, como por ejemplo el tiempo de sesión que tendrá cada uno de los usuarios que ingrese, definir detalles

claves de periodos y además se definirán parámetros complementarios que caracterizan al sistema UPS.

El administrador podrá crear, borrar y actualizar los parámetros.

- ✓ **Reporte**

Este proceso permite imprimir y exportar en formato hojas de cálculo, Word o PDF, mostrando los espacios físicos de un campus.

- ✓ **Gestión tipo de estructura**

Mediante esta función el administrador podrá gestionar el tipo de estructura o espacio físico al cual pertenece, este puede ser un bloque, aula, campus y otros.

- ✓ **Gestión tipo de aula**

Mediante esta función el administrador podrá gestionar el tipo de aula que puede existir en la UPS y puede ser regular, CECASIS, laboratorios.

- ✓ **Gestión estado de horario**

Mediante esta función el administrador podrá gestionar los diferentes estados que puede tener un horario y pueden ser cruce, correcto y otros.

- **Características de usuarios**

El software ha sido desarrollado para tres tipos de usuarios. Cada uno de los usuarios tendrá sus privilegios y restricciones según su perfil dentro del sistema.

- ✓ Administrador: Persona encargada de gestionar el sistema, accediendo a las funciones del Módulo Administrativo.

- **Restricciones**

Los módulos manejados permiten el ingreso solo del administrador. El administrador puede ingresar, actualizar y borrar la información que desee.

La creación del nuevo periodo se la podrá realizar cada seis meses.

El sistema debe ser instalado en los servidores de la UPS, utilizar un gestor de base de datos, un servidor de aplicaciones y su acceso deberá ser a través de la intranet de la institución.

- **Suposiciones y dependencias**

Este documento de requerimientos se basa en la estructura actual de la UPS Sede Quito, permitiendo saber que espacios físicos están disponibles y ayudando a su correcta utilización.

- **Requisitos futuros**

Este sistema podrá ser implantado para otras sedes de la universidad.

1.2.5 Requerimientos específicos.

Requerimientos de interfaz externo

- **Interfaces de usuario**

Las interfaces de usuario serán amigables y fáciles de usar para el usuario.

- **Interfaces de hardware**

La visualización del sistema debe respetar los estándares actuales de resolución (1024 x 768 pixeles).

- **Interfaces de software**

El módulo administración debe permitir la integración con otros módulos pertenecientes al Sistema UPS Schedule.

1.2.6 Requerimientos funcionales.

Para definir los requerimientos funcionales se ha utilizado la plantilla organizada por clase de usuarios.

- **Administrador del sistema**

Tabla 2. Requerimiento funcional 1.1: Gestión de parámetros

Descripción	Agregar, modificar y eliminar un parámetro para uso en todo el sistema.
Precondición	Estructura
Entrada	Estructura, nombre, valor, estado.
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 3. Requerimiento funcional 1.2: Gestión de estructura

Descripción	Agregar, modificar y eliminar un espacio físico.
Precondición	Tipo de estructura, tipo de aula, usuario
Entrada	Datos de la estructura según el tipo de estructura: nombre, dirección, total de pisos, área, número de pisos, capacidad, observaciones
Proceso	Datos modificados en la base de datos.
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 4. Requerimiento funcional 1.3: Gestión de nuevo periodo

Descripción	Generación de nuevo periodo
Precondición	Exista un periodo creado y activo
Entrada	Ninguna
Proceso	Duplicar datos existentes de periodo, horario, distributivo.
Salida	Mensaje de aceptación de nuevo periodo creado/ Error.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 5. Requerimiento funcional 1.4: Reporte

Descripción	Reporte estructura
Precondición	Exista una estructura campus ingresada
Entrada	Ninguna
Proceso	Genera un reporte en PDF, Excel, Word

Salida	Documento del tipo escogido.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 6. Requerimiento funcional 1.5: Gestión de tipo estructura

Descripción	Agregar, modificar y eliminar un tipo de estructura para uso de una estructura
Precondición	Exista un tipo estructura padre
Entrada	nombre, id_padre
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 7. Requerimiento funcional 1.6: Gestión de estado

Descripción	Describe un estado de las tablas tbl_parámetro y tbl_estructura y los estados pueden ser activo, inactivo y otros
Precondición	Ninguna
Entrada	Nombre
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 8. Requerimiento funcional 1.7: Gestión de estado horario

Descripción	Agregar, modificar y eliminar un estado que puede tener un horario
Precondición	Ninguna
Entrada	nombre, estado
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 9. Requerimiento funcional 1.8: Gestión de tipo aula

Descripción	Agregar, modificar y eliminar un tipo de aula que existe en la UPS
Precondición	Ninguna
Entrada	nombre, observaciones
Proceso	Datos modificados en el sistema
Salida	Mensaje de aceptación / Error

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

1.2.7 Requisitos de rendimiento.

- El sistema deberá estar disponible (24/7) 24 horas y / 7 días a la semana.

- Deberá soportar 10 usuarios conectados.

1.2.8 Restricciones del diseño.

Las herramientas y plataformas usadas deberán ser 100% en software libre.

1.2.9 Atributos de sistemas de software.

Seguridad

El sistema deberá permitir el ingreso a usuarios autorizados, este módulo será manejado por el usuario administrador.

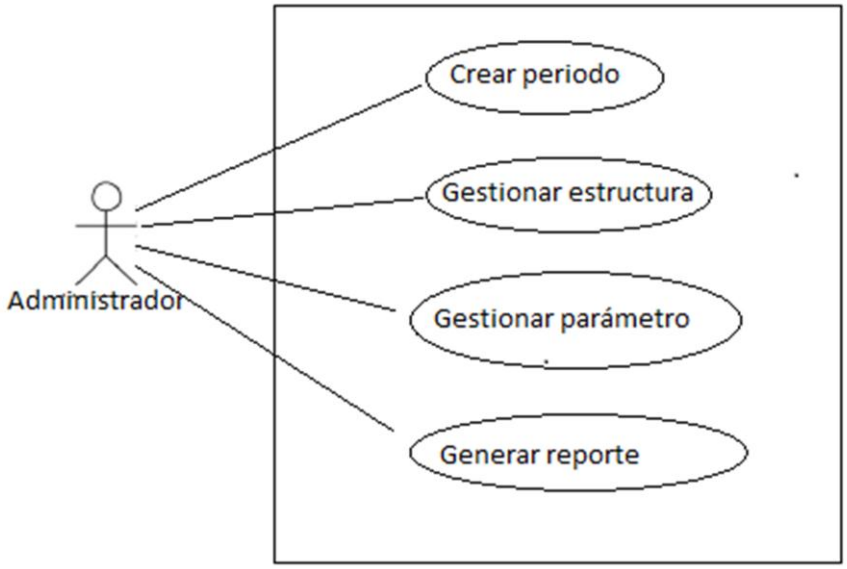
1.3 Diseño

El diseño del sistema podrá dar una solución de forma automática a un proceso repetitivo que se ha venido dando en la UPS, permitiendo ganar tiempo en los procesos al iniciar un nuevo periodo.

1.3.1 Diagramas de casos de uso.

En los diagramas de casos de uso se podrá visualizar las principales funciones del módulo.

Figura 1. Caso de uso 1. Administrador



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

El actor que interviene en la figura 1, es el administrador del sistema y muestra las diferentes funciones que puede gestionar en la aplicación.

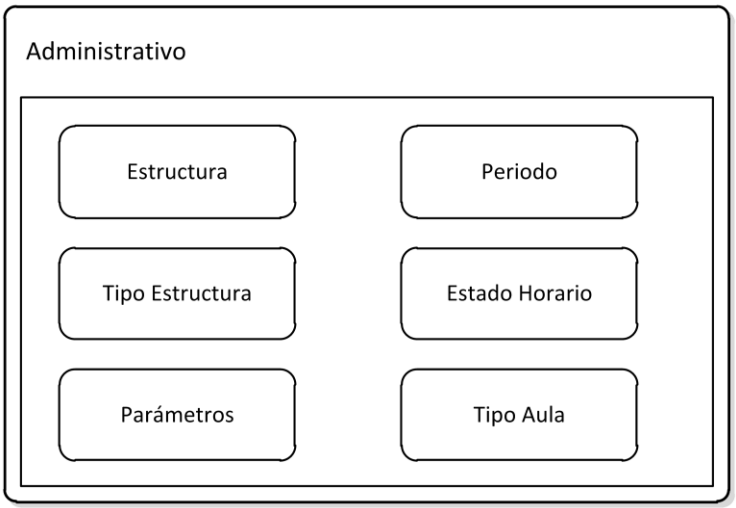
Cabe mencionar que las funciones de esta figura solo serán gestionadas por el administrador del sistema.

1.3.2 Diagrama de bloques en general.

El sistema consta de diferentes módulos, pero en base a los requerimientos y al caso de uso se define el Módulo Administrativo.

El Módulo Administrativo contiene sub módulos funcionales que implementarán los requerimientos planteados por los usuarios.

Figura 2. Diagrama de bloques general



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

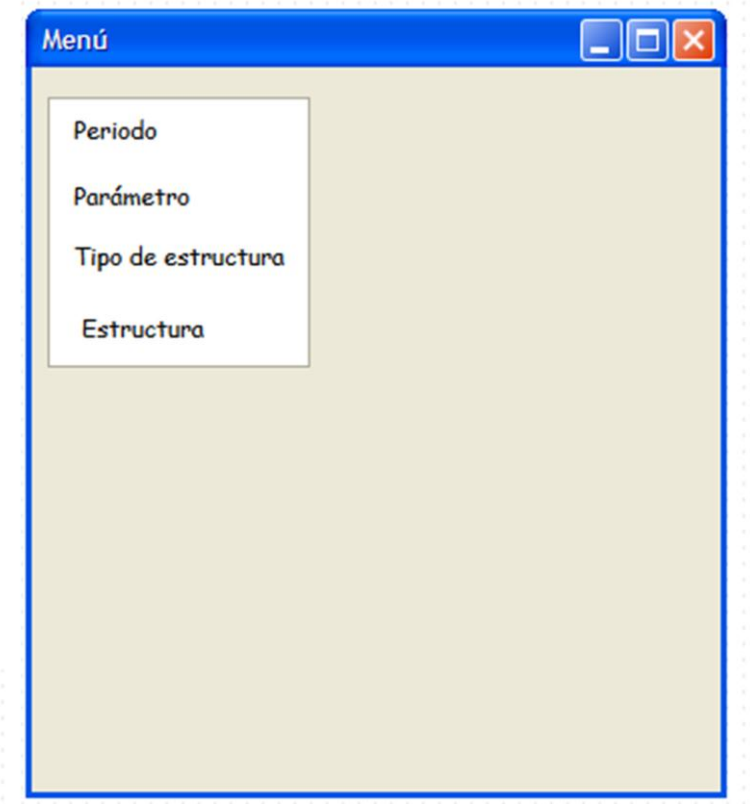
1.3.3 Interfaces de usuario.

Para el diseño de las interfaces del usuario se ha considerado criterios de simplicidad, seguridad, para que sean agradables para los usuarios.

- **Opciones del Módulo Administrativo:**

En la pantalla principal se podrá observar un menú con las opciones del módulo administrativo.

Figura 3. Opciones del Módulo Administrativo



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

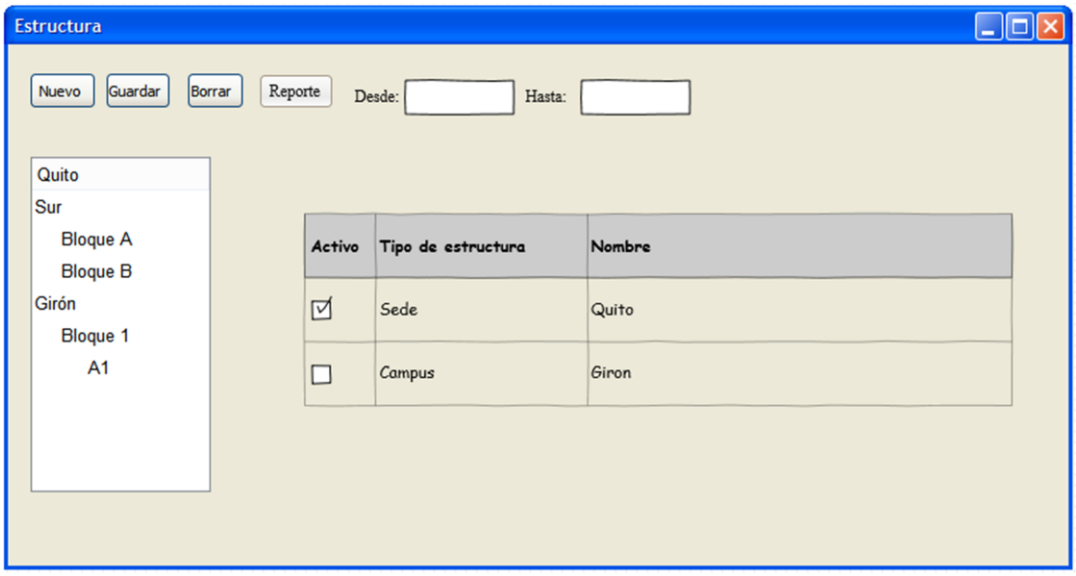
- **Interfaz estructura:**

Esta pantalla se observará las opciones para gestionar una estructura, para crear, modificar, eliminar. A está pantalla tendrá acceso únicamente el administrador del sistema.

Además, se puede observar que cuenta con un árbol que muestra la información en forma jerárquica de una estructura, clasificándola por el tipo de estructura al que pertenece cada espacio físico y facilitando la interpretación.

Existe una opción para generar un reporte, el cual permita mostrar la información de una estructura según el campus al cual pertenece. Este reporte podrá generarse en tres tipos de formatos PDF, Excel, Word

Figura 4. Interfaz estructura

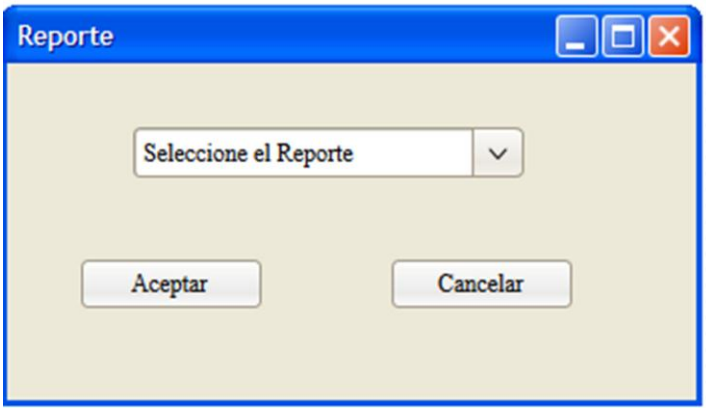


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz del reporte de estructura:**

En la siguiente figura muestra la pantalla del reporte, que crea el reporte según el campo de la estructura.

Figura 5. Interfaz del reporte de estructura

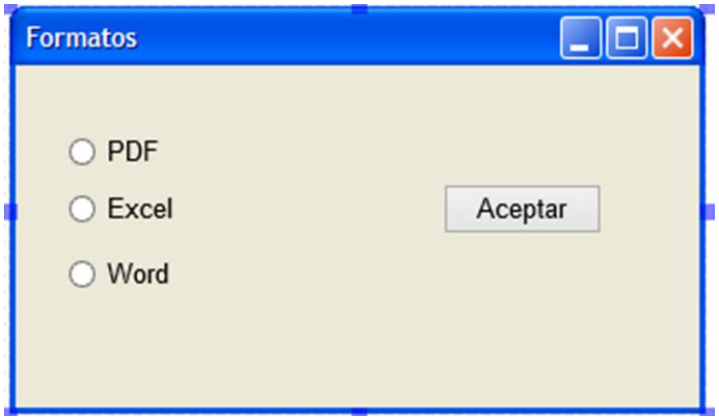


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Formatos en los que se va a realizar el reporte:**

Muestra los formatos con los cuales se pueden realizar los reportes.

Figura 6. Selección de formatos del reporte

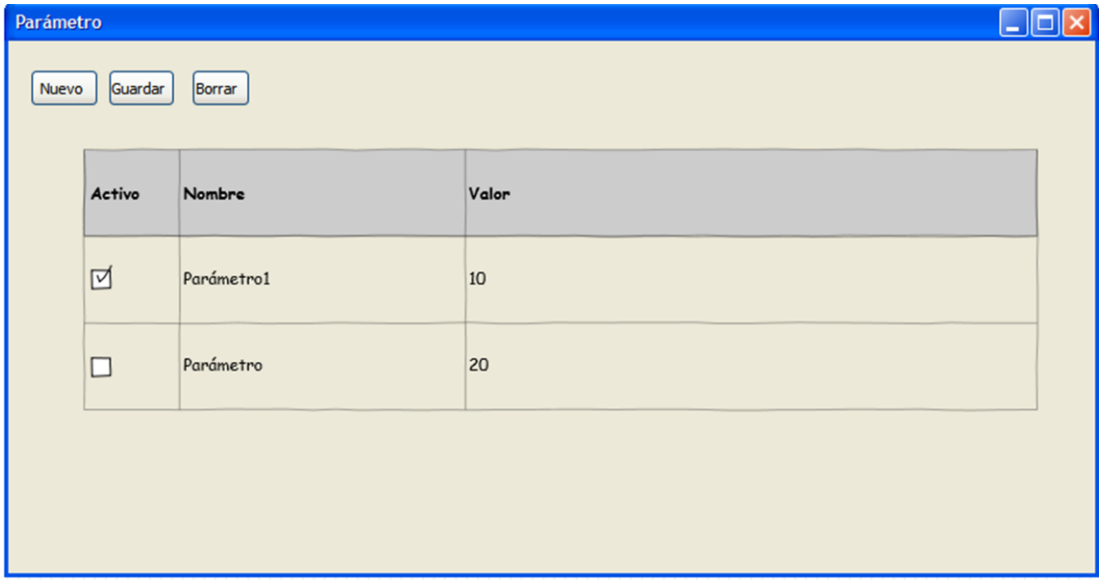


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz parámetro:**

Está interfaz contiene una barra para gestionar un parámetro, se puede crear, modificar y eliminar un parámetro.

Figura 7. Interfaz parámetro



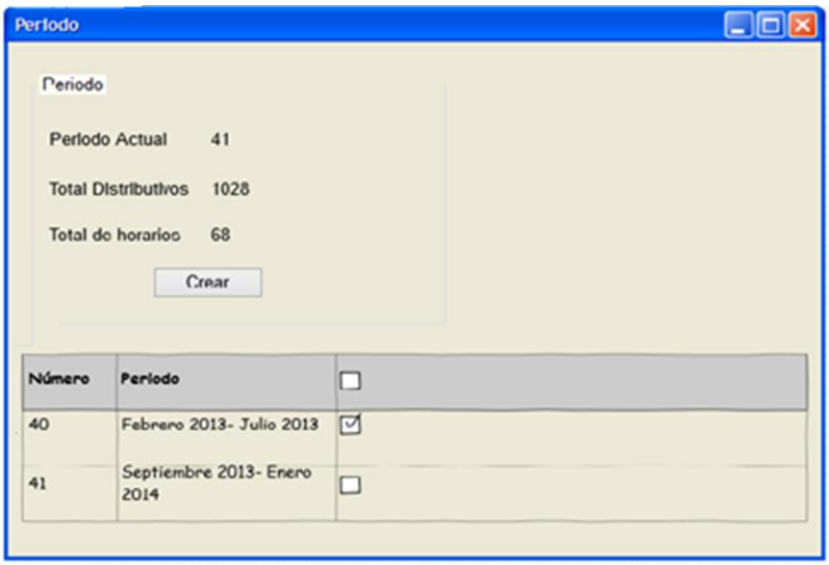
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz periodo:**

En está interfaz se puede observar el periodo activo con sus características, además se puede observar una tabla con todos los periodos existentes ya sean activos o inactivos.

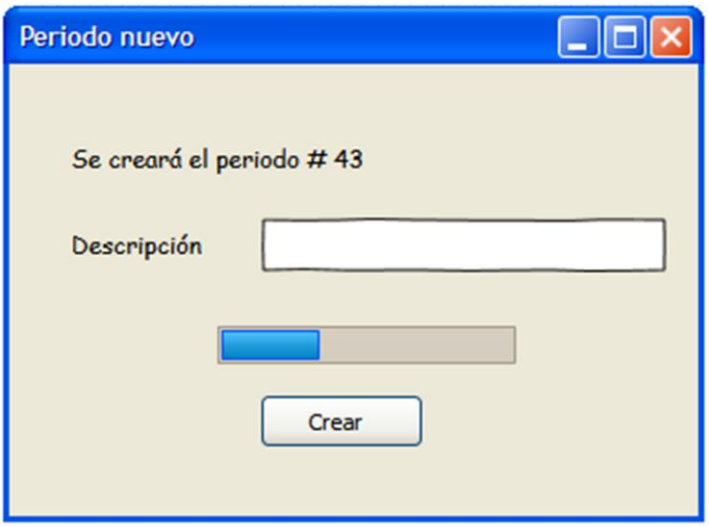
Esta interfaz crea un nuevo periodo, ingresando la descripción del periodo nuevo.

Figura 8. Interfaz periodo



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 9. Interfaz creación de nuevo periodo

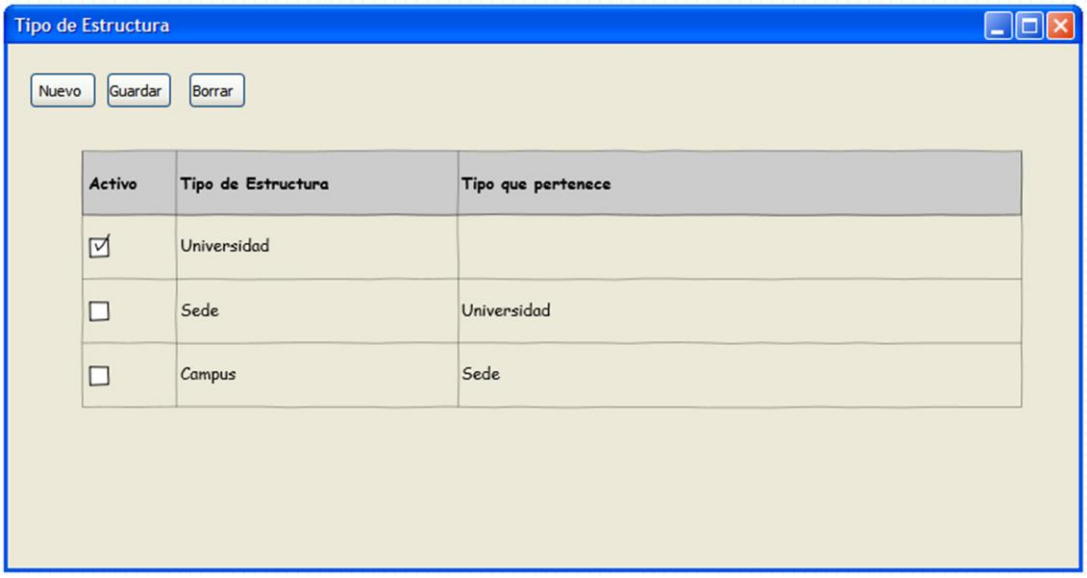


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz tipo de estructura:**

Está interfaz gestiona un tipo de estructura, también crea, modifica y elimina.

Figura 10. Interfaz tipo de estructura

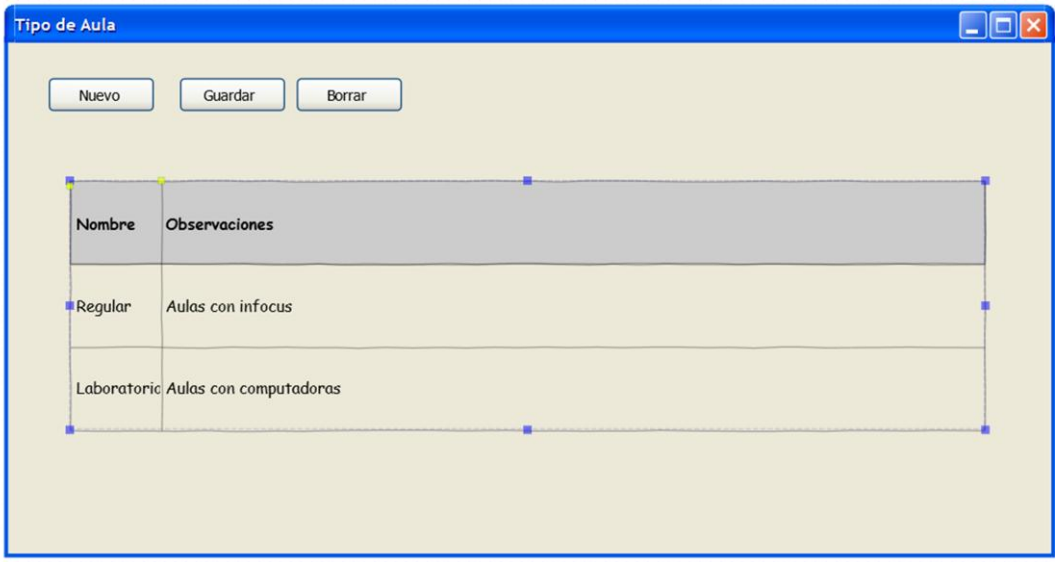


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz tipo de aula:**

Está interfaz gestiona un tipo de aula, crea, modifica y elimina.

Figura 11. Interfaz tipo de aula

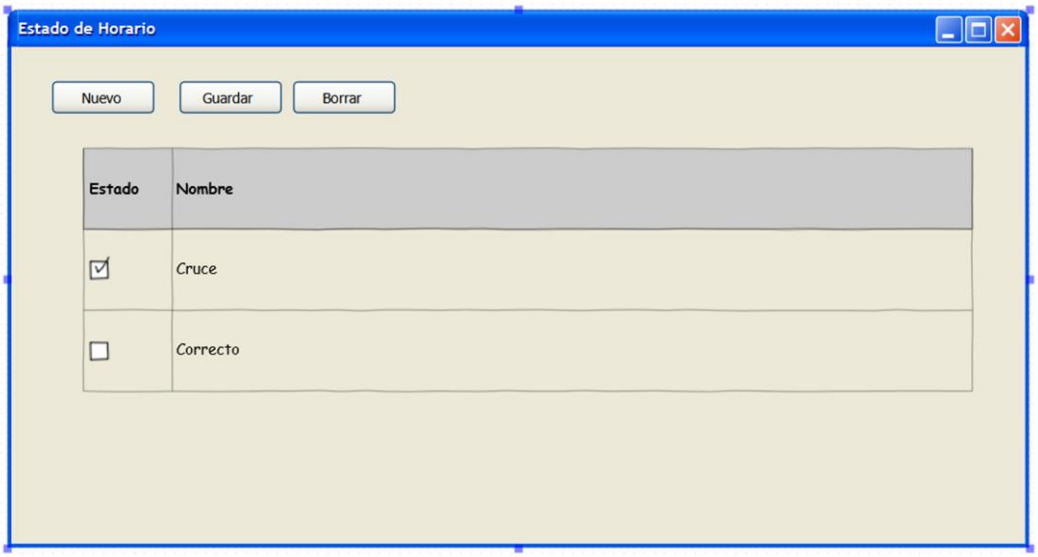


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Interfaz estado de horario:**

Está interfaz gestiona un estado de horario, también crea, modifica y elimina.

Figura 12. Interfaz estado de horario



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

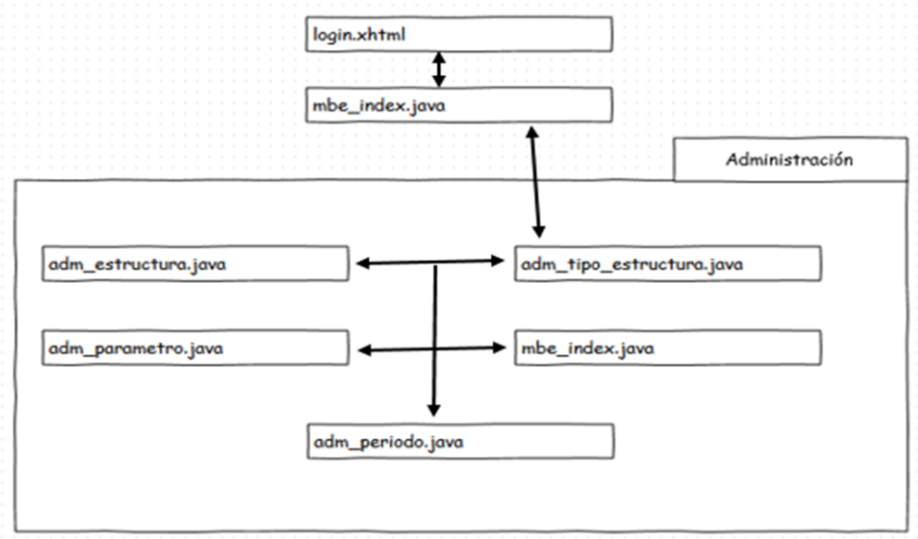
1.3.4 Mapa de navegación.

Mapa de navegación para módulo administración

Los usuarios ingresan por la misma página de autenticación.

La página de bienvenida muestra las opciones proporcionadas por el sistema, desde aquí se podrá gestionar los diferentes módulos proporcionados por el sistema UPS Schedule, a continuación se muestra el mapa de navegación para el módulo administración.

Figura 13. Mapa de navegación del Módulo Administración



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

En la Figura 13, se observa el mapa de navegación del módulo Administración en el cual el administrador debe ingresar al sistema siguiendo los siguientes pasos:

- Ingresar el usuario y la contraseña en la página Login.java.
- Si los datos están correctos ingresará al sistema y podrá observar la página principal Index.jsf.
- En la página Index.jsf el usuario administrador podrá navegar por el menú administración que contiene diferentes opciones.
- El usuario administrador podrá administrar estructuras, parámetros, tipo de estructura, tipos de aula, estados de horarios y periodo.
- Adm_estructura.java administra una estructura (crear, modificar y eliminar).
- Adm_tipo_estructura.java administra el tipo al que pertenece una estructura (crear, modificar y eliminar).
- Adm_parámetro.java administra los parámetros que va a usar el sistema (crear, modificar y eliminar).

- Adm_tipo_aula.java administra el tipo de aula existente en la UPS (crear, modificar y eliminar).

- Adm_estado_horario.java administra el estado del horario (crear, modificar y eliminar).

- Adm_crear_periodo.java crea un nuevo periodo utilizando los distributivos y los horarios del último periodo existente activo.

1.3.5 Base de datos.

La base de datos pertenece a un proyecto mucho más grande del Sistema UPS Schedule para la Universidad Politécnica Salesiana, el módulo en el que se ha trabajado utiliza las siguientes tablas:

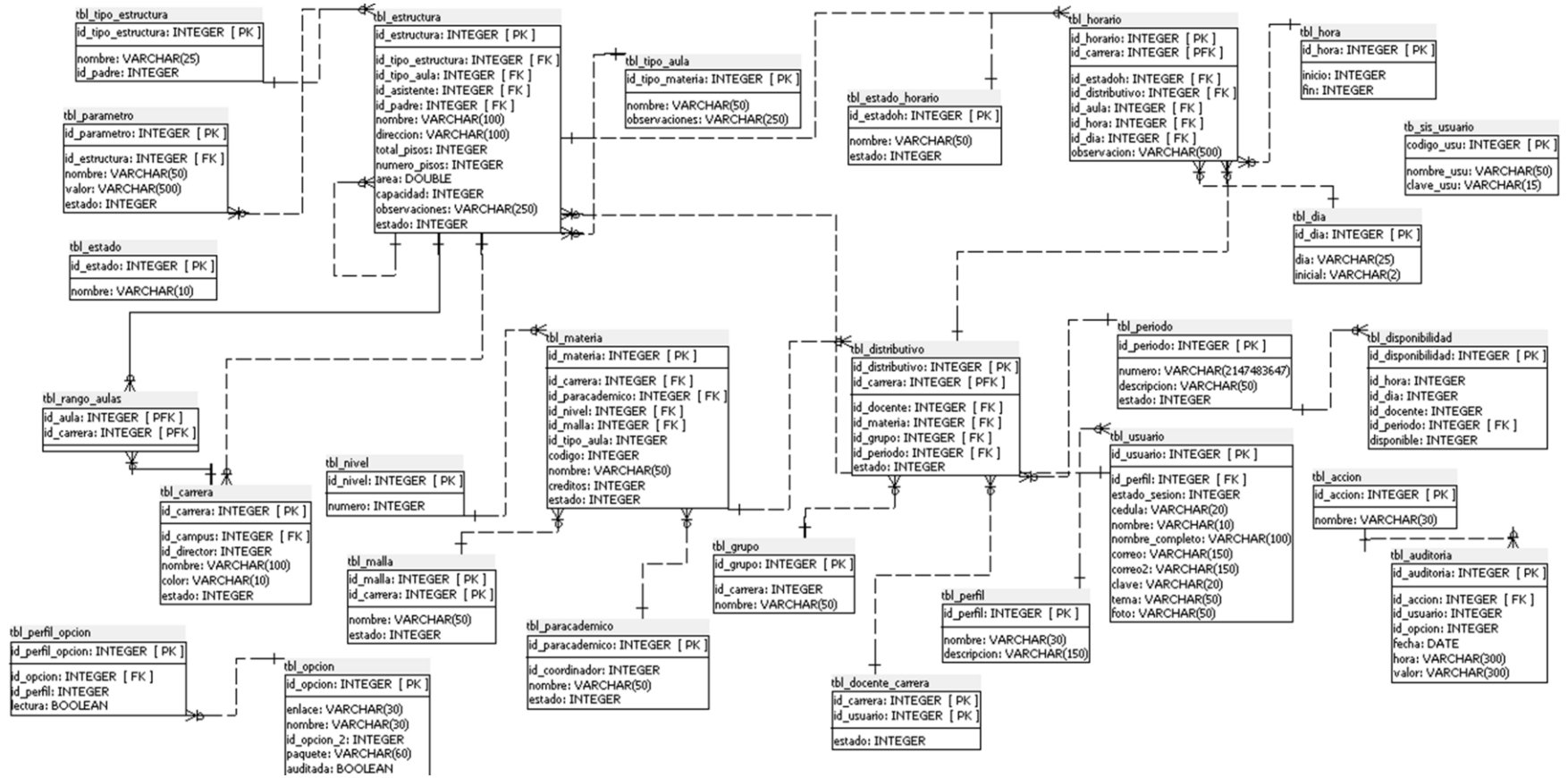
- tbl_parámetro
- tbl_periodo
- tbl_tipo_estructura
- tbl_estructura
- tbl_distributivo
- tbl_horario
- tbl_estado
- tbl_estado_horario
- tbl_tipo_aula

Tabla 10. Descripción de las tablas de la base de datos

Nombre	**Descripción**
tbl_estructura	Muestra la información de un espacio físico.
tbl_periodo	Muestra la información de un semestre.
tbl_tipo_estructura	Muestra la información del tipo de estructura a la que pertenece una estructura.
tbl_parámetro	Muestra la información de las constantes que van a ser usadas durante la ejecución del sistema.
tbl_distributivo	Contiene la información de las materias asignadas a cada docente y el espacio físico asignado.
tbl_horario	Contiene la información de los horarios de clases.
tbl_estado	Tabla maestra indica el estado de las tablas que heredan.
tbl_tipo_aula	Muestra la información de los tipos de aulas que existen en la UPS
tbl_estado_horario	Muestra la información de los estados que pueden tener los horarios

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 14. Diagrama de la base de datos del sistema UPS Schedule



Fuente: Tesistas “Sistema UPS Schedule de la Universidad Politécnica Salesiana sede Quito”

CAPÍTULO 2

CONSTRUCCIÓN Y PRUEBAS

2.1 Plataformas y herramientas

El sistema UPS Schedule es multiplataforma, es desarrollado en Java, puede ser ejecutado en cualquier sistema operativo, ya sea Mac, Windows, Linux, entre otros; porque no depende del lenguaje de máquina del CPU, maneja una máquina virtual (JVM), la misma que interpreta y ejecuta el código Java. Existe muchas JVM, una para cada tipo de CPU y sistema operativo.

Para el desarrollo del sistema UPS Schedule y los módulos que lo conforman se ha tomado en cuenta las siguientes herramientas:

- Glassfish, servidor de aplicaciones.
- JDK, herramienta para desarrollo de Java.
- PostgresSQL, servidor de base de datos.

2.1.1 NetBeans IDE 7.2.1.

NetBeans es un entorno de desarrollo de código abierto, multiplataforma, multilingüe, desarrollado principalmente para Java pero, puede ser utilizado por otros lenguajes de programación. Posee varias librerías que permiten el desarrollo de aplicaciones a partir de componentes de software denominados módulos. Estos módulos son un conjunto de clases de Java que interactúan con las API's (Interfaz de programación de aplicaciones) de NetBeans.

2.1.2 JDK.

Herramienta desarrollada por la empresa Sun Microsystems, que crea compila y ejecuta aplicaciones Java, se la puede conseguir de forma gratuita en su portal web en su página oficial, cada JDK viene con su versión Java.

2.1.3 Power Architect.

Es una herramienta de modelado de datos diseñada para permitir a los usuarios crear sus modelos de datos de manera fácil y simplificando esfuerzo, así como también realizar ingeniería inversa de las bases de datos existentes, realizar creación de perfiles de datos fuente, conectarse a base de datos específicas y generar automáticamente los metadatos de ETL "Extract, Transform and Load (Extraer, transformar y cargar en castellano) proceso que permite a las organizaciones mover datos desde múltiples fuentes, reformatearlos y limpiarlos, y cargarlos en otra base de datos." (Espinoza, 2010)

2.1.4 PostgreSQL.

Es un Sistema Gestor de Bases de Datos orientado a objetos de libre distribución, publicado bajo la licencia BSD (Berkeley Software Distribution).

El proyecto de Postgres, como varios de los proyectos de código abierto, es desarrollado por una comunidad que trabaja de manera desinteresada, altruista, sin fines de lucro y apoyados por Organizaciones Comerciales, dicha comunidad se denomina PGDG (PostgreSql Global Development Group).
Algunas de sus principales características son:

- Alta concurrencia que logra mediante un sistema denominado MVCC (Acceso Concurrente Multiversión por su siglas en inglés).
- Variedad de tipos nativos.
- Permite crear disparadores o Triggers.

2.1.5 JavaServer Faces JSF.

Es un framework de Java basado en web que simplifica el desarrollo de interfaces en aplicaciones de Java EE. Utiliza JavaServer Page (JSP) como la tecnología que permite el despliegue de páginas, pero también puede utilizar otras tecnologías como XUL (lenguaje XML para interfaz de usuario por sus siglas en inglés).

JSF incluye conjunto de API's (Interfaz de programación de aplicaciones), como componentes de la interfaz de usuario, maneja eventos, valida entradas, define esquemas de navegación de las páginas, entre otros. Proporciona un conjunto de

componentes para la interfaz de usuario incluyendo elementos HTML. Posee dos bibliotecas de etiquetas personalizadas para JSP que permiten mostrar una interfaz JSF dentro de una página JSP. (Perez, 2006)

2.1.6 PrimeFaces

Es una librería de componentes visuales de código abierto para JavaServer Faces (JSF), que posee un conjunto de componentes que facilitan la creación de aplicaciones web. PrimeFaces está bajo la licencia de Apache License V2.

Algunas características principales son:

- Soporte nativo de Ajax incluyendo Push/Comet.
- Kit para crear aplicaciones web móviles.
- Es compatible con otras librerías de componentes como RichFaces.
- Proyecto open source activo y estable.

Tabla 11. Comparativo con otras librerías (Lerma, 2010)

Característica	**ICEfaces**	**RichFaces**	**Primefaces**
Soporte de Ajax	Es transparente para el desarrollador, lo implementa de forma nativa en todos los componentes mediante la propiedad partialSubmit	Tenemos que hacer uso de Ajax4JSF, que no es tan transparente para el desarrollador, puesto que, además de introducir los componentes de RichFaces, tenemos que añadir componentes no visuales de la librería Ajax4JSF.	Es transparente para el desarrollador, aunque para activarlo deben utilizarse atributos específicos para lanzar un método del servidor y para indicar los componentes a actualizar.
Librerías en las que se basan	Usa el soporte de prototypejs, aunque la parte de Ajax la ha rescrito y para los efectos visuales utilizan script.aculo.us.	Usa el soporte de prototypejs y script.aculo.us, aunque soporta también jquery.	Utiliza el soporte de jQuery y jQuery UI para los efectos visuales.
Personalización de la interfaz de usuario	Incorpora el concepto de skins y distribuye 3 temas.	Incorpora el concepto de skins y distribuye 12 temas, aunque se pueden encontrar más en el repositorio de SNAPSHOTS.	Incorpora el concepto de skins, utilizando ThemeRoller, y dispone de 26 temas prediseñados.
Número de	Tiene 79 componentes en	Tiene 212 componentes	Tiene más de 90

(continúa)

Tabla 12. Comparativo con otras librerías (continuación)

componentes	la versión básica, a los que hay que sumar 32 de la versión empresarial, esta última es de pago. La percepción es que estan inviertiendo esfuerzos en mejorar la versión empresarial y, como es lógico, esperan obtener beneficio económico por ello.	entre los propios de RichFaces y los de Ajax4JSF. Con RichFaces todos los componentes son OpenSource y podemos usar un Pick List sin contratar nada, sin embargo, con ICEfaces sin queremos un Dual List o pagamos o lo implementamos nosotros.	componentes OpenSource, algunos muy avanzados como el HTMLEditor. Además dispone de un kit para crear interfaces web para teléfonos móviles.
Licencia	MPL 1.1, que cubre la LGPL V 2.1. Si bien disponen de una versión empresarial con licencia comercial.	LGPL V 2.1. en su totalidad.	Apache License V2
Relevancia	Ha sustituido a Woodstock como librería de componentes de referencia de Sun para el desarrollo de aplicaciones RIA. Se distribuye, por defecto, con NetBeans.	Es la librería de componentes visuales de Jboss, se integra, por defecto con Jboss Seam, aunque éste también soporta ICEfaces.	Ha sido una de las primeras librerías capaces de integrarse con JSF 2 y viene pisando fuerte debido a la diversidad y calidad de sus componentes. Puede utilizarse junto a Richfaces, pero no es compatible con ICEfaces.

2.1.7 Ajax.

Es una técnica desarrollo web proviene de sus siglas en inglés (Asynchronus JavaScript And XML), además de ser una tecnología asíncrona es una técnica válida para múltiples plataformas y utilizable en varios sistemas operativos y navegadores.

Ajax no es una tecnología por sí misma proviene de un conjunto de tecnologías que son las siguientes:

- XHTML y CSS
- DOM
- XML, XSLT y JSON,

- XMLHttpRequest
- JavaScript

2.1.8 GlassFish.

Es un servidor de aplicaciones, multiplataforma, de libre distribución desarrollado por Sun Microsystems, implementa tecnologías definidas en plataformas Java EE, permite implementar y ejecutar aplicaciones que siguen esta especificación.

Existe una versión comercial que es licenciada y una versión bajo licenciamiento CDDL y GNU GPL. GlassFish tiene como base al servidor Sun Java System Application Server que es un dericado de Apache Tomcat.

2.1.9 JUnit.

Es un conjunto de clases, que son utilizadas para realizar pruebas unitarias en aplicaciones Java, permite la ejecución de clases de una forma controlada para evaluar el funcionamiento de uno o varios métodos específicos y observar su comportamiento en función de valores determinados de entrada y obtener valores de retorno esperados.

2.1.10 JMeter.

Es una herramienta de pruebas de carga para analizar y medir el desempeño en aplicaciones prioritariamente en aplicaciones orientadas a la web. Desarrollado por Apache, también puede ser utilizado para obtener pruebas unitarias, pero mayoritariamente es catalogado como una herramienta de generación de carga.

2.1.11 JasperReports.

Es una herramienta de creación de informes y estadísticas que se compone de un conjunto de librerías Java, desarrollada por JasperSoft bajo la licencia libre GNU. Es el motor de informes Java más utilizado en todo el mundo, combina fuentes de datos y produce documentos ya sea para visualización, impresión o exportación a una variedad de formatos como por ejemplo PDF, HTML, XLS, entre otros.

Puede ser usado en gran variedad de aplicaciones Java incluyendo aplicaciones web o J2EE, JasperReport se usa por lo general con iReport para la edición de informes.

2.1.12 iReport.

Es un diseñador de informes visuales intuitivo y fácil de usar para JasperReports desarrollado en Java. Permite que los usuarios corrijan visualmente los reportes. iReport está integrado con JFreeChart una biblioteca gráfica OpenSource más difundida para Java.

2.1.13 SourceForge.

Es un sitio web que permite a los desarrolladores de software controlar y gestionar varios proyectos de software y que actúa como un repositorio de código fuente.

Es comercializado por Dice Holdings desde el 18 de septiembre del 2013. Es el principal repositorio y biblioteca de proyectos desarrollados en plataformas libres. Integra varias aplicaciones de software libre, tales como PostgreSQL, 7-Zip, FileZilla entre otras.

2.1.14 Pencil.

Es una herramienta de software libre de código abierto, que permite diseñar gráficos en dos dimensiones, fácil e intuitivo de uso para desarrolladores de software.

La interfaz gráfica posee herramientas básicas, pero a la vez útiles como poner en formato de mapa de bits o vectorial, paleta de colores que vienen a sustituir a otros editores gráficos como Paint.

2.2 Estructura de paquetes y clases

Para la aplicación UPS Schedule se ha aplicado la tecnología JSF de JAVA y utilizado el siguiente patrón MVC (Modelo – Vista – Controlador) para crear una separación lógica y de datos entre cada una de las clases que conforman el sistema.

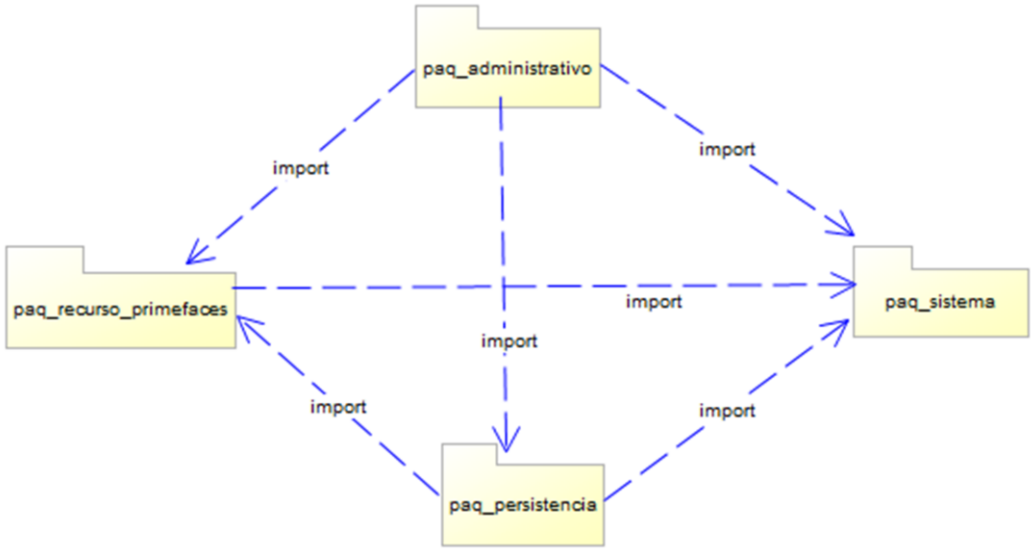
Las clases han sido agrupadas en paquetes dependiendo de la función que cumplen, estos paquetes son los siguientes:

- paq_administrativo
- paq_persistencia

- paq_recurso_primefaces
- paq_sistema

Mediante la siguiente figura se podrá visualizar y comprender la relación que existe entre cada uno de los paquetes.

Figura 15. Diagrama de paquetes.



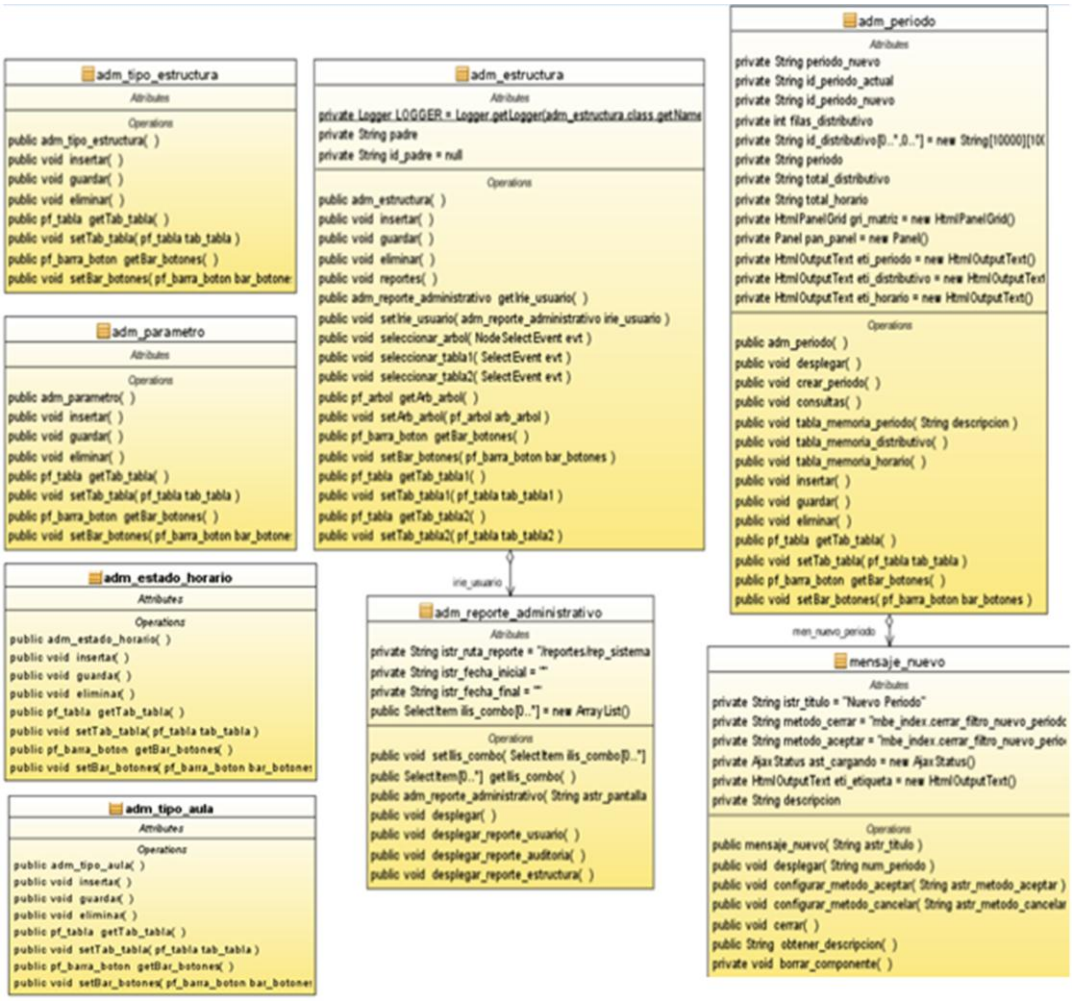
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

A continuación se detalla las clases que conforman los paquetes y su funcionalidad.

2.2.1 Paquete: paq_administrativo.

Este es el principal paquete del módulo administrativo ya que en este se alojan clases que definen la estructura y adicional la creación de nuevos periodos lectivos. Dentro de este paquete se encuentran seis clases que se describen a continuación.

Figura 16. Diagrama de clases del paquete paq_administrativo.

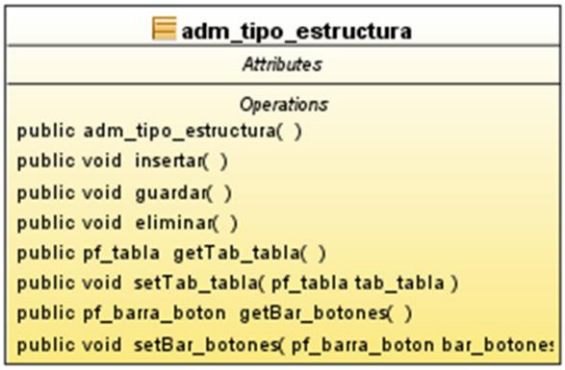


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_tipo_estructura

Aquí se encuentran todos los métodos que permiten la administración de estructuras que dispone la universidad dentro del sistema, en el caso de estos registros se puede agregar, modificar o eliminar la información almacenada en la base de datos.

Figura 17. Clase adm_tipo_estructura.

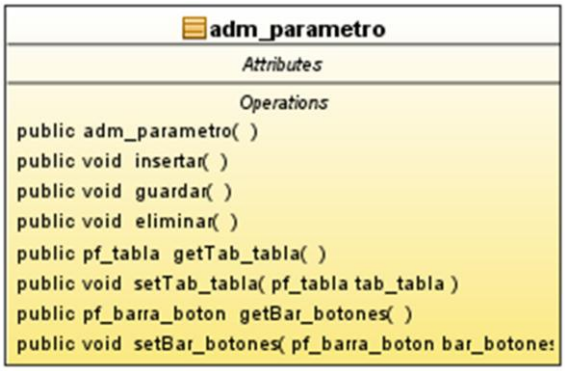


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_parametro

Esta clase obtiene y establece los parámetros del sistema que también se almacenan en la base de datos. En este caso los parámetros pueden ser modificados por lo que el administrador puede agregar o eliminar registros de acuerdo a la necesidad de otros módulos.

Figura 18. Clase adm_parametro.



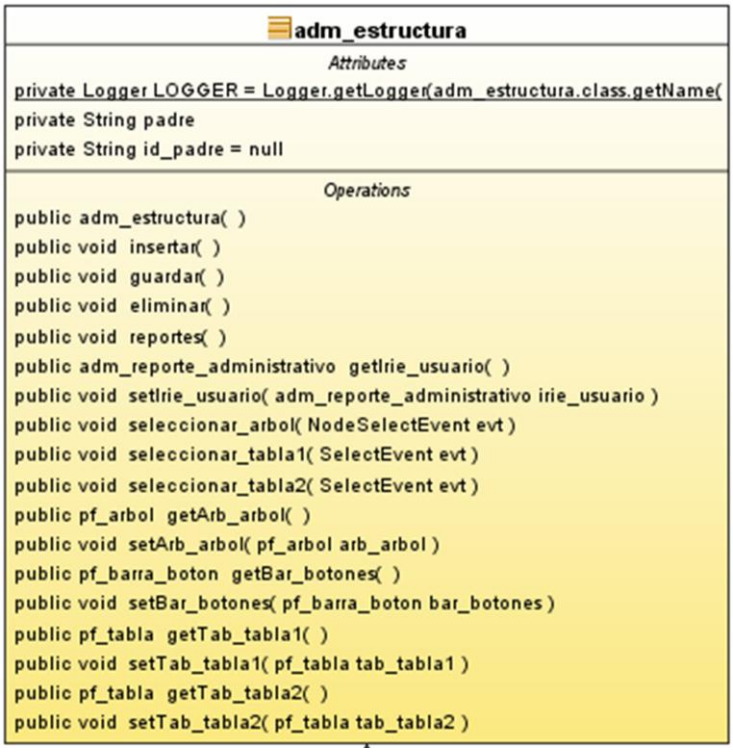
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_estructura

Esta clase administra el árbol de estructuras de la universidad es la principal clase del Módulo Administrativo. Los registros ingresados pueden agregar, modificar o

eliminar la información almacenada en la base de datos. Visualiza de una forma fácil los campus asociados a la sede.

Figura 19. Clase adm_estructura.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Entre los principales métodos tenemos los siguientes:

- **seleccionar_arbol:** Visualiza el árbol jerárquico de acuerdo a las estructuras de la universidad.

  A continuación se muestra el código más relevante del método en donde se filtra los campos de acuerdo a las estructuras.

Figura 20. Código del método seleccionar_arbol.

```
tab_tabla1.getColumna("id_tipo_estructura").configurar_combo("tbl
    _tipo_estructura", "id_tipo_estructura", "nombre", " where
    id_tipo_estructura = " + lstr_tipo_estructura);
tab_tabla1.setRows(20);

tab_tabla1.getColumna("id_tipo_aula").configurar_combo("tbl_tipo_
    aula", "id_tipo_materia", "nombre", "");
tab_tabla1.setRows(20);

tab_tabla1.getColumna("id_asistente").configurar_combo("tbl_usuar
    io", "id_usuario", "nombre", "");
tab_tabla1.setRows(20);
```

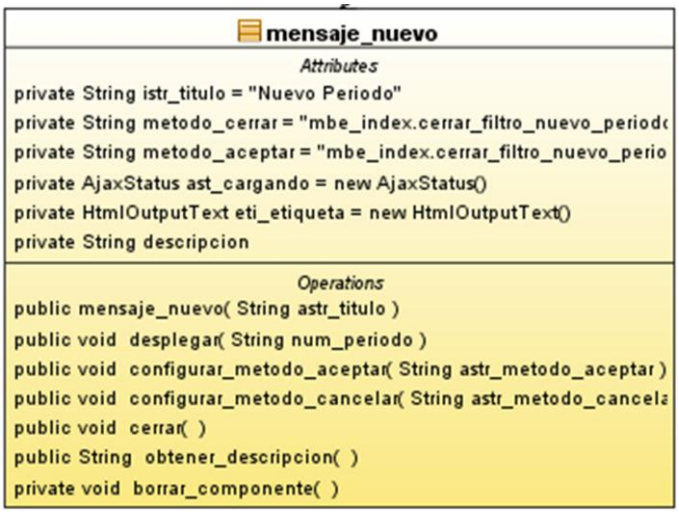
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **insertar:** Inserta los registros de acuerdo a la ubicación en la que se encuentre en el árbol.

Clase: mensaje_nuevo

Esta clase visualiza una pantalla donde se observará el número de periodo que se va crear y registra la descripción del nuevo periodo.

Figura 21. Clase mensaje_nuevo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_periodo

Por medio de esta clase se puede generar un nuevo periodo lectivo, se encarga de duplicar los registros de tablas específicas de la base de datos para dar comienzo al nuevo periodo, esta clase será administrada solo por el administrador del sistema.

Figura 22. Clase adm_periodo.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Entre los métodos más importantes están los siguientes:

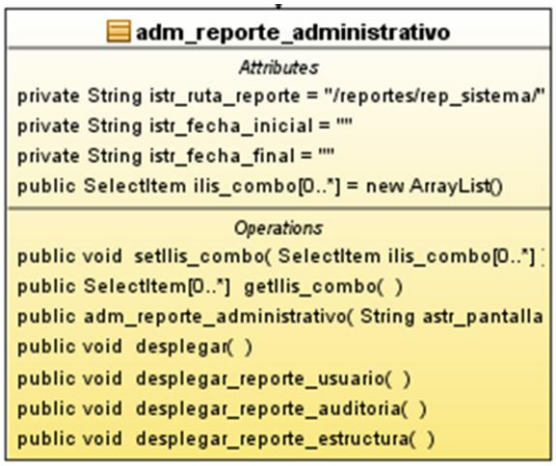
- **desplegar:** muestra la información del periodo que se va a crear.
- **consultas:** obtiene los datos del periodo, el número total de horarios y distributivos.
- **tabla_memoria_periodo:** crea un nuevo periodo con los datos del último periodo activo.

- **tabla_memoria_distributivo:** crea los distributivos del nuevo periodo, utilizando los distributivos del periodo anterior.
- **tabla_memoria_horario:** crea los horarios del nuevo periodo, utilizando los horarios de periodo anterior.

Clase: adm_reporte_administrativo

Es una sub clase que hereda de la clase pf_reporte.java del paquete paq_recurso_primefaces y contiene los métodos que permiten visualizar el reporte general de estructuras.

Figura 23. Clase adm_reporte_administrativo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Entre los métodos más importantes están los siguientes:

- **desplegar:** genera una pantalla para seleccionar el reporte para este caso administrativo.
- **desplegar_reporte_estructura:** agrega los parámetros al reporte, genera el reporte de acuerdo al formato escogido.

Figura 24. Código del método desplegar_reporte_estructura.

```
public void desplegar_reporte_estructura() throws JRException{
   if (isli_lista_reporte.isVisible()) {
       isli_lista_reporte.setVisible(false);
       pf_seleccion_formato seleccion_formato =
      (pf_seleccion_formato)
      FacesContext.getCurrentInstance().getViewRoot().findComponent(
      "formulario:isfo_formato");
```

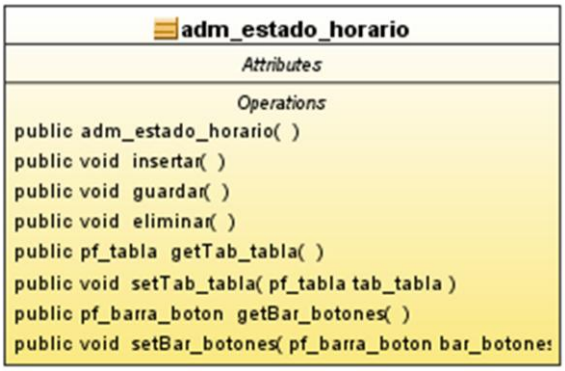
```
        isfo_formato = seleccion_formato;
        isfo_formato.setId("isfo_formato");
        isfo_formato.configurar("Formato",
       "mbe_index.clase.irie_usuario.desplegar_reporte_estructura");

       isfo_formato.configurar_accion_cancelar("mbe_index.cerrar_form
       ato");
             isfo_formato.desplegar();

} else if (isfo_formato.isVisible()) {
       Map lmap_parámetro = new HashMap();
       lmap_parámetro.put("estructura", null);
       String lstr_formato = isfo_formato.obtener_formato_reporte();
       isfo_formato.generar_reporte(lstr_formato, istr_ruta_reporte +
       "estructura1.jasper", lmap_parámetro);

         }
    }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_estado_horario

Esta clase administra los estados de los horarios que se almacenan en la base de datos. En este caso los estados pueden ser modificados por lo que el administrador puede agregar o eliminar registros.

Figura 25. Clase adm_estado_horario

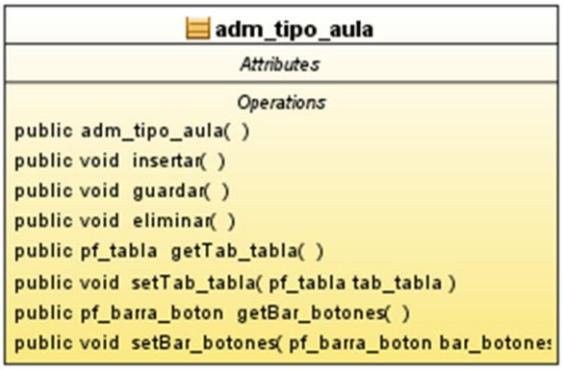


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Clase: adm_tipo_aula

Esta clase administra los tipos de aulas que existen en el sistema. Para este caso los tipos de aula pueden ser modificados por lo que el administrador puede agregar o eliminar registros de acuerdo a la necesidad de otros módulos.

Figura 26. Clase adm_tipo_aula



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3 Implementación de base de datos

A continuación se describe el diccionario de datos, con las tablas que interviene el módulo administrativo.

2.3.1 tbl_periodo: Muestra la información de un semestre**.**

Tabla 13. Descripción de la tabla tbl_periodo

Campo	Descripción
id_periodo	Es el identificador del periodo y es auto incrementable.
Número	Es el nombre pero en número que se da al periodo
descripción	Muestra el mes y el año en que inicia y finaliza el periodo. Ej. MARZO 2013 - JULIO 2013
Estado	Muestra el estado en el que está el periodo y puede ser activo o inactivo.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.2 tbl_distributivo: Contiene la información de las materias asignadas a cada docente y el espacio físico asignado.

Tabla 14. Descripción de la tabla tbl_distributivo

Campo	Descripción
id_distributivo	Es el identificador del distributivo y es auto incrementable
id_carrera	Identificador de la carrera a la que pertenece
id_docente	Identificador del docente al que está asignado
id_materia	Identificador de la materia asignada.
id_grupo	Identificador al grupo asignado
id_periodo	Identificador al periodo asignado
Estado	Muestra el estado en el que está el periodo y puede ser activo o inactivo.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.3 **tbl_horario:** Contiene la información de los horarios.

Tabla 15. Descripción de la tabla tbl_horario

Campo	**Descripción**
id_horario	Es el identificador de la tabla.
id_carrera	Es el identificador de la carrera a la cual está asignado el horario.
id_estadoh	Es el identificado en los cuales puede encontrarse el horario.
id_distributivo	Es el identificador del distributivo al cual pertenece el horario.
id_aula	Es el identificador de un aula asignada en ese horario.
id_hora	Es el identificador de la hora.
id_dia	Es el identificador del día.
Observación	Es una breve descripción del horario.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.4 **tbl_parámetro:** Muestra la información de las constantes que van a ser usadas durante la ejecución del sistema.

Tabla 16. Descripción de la tabla tbl_parametro

Campo	**Descripción**
id_parámetro	Es el identificador del parámetro y es auto incrementable.
id_estructura	Es el identificador de la tabla estructura permitiendo la relación entre las dos tablas.
Nombre	Es el nombre del parámetro.
Valor	Es el valor que va a utilizar el parámetro en el sistema.
Estado	Muestra el estado en el que está el parámetro y puede ser activo o inactivo

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.5 **tbl_estructura:** Muestra la información de un espacio físico.

Tabla 17. Descripción de la tabla tbl_estructura

Campo	**Descripción**
id_estructura	Es el identificador de la tabla y es auto incrementable.
id_tipo_estructura	Es el identificador de la tabla tipo_estructura describe el tipo de estructura a la que pertenece. Ej. sede, campus, bloque, etc.
id_tipo_aula	Identificador del tipo de aula.
id_asistente	Identificador del tipo de usuario.
id_padre	Identificador del padre que tiene para la recursividad.
Nombre	Nombre del espacio físico.
dirección	Lugar donde está ubicado.
total_pisos	Número total de pisos que existe en el edificio.

numero_pisos	Piso donde se encuentra el aula.
Área	Capacidad que tiene un aula.
Capacidad	Número de estudiantes que pueden usar en un aula.
observaciones	Descripción breve de la estructura.
Estado	Muestra el estado en el que está el parámetro y puede ser activo o inactivo.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.6 tbl_estado: Es una tabla maestra que muestra el estado de los registros de las tablas que heredan de esta.

Tabla 18. Descripción de la tabla tbl_estado

Campo	Descripción
id_estado	Es el identificador del estado, campo de tipo integer
Nombre	Es el identificador del estado ya sea este activo o inactivo

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.7 tbl_tipo_aula: Muestra la información de los diferentes tipos de aulas que existen en la UPS.

Tabla 19. Descripción de la tabla tbl_tipo_aula

Campo	Descripción
id_tipo_materia	Es el identificador del tipo de aula
Nombre	Es el nombre que describe al tipo de aula
observaciones	Es una descripción de un tipo de aula.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.3.8 tbl_estado_horario: Muestra la información de los estados que puede tener un horario.

Tabla 20. Descripción de la tabla tbl_estado_horario

Campo	Descripción
id_estadoh	Es el identificador del estado de un horario
Nombre	Es el nombre que describe del estado de un horario
Estado	Es el estado en el que se puede encontrar un horario

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.4 Codificación

Cabe mencionar que a medida que se va desarrollando el módulo administrativo, es necesario que por cada línea de código que se genere ya sea esta una clase, un método o incluso una variable se debe comentar el código más relevante o que amerite una explicación.

El motivo por el cual se debe comentar, es hacer más comprensible al desarrollador, teniendo la visión de que a futuro manipularán nuevos desarrolladores; para que así el crecimiento de la aplicación no dependa solo del programador inicial.

Para el desarrollo de este proyecto se tomó en cuenta dos lógicas: lógica de la aplicación y la lógica del negocio ambas desarrolladas en JSF.

2.4.1 Lógica de la aplicación.

Aquí se indica las clases, funciones, métodos y variables más importantes del módulo administrativo.

La clase "adm_estructura.java" es la más significativa del módulo, ya que comunica con el FrontEnd, para el envío y recepción de registros.

Figura 27. Código de variables de la clase adm_estructura.java.

```
private final static Logger LOGGER =
Logger.getLogger(adm_estructura.class.getName());
//Objeto que enlaza los reportes con la pantalla
private adm_reporte_administrativo irie_usuario = new
adm_reporte_administrativo("estructura");

private sis_soporte soporte = new sis_soporte();
private pf_tabla tab_tabla1 = new pf_tabla();
private pf_arbol arb_arbol = new pf_arbol();
private pf_barra_boton bar_botones = new pf_barra_boton();
private pf_panel div_division = new pf_panel();
private pf_panel_grupo gru_pantalla = new pf_panel_grupo();
private String padre;
private String id_padre = null;
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Cada una de estas variables necesita sus métodos getter y setter para que puedan leerlos; esto se puede ver a continuación:

Figura 28. Código getter y setter de la clase adm_estructura.java.

```
public pf_arbol getArb_arbol()
    {
        padre =arb_arbol.getValorSeleccionado();
        return arb_arbol;
    }

public void setArb_arbol(pf_arbol arb_arbol)
    {
        this.arb_arbol = arb_arbol;
    }

public pf_barra_boton getBar_botones()
    {
        return bar_botones;
    }
public void setBar_botones(pf_barra_boton bar_botones)
    {
        this.bar_botones = bar_botones;
    }
public pf_tabla getTab_tabla1()
    {
        return tab_tabla1;
    }
public void setTab_tabla1(pf_tabla tab_tabla1)
    {
        this.tab_tabla1 = tab_tabla1;
    }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Los métodos principales, que interactúa el Front y la clase son:

Insertar()

Método que permite la inserción de registros de acuerdo a la ubicación en la que se encuentre en el árbol esto dependerá del id_padre asociado.

Figura 29. Código del método insertar() de la clase adm_estructura.java.

```
public void insertar()
 {
   try{
        id_padre = arb_arbol.getValorSeleccionado();
        if(id_padre != null)
          {
            pf_tabla ltab_memoria=new pf_tabla();
            ltab_memoria.setSql("Select  *  from  tbl_estructura
where id_estructura = " + Integer.parseInt(id_padre));
            ltab_memoria.getSql();
            ltab_memoria.ejecutar_sql();
            int fila = ltab_memoria.getTotalFilas();
            String id_tipo_estructura_padre =
ltab_memoria.getValor(fila - 1,"id_tipo_estructura");
            ltab_memoria=new pf_tabla();
```

```
                ltab_memoria.setSql("Select * from
        tbl_tipo_estructura where id_padre = " +
        Integer.parseInt(id_tipo_estructura_padre));
                ltab_memoria.getSql();
                ltab_memoria.ejecutar_sql();
                fila = ltab_memoria.getTotalFilas();
                if (fila == 0) {
                soporte.crear_error("AVISO","Esta estructura no
        tiene hijos");
              }
                 else{

        tab_tabla1.getColumna("id_tipo_estructura").configurar_combo("
        tbl_tipo_estructura", "id_tipo_estructura", "nombre", " where
        id_padre = " + id_tipo_estructura_padre);
                tab_tabla1.setRows(20);
                soporte.obtener_tabla_foco().insertar();
                 }
               }
                else{
        tab_tabla1.getColumna("id_tipo_estructura").configurar_combo("
        tbl_tipo_estructura", "id_tipo_estructura", "nombre", " where
        id_padre is null ");
        tab_tabla1.setRows(20);
            }
                }
         catch(Exception e ){
            soporte.crear_error(e.getMessage(), "adm_estructura en el
        método insertar() ");
               }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Seleccionar_arbol ()

Método que muestra la información de manera jerárquica, representado en forma de árbol.

Figura 30. Código del método seleccionar_arbol() de la clase adm_estructura.java.

```
    public void seleccionar_arbol(NodeSelectEvent evt)
        {

    tab_tabla1.ejecutarValorPadre(arb_arbol.getValorSeleccionado());
            id_padre = arb_arbol.getValorSeleccionado();
            String lstr_tipo_estructura =
    tab_tabla1.getValor("id_tipo_estructura");
    tab_tabla1.getColumna("id_estructura").setVisible(false);
    tab_tabla1.getColumna("id_tipo_estructura").setVisible(true);
    tab_tabla1.getColumna("id_tipo_aula").setVisible(true);
    tab_tabla1.getColumna("id_asistente").setVisible(true);
    tab_tabla1.getColumna("id_padre").setVisible(true);
    tab_tabla1.getColumna("nombre").setVisible(true);
    tab_tabla1.getColumna("estado").setVisible(true);
    tab_tabla1.getColumna("observaciones").setVisible(true);
    tab_tabla1.getColumna("direccion").setVisible(true);
    tab_tabla1.getColumna("total_pisos").setVisible(true);
```

```
    tab_tabla1.getColumna("numero_pisos").setVisible(true);
    tab_tabla1.getColumna("area").setVisible(true);
    tab_tabla1.getColumna("capacidad").setVisible(true);
    tab_tabla1.getColumna("id_tipo_estructura").configurar_combo("tb
    l_tipo_estructura", "id_tipo_estructura", "nombre", " where
    id_tipo_estructura = " + lstr_tipo_estructura);
    tab_tabla1.setRows(20);
    tab_tabla1.getColumna("id_tipo_aula").configurar_combo("tbl_tipo
    _aula", "id_tipo_materia", "nombre", "");
    tab_tabla1.setRows(20);
    tab_tabla1.getColumna("id_asistente").configurar_combo("tbl_usua
    rio", "id_usuario", "nombre", "");
    tab_tabla1.setRows(20);
    if(lstr_tipo_estructura.equalsIgnoreCase("1")){
    tab_tabla1.getColumna("id_tipo_aula").setVisible(false);
    tab_tabla1.getColumna("id_asistente").setVisible(false);
    tab_tabla1.getColumna("direccion").setVisible(false);
    tab_tabla1.getColumna("total_pisos").setVisible(false);
    tab_tabla1.getColumna("numero_pisos").setVisible(false);
    tab_tabla1.getColumna("area").setVisible(false);
    tab_tabla1.getColumna("capacidad").setVisible(false);
     }else
       if(lstr_tipo_estructura.equalsIgnoreCase("2")){
       tab_tabla1.getColumna("id_tipo_aula").setVisible(false);
       tab_tabla1.getColumna("id_asistente").setVisible(false);
      tab_tabla1.getColumna("direccion").setVisible(false);
      tab_tabla1.getColumna("total_pisos").setVisible(false);
      tab_tabla1.getColumna("numero_pisos").setVisible(false);
      tab_tabla1.getColumna("area").setVisible(false);
      tab_tabla1.getColumna("capacidad").setVisible(false);
    }else

    if(lstr_tipo_estructura.equalsIgnoreCase("3")){
      tab_tabla1.getColumna("id_tipo_aula").setVisible(false);
      tab_tabla1.getColumna("id_asistente").setVisible(false);
      tab_tabla1.getColumna("direccion").setVisible(true);
      tab_tabla1.getColumna("total_pisos").setVisible(false);
      tab_tabla1.getColumna("numero_pisos").setVisible(false);
      tab_tabla1.getColumna("area").setVisible(false);
      tab_tabla1.getColumna("capacidad").setVisible(false);

    }else
      if(lstr_tipo_estructura.equalsIgnoreCase("4")){
      tab_tabla1.getColumna("id_tipo_aula").setVisible(false);
      tab_tabla1.getColumna("id_asistente").setVisible(false);
      tab_tabla1.getColumna("direccion").setVisible(false);
      tab_tabla1.getColumna("total_pisos").setVisible(true);
      tab_tabla1.getColumna("numero_pisos").setVisible(true);
      tab_tabla1.getColumna("area").setVisible(false);
      tab_tabla1.getColumna("capacidad").setVisible(false);
    }else{
      tab_tabla1.getColumna("id_tipo_aula").setVisible(true);
      tab_tabla1.getColumna("id_asistente").setVisible(false);
      tab_tabla1.getColumna("direccion").setVisible(false);
      tab_tabla1.getColumna("total_pisos").setVisible(false);
      tab_tabla1.getColumna("numero_pisos").setVisible(false);
      tab_tabla1.getColumna("area").setVisible(true);
      tab_tabla1.getColumna("capacidad").setVisible(true);
       }
    }
```

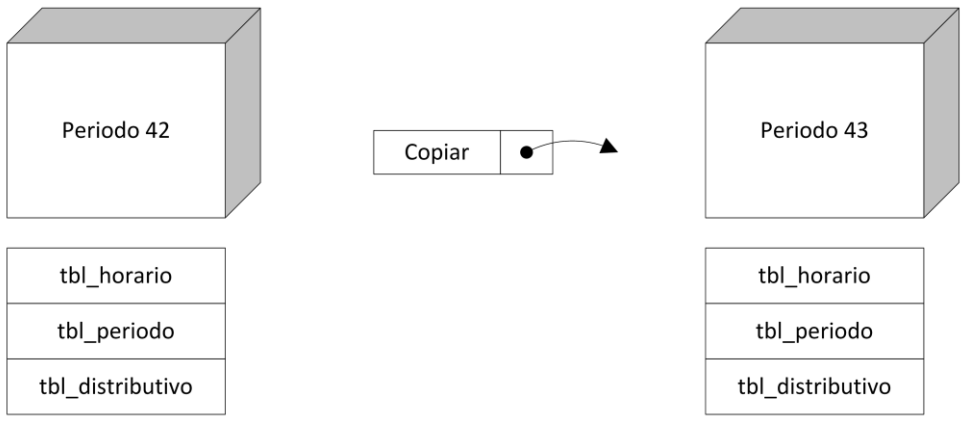
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.4.2 Lógica de negocio.

Para el manejo de la lógica de negocio, se lo hace desde JSF, se controla desde el mismo programa, ya que si hubiera una migración hacia otro motor de base de datos sería mínimo el impacto en la migración.

La clase "adm_periodo.java" es la encargada de iniciar un nuevo periodo en base a la información precedente del último periodo lectivo. Se encarga de duplicar los registros de ciertas tablas (tbl_distributivo, tbl_periodo, tbl_horario) y generar un nuevo periodo.

Figura 31. Proceso de copia clase "adm_periodo"



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

En la figura 32, se puede identificar las siguientes variables:

Figura 32. Código de variables de la clase adm_periodo.java.

```
private pf_barra_boton bar_botones = new pf_barra_boton();
private sis_soporte soporte = new sis_soporte();
private pf_tabla tab_tabla = new pf_tabla();
private pf_panel div_division = new pf_panel();
private pf_panel_grupo gru_pantalla = new pf_panel_grupo();
private String periodo_nuevo,id_periodo_actual,id_periodo_nuevo;
private cla_conexion cl= new cla_conexion();
private int tamaño_distributivo;
private String[][] id_distributivo= new String[10000][1000];
private String numero_periodo_actual,
total_distributivo,total_horario;
private HtmlPanelGrid gri_matriz = new HtmlPanelGrid();
private Panel pan_panel = new Panel();
private HtmlOutputText eti_periodo = new HtmlOutputText();
private HtmlOutputText eti_distributivo = new HtmlOutputText();
private HtmlOutputText eti_horario = new HtmlOutputText();
private pf_boton bot_CPeriodo = new pf_boton();
private pf_grafico ima_cargando = new pf_grafico();
private mensaje_nuevo men_nuevo_periodo = new
mensaje_nuevo("Crear nuevo periodo");
private pf_tabla ltab_memoria = new pf_tabla();
private pf_ajax aja_nuevo_periodo= new pf_ajax();
private String consulta_periodo;
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Los métodos principales, que interactúa el Front y la clase son:

Desplegar()

Método que muestra la información del periodo que se va a crear e ingresar la descripción del nuevo periodo.

Figura 33. Código del método desplegar() de la clase adm_periodo.java.

```
public void desplegar() {

String x1="Select * from tbl_periodo where id_periodo=
'"+id_periodo_actual+" '";
 ltab_memoria.setSql(x1);
  ltab_memoria.ejecutar_sql();
  periodo_nuevo = "" +
(Integer.parseInt(numero_periodo_actual) + 1);
  mensaje_nuevo seleccion_nuevo_periodo = (mensaje_nuevo)
FacesContext.getCurrentInstance().getViewRoot().findComponent
("formulario:men_nuevo_periodo");
if (!seleccion_nuevo_periodo.isVisible()) {
   men_nuevo_periodo = seleccion_nuevo_periodo;
   men_nuevo_periodo.setId("men_nuevo_periodo");
   men_nuevo_periodo.configurar_metodo_aceptar("mbe_index.cla
se.crear_periodo");
   men_nuevo_periodo.configurar_metodo_cancelar("mbe_index.ce
rrar_men_nuevo_periodo");
              men_nuevo_periodo.desplegar(periodo_nuevo);

        }
  }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultas()

Método que obtiene los datos del periodo, el número total de horarios y distributivos del último periodo activo.

Figura 34. Código del método consultas() de la clase adm_periodo.java.

```
public void consultas() {

        id_periodo_actual=""+(cl.consultar("select max(
id_periodo) from tbl_periodo where estado='1' ")).get(0);
        numero_periodo_actual= ""+ (cl.consultar("select
numero from tbl_periodo where id_periodo= '" +
id_periodo_actual + "'")).get(0);
        total_distributivo= ""+(cl.consultar("select
count(d.id_distributivo) from tbl_distributivo d,  tbl_periodo
p where d.id_periodo=p.id_periodo and p.id_periodo='" +
id_periodo_actual + "'")).get(0);
        total_horario= ""+(cl.consultar("select
count(h.id_horario) from tbl_distributivo d, tbl_horario h,
tbl_periodo p where h.id_distributivo= d.id_distributivo and
d.id_periodo=p.id_periodo and p.id_periodo='" +
id_periodo_actual + "'")).get(0);

    }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla_memoria_periodo()

Método que crea un nuevo periodo con los datos del último periodo activo y en este periodo hace una actualización en el campo estado con valor 0 de inactivo.

Figura 35. Código del método tabla_memoria_periodo() de la clase adm_periodo.java.

```
public void tabla_memoria_periodo(String descripcion) {
 try {
 id_periodo_nuevo=""+(Integer.parseInt(id_periodo_actual)+1);
 periodo_nuevo=""+(Integer.parseInt(numero_periodo_actual)+1);
 consulta_periodo = "Insert into tbl_periodo( id_periodo,numero,
descripcion, estado) VALUES ('" + id_periodo_nuevo + "','" +
periodo_nuevo + "','" + descripcion + "','" + 1 + "')";
cl.ejecutar_sql(consulta_periodo);

   } catch (Exception e) {
   soporte.crear_error(e.getMessage(), "adm_periodo en el método
insertar() ");
   }
 }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla_memoria_distributivo()

Método que crea los distributivos del nuevo periodo, utilizando los distributivos del periodo anterior.

Figura 36. Código del método tabla_memoria_distributivo() de la clase adm_periodo.java.

```
public void tabla_memoria_distributivo() {

String id_carrera, id_docente, id_materia, id_grupo, estado;
pf_tabla ltab_memoria_distr = new pf_tabla();
ltab_memoria_distr.setSql("Select * from tbl_distributivo where
id_periodo=" + id_periodo_actual );
ltab_memoria_distr.ejecutar_sql();

tamaño_distributivo=Integer.parseInt(""+(cl.consultar("select
count( id_distributivo) from tbl_distributivo where id_periodo=" +
id_periodo_actual)).get(0));
int id_nuevo_distributivo=
Integer.parseInt(""+(cl.consultar("select max( id_distributivo)
from tbl_distributivo  ")).get(0));
for (int i = 0; i < tamaño_distributivo; i++) {
 id_nuevo_distributivo=id_nuevo_distributivo+1;
 String
id_distributivo_actual=ltab_memoria_distr.getValor(i,"id_distributi
vo");
 id_carrera = ltab_memoria_distr.getValor(i, "id_carrera");
 id_docente = ltab_memoria_distr.getValor(i, "id_docente");
 id_materia = ltab_memoria_distr.getValor(i, "id_materia");
 id_grupo = ltab_memoria_distr.getValor(i, "id_grupo");
 estado = ltab_memoria_distr.getValor(i, "estado");
 String x = "Insert into
tbl_distributivo(id_distributivo,id_carrera, id_docente,
id_materia, id_grupo,id_periodo,estado) VALUES (" +
id_nuevo_distributivo + "," + id_carrera + "," + id_docente + "," +
id_materia + "," + id_grupo + "," + id_periodo_nuevo + "," + estado
+ ")";
            cl.ejecutar_sql(x);
            id_distributivo[i][0] = id_distributivo_actual;
            id_distributivo[i][1] = ""+id_nuevo_distributivo;
     }
 }
```

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla_memoria_horario()

Método que crea los horarios del nuevo periodo, utilizando los horarios de periodo anterior y de los distributivos anteriores del mismo.

Figura 37. Código del método tabla_memoria_horario() de la clase adm_periodo.java.

```
 public void tabla_memoria_horario() {

 String id_horario="",id_carrera = "", id_estado_h =
 "",id_distributivo_actual="",id_distributivo_nuevo="", id_aula =
 "", id_hora = "", id_dia = "", observacion = "";
 pf_tabla ltab_memoria_horario = new pf_tabla();
 String x=("select d.id_carrera,h.id_estadoh,
 d.id_distributivo,h.id_aula,h.id_hora, h.id_dia, h.observacion
 from tbl_distributivo d, tbl_periodo p, tbl_horario h where
 d.id_periodo=p.id_periodo and
 h.id_distributivo=d.id_distributivo and p.id_periodo=
 "+id_periodo_actual);
 ltab_memoria_horario.setSql(x);
 ltab_memoria_horario.ejecutar_sql();
 int id_nuevo_horario=Integer.parseInt(""+(cl.consultar("select
 max( id_horario) from tbl_horario")).get(0));
   int tamaño=Integer.parseInt(""+(cl.consultar("select count(
id_horario) from tbl_distributivo d, tbl_periodo p, tbl_horario h
              where d.id_periodo=p.id_periodo and
      h.id_distributivo=d.id_distributivo and p.id_periodo=
                  "+id_periodo_actual)).get(0));

 for (int i = 0; i < tamaño; i++) {
 id_nuevo_horario= id_nuevo_horario+1;
 id_carrera = ltab_memoria_horario.getValor(i, "id_carrera");
 id_estado_h = ltab_memoria_horario.getValor(i, "id_estadoh");
 id_distributivo_actual= ltab_memoria_horario.getValor(i,
 "id_distributivo");
 id_aula = ltab_memoria_horario.getValor(i,"id_aula");
 id_hora = ltab_memoria_horario.getValor(i,"id_hora");
 id_dia = ltab_memoria_horario.getValor(i,"id_dia");
 observacion = ltab_memoria_horario.getValor(i,"observacion");
 for(int j=0; j<tamaño_distributivo;j++){

 if(id_distributivo_actual.equalsIgnoreCase(id_distributivo[j][0]
 )){

 id_distributivo_nuevo=id_distributivo[j][1];
 j=tamaño_distributivo;
      }
   }
String x1 = "Insert into tbl_horario(id_horario,id_carrera,
id_estadoh, id_distributivo,id_aula,id_hora,id_dia,observacion)
VALUES ('" + id_nuevo_horario + "','" + id_carrera + "','" +
id_estado_h + "','" + id_distributivo_nuevo + "','" + id_aula +
"','" + id_hora + "','" + id_dia + "','" + observacion + "')";
 cl.ejecutar_sql(x1);
    }

 }
```

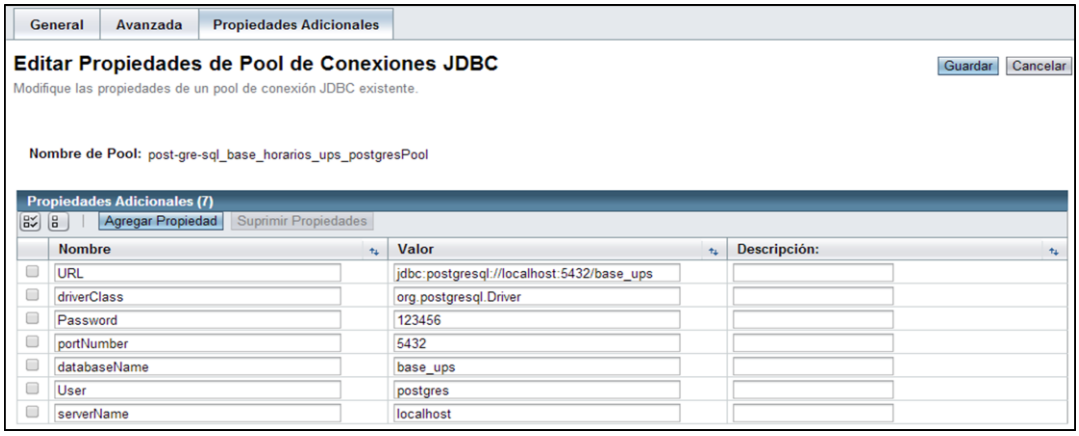
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.4.3 Configuración del servidor.

Para que la aplicación sea ejecutada exitosamente es necesario configurar ciertos parámetros que permitirán su correcto desempeño, a continuación de detalla las configuraciones en el servidor GlassFish:

1. Para poder realizar las configuraciones en el servidor es necesario acceder a la consola web del servidor, mediante el siguiente link http://localhost:4848/common/index.jsf, donde se procederá con la creación del Pool de conexión JDBC con la base de datos, configurando tanto el nombre del Pool como las propiedades que se exponen a continuación.

Figura 38. Propiedades del Pool de conexiones JDBC



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2. Una vez creado el Pool de conexión se procede con el siguiente requerimiento que es la generación de un nuevo Recurso JDBC donde se asignará el Pool anteriormente creado de la siguiente manera.

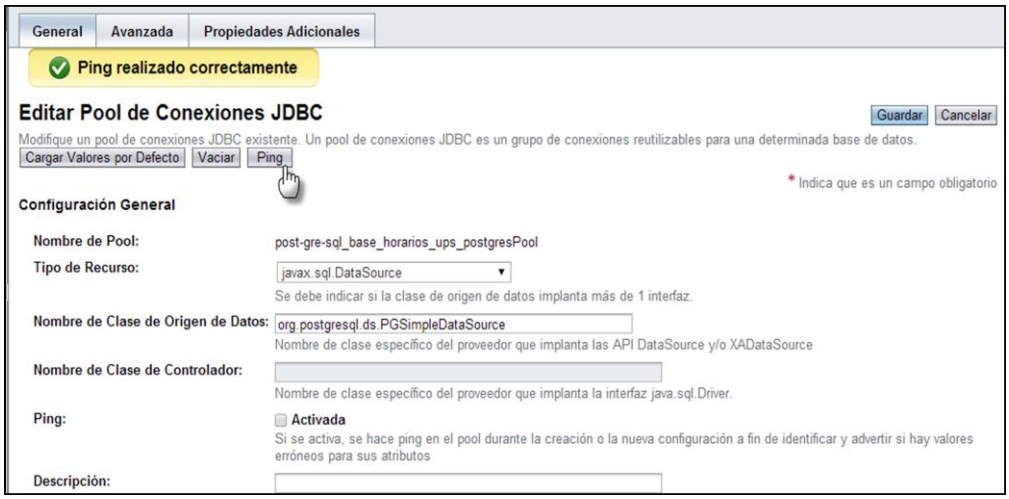
Figura 39. Creación nuevo recurso JDBC.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

3. Para que la conexión sea exitosa con PostgreSQL es necesario copiar una librería ".jar" de acuerdo a la versión del motor de base de datos utilizada en la dirección del dominio de GlassFish donde se publicará la aplicación. Para el caso se utilizará la siguiente librería "postgresql-9.1-901.jdbc4.jar".

   De esta forma la conexión será exitosa entre el servidor de GlassFish y la base de datos PostgresSQL como se puede apreciar en la siguiente Figura:

Figura 40. Prueba de conexión desde GlassFish hacia PostgreSQL.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.5 Pruebas

Una vez concluido la construcción del módulo se la ha sometido a tres tipos de pruebas:

- Pruebas de unitarias
- Pruebas de caja negra
- Pruebas de rendimiento
    - Pruebas con tres navegadores
    - Pruebas con tres usuarios
    - Pruebas remotas

Los resultados de las pruebas se los va a mostrar en una tabla para verificar que el sistema cumple con los requerimientos planteados.

2.5.1 Pruebas unitarias.

Las pruebas unitarias permiten evaluar las funciones principales del módulo administrativo.

Para ello se ingresan los valores de entrada para evaluar la respuesta y el tiempo que estos toman en ejecutarse.

Para realizar las pruebas unitarias se utilizará JUNIT 4.0 el mismo que ya viene instalado en la versión NetBeans 7.2.1 por defecto.

Para ello se procede a escoger la o las clases que se va a realizar la prueba para el caso se ha seleccionado y evaluado la clase “adm_periodo.java”.

Al realizar las pruebas con los métodos seleccionados se obtiene los siguientes resultados.

Tabla 21. Tablas de resultados pruebas unitarias

N°	Función	Entrada	Tiempo (ms)	Resp.	Observaciones
1	consultas()	ninguna	55,5	Ok	Resultado esperado
2	tabla_memoria_periodo(descripcion)	String descripcion	6,8	Ok	Resultado esperado
3	tabla_memoria_distributivo()	Ninguna	675,7	Ok	Resultado esperado
4	tabla_memoria_horario()	Ninguna	2959,1	Ok	Resultado esperado
5	cl.ejecutar_sql(aPeriodo)	Update	2	Ok	Resultado esperado

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

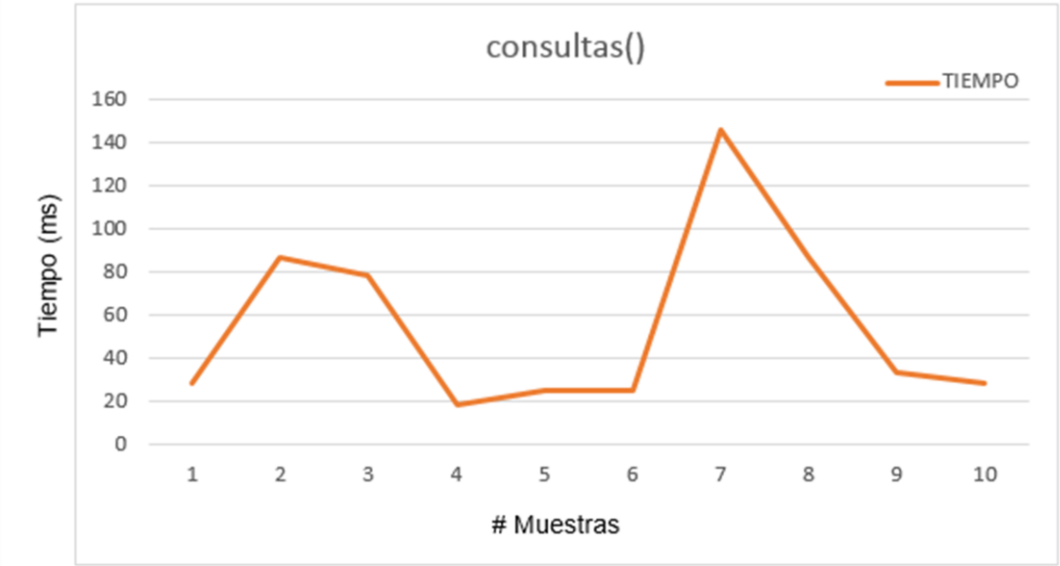
Se ha realizado 10 pruebas unitarias de las funciones obteniendo los siguientes resultados en ms:

Tabla 22. Resultados pruebas unitarias tomando 10 muestras.

Función	Muestra (ms)										Prom.
	#1	#2	#3	#4	#5	#6	#7	#8	#9	#10	
consultas()	28	87	78	18	25	25	146	87	33	28	55,5
tabla_memoria_periodo(descripcion)	7	7	7	4	3	4	4	2	23	7	6,8
tabla_memoria_distributivo()	587	614	812	538	528	588	527	533	834	1194	675,5
tabla_memoria_horario()	3823	2766	3417	2706	2640	2120	2684	3033	3220	3092	2950,1
cl.ejecutar_sql(aPeriodo)	7	1	1	1	1	1	2	1	1	2	1,8

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

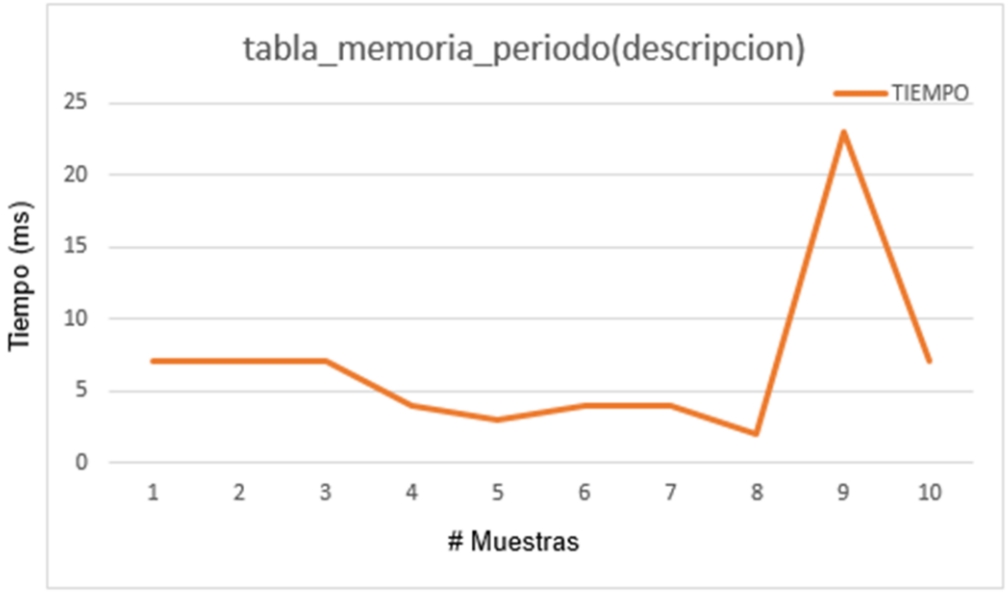
Figura 41. Resultados de las diez muestras en el método consultas() en función al tiempo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

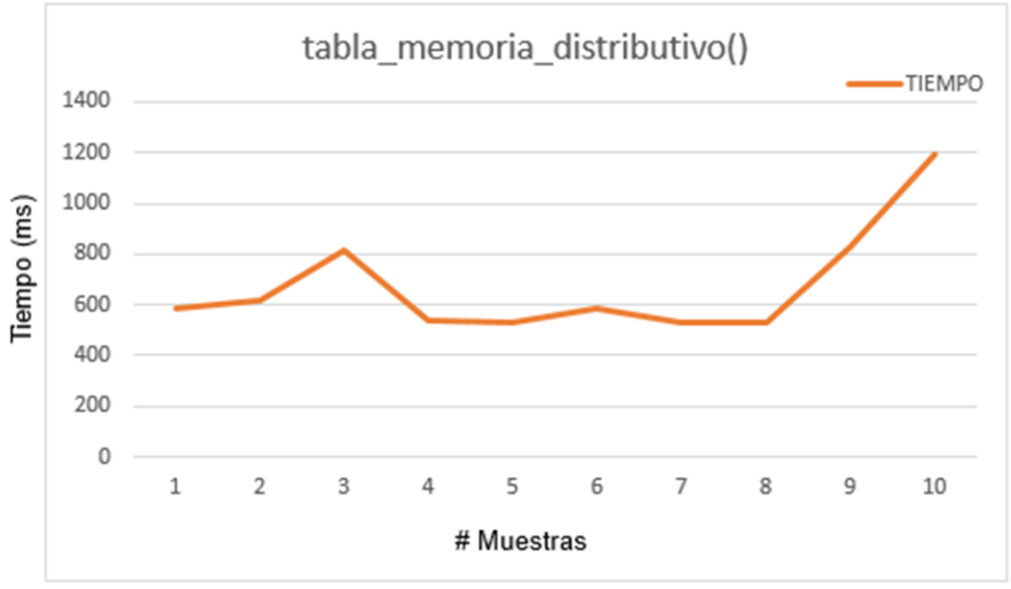
A continuación se indican las gráficas de los resultados de las pruebas unitarias y la variación del tiempo:

Figura 42. Resultados de las diez muestras en el método tabla_memoria_periodo(descrpcion) en función al tiempo.



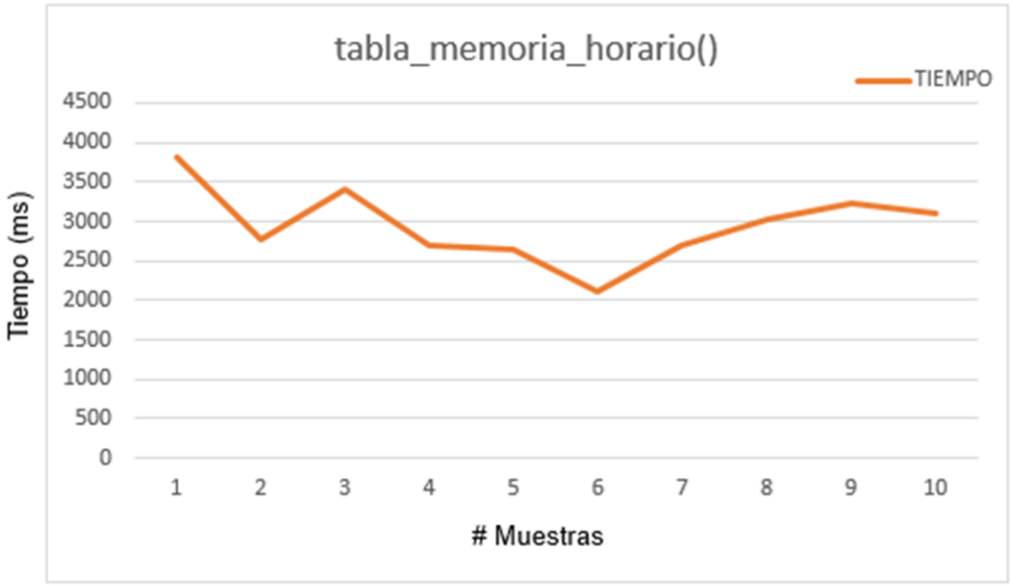
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 43. Resultados de las diez muestras en el método tabla_memoria_distributivo() en función al tiempo.



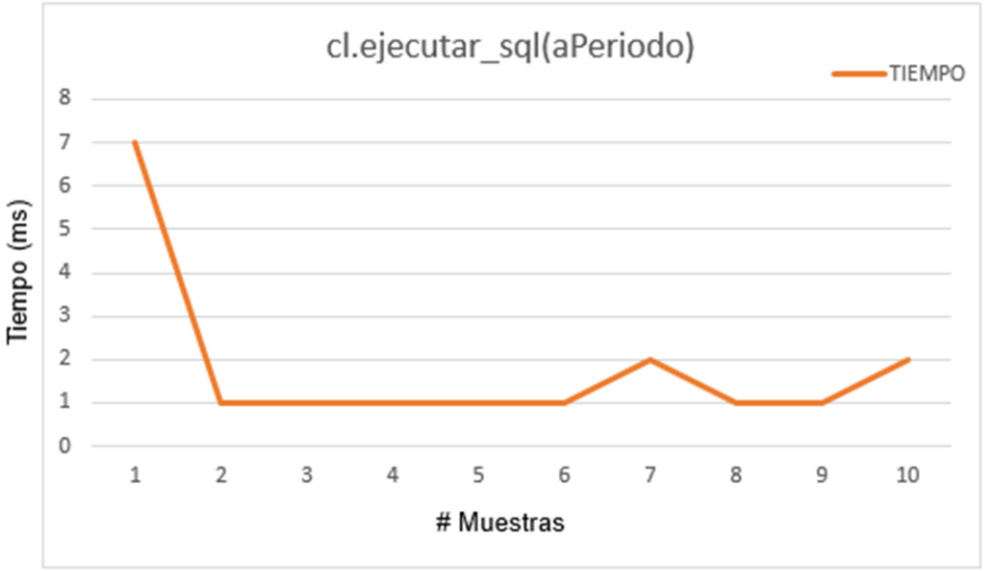
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 44. Resultados de las diez muestras en el método tabla_memoria_horario() en función al tiempo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 45. Resultados de las diez muestras en el método cl.ejecutar_sql() en función al tiempo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.5.2 Pruebas de caja negra

Las pruebas de caja negra permiten evaluar los procesos realizados por el usuario, que la funcionalidad sea correcta y que los resultados sean los esperados.

Tabla 23. Resultados pruebas contra requerimientos.

N°	Nombre de la prueba	Req.	Pasos	Ok	Observaciones
1	Agregar nueva parámetro	1.1	1. Del menú principal escoger la opción de parámetro. 2. A continuación en la barra de botones seleccionamos el icono Nuevo. 3. Ingresar los datos del nuevo parámetro. 4. Clic en el botón Guardar.	Sí	
2	Modificar parámetro	1.1	1. Del menú principal escoger la opción de parámetro. 2. Seleccionar el ítem a modificar. 3. Modificar los datos deseados. 4. Clic en el botón Guardar.	Sí	
3	Eliminar parámetro	1.1	1. Del menú principal escoger la opción de parámetro. 2. Seleccionar el ítem a borrar. 3. Clic en el botón Eliminar.	Sí	
4	Agregar nueva estructura	1.2	1. Del menú principal escoger la opción de estructura. 2. Seleccionar el icono Nuevo. 3. Ingresar los datos de la nueva estructura. 4. Clic en el botón Guardar.	Sí	
5	Modificar una estructura	1.2	1. Del menú principal escoger la opción de estructura. 2. Seleccionar el ítem a modificar. 3. Modificar los datos. 4. Clic en el botón Guardar.	Sí	
6	Eliminar una estructura	1.2	1. Seleccionar la opción estructura del menú principal. 2. Seleccionar el ítem a eliminar. 3. Clic en el botón Eliminar.	Sí	
7	Crear un nuevo periodo	1.3	1. Seleccionar la opción periodo del menú principal. 2. Clic en el botón Nuevo periodo. 3. Ingresar la descripción del Nuevo periodo ejemplo: MARZO 2013 - JULIO 2013 5. Clic en el botón Aceptar. 6. Esperar que se cree el nuevo periodo.	Sí	El proceso realizado tiene una duración de 4084,666667 ms con respecto a tres carreras, y puede variar a las carreras existentes.

N°	Nombre de la prueba	Req.	Pasos	Ok	Observaciones
8	Crear reporte	1.4	1. Selecciona la opción estructura. 2. Clic en el botón crear reporte 3. Seleccionar el tipo de formato en el que se va a crear 4. Clic en el botón Aceptar	Sí	
9	Crear tipo estructura	1.5	1. Seleccionar la opción de tipo estructura. 2. Clic en el botón Nuevo estructura. 3. Ingresar los datos del tipo de estructura. 4. Clic en el botón guardar.	Sí	
10	Modificar un tipo de estructura	1.5	1. Seleccionar la opción de tipo estructura. 2. Modificar los datos los datos deseados. 3. Clic en Guardar.	Sí	
11	Eliminar un tipo de estructura	1.5	1. Seleccionar la opción de tipo estructura. 2. Seleccionar el ítem a eliminar. 3. Clic en eliminar.	Sí	
12	Crear estado horario	1.7	1. Seleccionar la opción de estado horario. 2. Clic en el botón estado horario. 3. Ingresar los datos del estado horario. 4. Clic en guardar.	Sí	
13	Modificar un estado horario	1.7	1. Seleccionar la opción de estado horario. 2. Modificar los datos los datos deseados. 3. Clic en Guardar.	Sí	
14	Eliminar un estado horario	1.7	1. Seleccionar la opción de estado horario. 2. Seleccionar el ítem a eliminar. 3. Clic en eliminar.	Sí	
15	Crear tipo aula	1.8	1. Seleccionar la opción de tipo aula. 2. Clic en el botón Nuevo tipo aula. 3. Ingresar los datos del tipo aula. 4. Clic en guardar.	Sí	
16	Modificar un tipo aula	1.8	1. Seleccionar la opción de tipo aula. 2. Modificar los datos los datos deseados. 3. Clic en Guardar.	Sí	
17	Eliminar un tipo aula	1.8	1. Seleccionar la opción de tipo aula. 2. Seleccionar el ítem a eliminar. 3. Clic en eliminar.	Sí	

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2.5.3 Pruebas de rendimiento

Las pruebas de rendimiento evalúa el desempeño del módulo de asignación de espacios físicos, y las funciones que desempeña. Para ello se han realizado las siguientes pruebas:

Pruebas con tres usuarios

Se debe considerar que el módulo de Administrativo va a ser usado por usuarios con perfil administrador cabe recalcar que el número de usuarios con este perfil no será mayor a tres usuarios. Ante esta aclaración, los resultados obtenidos con la herramienta JMeter son los siguientes:

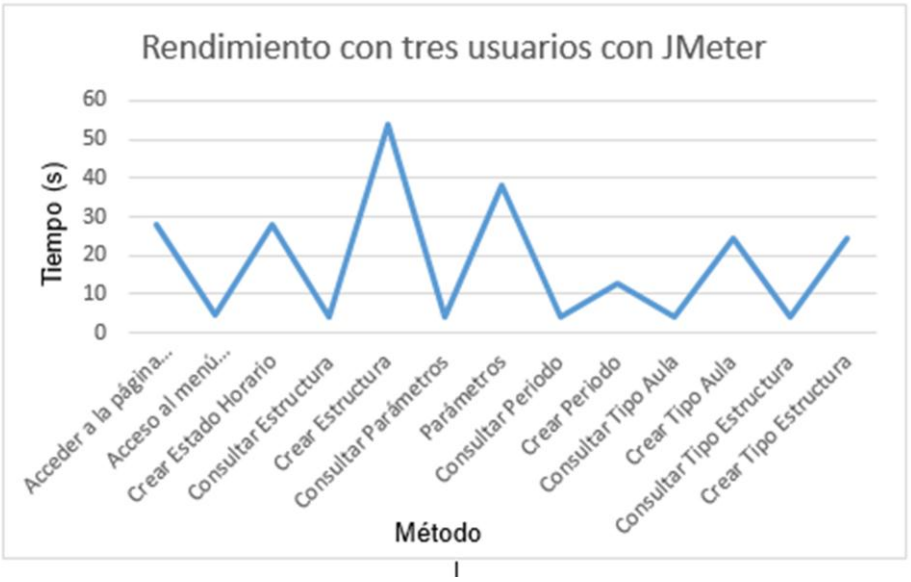
Tabla 24. Resultados de las pruebas de rendimiento con la herramienta JMeter con cinco muestras.

N°	Método	Carga	Respuesta	Promedio Tiempo (s)	Observaciones
1	Persistencia de aplicación	3 usuarios	OK	236,5	Respuesta satisfactoria
2	Acceder a la página principal	3 usuarios	OK	27,9	Respuesta satisfactoria
3	Acceso al menú administrativo	3 usuarios	OK	4,5	Respuesta satisfactoria
4	Crear estado horario	3 usuarios	OK	28,1	Respuesta satisfactoria
5	Consultar estructura	3 usuarios	OK	4,4	Respuesta satisfactoria
6	Crear estructura	3 usuarios	OK	53,9	Respuesta satisfactoria
7	Consultar parámetros	3 usuarios	OK	4,4	Respuesta satisfactoria
8	Parámetros	3 usuarios	OK	38,4	Respuesta satisfactoria
9	Consultar periodo	3 usuarios	OK	4,4	Respuesta satisfactoria
10	Crear periodo	3 usuarios	OK	12,7	Respuesta satisfactoria
11	Consultar tipo aula	3 usuarios	OK	4,4	Respuesta satisfactoria
12	Crear tipo aula	3 usuarios	OK	24,5	Respuesta satisfactoria
13	Consultar tipo estructura	3 usuarios	OK	4,4	Respuesta satisfactoria
14	Crear tipo estructura	3 usuarios	OK	24,5	Respuesta satisfactoria

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Los resultados que son expuestos en la tabla 24, se han obtenido de las pruebas realizadas con la herramienta JMeter verificar anexo 3.

Figura 46. Gráfica de resultados con tres usuarios utilizando la herramienta JMeter.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Pruebas con exploradores

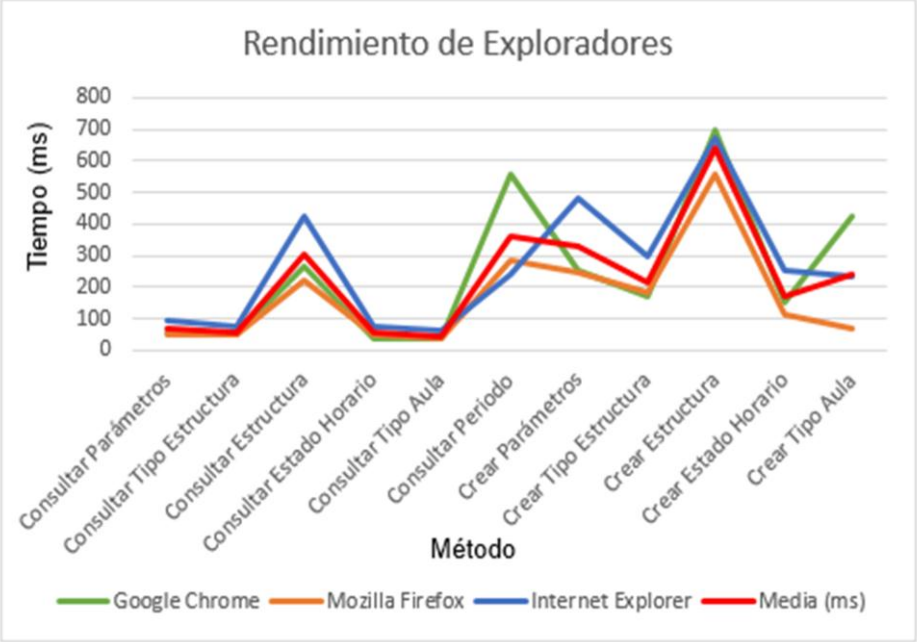
A continuación se presenta los resultados que se obtuvieron al realizar las pruebas de rendimiento en los exploradores más conocidos como son Google Chrome, Mozilla Firefox e Internet Explorer.

Tabla 25. Resultados obtenidos con los diferentes exploradores.

	Exploradores tiempo (ms)			Media (ms)	Respuesta
Nombre	**Google Chrome**	**Mozilla Firefox**	**Internet Explorer**		
Consultar parámetros	53	51	94	66	Ok
Consultar tipo estructura	48	50	78	58,67	Ok
Consultar estructura	264	220	422	302	Ok
Consultar estado horario	39	47	78	54,67	Ok
Consultar tipo aula	37	34	62	44,33	Ok
Consultar periodo	557	284	239	360	Ok
Crear parámetros	256	249	484	329,67	Ok
Crear tipo estructura	169	186	298	217,67	Ok
Crear estructura	699	555	672	642	Ok
Crear estado horario	154	116	250	173,33	Ok
Crear tipo aula	422	71	234	242,33	Ok
Crear periodo	4268	4020,9	3965,1	4084,67	Ok

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 47. Gráfico lineal de rendimiento de los exploradores utilizados.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

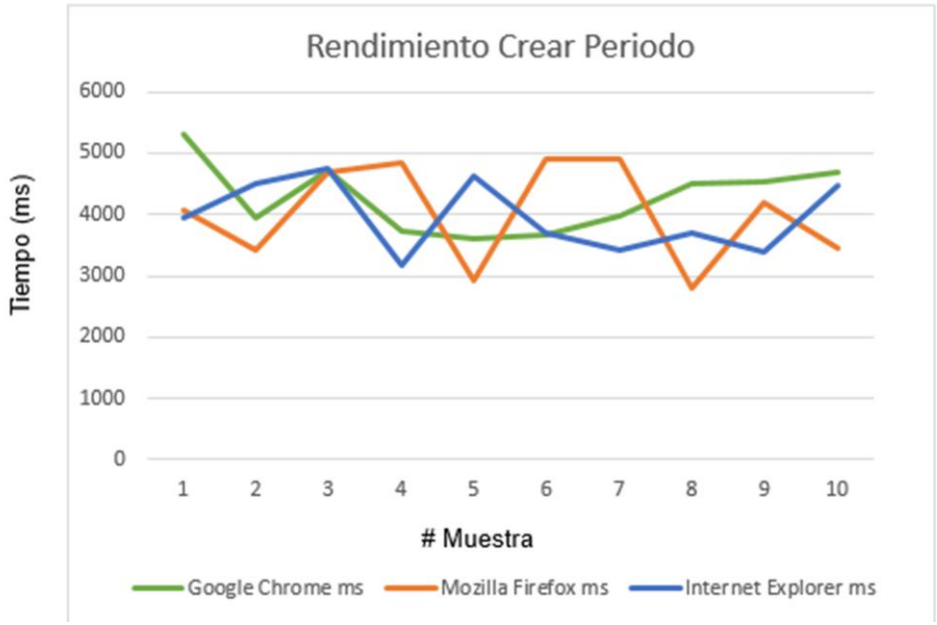
Para la opción crear periodo perteneciente al módulo administrativo se ha realizado un análisis con 10 muestras en distintos exploradores, ya que ésta genera la duplicidad de varias tablas y es preciso determinar el tiempo con valores reales ingresados, obteniendo los siguientes resultados:

Tabla 26. Tiempos obtenidos en la opción crear periodo con 10 muestras en distintos exploradores.

Crear periodo					
# Muestra	**Google Chrome (ms)**	**Mozilla Firefox (ms)**	**Internet Explorer (ms)**	**Promedio (ms)**	**Resultado**
1	5316	4079	3937	4444	Ok
2	3934	3433	4495	3954	Ok
3	4723	4687	4744	4718	Ok
4	3714	4843	3157	3904,67	Ok
5	3599	2917	4616	3710,67	Ok
6	3659	4897	3704	4086,67	Ok
7	3988	4897	3420	4101,67	Ok
8	4509	2797	3704	3670	Ok
9	4544	4197	3398	4046,33	Ok
10	4694	3462	4476	4210,67	Ok

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 48. Gráfico lineal de resultados de rendimiento opción crear periodo en varios exploradores.

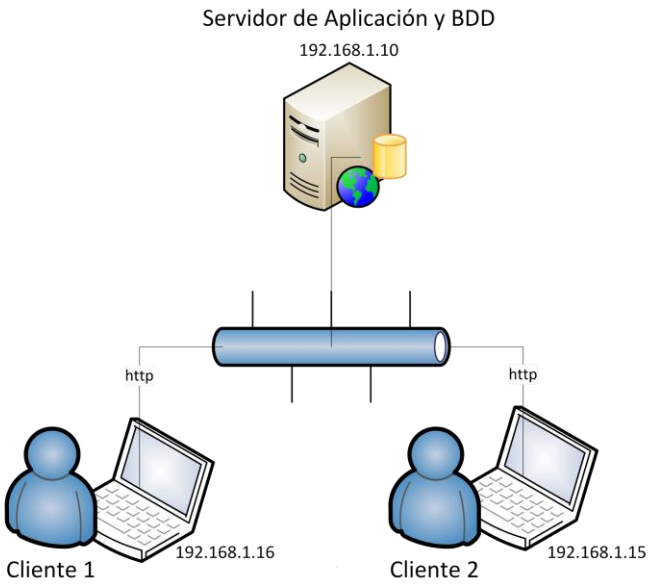


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Pruebas Cliente / Servidor

Para las pruebas de cliente/servidor se ha utilizado el explorador Google Chrome y se ha realizado cinco pruebas de la aplicación al servidor que se encontraba en otros equipos.

Figura 49. Diseño de las pruebas cliente/servidor



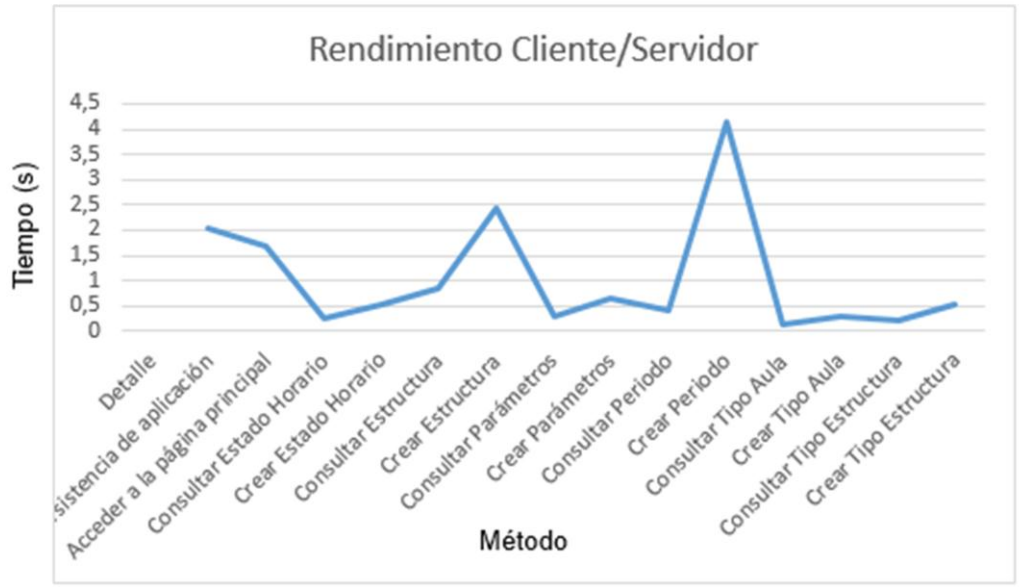
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Tabla 27. Tiempos obtenidos desde los exploradores de los clientes.

N°	Método	Muestra (s)					Promedio
		# 1	# 2	# 3	# 4	# 5	
1	Acceder a la página principal	1,58	1,91	3,65	1,41	1,61	2,032
2	Acceso al menú administrativo	1,74	1,66	1,75	1,73	1,58	1,692
3	Consultar estado horario	0,1	0,018	0,77	0,15	0,19	0,2456
4	Crear estado horario	0,38	1,12	0,36	0,36	0,38	0,52
5	Consultar estructura	0,63	2,21	0,73	0,35	0,3	0,844
6	Crear estructura	2,58	3,07	2,87	1,58	2,04	2,428
7	Consultar parámetros	0,24	0,26	0,32	0,288	0,42	0,3056
8	Parámetros	0,58	0,78	0,52	0,67	0,65	0,64
9	Consultar periodo	0,31	0,44	0,36	0,45	0,5	0,412
10	Crear periodo	4,84	3,14	4,02	3,83	4,8	4,126
11	Consultar tipo aula	0,16	0,14	0,11	0,11	0,14	0,132
12	Crear tipo aula	0,27	0,2	0,37	0,26	0,36	0,292
13	Consultar tipo estructura	0,17	0,16	0,17	0,43	0,13	0,212
14	Crear tipo estructura	0,43	0,74	0,4	0,56	0,5	0,526

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 50. Gráfica lineal del rendimiento de las pruebas cliente/servidor.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

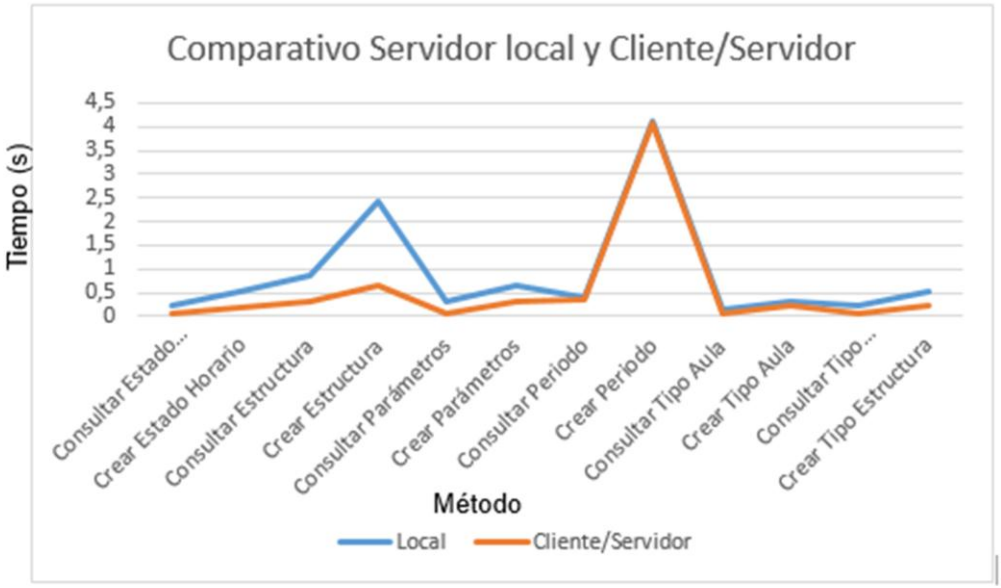
A continuación se indica una tabla comparativa con los tiempos obtenidos entre el rendimiento local del servidor y el rendimiento de las pruebas cliente/servidor.

Tabla 28. Comparativo del rendimiento entre el servidor local y cliente/servidor.

		Tiempo (s)	
N°	**Método**	**Local (3 usuarios)**	**Cliente/Servidor (2 usuarios)**
3	Consultar estado horario	0,25	0,05
4	Crear estado horario	0,52	0,17
5	Consultar estructura	0,84	0,30
6	Crear estructura	2,42	0,64
7	Consultar parámetros	0,31	0,06
8	Crear parámetros	0,64	0,33
9	Consultar periodo	0,41	0,36
10	Crear periodo	4,12	4,08
11	Consultar tipo aula	0,13	0,04
12	Crear tipo aula	0,29	0,24
13	Consultar tipo estructura	0,21	0,06
14	Crear tipo estructura	0,52	0,22

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 51. Diagrama lineal del comparativo de rendimiento entre el servidor local y cliente/servidor.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

CAPÍTULO 3

INTEGRACIÓN

Para realizar la integración entre los diferentes módulos fue necesario realizar configuraciones en el IDE de desarrollo y determinar que librerías son necesarias en la aplicación, como también cambios en la estructura original de la base de datos de esta forma minimizar incongruencias.

En los siguientes apartados se describe como se integró con los diferentes módulos.

3.1 Integración a nivel PostgreSQL

Los cambios que realizaron a nivel de la base de datos fueron aplicados de la siguiente forma:

- Unificación de tablas en la base de datos
- Validación campos nuevos en cada tabla.
- Definir nuevas secuencias de acuerdo a cada módulo.
- Integración de código.
- Configuración de reportes.

El módulo Administrativo debe ser integrado con el módulo de "Usuario", en la figura 52, se muestra el diagrama de despliegue e integración, los procesos y actividades más importantes para realizar la tarea en mención:

3.2 Definición de IDE

Al existir la libertad de que los grupos puedan seleccionar las herramientas de desarrollo para el proyecto "UPS Schedule" se generó la primera complicación, que es definir el IDE de desarrollo.

Las herramientas para el desarrollo por parte de los grupos son NetBeans y Eclipse.

Para el caso se utilizó NetBeans ya que por su facilidad de organización y compatibilidad con el resto de grupos fue escogida como la mejor opción para el desarrollo. Dentro de las principales ventajas se puede destacar:

- Reutilización de módulos.
- Uso de la herramienta Update Center Module.
- Instalación y actualización simple.
- Incluye Templates y Wizards.

3.3 Levantamiento de servidores para la integración

Cuando se haya definido el IDE para el desarrollo, se debe continuar con la configuración del servidor de aplicaciones con los recursos necesarios para la correcta ejecución de los módulos del sistema.

Para ello se procede con la configuración de la conexión con la base de datos y el servidor de aplicación como fue descrito anteriormente en el capítulo 2 (2.4.3).

3.4 Configuración en el desarrollo

Cuando se haya finalizado con la configuración del servidor de aplicaciones es necesario configurar la herramienta de desarrollo ya sea Netbeans o Eclipse de acuerdo a los requerimientos cada integrante del módulo. Para ello se debe tomar en cuenta lo siguiente:

- Instalación de librerías de acuerdo al IDE seleccionado.
- Instalar JReports y librerías correspondientes
- Resolución de dependencias.

Una vez instaladas y actualizadas las librerías necesarias para cada uno de los módulos que conforman el sistema UPS Schedule y así obtener un correcto funcionamiento en el desarrollo.

3.5 Cambios principales en el código

Durante el proceso de integración se generaron inconsistencias que impidieron la ejecución directa de los módulos, por lo que fue necesario realizar cambios, principalmente en la forma de la conexión a la base de datos.

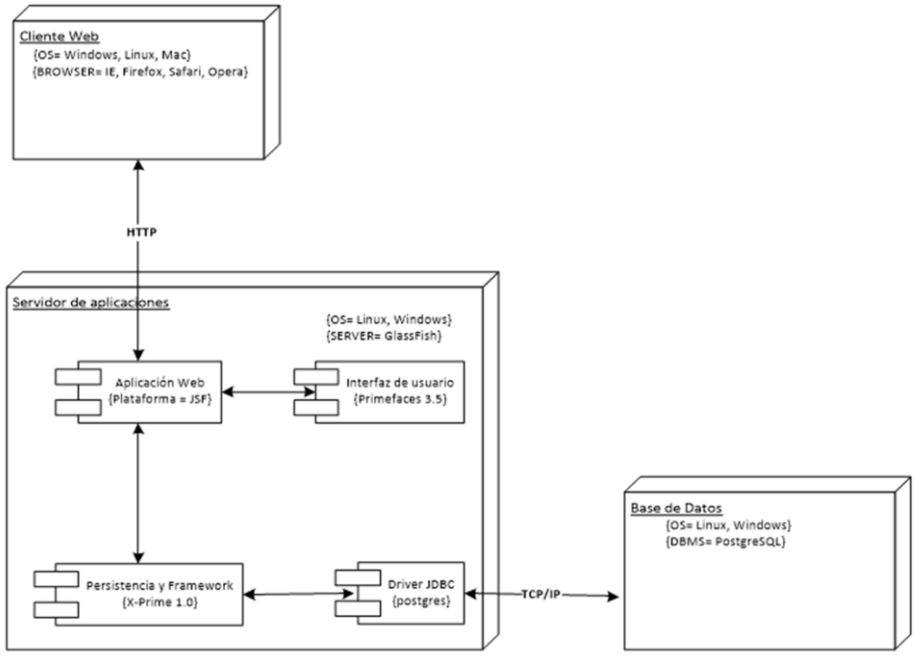
Luego se realizó la revisión del código generado por cada uno de los módulos tomando en cuenta los estándares que se definieron en un comienzo. Posteriormente se procedió a copiar los paquetes generados por cada módulo.

Previo a la ejecución de la aplicación, fue necesario configurar la pantalla principal (index.jsf) con las nuevas opciones empleadas en cada módulo. Una vez estandarizado paquetes, código y opciones nuevas se ejecutó sin inconvenientes el sistema.

3.6 Diagrama de despliegue

La figura 52 se muestra la interrelación de los diferentes componentes del sistema a través de un diagrama de despliegue. Tanto el servidor de aplicaciones, como la base de datos podrían funcionar sobre el mismo computador.

Figura 52. Diagrama de despliegue



Fuente: Tesistas “Sistema UPS Schedule de la Universidad Politécnica Salesiana sede Quito”

CONCLUSIONES

1. Se ha obtenido un módulo de administración de espacios físicos de acuerdo a los requerimientos la Universidad Politécnica Salesiana sede Quito, que constituye un gran aporte para el personal administrativo que conforma la institución, utilizando tecnologías web que se encuentran disponibles en el mercado. Situación que permitirá a corto plazo el ahorro de tiempo, de recursos humanos y económicos.

2. Durante el desarrollo y las pruebas realizadas, se evidenció que las herramientas escogidas por los integrantes del grupo, esto es GlassFish 3.1, Java y PostgreSQL 9.1 resultaron ser las mejores opciones para el desarrollo del proyecto, puesto que permitió una rápida implementación y por otro lado son herramientas de libre acceso que mejora de manera significativa el uso de una mayor cantidad de usuarios.

3. Las pruebas de rendimiento desarrolladas en el método que se encarga de la generación de un nuevo periodo lectivo en el módulo administrativo, demuestra que es capaz de generar periodos nuevos sin tiempos largos de espera que se aproximan a los 5 segundos con datos de tres carreras ingresadas.

4. Durante la pruebas realizadas sobre el módulo administrativo se obtuvo tiempos de respuesta óptimos sin afectar el rendimiento del sistema y siendo imperceptible para el usuario.

5. Las herramientas utilizadas para el desarrollo de este proyecto, bajo arquitecturas de CPU: x86, x64 bits resultaron ser compatibles y operativos con los siguientes sistemas operativos: Windows, Linux, Unix y Mac.

RECOMENDACIONES

1. Ya que este proyecto forma parte de un sistema grande se recomendable mantener un estándar en los paquetes, clases, y tablas desarrollar para facilitar la integración y comprensión con el resto de módulos.

2. Profundizar el análisis de la aplicación de los módulos del sistema por parte de los tesistas de manera conjunta, con la finalidad de propiciar espacios de intercambio de experiencias, y que permitan mejorar la experticia en el manejo integral del sistema.

3. Una vez validado el Sistema Administrativo como parte del Sistema UPS Schedule, es imperativo que los resultados alcanzados sean socializados a alumnos, docentes y personal administrativo en su conjunto, para que de manera inmediata lo puedan poner en práctica para beneficio de la UPS.

4. Es necesario que se valide en su conjunto los módulos del sistema UPS Schedule y se capacite sobre su uso a los docentes, estudiantes y administrativos, para ver de manera objetiva e integral las bondades del sistema, tendientes a mejorar en los próximos periodos lectivos el uso de aulas, laboratorios, audiovisuales, etc. en las carreras que se imparten en la universidad de forma automática.

5. Evaluar de manera periódica el uso del sistema, y en base a los informes técnicos de aplicación implementar los reajustes necesarios que permitan tener vigente su operatividad y utilidad.

LISTA DE REFERENCIAS

Pineda, I. (2011). *Sistema Inteligente de Soporte en la Generación de Horarios Académicos para la Carrera de Ingeniería en Sistemas de la Universidad Politécnica Salesiana.* Tesis de Ingeniería en Sistemas, Universidad Politécnica Salesiana, Quito, Ecuador.

Jacome, H. Rios, A. (2003). *Diseño e Implementación de un Sistema de Administración para los Laboratorios de Informática de la Universidad Politécnica Salesiana campus Sur.* Tesis de Ingeniería en Sistemas, Universidad Politécnica Salesiana, Quito, Ecuador.

Russell, S. Norving, P. (2004). *Inteligencia Artificial – Un Enfoque Moderno*. 2nd Edición. España: Pearson Educación.

Booch, G. (1996). *Análisis y Diseño Orientado a Objetos con Aplicaciones.* 2nd Edición. México, DF: Addison Wesley.

Rumbaugh, J., Blaha, M., Premerlani, W., Eddy, F. y Lorensen, W. (1996). *Modelado y Diseño Orientado a Objetos.* 1ra Edición. Madrid: Prentice Hall.

Martin, J. Odell, J. (1994). *Análisis y Diseño Orientado a Objetos.* México, DF: Prentice Hall.

Jacobson, I. (1998). *Object Oriented Software Engineering.* Boston, MA: Addison-Wesley.

Quero Catalinas, E. (2003). P*rogramación en Lenguajes estructurados.* Madrid: Thomson Editores.

Schmuller, J. (2003). *Aprendiendo UML en 24 horas.* 1ra Edición. México, DF: Pretince Hall.

Tepper, P. Murphy, B. (2010). *Persistence with Hibernate.* 1ra Edición. Estados Unidos: Apress.

Mann, K. (2005). *Java Server Faces in Action.* 1ra Edición. Estados Unidos: Manning Publications.

Bonilla, B. "*Validación y conversiones en JSF*", 2006, 2006, Recuperado el 24 de abril 2014 de;

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=convValidJSF

Perez, A. "*Introducción a JSF (Java Server Faces)*", 2006, 2006, Recuperado el [24 de abril 2014 de;

http://www.desarrolloweb.com/articulos/2380.php

Lerna, E. "*Adictos al trabajo*", 2010, 2010, Recuperado el 30 de junio 2014 de;

http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=introduccionPrimefaces

Giftsam, "*Step by step tutorial to setup Primefaces in Netbeans*", 2009, 2009 Recuperado el 10 de mayo 2014 de;

http://www.techbrainwave.com/?p=198

Espinosa, R. "*Data Prix*", 2010, 2010, Recuperado el 28 de mayo 2014 de;

http://www.dataprix.com/blogs/respinosamilla/herramientas-etl-que-son-para-que-valen-productos-mas-conocidos-etl-s-open-sour

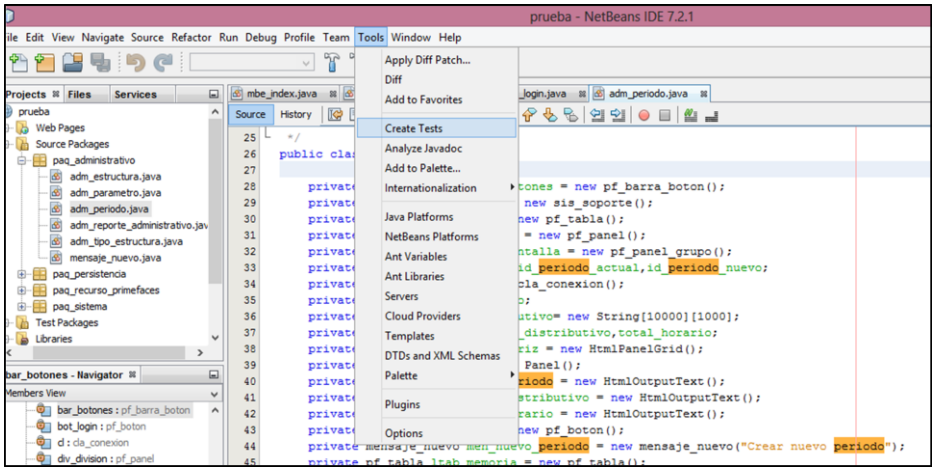
ANEXOS

Anexo 1. Proceso para la creación de pruebas unitarias en Netbeans 7.1.2 con la herramienta JUnit.

A continuación se detalla el proceso:

1. Ir a Tools y seleccionar “Create Test”.

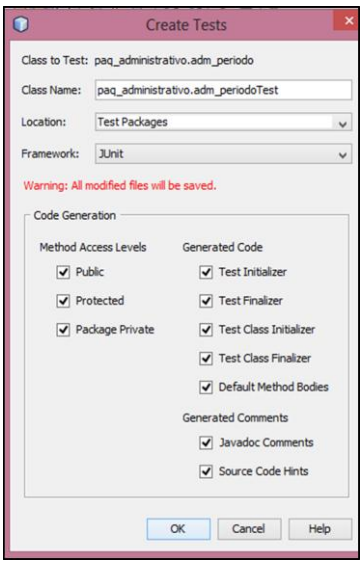
Figura 1. Creación de pruebas unitarias.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2. Se desplegará la siguiente pantalla y seleccionar “Ok”.

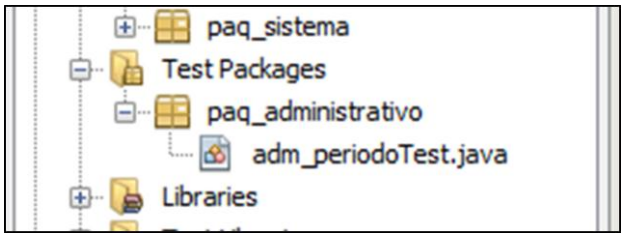
Figura 2. Opciones generales para la creación de pruebas unitarias.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

3. A continuación se generará un nuevo paquete.

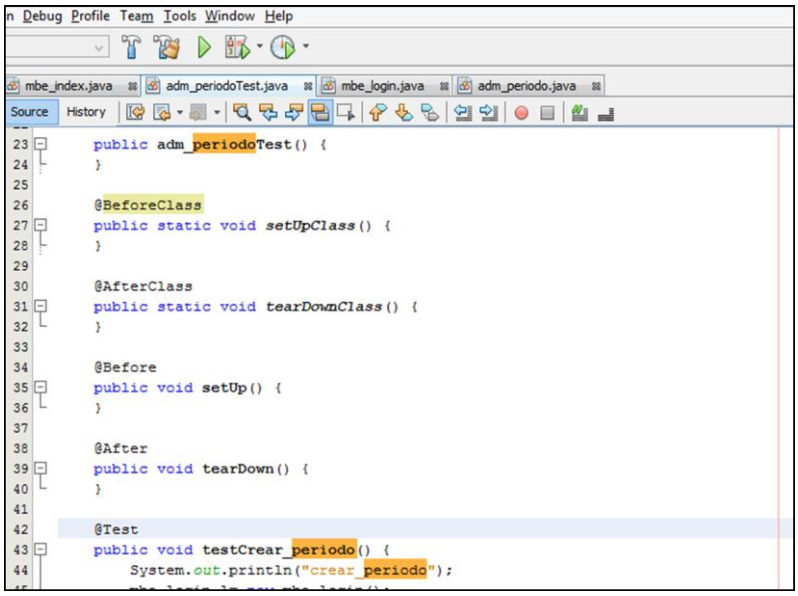
Figura 3. Ubicación paquete de pruebas.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

4. Se procede con la modificación de métodos que serán utilizados para las pruebas necesarias.

Figura 4. Clase adm_periodoTest.



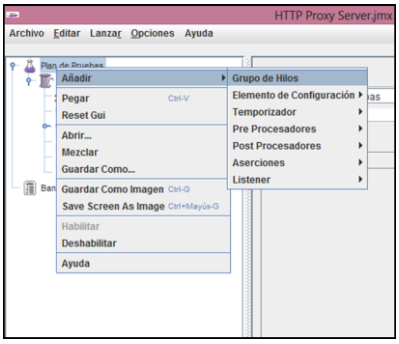
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Anexo 2. Proceso para creación de pruebas de estrés.

Para realizar las pruebas de estrés se ha utilizado la herramienta JMeter, donde a continuación se detalla el uso de la misma:

1. En primer lugar se debe agregar un grupo de hilos con tres usuarios.

Figura 1. Configuración JMeter agregar grupo de hilos.



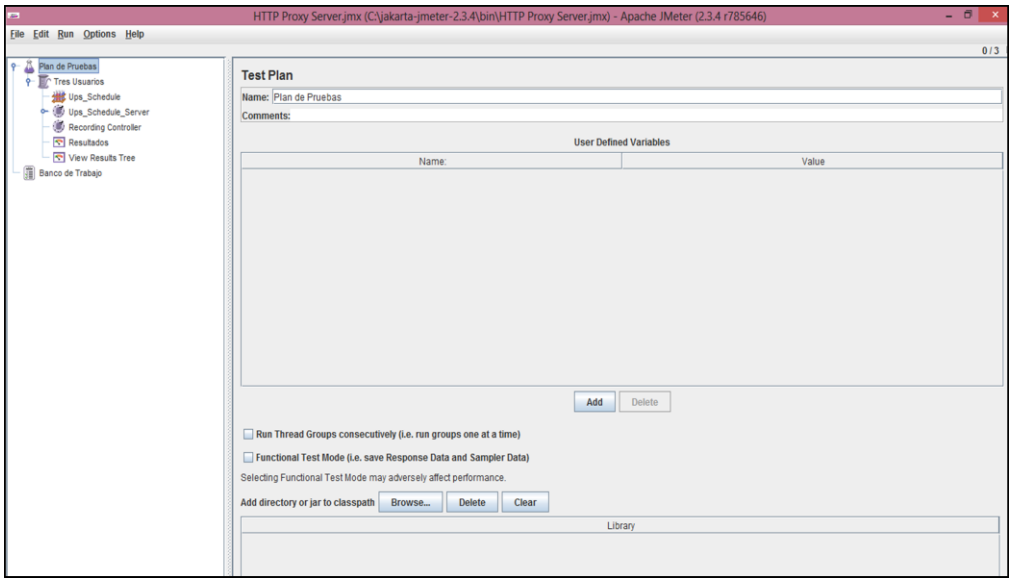
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Figura 2. Configuración Jmeter para tres usuarios administradores.

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

2. Al grupo de hilos se procede a agregar los siguientes elementos.

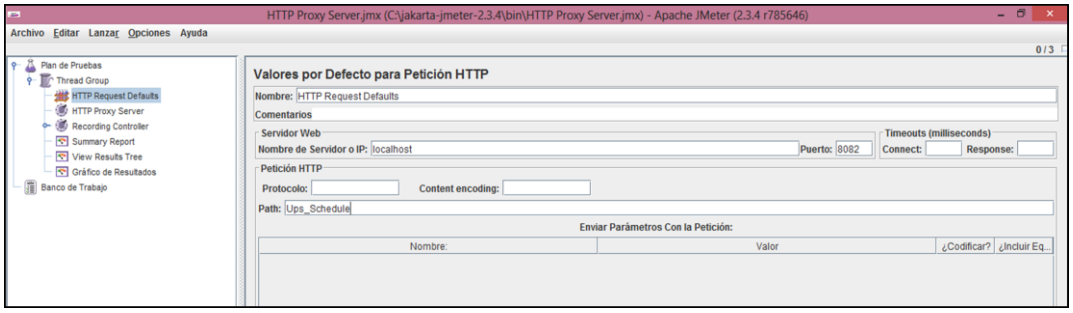
Figura 3. Configuración JMeter agregar elementos para pruebas.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

3. A continuación se configura la opción HTTP Request Defaults para la petición a la aplicación

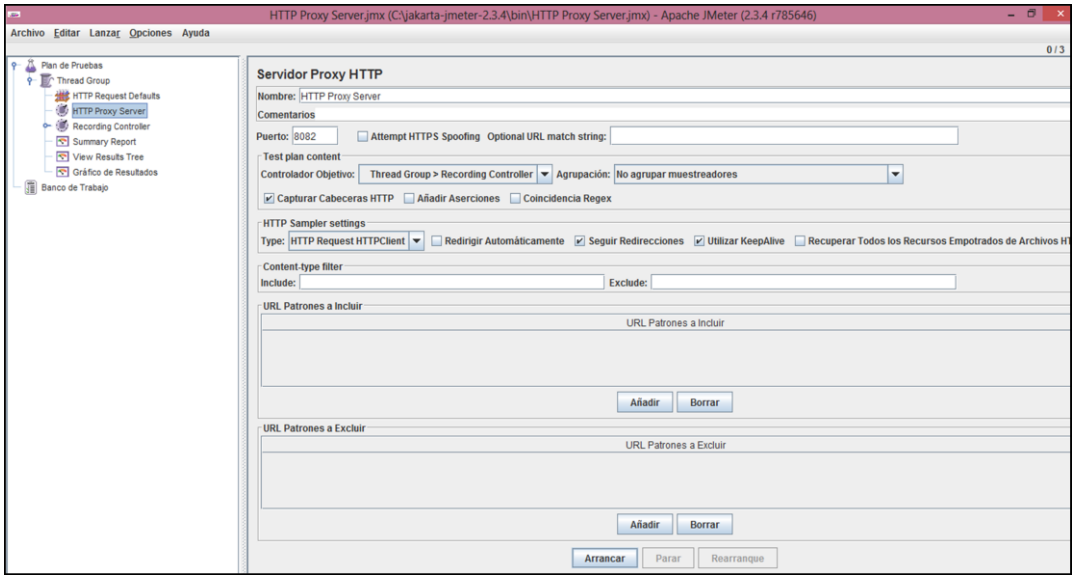
Figura 4. Configuración JMeter del elemento HTTP Request Default.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

4. A continuación se configura la opción HTTP Proxy Server y hacer clic en Arrancar para crear el escenario.

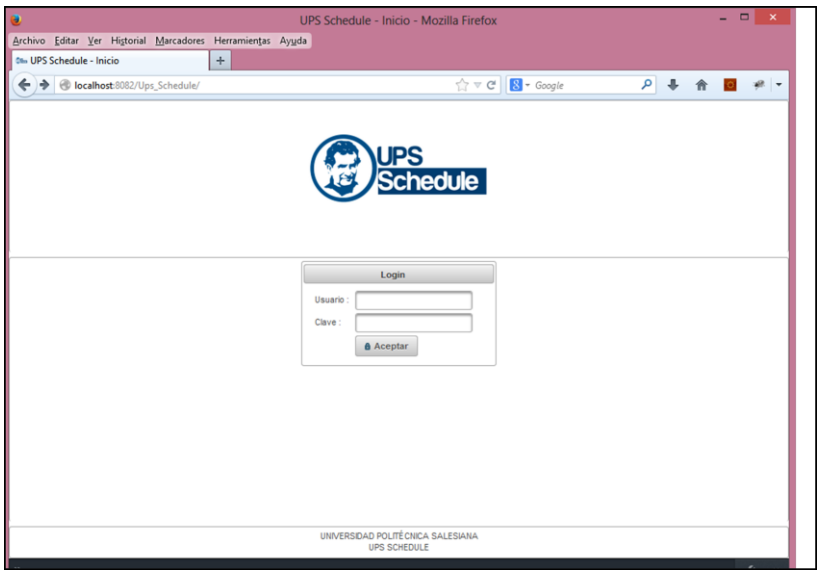
Figura 5. Configuración JMeter del elemento HTTP Proxy Server.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

5. Una vez realizado las configuraciones necesarias, probar la aplicación.

Figura 6. Pantalla principal del sistema UPS Schedule.



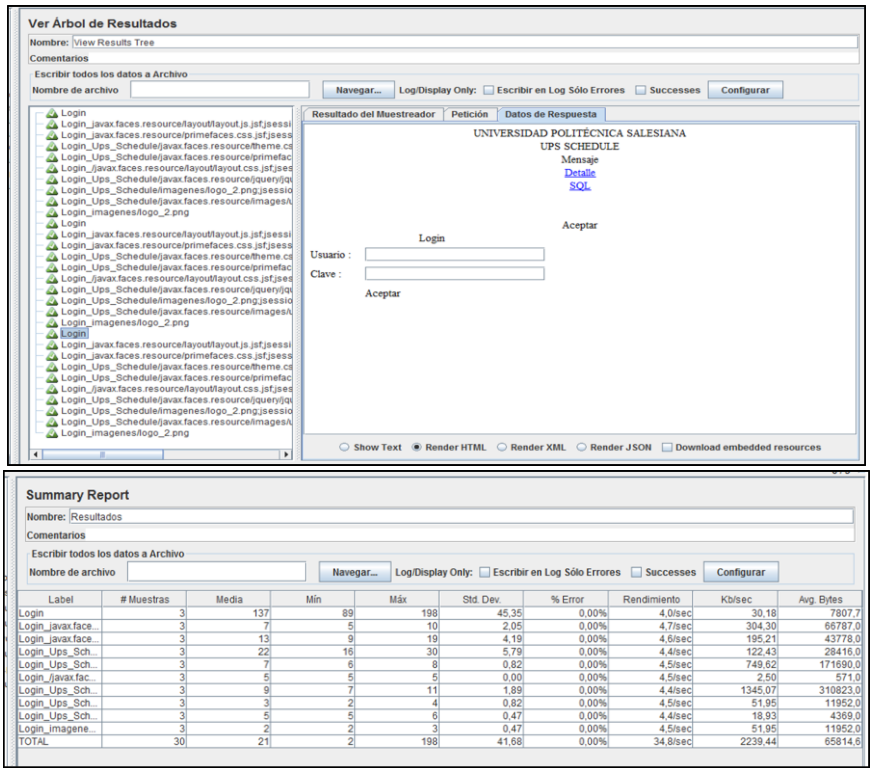
Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Anexo 3. Obtención de resultados de los métodos con la herramienta JMeter.

Los resultados obtenidos de las pruebas realizadas con tres usuarios administrativos en JMeter fueron los siguientes:

Pantalla de login

Figura 1. Resultados Pantalla "login" con la herramienta JMeter.

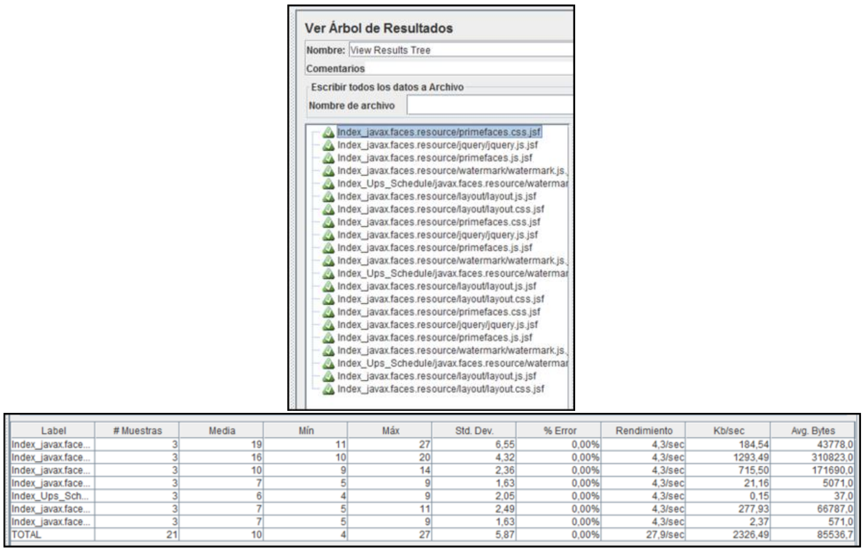


Label	# Muestras	Media	Mín	Máx	Std. Dev.	% Error	Rendimiento	Kb/sec	Avg. Bytes
Login	3	137	89	198	45,35	0,00%	4,0/sec	30,18	7807,7
Login_javax.face...	3	7	5	10	2,05	0,00%	4,7/sec	304,30	66787,0
Login_javax.face...	3	13	9	19	4,19	0,00%	4,6/sec	195,21	43778,0
Login_Ups_Sch...	3	22	16	30	5,79	0,00%	4,4/sec	122,43	28416,0
Login_Ups_Sch...	3	7	6	8	0,82	0,00%	4,5/sec	749,62	171690,0
Login_/javax.fac...	3	5	5	5	0,00	0,00%	4,5/sec	2,50	571,0
Login_Ups_Sch...	3	9	7	11	1,89	0,00%	4,4/sec	1345,07	310823,0
Login_Ups_Sch...	3	3	2	4	0,82	0,00%	4,5/sec	51,95	11952,0
Login_Ups_Sch...	3	5	5	6	0,47	0,00%	4,4/sec	18,93	4369,0
Login_imagene...	3	2	2	3	0,47	0,00%	4,5/sec	51,95	11952,0
TOTAL	30	21	2	198	41,68	0,00%	34,8/sec	2239,44	65814,6

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Ingreso de usuario y contraseña

Figura 2. Resultados obtenidos por JMeter del ingreso de usuario y contraseña del sistema.

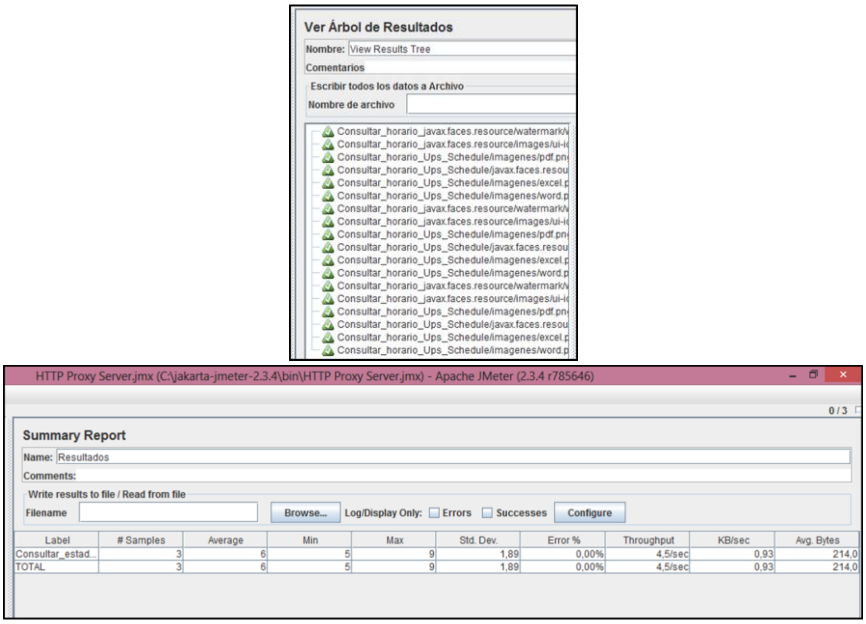


Label	# Muestras	Media	Mín	Máx	Std. Dev.	% Error	Rendimiento	Kb/sec	Avg. Bytes
Index_javax.face...	3	19	11	27	6,55	0,00%	4,3/sec	184,54	43778,0
Index_javax.face...	3	16	10	20	4,32	0,00%	4,3/sec	1293,49	310823,0
Index_javax.face...	3	10	9	14	2,36	0,00%	4,3/sec	715,50	171690,0
Index_javax.face...	3	7	5	9	1,63	0,00%	4,3/sec	21,16	5071,0
Index_Ups_Sch...	3	6	4	9	2,05	0,00%	4,3/sec	0,15	37,0
Index_javax.face...	3	7	5	11	2,49	0,00%	4,3/sec	277,93	66787,0
Index_javax.face...	3	7	5	9	1,63	0,00%	4,3/sec	2,37	571,0
TOTAL	21	10	4	27	5,87	0,00%	27,9/sec	2326,49	85536,7

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultar estado horario

Figura 3. Resultados obtenidos por JMeter de consultar estado horario.

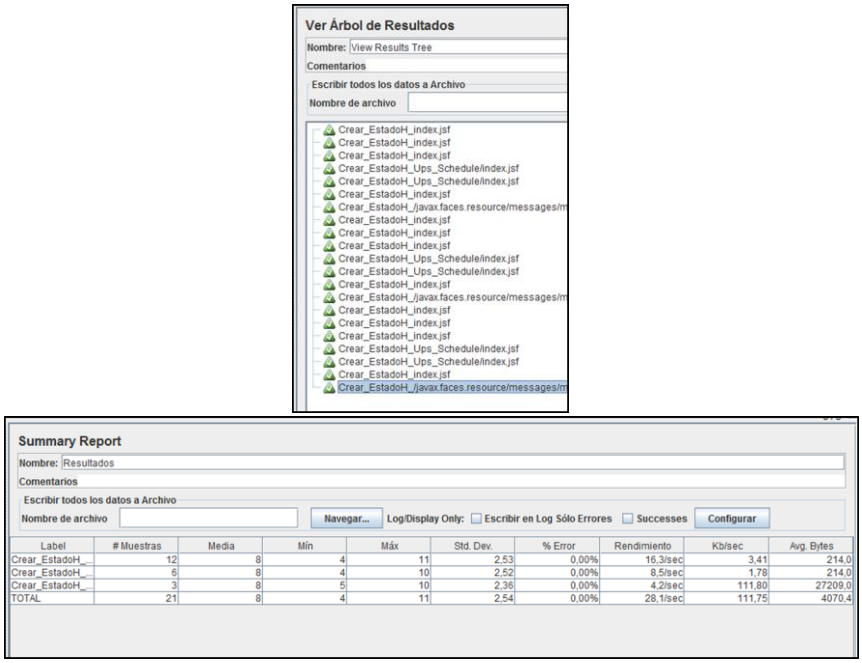


Label	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	KB/sec	Avg. Bytes
Consultar_estad...	3	6	5	9	1,89	0,00%	4,5/sec	0,93	214,0
TOTAL	3	6	5	9	1,89	0,00%	4,5/sec	0,93	214,0

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear estado horario

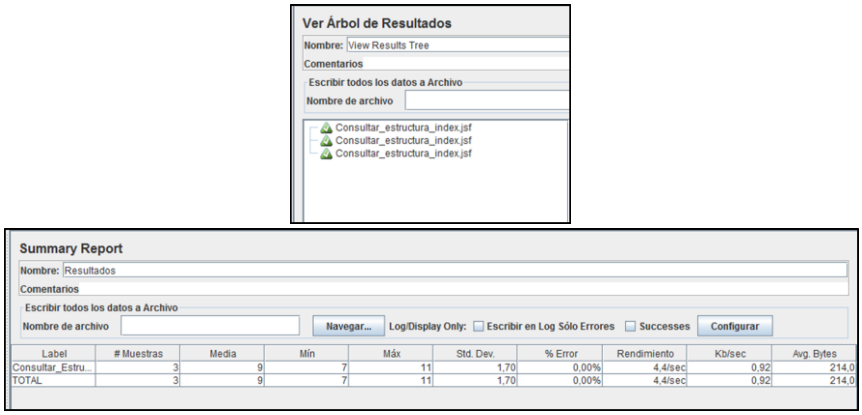
Figura 4. Resultados obtenidos por JMeter de crear estado horario.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultar estructura

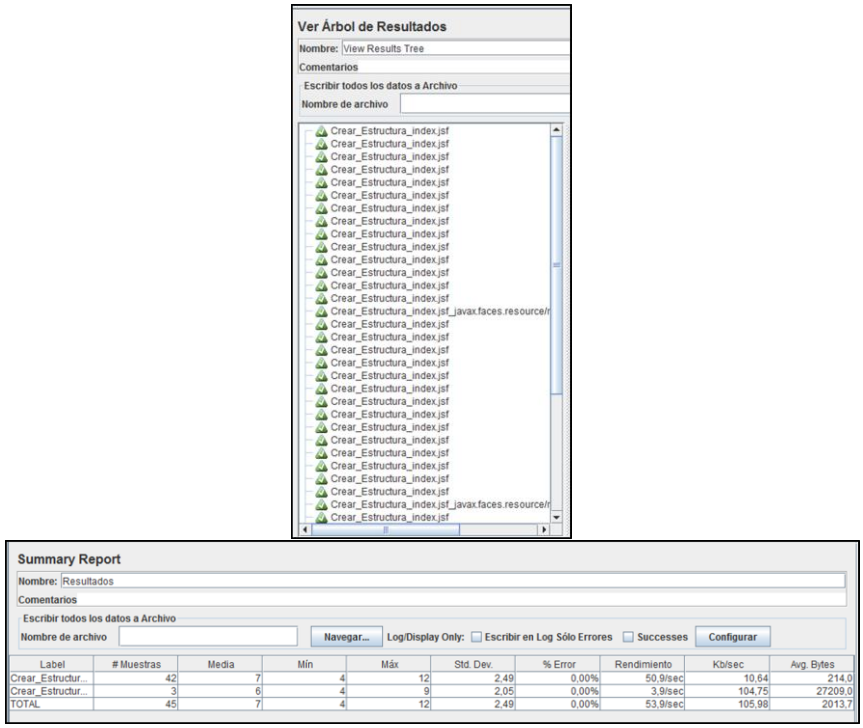
Figura 5. Resultados obtenidos por JMeter de consultar estructura.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear estructura

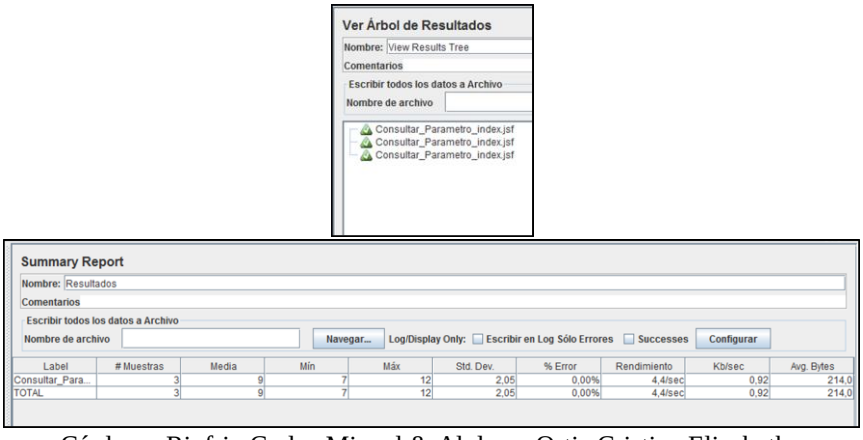
Figura 6. Resultados obtenidos por JMeter de crear estructura.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultar parámetro

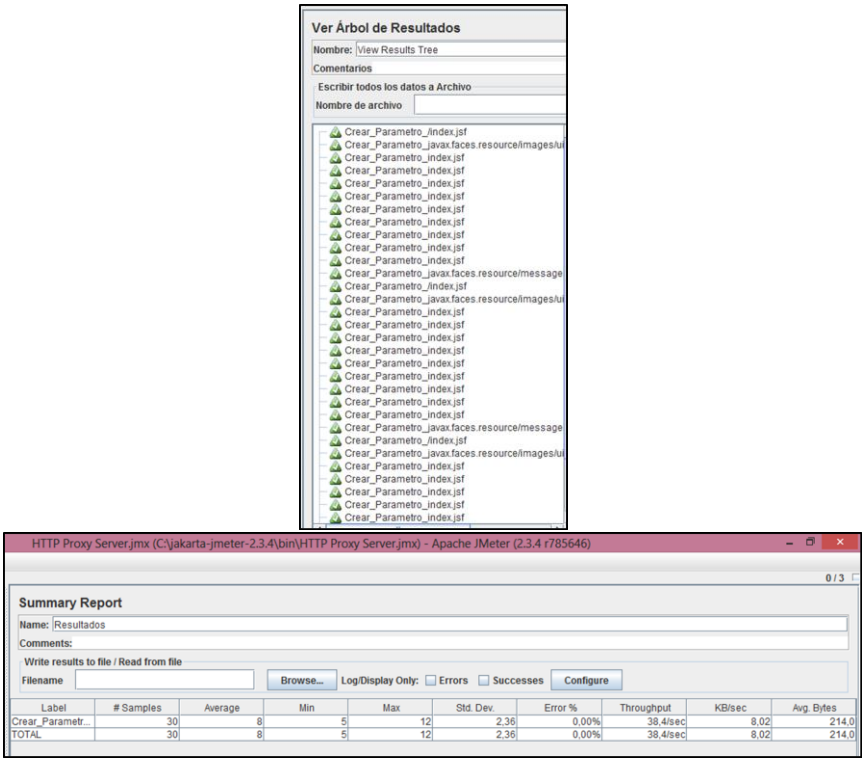
Figura 7. Resultados obtenidos por JMeter de consultar parámetro.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear parámetro

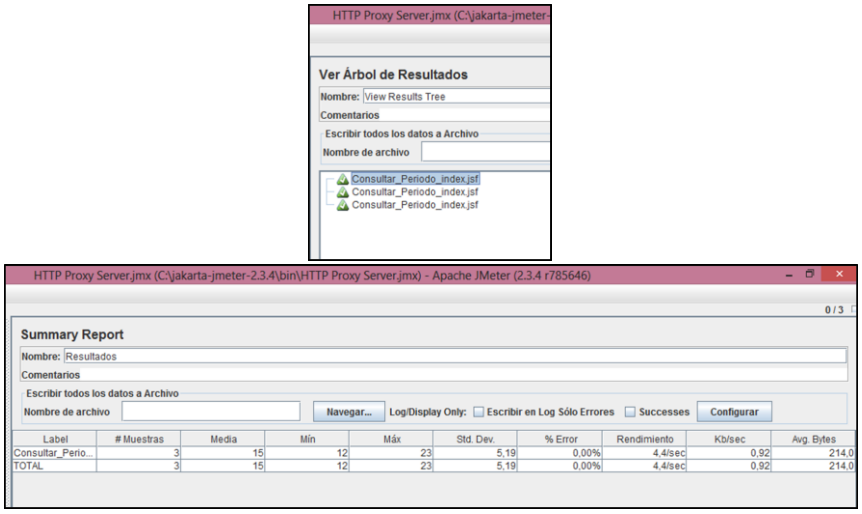
Figura 8. Resultados obtenidos por JMeter de crear parámetro.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultar periodo

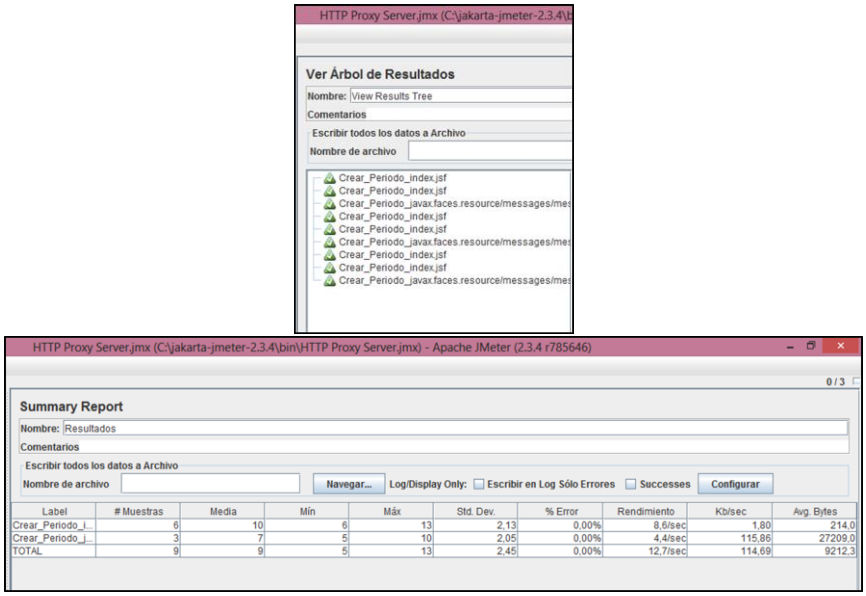
Figura 9. Resultados obtenidos por JMeter de consultar periodo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear periodo

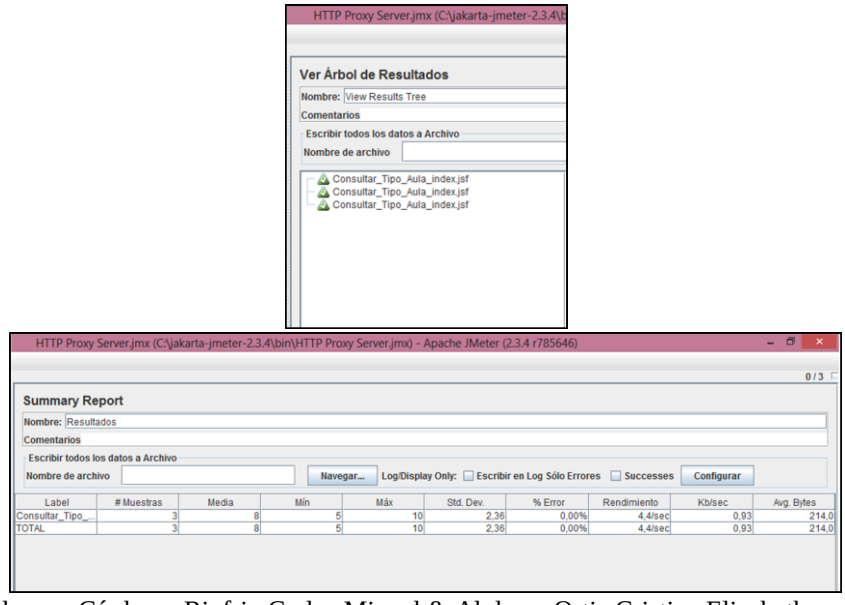
Figura 10. Resultados obtenidos por JMeter de crear periodo.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consultar tipo aula

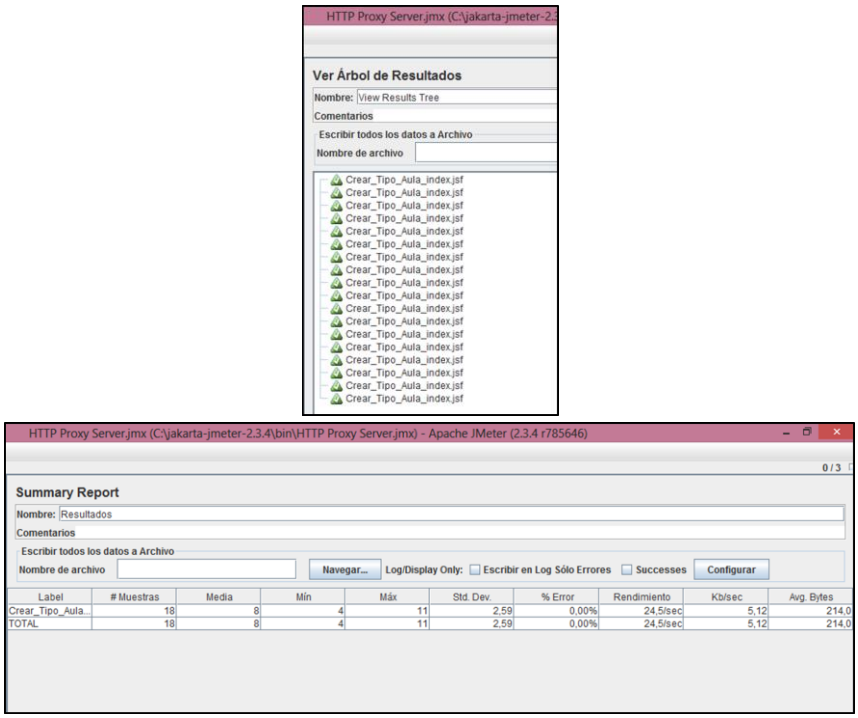
Figura 11. Resultados obtenidos por JMeter de consultar tipo aula.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear tipo aula

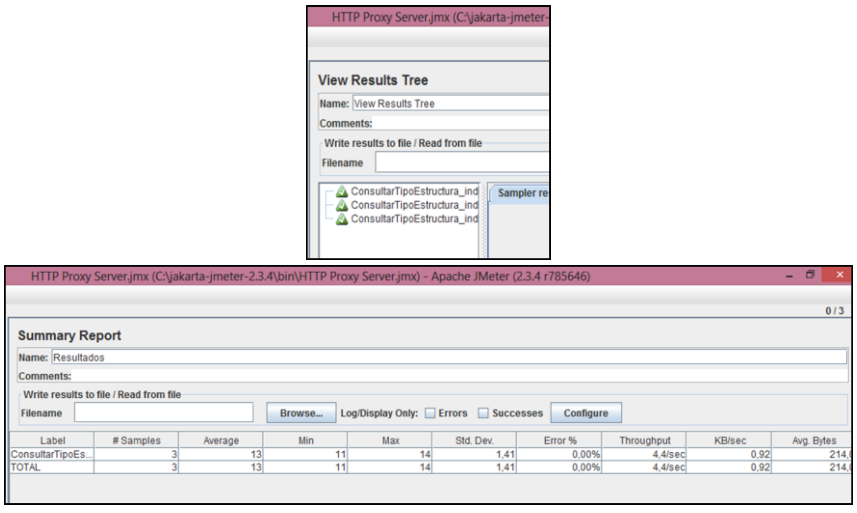
Figura 12. Resultados obtenidos por JMeter de crear tipo aula.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Consulta tipo estructura

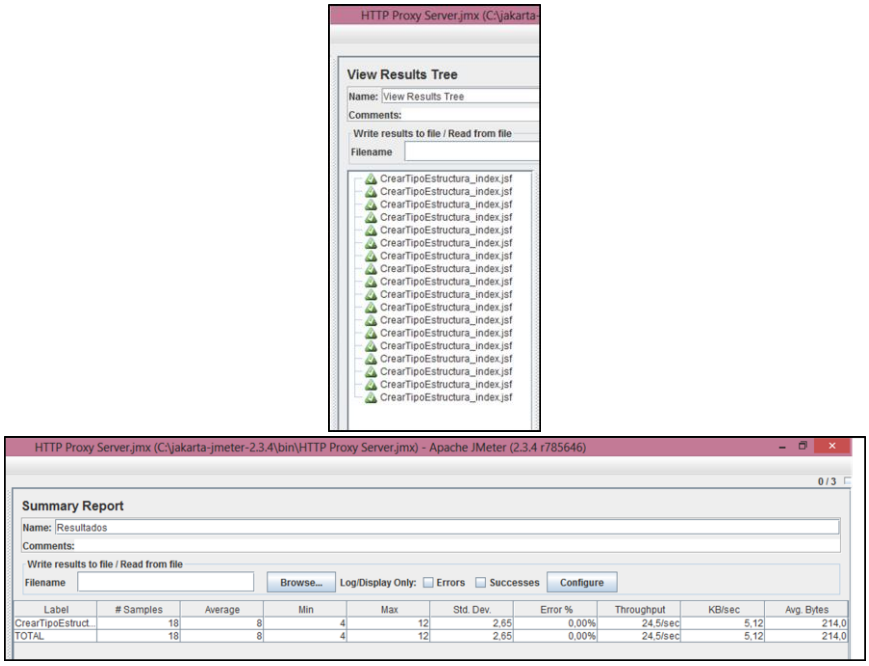
Figura 13. Resultados obtenidos por JMeter de consultar tipo estructura.



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Crear tipo estructura

Figura 14. Resultados obtenidos por JMeter de crear tipo estructura.

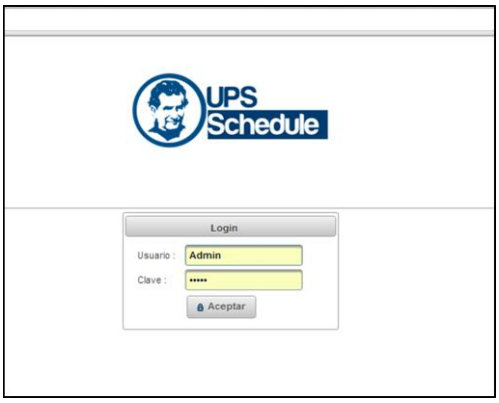


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Anexo 4. Manual de usuario.

La siguiente pantalla permite ingresar a la aplicación, en esta pantalla se debe ingresar el usuario y la clave proporcionada por el Administrador del sistema.

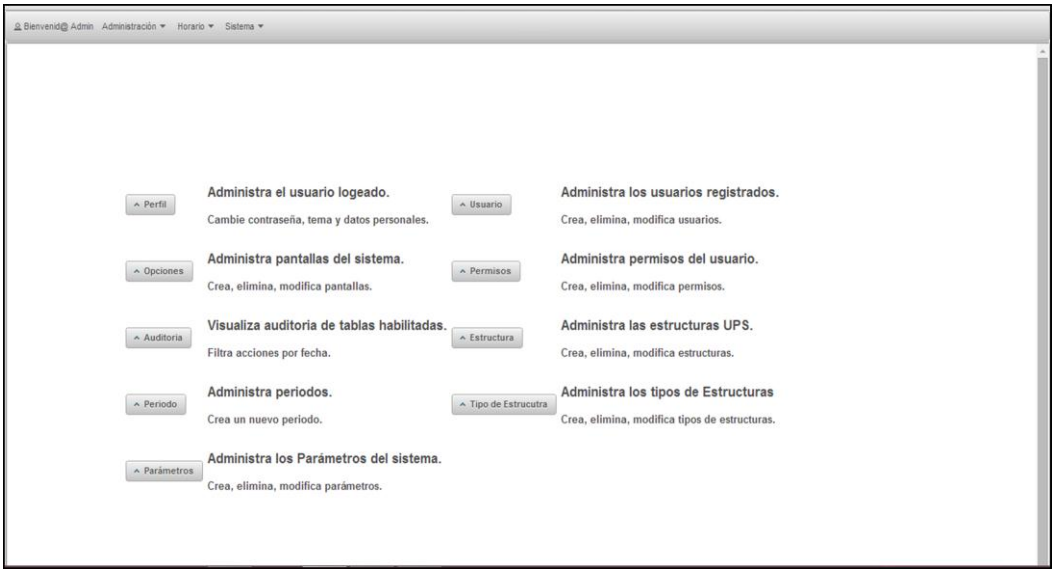
Figura 1. Pantalla principal del sistema



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Cuando se ingresa el usuario y la clave correcta se va a ingresar al sistema, y permitirá ver la pantalla principal con las diferentes opciones.

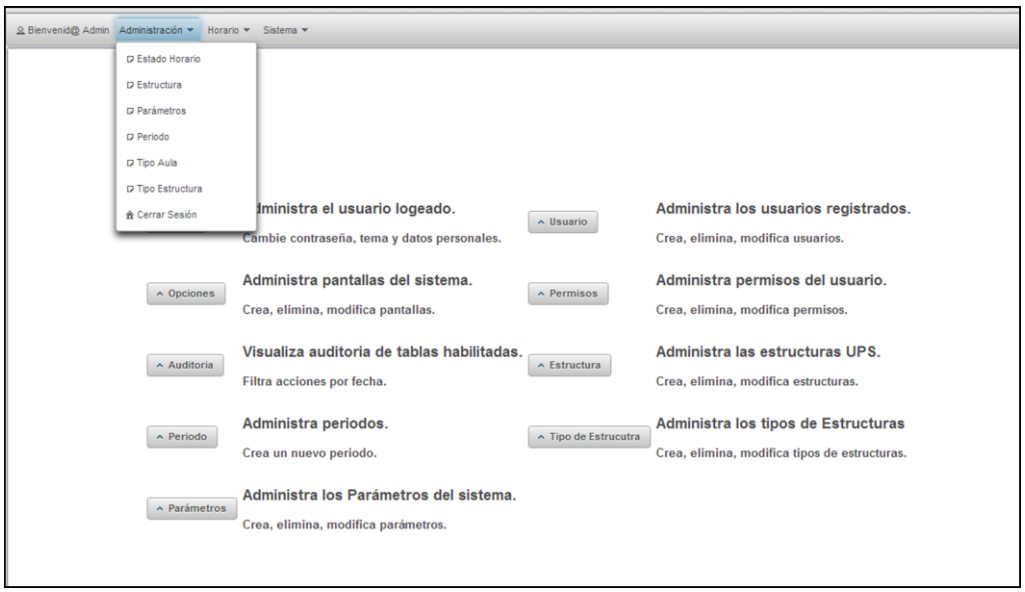
Figura 2. Pantalla Principal



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

A continuación se muestra las opciones del Módulo Administrativo.

Figura 3. Opciones del módulo administrativo

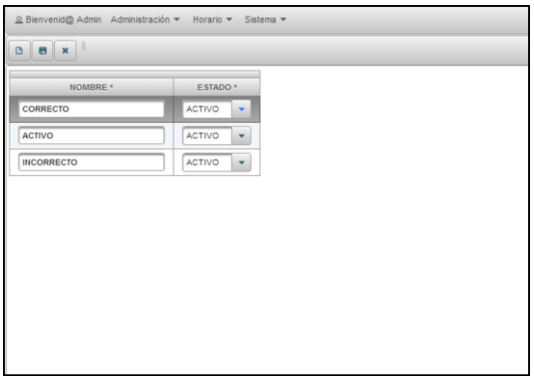


Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Opciones del sistema del módulo Administrativo:

- **Estado horario**: Esta opción muestra el estado en el que se puede encontrar un horario, en está pantalla se puede crear, eliminar, modificar el estado del horario.

Figura 4. Pantalla estado horario



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

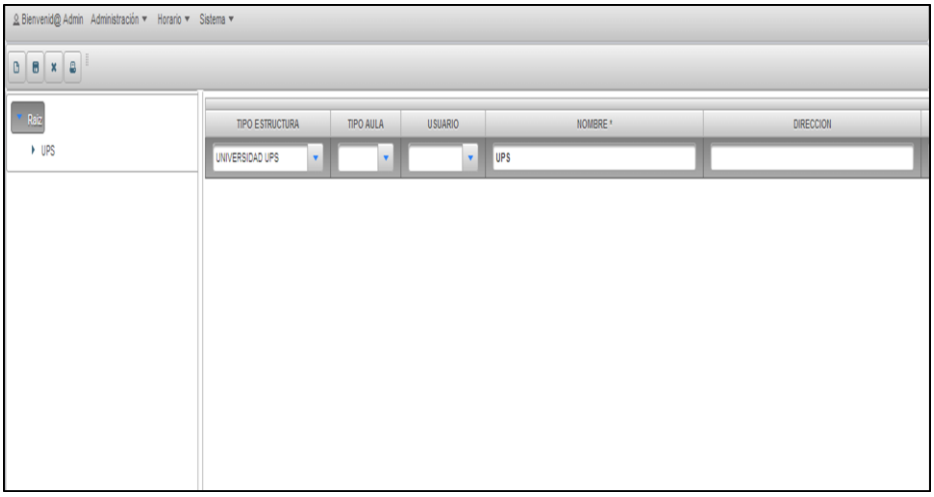
 **Nuevo**: Esta opción crea un nuevo ítem con el nombre y el estado (Activo e inactivo) del horario.

 **Modificar**: Esta opción modifica el nombre o el estado de un horario.

 **Eliminar**: Esta opción elimina un ítem de un estado de horario.

- **Estructura:** Esta opción gestiona una estructura la cual es un espacio físico, esta pantalla muestra la información de una estructura que puede ser una sede, campus, edificio, aula, laboratorio, entre otros.

Figura 5. Pantalla estructura



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

 **Nuevo:** Esta opción crea una estructura con la información requerida dependiendo de la estructura.

 **Modificar:** Esta opción modifica la información de una estructura.

 **Eliminar:** Esta opción elimina un ítem de una estructura.

- **Parámetros:** Esta opción gestiona un parámetro que va a ser usado durante la ejecución del sistema.

Figura 6. Pantalla parámetros

Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

 **Nuevo:** Esta opción crea un parámetro con la estructura, nombre y valor que va a usar.

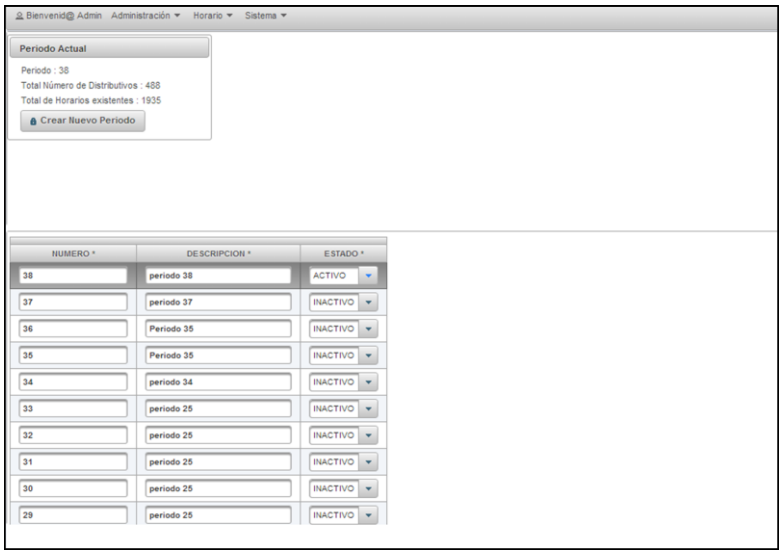
 **Modificar:** Esta opción modifica la información de un parámetro.

 **Eliminar:** Esta opción elimina un ítem de un parámetro.

 **Reporte:** Esta opción crea un reporte de los campus y su estructura.

- **Periodo:** Esta opción crea un nuevo periodo basándose con la información del periodo anterior activo, además muestra la información de los periodos anteriores.

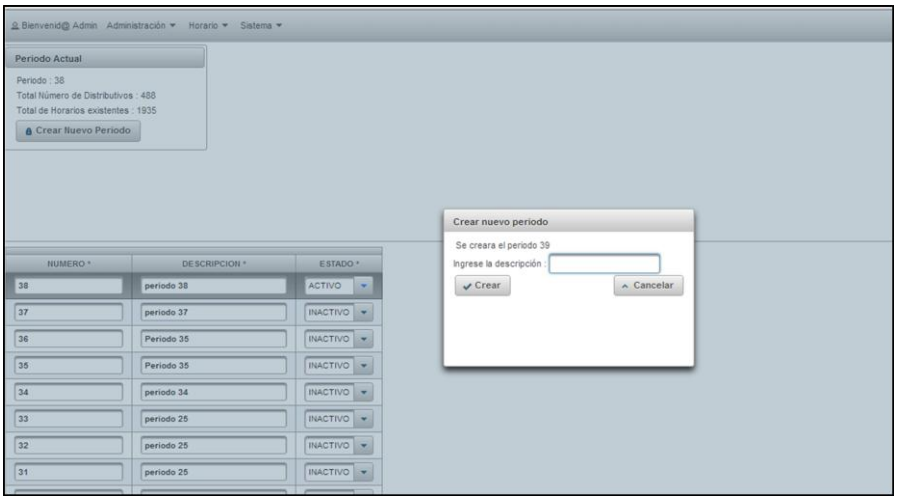
Figura 7. Pantalla periodo



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

Cuando se escoge la opción de crear un nuevo periodo aparecerá la siguiente pantalla, en la cual se debe ingresar la descripción del nuevo periodo, este campo no deberá ser nulo.

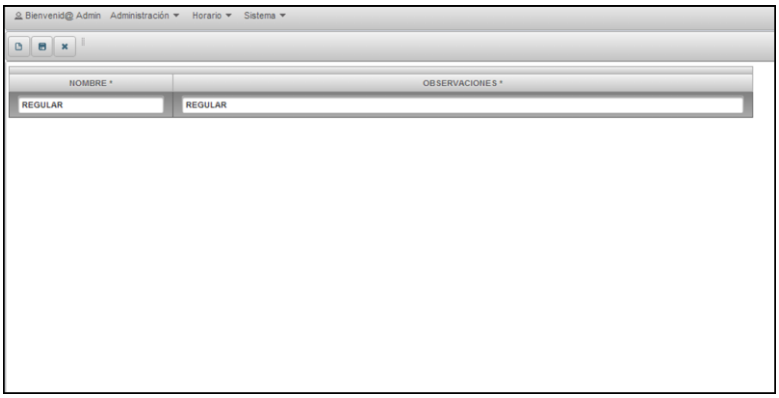
Figura 8. Pantalla crear nuevo periodo



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

- **Tipo aula:** Esta opción gestiona la información de un tipo de aula.

Figura 9. Pantalla tipo aula



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

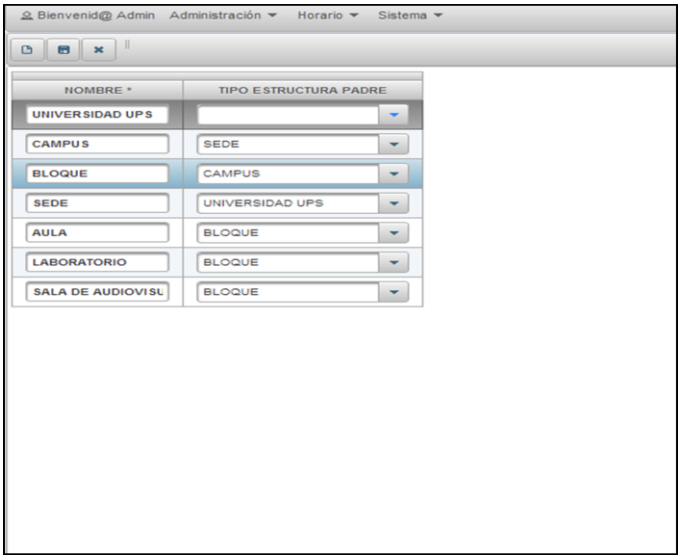
 **Nuevo:** Esta opción crea un tipo de aula con el nombre y la observación.

 **Modificar:** Esta opción modifica la información de un tipo de aula.

 **Eliminar**: Esta opción elimina un ítem de un tipo de aula.

- **Tipo estructura:** Esta opción gestiona un tipo de estructura.

Figura 10. Pantalla tipo estructura



Elaborado por: Cárdenas Riofrio Carlos Miguel & Alulema Ortiz Cristina Elizabeth

 **Nuevo:** Esta opción crea un tipo estructura con el nombre y el tipo de estructura Padre.

 **Modificar:** Esta opción modifica la información de un tipo estructura.

 **Eliminar:** Esta opción elimina un ítem de un tipo estructura.