UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE QUITO

CARRERA: INGENIERÍA DE SISTEMAS

Trabajo de titulación previo a la obtención del título de:
INGENIERA E INGENIERO DE SISTEMAS

TEMA:
DISEÑO, CONSTRUCCIÓN E IMPLEMENTACIÓN DE UNA APLICACIÓN WEB PARA LA GESTIÓN DE LAS ACTIVIDADES INTERNAS DEL LABORATORIO DE FÍSICA DE LA UNIVERSIDAD POLITÉCNICA SALESIANA SEDE QUITO CAMPUS SUR

AUTORES:
MYRIAN MAGDALENA GUIJARRO COPO
JORGE DANIEL JARAMILLO NARVÁEZ

DIRECTOR:
WASHINGTON ARSENIO RAMÍREZ MONTALVAN

Quito, julio de 2015

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 titulación y su reproducción sin fines de lucro.

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

Quito, julio de 2015

_____ _____

Myriam Magdalena Guijarro Copo
C.C. 172104089-5

Jorge Daniel Jaramillo Narváez
C.C. 172182096-5

DEDICATORIA

A Dios, por haberme permitido terminar mi carrera.

A mis padres, que de manera incondicional me prodigaron fortaleza, determinación, a través de la palabra de Dios, y de su amor para que no diera oportunidad a la derrota de rendirme.

Gracias a ustedes queridos padres por no mirar mis innumerables faltas sino apoyarme y levantarme en cada caída, e instarme a avanzar cada momento.

A mi mejor amigo, Jorgito, por haber caminado conmigo en esta travesía, por su paciencia, serenidad, sabiduría ,amistad, aunque en el camino tuvimos adversidades permanecimos juntos gracias querido amigo por todo, ya que para mí más que un amigo eres como mi hermano.

Finalmente a mi hermano Wellington por recordarme que los sueños se consiguen a base de esfuerzo y dedicación, caminando mirando mi objetivo que aunque era difícil lo iba a lograr. Gracias por haberme dado ánimo y estar siempre a mi lado.

Myrian Magdalena Guijarro Copo

A mi familia por el ejemplo, esfuerzo y apoyo constantes demostrados durante toda mi vida académica.

A Valeria, mi compañera y soporte en todo momento.

Jorge Daniel Jaramillo Narváez

AGRADECIMIENTO

A la Universidad Politécnica Salesiana, por aportar con todo el conocimiento y herramientas para lograr la culminación de nuestra carrera.

A nuestro tutor y amigo Ing. Washington Ramírez, por su acertada guía durante el desarrollo de este proyecto y llevarnos a sobrepasar nuestro límite al mostrarnos que siempre hay algo más para dar. Por su dedicación, tiempo y compromiso. Por haber sido un miembro más del equipo de trabajo y caminar siempre a nuestro lado, mostrándonos las mejores alternativas para alcanzar nuestro objetivo.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

RESUMEN

El presente proyecto de titulación comprende las fases de análisis, diseño, construcción e implementación de un sistema informático orientado a la web para la gestión de las actividades académicas y administrativas del Laboratorio de Física de la Universidad Politécnica Salesiana, sede Quito Campus sur.

En primera instancia se describe y analiza la situación actual del Laboratorio y la necesidad del desarrollo del sistema propuesto, tomando en cuenta que el uso de las tecnologías de la información y comunicación, como una herramienta de apoyo para los procesos de enseñanza aprendizaje, es de importancia vital para los docentes en la actualidad. También se incluye como parte fundamental del análisis la normativa del CEAACES para el diseño de evaluaciones en base a reactivos que permiten valorar conocimientos concretos y específicos en los estudiantes.

Partiendo de esta situación, el proyecto se orienta hacia el desarrollo de un sistema web mediante el uso de una metodología ágil de desarrollo como lo es Scrum, esta metodología permite que todos los involucrados en el proyecto vean rápidamente avances concretos en el desarrollo del producto final, a la vez que facilita la rápida adaptación del equipo de desarrollo a los cambios en los requerimientos del proyecto sin afectar la dirección del mismo.

Al finalizar todas las fases de desarrollo se obtendrá una aplicación robusta y estable en JSF que permitirá gestionar de mejor manera las actividades del Laboratorio de Física y que sirva como base para el desarrollo de futuros proyectos de la Universidad Politécnica Salesiana.

ABSTRACT

This degree project comprises the phases of analysis, design, construction and implementation of a web-oriented computer system for management of academic and administrative activities of the Physics Laboratory at the South Campus of the Politécnica Salesiana University, located in Quito.

In the first instance it describes and analyzes the current situation of the Laboratory and the need for the development of the proposed system, taking into account that the use of information and communication technologies, as a support tool for teaching and learning processes, is of vital importance for teachers today. Also it is included as a fundamental part of analysis the CEAACES regulations for designing evaluations based on simple questions that value concrete and specific knowledge in students.

Based on this, the project aims at developing a web-oriented system using an agile development methodology such as Scrum, this methodology allows everyone involved in the project to quickly see concrete progress in the development of the final product while facilitating rapid adaptation of the development team to changes in the project requirements without affecting the direction itself.

Upon completion of all phases of development, a robust and stable JSF application will be obtained, that will allow better management of the Physics Laboratory activities and will serve as a basis for the development of future projects of the Politécnica Salesiana University.

INTRODUCCIÓN

Los planteamientos académicos desde sus inicios fueron realizados en la pizarra del salón de clases, donde escasamente se contaba de alguna tecnología moderna, que apoyara integralmente el proceso académico, Después con la evolución de la informática, el software permitió facilitar el trabajo de los docentes y de los alumnos para realizar sus tareas (Rosado & Herreros, 2005)

Simultáneamente fue el aparecimiento de sistemas multimedia que tuvieron una importante aportación, además de las herramientas interactivas que fomentan la participación activa y el autoaprendizaje del alumno. La forma de enseñanza tradicional sin un soporte tecnológico virtual que la acompañe se vuelve obsoleto e ineficaz, donde prima el replanteamiento de la forma de enseñanza, en donde la interacción de los estudiantes es clave para la reconstrucción de los conocimientos que habitualmente se transmiten, previamente elaborados.

El proceso de transformación académico debe ser normado de acuerdo a organismos de control. A través de la creación del Consejo de Acreditación y Aseguramiento de la Calidad de la Educación Superior (CEAACES), se pretende delimitar las bases y criterios para alcanzar esta excelencia, mediante lineamientos que se deben seguir, como la selección del tipo de preguntas dadas por el CEAACES, reactivos que se deben utilizar al elaborar preguntas para las evaluaciones que garanticen el cumplimiento de los objetivos de las pruebas académicas. (Solis Murillo, 2013)

En cuanto al proceso de evaluación, el CEAACES plantea la inclusión de reactivos; evaluaciones en base a preguntas de selección, agrupados en varios tipos según como están estructurados: simple, ordenamiento, relación de columnas, completamiento, elección de elementos y dependiente de contexto. Según la guía para la construcción de reactivos del CEAACES, todos los tipos comparten los mismos elementos: base o cuerpo donde se expresa la situación o problema y opciones de respuesta. El docente, al momento de construir sus evaluaciones, tiene la libertad de seleccionar entre los distintos tipos de reactivo según su criterio.

Los avances experimentados en los últimos años en lo que podríamos denominar tecnologías en la educación, nos ha llevado a la evolución constante en el uso y manejo de herramientas docentes. A medida que las asignaturas van adquiriendo un

carácter tecnológico, la clase magistral ya no satisface las expectativas de los alumnos y las propuestas académicas que requieren los alumnos necesitan de mayor desarrollo y complejidad, por lo que hay que recurrir al desarrollo de aplicaciones de enseñanza-aprendizaje y el uso de las Tecnologías de la Información y Comunicación (TIC), optimizando recursos tales como tiempo y costos (Rothamel, 2014), para los laboratorios de la materia de física y en materias afines a las carreras de ingeniería.

En la última década, el internet y sus tecnologías emergentes han provocado el cambio tecnológico más importante en cuanto a redes de comunicación se refiere, esto abre un camino en todas las áreas de investigación mediante el uso de herramientas para el intercambio de información.

En la actualidad y fundamentalmente en las carreras técnicas, como las ingenierías, es necesario realizar un elevado número de prácticas de forma que el estudiante adquiera los conocimientos teórico-prácticos suficientes como complemento de sus estudios.

En este momento las TIC han abierto la puerta a otra forma de docencia, entre las que podemos destacar la basada en la utilización del internet. Ya que para todo tipo de aplicaciones educativas, las TIC son medios y no fines. Es decir, son herramientas y materiales de construcción que facilitan el aprendizaje, el desarrollo de habilidades y distintas formas de aprender, estilos y ritmos de los alumnos.

En las universidades de Argentina, la ley de Educación Superior contempla la necesidad de un replanteamiento y enriquecimiento de la enseñanza universitaria, a fin de hacerla socialmente más atractiva para los jóvenes estudiantes. (Art.44 Ley de educación superior Argentina), donde se determinó que existía una imperiosa necesidad de mejorar los laboratorios de enseñanza de la materia de física y sugiriendo que habría que considerar distintos programas para mejorar la calidad de la enseñanza de pregrado y postgrado en estas universidades, proponiendo nuevas alternativas e incentivando a sus estudiantes a cumplir los objetivos que se han propuesto.

Durante el XI Congreso Internacional sobre Tecnologia y Educación a Distancia, Monge-Nájera, Rossi y Méndez-Estrada afirman que los primeros laboratorios

virtuales, en América Latina, comenzaron a desarrollarse en el año 1997 en el Centro de Investigación Académica de la Universidad Estatal a Distancia de Costa Rica, siendo estos los primeros laboratorios virtuales para enseñanza a distancia a nivel mundial, cuatro años más tarde se emprenden nuevos proyectos académicos desarrollados en Estados Unidos y Canadá. En América Latina el único organismo que ha realizado proyectos de esta índole es la UNED. Los autores hacen referencia al incremento en el número de proyectos semejantes y que la mayoría se refieren al área de física, aunque también los hay de química y biología; en su mayor parte tratándose de simulaciones mediante software escrito en Java (Monge Nájera, Rivas Rossi, & Méndez Estrada, 2002).

Algunos ejemplos de este tipo de simulaciones para el desarrollo de micro prácticas en física pueden verse en el portal web diseñado en la Universidad de Oregón y también en el sistema desarrollado en el Centro Nacional de Información y Comunicación Educativa de Madrid. Si bien, ambos ejemplos tratan de aplicaciones interactivas que permiten a los estudiantes desarrollar experimentos de forma virtual, el enfoque del trabajo realizado por la institución española es más completo ya que resalta la parte teórica y de conocimientos básicos previo al desarrollo del ejemplo práctico.

Durante los últimos años, grupos de diferentes universidades españolas han trabajado en el desarrollo de nuevas estrategias docentes para la enseñanza; en el ámbito de la educación no presencial, enfocándose en el área de las prácticas de laboratorio mediante el diseño de ambientes interactivos y el uso de internet en el mundo académico como una herramienta adicional que permite una relación más amplia entre el docente y el estudiante.

En esta línea de investigación académica también se evidencia el desarrollo del laboratorio remoto basado en internet para la docencia de las asignaturas de Automática (Domínguez, Reguera, & Fuertes, 2005), como una solución necesaria para transformar todo el sistema académico.

Es por ello que el uso de un sistema; que funcione en cualquier sistema operativo y permita automatizar la gestión de los laboratorios de física, es una necesidad actual

cuyo objetivo básico es lograr un producto tan bueno como los de los países más avanzados en docencia de la materia, a un costo menor.

CAPÍTULO 1
SITUACIÓN GENERAL Y PLANTEAMIENTO DEL PROYECTO

1.1 Descripción del problema

La Universidad Politécnica Salesiana sede Quito campus sur cuenta con un laboratorio para la materia de física, donde se realizan actividades como: toma de evaluaciones a los estudiantes, elaboración de prácticas de laboratorio, entrega de informes, préstamo de materiales, etc. El control de todas estas actividades se lleva a cabo de forma manual.

Para entender de mejor manera las actividades del laboratorio se las puede dividir en dos grupos: gestión académica y gestión administrativa.

La gestión académica incluye todas las actividades relacionadas con el estudiante de la materia del laboratorio de física, quien para realizarlas es asignado a un grupo de acuerdo a la materia y la carrera. Este registro se lleva a cabo mediante una hoja de cálculo.

El docente divide al grupo en mesas de trabajo que tienen un número máximo de estudiantes. Cada mesa de trabajo debe designar a uno de sus integrantes como líder, el cual tendrá la responsabilidad de evaluar a sus compañeros en función del desempeño mostrado durante el desarrollo de la práctica de laboratorio.

Las prácticas de laboratorio están acompañadas de su respectivo material de apoyo, una guía impresa para los estudiantes donde está toda la información necesaria: objetivos, marco teórico, materiales y procedimientos para realizar las prácticas.

Las evaluaciones a los estudiantes son de tres tipos: coloquio, defensa e informe. Al iniciar cada clase se evalúa defensa y coloquio, lo que resta tiempo a los estudiantes para desarrollar su trabajo. La defensa es una evaluación sobre la práctica de laboratorio realizada en la clase anterior, el coloquio evalúa los conocimientos del estudiante antes de la práctica que va a realizar y finalmente el informe es un documento que resume el trabajo realizado por todos los miembros de la mesa de trabajo, es entregado la clase posterior a la realización de la práctica.

El 100% de estas evaluaciones son elaboradas en papel, esto implica que cada semana los docentes deben emplear un tiempo excesivo en el proceso de revisar y calificar cada hoja. Como resultado, existe una gran cantidad de papel mal utilizado además del tiempo adicional invertido en la tarea.

Las notas son manejadas de forma manual mediante un formato prediseñado en una hoja de cálculo, donde se ingresan los valores obtenidos en cada evaluación. La nota de la práctica de laboratorio se calcula sumando los aportes obtenidos en coloquio, defensa y el promedio entre el informe con la evaluación de líder.

La gestión administrativa comprende las actividades relacionadas con el inventario: préstamos, devoluciones, pérdidas y mantenimiento de los materiales y equipos, además del registro del mobiliario disponible en el laboratorio. El control de estas actividades se realiza de forma manual a través de una hoja de cálculo donde se registra cada una de estas.

Los equipos y materiales pueden ser solicitados por usuarios internos o externos al laboratorio. El control de los préstamos se realiza mediante registro manual, donde el usuario debe llenar la siguiente información: nombre del solicitante, cedula, tipo de bien, fecha de préstamo, fecha de devolución y estado.

En el inventario del laboratorio de la materia de física se lleva un control minucioso de cada equipo o elemento y la cantidad disponible, discriminado de acuerdo al estado en buenos, malos y regulares; siendo los malos aquellos que ya no se usan por cualquier motivo. El control del mobiliario toma en cuenta la siguiente información: cantidad, descripción y estado del elemento.

1.2 Justificación

El laboratorio de la materia de física tiene una importancia fundamental dentro del currículo académico de las diferentes carreras ya que apoya y fortalece de manera teórico práctico los conocimientos del estudiante.

Para resolver la problemática ya descrita es necesario diseñar y construir un sistema orientado a la web que gestione las actividades internas del laboratorio de la materia de física de la Universidad Politécnica Salesiana, sede Quito campus Sur; constará de dos módulos que engloben las actividades tanto de la gestión académica como de la

gestión administrativa. Este sistema funcionará de forma independiente a las plataformas virtuales con que cuenta la Universidad Politécnica Salesiana.
Al tener un sistema que gestione todas estas actividades, el proceso se vuelve más eficiente y rápido, con esto se logra que el docente distribuya el tiempo de mejor manera y los estudiantes puedan desarrollarse de manera integral.

1.3 Objetivos

1.3.1 Objetivo general

Diseñar, construir e implementar un sistema informático orientado a la web para la automatización de las actividades internas del Laboratorio de Física de la Universidad Politécnica Salesiana sede Quito campus Sur.

1.3.2 Objetivos específicos

- Diseñar una aplicación web para la gestión administrativa y académica del Laboratorio de física de la Universidad Politécnica Salesiana sede Quito campus Sur.

- Construir los módulos de gestión académica y administrativa según los requerimientos establecidos por el coordinador del Laboratorio de física de la Universidad Politécnica Salesiana sede Quito campus Sur.

- Elaborar un video tutorial de una práctica de laboratorio segmentada en tres partes: introducción, desarrollo, resultados.

- Aplicar las pruebas suficientes, previa la implementación del sistema terminado, que garanticen su correcto funcionamiento.

1.4 Alcance

El desarrollo de este proyecto está limitado a las actividades que se llevan a cabo dentro del Laboratorio de Física de la Universidad Politécnica Salesiana sede Quito Campus sur. Como se indica previamente el sistema cuenta con dos módulos que abarcan todos los procesos principales del laboratorio: gestión académica y gestión administrativo.

- **Módulo de gestión académica**: está enfocado en desarrollo de las actividades del estudiante antes, durante y después de la elaboración de las respectivas prácticas de laboratorio; dentro de este módulo se incluyen los siguientes componentes:

a. Asignación de estudiantes a grupos de trabajo.

b. Consulta a repositorio de documentos y material de apoyo; propios del laboratorio de física, relacionados al trabajo de los estudiantes.

c. Evaluación, previa y posterior al desarrollo de las actividades de las prácticas de laboratorio, siguiendo los modelos de preguntas propuestos por el CEAACES; entidad reguladora de la educación superior en el Ecuador, que plantea el uso de evaluaciones en base a reactivos.

d. Informes de laboratorio en línea, que permitan al estudiante desarrollar su trabajo en la web, mediante la carga de archivo.

- **Módulo de gestión administrativa:** está dirigido a los docentes y ayudantes del laboratorio de física, con el fin de facilitar su trabajo y optimizar tiempo y recursos, brindándoles la oportunidad de enfocarse más en el desempeño académico de sus estudiantes, éste módulo se compone de los siguientes elementos:

e. Administración de perfiles y usuarios.

f. Asignación de tareas a los estudiantes.

g. Envío de notificaciones a los estudiantes vía correo electrónico.

h. Gestión de inventario de equipos y enseres, el sistema permitirá ingresar información de los elementos del laboratorio para llevar un mejor control de los mismos además de un registro de los préstamos de equipos y materiales de laboratorio.

Adjunto al sistema, en la sección de material de apoyo, se incluirá un video de ejemplo con el proceso de una práctica de laboratorio, dividido en tres partes: introducción, desarrollo y resultados. Este video servirá como material de referencia

para futuros trabajos realizados por los estudiantes y los docentes tendrán la capacidad de cargar en el sistema los nuevos videos desarrollados.

Para garantizar el correcto funcionamiento del sistema se realizarán las respectivas pruebas de aceptación y pruebas de rendimiento.

- **Pruebas de aceptación**

Con las pruebas de aceptación se verifica si el producto cumple con las especificaciones funcionales y técnicas del cliente. Se la realiza en un entorno de características similares al de producción, en conjunto con el usuario final.

Permiten asegurar que el producto de software cumpla con los requerimientos planteados. Una vez finalizadas las pruebas de validación el cliente entregará un documento de conformidad con el producto terminado.

- **Pruebas de rendimiento**

Aseguran que no existan procesos en el código de la aplicación que provoquen fallos debido a posibles demoras o consumo excesivo de memoria por parte de la aplicación. De esta forma se evita fallos a futuro debido a errores en la aplicación causados por procesos lentos, funciones redundantes complejas.

CAPÍTULO 2
MARCO TEÓRICO

Este capítulo incluye una descripción de varios grupos de herramientas, lenguajes y metodologías disponibles en el mercado para el desarrollo de software; tomando en cuenta que el objetivo del presente proyecto es la construcción de un sistema orientado a la web, los grupos que se analizará son:

- Metodología de desarrollo de software.
- Lenguaje de programación.
- Servidor de aplicaciones.
- Sistema Gestor de Base de Datos.

Se ha incluido al final de cada grupo mencionado, un cuadro comparativo de sus respectivas herramientas con el propósito de tener una visión más clara de la elección para el desarrollo del presente trabajo.

2.1 Metodologías para el desarrollo de software

Una metodología de desarrollo de software es un conjunto de herramientas, prácticas y procesos usados para estructurar, planear y controlar la producción de sistemas de información (Pressman, 2010).

Seguir una metodología permite a los desarrolladores organizar el trabajo de tal manera que la obtención del producto final sea de alta calidad y satisfaga las necesidades del cliente. Existen dos modelos, comparados en la Tabla 1, bajo los que se agrupan las diferentes metodologías: el tradicional y el ágil.

- El enfoque tradicional se direcciona a través de documentación, las metodologías de este grupo son ideales para la elaboración de proyectos grandes ya que tienen una gran cantidad de roles y se concentran en la arquitectura de software mediante el uso de modelos. Una vez que se ha establecido el punto de llegada del proyecto, es modelo metodológico tradicional presente.

- Las metodologías ágiles, por el contrario, se enfocan en el desarrollo. Uno de los preceptos fundamentales de las metodologías ágiles es conseguir un software que funcione en lugar de obtener una documentación extensa sobre el proyecto (Agile Alliance, 2015).

Tabla 1. Metodologías ágiles vs metodologías tradicionales de desarrollo de software

Metodologías ágiles	Metodologías tradicionales
Se basan en procesos provenientes de prácticas de producción de código	Basadas en Normas provenientes de estándares seguidos por el entorno de desarrollo
Orientadas hacia las necesidades del cliente	Orientadas hacia el proceso del software
Preparadas para responder a cambios durante el proyecto	Presentan resistencia a los cambios
Proceso menos controlado, con pocos principios	Proceso mucho más controlado, con numerosas políticas/normas
El cliente es parte del equipo de desarrollo	El cliente interactúa con el equipo de desarrollo mediante reuniones
Pocos roles	Más roles
Menos énfasis en la arquitectura del software	La arquitectura de software es esencial
Genera poca documentación	Produce una gran cantidad de documentación
No existe un contrato previo y de existir, es flexible ante los cambios	Existe un contrato previo

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

A continuación se describe de manera breve tres de las metodologías de desarrollo de software más utilizadas en el medio académico, con el fin de ampliar lo que se ha expuesto sobre las metodologías ágiles y tradicionales.

2.1.1 Metodología RUP

El Proceso Unificado de Rational, RUP por sus siglas en inglés, es una metodología de desarrollo de software de tipo iterativo e incremental, guiada por casos de uso y centrada en la arquitectura (Kruchten, 2004).

Ideado por la empresa Rational; actualmente propiedad de IBM, la metodología RUP es un proceso de ingeniería para guiar a las organizaciones de desarrollo de software brindándoles un entorno de trabajo bien definido y documentado, que trabaja en dos ejes: horizontal y vertical (IBM, 2005).

El eje horizontal es el aspecto dinámico del proyecto y está representado por las fases, iteraciones y entregables del proyecto. El eje vertical describe la parte estática

del proceso de desarrollo y se define mediante varios componentes como las actividades, roles e instrumentos del proyecto. Bajo estos ejes RUP divide al proyecto en cuatro fases dentro de cada una de las cuales se ejecutan una o varias iteraciones, según la dimensión del proyecto. El portal web de IBM contiene una sección con la documentación técnica de la metodología RUP.

2.1.2 Metodología SCRUM

SCRUM es un proceso ágil y liviano que sirve para administrar y controlar el desarrollo de software. El desarrollo se realiza en forma iterativa e incremental (una iteración es un ciclo corto de construcción repetitivo). "Cada ciclo o iteración termina con una pieza de software ejecutable que incorpora nueva funcionalidad" (Schwaber & Sutherland, 2015)

SCRUM se enfoca en el trabajo en función del valor que tenga para el negocio, maximizando la utilidad de lo que se construye y el retorno de inversión.

Está diseñado para adaptarse a los cambios en los requerimientos, por ejemplo en un mercado de alta competitividad. Los requerimientos y las prioridades se revisan y ajustan durante el proyecto en intervalos muy cortos y regulares. De esta forma se puede adaptar en tiempo real el producto que se está construyendo a las necesidades del cliente. Se busca entregar software que realmente resuelva las necesidades, aumentando la satisfacción del cliente (Figueroa, Solís, & Cabrera, 2013).

La gestión de un proyecto SCRUM define cuáles son las características que debe tener el producto a construir (qué construir, qué no y en qué orden) y permite remover cualquier obstáculo que pudiera entorpecer la tarea del equipo de desarrollo. Se busca que los equipos sean lo más efectivos y productivos posible. "SCRUM tiene un conjunto de reglas muy pequeño y muy simple y está basado en los principios de inspección continua, adaptación, auto-gestión e innovación." (Canos, Letelier, & Penadés, 2003).

2.1.3 Metodología XP

Esta metodología tiene como base la simplicidad, para dar solución a un problema se lo hace de la manera más simple. El éxito de la metodología es que se entrega el producto de software que se necesita en el momento en que se necesita en lugar de entregar todo lo que posiblemente se desea planificando a futuro (Wells, 2013). Al

estar dentro del grupo de metodologías ágiles de programación; uno de los objetivos principales de XP es la rápida adaptabilidad del producto a los posibles cambios lo que se traduce en la satisfacción del cliente. Con esta metodología, el grupo de desarrolladores debe tomar en cuenta que durante el desarrollo puede aparecer la necesidad de crear, eliminar o modificar funcionalidades a su producto.

Para lograr la satisfacción del cliente, el equipo de desarrollo bajo XP debe asegurar que el trabajo se lleve a cabo de una manera sencilla y entendible. Sin importar la funcionalidad que se requiera en el sistema el diseño debe ser simple y de fácil uso, el código debe ser entendible, programando sólo lo necesario y lo que se utilizará. Se debe entregar al cliente lo que necesita (Universidad ORT Uruguay, 2003).

Tabla 2. Comparación entre RUP, SCRUM y XP como metodologías de desarrollo de software

	RUP	**SCRUM**	**XP**
Tipo de metodología	Metodología tradicional, basada en casos de uso	Metodología de desarrollo ágil basada en desarrollo incremental del producto	Metodología ágil basada en la retroalimentación continúa entre cliente y equipo de desarrollo
Tiempo aproximado entre iteraciones	No aplica. Depende de la dimensión del proyecto	2 – 4 Semanas	1 – 3 Semanas
Organización del trabajo	Organiza equipos especializados en cada etapa del desarrollo	Cada miembro del equipo trabaja de forma independiente a los demás miembros	Los desarrolladores trabajan en pareja sobre el mismo código
Cambios a funcionalidades entregadas	No, una vez aceptado el producto no se aceptan cambios	No, una vez que el cliente ha expresado su conformidad con la porción de trabajo realizado, esa parte se "guarda" y no es susceptible de cambios	Si, cualquier tarea terminada puede ser objeto de modificaciones en cualquier etapa del proyecto. Incluso luego que funcione de manera correcta
Cambios durante iteraciones	Es susceptible de cambios, sin embargo, presenta resistencia mientras	No se aceptan cambios durante la iteración. Cualquier cambio debe esperar	Se aceptan cambios en cualquier momento de las iteraciones

	más avanzada sea la fase actual de desarrollo	hasta el final de la iteración actual para ser añadido a la lista de prioridades	
Orden de prioridad de los entregables	Entregables fijos, una vez definidos los entregables, no se modifica su orden	Entregables variables, el equipo de desarrollo sigue el orden de prioridad definido, pero durante el desarrollo este orden puede cambiar	Entregables fijos, el equipo de desarrollo debe seguir el orden de prioridades definido por el cliente

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tomando en cuenta las características de las diferentes metodologías presentadas en la Tabla 2, este proyecto será desarrollado utilizando SCRUM, ya que si bien XP permite la obtención de resultados en un tiempo más corto, es susceptible a cambios incluso en los elementos que ya han sido aceptados. En cuanto a RUP, requiere un equipo de desarrollo más grande para llevar a cabo las actividades a lo largo de las fases del proyecto. Otro punto a favor de SCRUM es la administración mediante una lista de objetivos priorizada que puede reajustarse al finalizar cada iteración, esto brinda al equipo de desarrollo un grado de flexibilidad en las tareas que realizará.

2.2 Lenguajes de programación

2.2.1 Java

Es un lenguaje de programación multiplataforma y orientado a objetos, publicado originalmente por Sun Microsystems en el año de 1995. Entre sus características principales, Java gestiona de manera automática el uso de memoria, es compatible con el modelo cliente-servidor, maneja multi-hilos, tiene mecanismos de seguridad y documentación integrados. Cuenta con el respaldo de una de las compañías de software más grandes del mundo, Oracle Corporation, que en 2010 adquirió a Sun Microsystems y desde entonces continúa con el desarrollo de las nuevas versiones de Java, actualmente en su versión 8 (Oracle Corporation, 2015).

Java es de tipo compilado lo que significa que su código fuente se escribe en un lenguaje de alto nivel que se traduce a un archivo ejecutable entendible para la

máquina. Esto reduce en gran medida el tiempo de ejecución ya que no es necesario repetir el proceso de compilación entre ejecución y ejecución, al mismo tiempo que lo transforma en un lenguaje multiplataforma ya que se ejecuta sobre su propia máquina virtual JVM (Java Virtual Machine).

2.2.2 PHP

Es un lenguaje de programación de código abierto y orientado a la web para la generación dinámica de páginas, mediante el uso de scripts que se ejecutan sobre un servidor. Las siglas PHP provienen del acrónimo inglés que se traduce como Procesador de Hipertexto PHP (The PHP Documentation Group, 2015).

PHP es un tipo de lenguaje interpretado que permite un tipado dinámico de datos, es decir que al momento de utilizar variables no es necesario especificar el tipo de dato, este es definido en tiempo de ejecución dependiendo del valor que tome en ese momento.

Al ser ejecutado en el servidor se transforma inmediatamente en un lenguaje multiplataforma, ya que las páginas que genera pueden visualizadas en cualquier navegador y sobre cualquier sistema operativo.

2.2.3 Python

Es un lenguaje de programación interpretado o de script, multiplataforma y orientado a objetos. Posee un tipado dinámico y fuertemente tipado; esto quiere decir que el manejo de los tipos de dato puede en la programación puede ser manejado de manera flexible (Python Software Foundation, 2015).

Python comparte algunas de las características de los lenguajes compilados ya que la primera vez que se ejecuta genera un archivo llamado bytecode, que contiene un pseudo código de máquina, de esta manera las siguientes llamadas al programa serán ejecutadas desde el archivo.

Una característica particular de Python es que maneja un lenguaje a muy alto nivel cercano al lenguaje natural, esto lo hace ideal para personas que se están iniciando en la programación.

Tabla 3. Comparación de lenguajes de programación

	JAVA	**PHP**	**PYTHON**
Multiplataforma	Sí	Sí	Sí
Orientado a objetos	Sí	Sí	Sí
Nivel de abstracción	Alto nivel	Alto nivel	Alto nivel
Máquina Virtual	JVM	HHVM	PVM
Herencia	Sí	Sí	Sí
Polimorfismo	Sí	Sí	Sí
Orientado exclusivamente al desarrollo de aplicaciones web	No	Sí	No
Capacidad de expansión de funcionalidad con librerías propias	Sí	Sí	Sí
Seguridad integrada	Sí Clases propias de java SecurityManager y compatibilidad con la mayoría de herramientas de seguridad	Sí Mediante configuración propia.	Sí Mediante configuración propia. Existen versiones de python con reglas de seguridad más rigidas (OWASP)
Gestión de uso de memoria	Sí Garbage collector	Sí Funciones de gestión de memoria	Sí Manejo de memoria dinámico
Tipo de licencia	GNU GPL v2	Open Source PHP License v3.01	Open Source

Tipado	Fuertemente tipado	Débilmente tipado	Dinámicamente tipado Fuertemente tipado
Interpretado o compilado	Compilado	Interpretado del lado del servidor	Interpretado

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

La Tabla 3, permite apreciar claramente las características de los tres lenguajes de programación. Se ha seleccionado Java para la construcción presente trabajo ya que el equipo de desarrollo está totalmente familiarizado con el lenguaje, lo que representa un menor tiempo invertido en capacitación y reduce la probabilidad de cometer errores de programación. Java ofrece las ventajas de un lenguaje orientado a objetos, multiplataforma y multihilos, que permite el desarrollo de un amplio tipo de aplicaciones de escritorio, web, cliente-servidor, distribuidas, orientadas a servicios, etc.

2.3 Framework de JSF

En cuanto al desarrollo de aplicaciones web, Java cuenta con JSF una tecnología y framework diseñado como un estándar (JSR 344) para la construcción de interfaces de usuario del lado del servidor (Oracle Corporation, 2015).

JSF posee una gran compatibilidad con variedad de librerías que facilitan el desarrollo de aplicaciones web. Estos conjuntos de librerías brindan a JSF cierto número de funcionalidades adicionales, de esta manera el desarrollador ocupa menos tiempo en la programación de los elementos visuales de su aplicación y puede concentrar sus esfuerzos en hacer que el producto final funcione correctamente.

Dentro de los frameworks más utilizados junto a JSF, descritos en la Tabla 4, se pueden citar tres: IceFaces, PrimeFaces y RichFaces.

Tabla 4. Frameworks compatibles con JSF

	IceFaces	**PrimeFaces**	**RichFaces**
Componentes disponibles	70 componentes	117 componentes	39 componentes
Dispone un tutorial de inicio	Si, existe una guía del distribuidor	Si, existe un manual de todos sus componentes, en línea y con varios ejemplos	No tiene una guía de inicio en su documentación
Documentación	Amplia. Requiere registro en la página del proveedor para tener acceso	Amplia. No requiere autenticación. Ejemplos claros y prácticos de cada uno de sus componentes y como integrarlo en una aplicación web	Amplia. Guía de usuario proporcionada por el proveedor
Análisis de tendencia	Según Google Trends, la tendencia sobre Primefaces Figura 4 ha ido en aumento desde su aparición mientras que la de los otros frameworks ha ido en descenso a raíz de la aparición de Primefaces		

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Análisis de interés sobre PrimeFaces

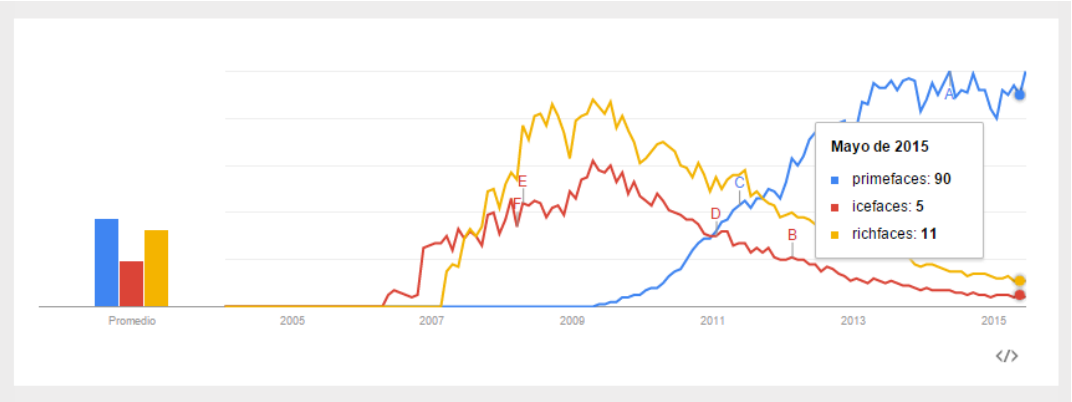


Figura 1. Análisis de interés sobre PrimeFaces. Fuente: Google Trends.

PrimeFaces es el framework más utilizado en el desarrollo de aplicaciones junto a JSF, la Figura muestra claramente la tendencia en el mercado respecto al uso de esta herramienta. Tiene a su disposición una gran variedad de componentes, documentación y ejemplos en línea, es robusto, liviano y fácil de implementar en

cualquier tipo de aplicación JSF, lo que lo hace ideal para utilizarla en el desarrollo de este proyecto.

2.4 Sistemas gestores de bases de datos

Un sistema de gestión de base de datos (SGBD) es un conjunto de programas que permite a los usuarios crear y mantener una base de datos. Por lo tanto, un SGBD es un software de propósito general que facilita el proceso de definir, construir y manipular bases de datos para diversas aplicaciones. La Tabla 5 describe las características de cada uno de los posibles sistemas de bases de datos seleccionados para este proyecto: Microsoft SQL Server, MySQL y PostgreSQL.

2.4.1 Microsoft SQL Server

Es un potente sistema de administración y análisis de bases de datos relacionales, distribuido por Microsoft. Se caracteriza por su facilidad de uso y cuenta con una gran variedad de herramientas para análisis de datos, gestión del servidor, almacenamiento en la nube y procesamiento de transacciones en línea (Microsoft, 2015).

Microsoft SQL Server permite la integración de servicios de cloud computing para la administración de bases de datos, gracias a nuevas características de como lo es su compatibilidad con el servicio Microsoft Azure. Con esta característica, las empresas tienen una nueva manera de alojar sus datos sin la necesidad de mantener una infraestructura especializada para esto a un costo relativamente menor.

2.4.2 MySQL

Es uno de los sistemas más importantes en cuanto a diseño y programación de bases de datos relacionales, multihilo y multiusuario. MySQL se caracteriza por la velocidad en el acceso a los datos, lo que se traduce en un tiempo de ejecución de consultas relativamente corto (Oracle Corporation, 2015).

En la actualidad es desarrollado por Oracle Corporation con un esquema de licencia dual; la licencia GNU GPL para todos los proyectos compatibles con este tipo de licencia como educativos o de software libre y una versión Enterprise licenciada privadamente orientada hacia implementaciones privativas.

En cuanto a seguridad, el esquema de MySQL maneja cuatro aspectos principales: autenticación integrada en el motor de base de datos, encriptación de tipo SSL/TSL entre cliente y servidor, autorización mediante control de privilegios a los usuarios y un firewall que genera automáticamente las reglas de acceso para cada aplicación.

2.4.3 PostgreSQL

PostgreSQL es un sistema gestor de base de datos de código abierto, distribuido bajo licencia BSD, lo que ha permitido mejoras más rápidas en relación a otros sistemas de bases de datos, es de los más potentes del mercado y pone a disposición de los usuarios, sin ningún costo adicional, funcionalidades similares a las de bases de datos comerciales (PostgreSQL, 2010). Su versión inicial bajo este nombre se liberó en al año 1996 y actualmente continua su desarrollo gracias a The PostgreSQL Global Development Group.

Utiliza un modelo cliente/servidor que se maneja mediante multiprocesos para garantizar la estabilidad del sistema ya que un posible fallo en uno de los procesos no afectará al resto del sistema.

Mediante un sistema denominado MVCC (Acceso concurrente multi-versión, por sus siglas en inglés) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos.

Tabla 5. Comparación entre sistemas gestores de bases de datos

	Microsoft SQL Server	**MySQL**	**PostgreSQL**
Desarrollador	Microsoft Corporation	Sun MycroSystems	The PostgreSQL Global Development Group
Tipo de licencia	Privada	Libre a nivel de usuario, para empresas tiene costo	BSD, de libre distribución
Multiplataforma	No	Sí	Sí
Versión actual	Sql Server 2014, Abril 2014	5.6.25, Mayo 2015	9.4.4, Junio 2015
Maneja esquemas de datos	Sí	Sí	Sí
Tipado de datos	Sí	Sí	Sí
Soporta XML	No especificado	Sí	No especificado

Métodos de acceso	OLE DB TDS ADO.NET JDBC ODBC	ADO.NET JDBC ODBC	Librería nativa en C ADO.NET JDBC ODBC
Métodos de Replicación	Si, dependiente de la versión del servidor	Maestro-Maestro y Maestro-Esclavo	Maestro-Esclavo
Concurrencia	Sí	Sí	Sí
Copias de seguridad en caliente	Sí	Sí	Sí
Métodos de autenticación	Autenticación de SQL Server (seguridad estándar) Autenticación de Windows (seguridad integrada)	Contraseña segura, Clear text, Nativa de Windows y SHA256	Autenticación LDAP, Kerberos, Confiable, GSSAPI, MD5 password, SSPI
Soporte de transacciones	Sí	Sí	Sí
Procedimientos alma cenados	Sí	Sí	Sí
Entorno gráfico de administración	SQL Server Management Studio	MySQL Workbench	pgAdmin
Encriptación	MD5 SHA SHA1	SSL/TLC llaves publicas Llaves privadas Firmas digitales	MD5 SSL

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Una de las características más importantes mencionadas en la Tabla 5, por la que se eligió PostgreSQL como gestor de base de datos en este proyecto es que está

diseñado para ambientes de alto volumen de información lo que mejora su rendimiento en comparación con otras herramientas. Utilizando la característica ya mencionada, MVCC, se consigue una mejor respuesta en grandes volúmenes de información, al mismo tiempo que permite que los accesos de lectura a la base de datos continúen recolectando datos consistentes durante la actualización de registros por parte de otros accesos con privilegios de lectura-escritura.

2.5 Servidor de aplicaciones

Es un dispositivo sobre el que se ejecuta una aplicación que acepta peticiones de múltiples clientes y se encarga de gestionar la mayor parte de la lógica de negocio de la misma. En cuanto al uso de tecnologías Java, los servidores de aplicaciones simplifican el desarrollo, implantación y gestión de las aplicaciones construidas bajo este lenguaje.

2.5.1 Apache TomEE

Es un conjunto de librerías que brindan a Apache Tomcat la compatibilidad necesaria para ejecutar aplicaciones Java EE. Tomcat es un contenedor de aplicaciones que implementa las especificaciones de Java Servlet Pages (The Apache Software Foundation, 2015).

En cuanto a su uso en ambientes de producción no es la primera elección para un gran número de desarrolladores que lo consideran como una herramienta mucho más orientada hacia un ambiente de pruebas donde se requiere un servidor liviano y que permita implementar aplicaciones de manera rápida.

2.5.2 Glassfish

Es un servidor de aplicaciones que implementa las tecnologías de Java EE. Glassfish fue inicialmente desarrollado por Sun Microsystems hasta que fue adquirido por Oracle.

Glassfish se caracteriza por la capacidad de dividir al servidor en dominios independientes, cada uno de los cuales puede funcionar bajo configuraciones diferentes (Oracle Corporation, 2013). Proporciona además una interfaz de configuración bastante intuitiva y amigable con el usuario.

Permite la creación y gestión de “pools de conexiones” los cuales pueden ser compartidos por varias aplicaciones trabajando sobre la misma base de datos. El proceso de implementación de las aplicaciones, denominado en Glassfish como

despliegue, es también muy sencillo, esto garantiza que el tiempo que una aplicación permanece fuera de línea al realizar cualquier cambio o actualización sea muy corto.

2.5.3 JBoss

Desarrollado por RedHat, este es un servidor de aplicaciones de código abierto basado en su totalidad en Java. Permite implementar aplicaciones empresariales y es compatible con el esquema de persistencia de Hibernate para el mapeo de bases de datos relacionales hacia modelos de objetos (Red Hat, Inc., 2007).

JBoss integra como marco de desarrollo una tecnología denominada Seam, la cual simplifica la creación de aplicaciones combinando elementos de Enterprise JavaBeans y Java Server Faces.

Tabla 6. Comparación entre Glassfish, Apache TomEE y JBoss

Característica	Glassfish	Apache TomEE	JBoss
Consolas de Administración con calidad Enterprise	Sí	No	No
Más de una instancia por nodo	Sí	No	No
Administración remota	Sí	No	No
Adaptabilidad de Firewall	Sí	No	No
Administración centralizada en despliegue de múltiples servidores	Sí	No	No
Soporte para JPA: Java Persistence API	Sí	No	Sí
cuenta con plugins para Eclipse y NetBeans 5.5	Sí	No	No
Hibernate: implementación Open Source para el manejo de persistencia	Sí	No	Sí
Arquitecturas multi-cluster.	Sí	No	No
Portabilidad: independiente del Sistema Operativo	Sí	Sí	Sí
Dominios. Distribución de aplicaciones	Sí	No	No
Compatible con Tecnologías web de Java (Servlet 3.0, JSP 2.2, JSF 2.0, JSTL 1.2, el 2.2)	Sí	Sí	Sí
JMS de calidad Enterprise	Sí	No	No

Arquitectura Microkernel basada en OSGI	Sí	No	No
Cuenta con equipos de soporte locales (Madrid) que pueden dar servicio en español en 24x7	Sí	No	No
Costo	Sin costo en su versión Open Source	Sin costo	Sin costo en su versión Open Source
Licencia	Common Development and Distribution License (CDDL) y GNU General Public License (GPL)	Open Source. Apache Licence v2.0	GNU General Public License (GPL)
Clustering	Sí	No	Sí
GlassFish incluye Grizzly (Basado en NIO de JAVA5)	Sí	No	No
Rendimiento. Benchmarks.	Sí	No	No
servidores certificados JAVA EE 5	Sí	No	No
Tiene el paquete completo JAVA SE + JAVA EE en plataformas Windows, Linux, y Solaris.	Sí	No	No
Facilidad de implementación	Sí	Sí	Sí

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

En cuanto a los servidores de aplicaciones, Glassfish es considerado como uno de los más robustos, tomando como referencia las características de la Tabla 6 por lo que se lo seleccionó como el ideal para el desarrollo de este trabajo. Permite una gestión centralizada mediante una consola de administración con un diseño bastante ordenado y funcional.

En la Figura 2 se observa la tendencia de búsquedas relacionadas, a mayo de 2015, en cuanto a los servidores de aplicaciones Glassfish y JBoss. Se puede apreciar claramente que el índice de búsqueda de JBoss deciende mientras que Glassfish va en aumento. A comparación, de estas dos grandes plataformas, Apache TomEE pasa desapercibido. La Figura 3 muestra un detalle ampliado de los valores asignados a

cada uno de los términos de búsqueda y se puede apreciar de mejor manera los resultados obtenidos a favor de Glassfish.

Diagrama de tendencia de búsqueda Glassfish vs JBoss

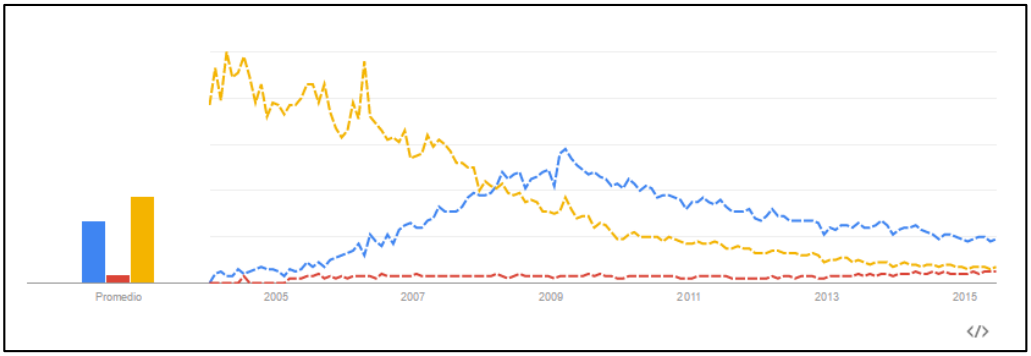


Figura 2. Tendencia de búsqueda Glassfish vs JBoss. Fuente: Google Trends

Detalle de diagrama de tendencia de búsqueda Glassfish vs JBoss

Figura 3. Detalle de tendencia Glassfish vs JBoss. Fuente: Google Trends.

A continuación se realiza una descripción, de manera más precisa, de las cualidades de cada herramienta de desarrollo con las que se trabajará para la construcción del producto final de este proyecto. Se ha dado una mayor importancia a la sección correspondiente al lenguaje de programación debido a que sus características son más relevantes para este trabajo que las de los demás componentes de desarrollo.

2.6 Scrum

Scrum es una metodología ágil para el desarrollo de software, de tipo incremental e iterativo y se basa en la capacidad de adaptarse rápidamente los cambios continuos.

Cada iteración lleva el nombre de Sprint y al finalizar añade una funcionalidad al producto, el mismo que es resultado de la visión y necesidades del cliente.

Scrum se basa en el empirismo, una teoría que afirma que el conocimiento viene de la experiencia y que la toma de decisiones se la hace en base a lo que se conoce. El enfoque iterativo de Scrum optimiza la predicción y prevención de riesgos. Scrum se fundamenta en tres pilares (Schwaber & Sutherland, 2015):

- **Transparencia:** los aspectos significativos del proyecto deben ser visibles a los responsables de su resultado. Esta transparencia requiere que sea definida en un estándar común con el propósito que cualquier observador pueda entender de qué se trata el proyecto.
- **Inspección**: los usuarios de Scrum deben inspeccionar constantemente todos los instrumentos y el progreso en pos de obtener el objetivo de la iteración actual.
- **Adaptación:** si un inspector determina que uno o varios aspectos del proyecto se desvían del objetivo y salen de los límites aceptables, debe llevarse a cabo un ajuste para evitar estas desviaciones.

Scrum tiene tres elementos fundamentales, que abarcan todo el proceso de desarrollo: los instrumentos, equipos de trabajo y los eventos; la metodología dirige las relaciones y la manera cómo interactúan uno con el otro.

2.6.1 Los instrumentos de Scrum

Los instrumentos son herramientas de la metodología que representan trabajo o valor, ofrecen transparencia y oportunidades para inspección y adaptación. Los instrumentos definidos por Scrum son: backlog de producto y backlog de sprint; están diseñados para maximizar la transparencia de la información clave y que todos los involucrados entiendan el propósito del instrumento.

- **Backlog de producto:**

Es una lista ordenada de todo lo que se puede necesitar en el producto y es la única fuente válida de requerimientos en caso de que algún cambio necesite hacerse en el producto. El propietario es el responsable del contenido y ordenamiento de este instrumento.

El Backlog de producto nunca está completo, la versión inicial solo establece los primeros y mejor entendidos requerimientos. Esta lista se adapta de igual forma que lo hace el producto final. Al finalizar el trabajo, el backlog del producto contiene las posibles mejoras que se realizarán en versiones futuras de la aplicación.

- **Backlog de Sprint:**

Es un listado similar al Backlog del Producto pero solo maneja los objetivos planteados para el Sprint o iteración actual, junto con un plan de las actividades necesarias para cumplir con cada uno de los ítems en el listado, resultando en un incremento del producto.

Este backlog también es cambiante a lo largo del sprint. El equipo de trabajo enriquece al backlog de sprint durante su trabajo ya que, mientras se van completando las tareas, se identifican nuevas características del proyecto y se entiende de mejor manera al producto terminado.

2.6.2 El equipo de SCRUM

Esta metodología maneja tres roles en el equipo de trabajo según la función que cumplen. Los equipos son auto-organizados y multifuncionales. Los equipos auto-organizados deciden la mejor manera para cumplir con su trabajo sin la necesidad de ser dirigidos por alguien ajeno al equipo. Al ser multifuncionales, los miembros del equipo tienen las competencias necesarias para llegar a su objetivo sin depender de terceros.

Los roles del equipo Scrum pueden resumirse en tres palabras: propiedad, control y desarrollo.

- **Propietario del Producto:** representa al cliente, a las personas que han solicitado el producto y por lo tanto a quien se entregará el resultado del proyecto. Es quien marca los requisitos y gestiona el backlog de producto.

  Para que este miembro del equipo tenga éxito, debe ser una persona con capacidad de toma de decisiones y que goce del respaldo de toda la organización. Su trabajo es visible en el contenido y orden del Backlog de producto. Puede delegar sus responsabilidades al equipo de desarrollo sin embargo, será responsable por el producto terminado.

- **Scrum Master:** es la persona responsable de que el proceso de Scrum se ejecute correctamente. Se asegura de que se respeten las reglas de la metodología y aísla al equipo de cualquier influencia externa.

  El Scrum Master trabaja como un mediador entre el equipo y el propietario. Debe asegurarse que los requisitos definidos por el propietario sean claros y concisos, mientras que debe guiar el enfoque del equipo de trabajo para que pueda crear productos con alto valor para el cliente.

- **El equipo de desarrollo:** es el grupo de personas que tiene la responsabilidad del desarrollo del producto, al final de cada Sprint deben entregar una característica o función nueva al producto terminado.

  Se auto-organiza, por esa razón no existe ningún rol de líder y se maneja bajo un esquema multidisciplinario ya que todos los miembros deben ser capaces de realizar las tareas de análisis, diseño, desarrollo, pruebas. Por esta misma cualidad, no existen subgrupos de desarrollo orientados hacia tareas específicas.

  Las competencias del equipo de desarrollo se miran como un todo, resultado de la suma de las competencias individuales de cada miembro del equipo.

2.6.3 Eventos de Scrum

Los eventos en Scrum permiten la creación de un ciclo regular que minimiza la necesidad de reuniones. Todos los eventos tienen un parámetro de tiempo que define el máximo que tomará la culminación de dicho evento; una vez iniciado no se puede acortar o alargar el tiempo planificado.

En total, Scrum considera cuatro eventos: sprint, planeación del sprint, scrum diario, revisión del sprint y retrospectiva del sprint. Cada uno se describe a detalle en la siguiente sección.

2.6.4 El proceso de Scrum

Para conseguir la satisfacción del cliente con el producto terminado, esta metodología ejecuta en varias repeticiones del proceso de desarrollo. El evento central de la metodología es el Sprint o iteración, usualmente dura como máximo un mes. La culminación del Sprint define la entrega de un incremento al producto terminado y mientras esto no suceda:

- No se aceptan cambios que puedan cambiar el Objetivo del Sprint.
- Las metas de calidad no se reducen.

El alcance del sprint puede ser clarificado o renegociado entre el Equipo de desarrollo y el Propietario en función de lo que se ha aprendido durante el Sprint.

Scrum inicia con una reunión entre las partes involucradas, donde se plantea una visión general del producto final y se genera una lista priorizada de objetivos donde constan todos los componentes del sistema a desarrollarse, esta lista toma el nombre de backlog del producto.

En la Figura 4, se muestra un esquema del proceso completo de Scrum cuya entrada son los requisitos del cliente reunidos en el backlog.

Diagrama del ciclo de vida de SCRUM

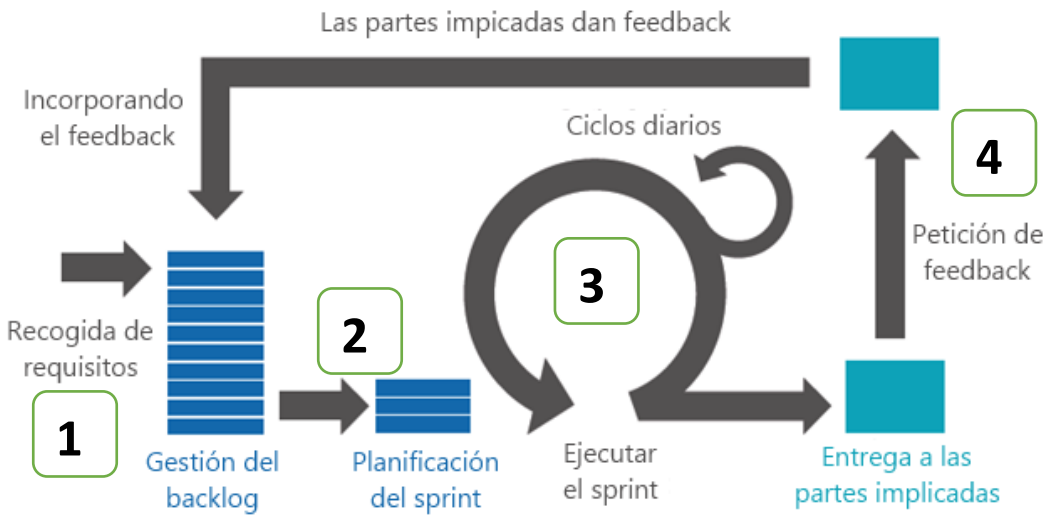


Figura 4. Ciclo de vida de SCRUM. Fuente: The Agile Alliance.

Siguiendo este diagrama, se puede dividir a todo el proceso de SCRUM en cuatro elementos principales:

(1) **Gestión del backlog:** las prioridades de cada ítem son definidas por el propietario del producto quien, al final de cada sprint, puede modificar dichas prioridades según su propio análisis o intereses. El equipo de trabajo contribuye a la reorganización de los objetivos realizando sugerencias al propietario antes de dar inicio al siguiente sprint.

(2) **Planificación del sprint:** se seleccionan los objetivos a cumplirse en el sprint actual mediante la colaboración del equipo completo: propietario, desarrolladores y Scrum master. El equipo de desarrolladores presenta al cliente las posibles dudas sobre los objetivos planteados, mientras que el Scrum master se asegura que el objetivo general de la reunión esté claro y no tome más del tiempo estrictamente necesario.

El resultado de la planificación debe responder estas preguntas:

- *¿Qué será entregado como resultado del siguiente sprint?*
- *¿Cómo se organizará el trabajo necesario para entregar el incremento?*

En base a los objetivos seleccionados se asignan, dentro del equipo, las tareas necesarias para lograr su culminación.

La "meta del Sprint" es un objetivo planteado por el equipo de desarrollo, en base al backlog, que provee una guía del por qué se está construyendo el incremento actual. Este objetivo puede ser la funcionalidad de los ítems del backlog seleccionados para el sprint actual.

Si durante la planeación del sprint el equipo de desarrollo detecta que el trabajo a realizarse es diferente del esperado, colaboran con el propietario del producto para negociar y redefinir el alcance del backlog.

(3) **Ejecución del sprint:** la ejecución del sprint incluye varias repeticiones diarias (Scrum diario) del mismo proceso, que inicia con una reunión entre los desarrolladores para revisar el avance general del sprint, analizar los posibles obstáculos en el proceso actual y realizar las correcciones respectivas que lleven al equipo al cumplimiento de los objetivos.

Durante el Scrum diario se responden tres preguntas:

- *¿Qué se hizo el día anterior para cumplir con los objetivos del sprint?*
- *¿Qué se hará hoy para ayudar al equipo a cumplir los objetivos del sprint?*
- *¿Se pude ver algún impedimento para que se cumpla el objetivo del sprint?*

Entre las ventajas del Scrum diario se puede citar que: mejora la comunicación del equipo, elimina la necesidad de otras reuniones, identifica y elimina los posibles obstáculos para el desarrollo, promueve la toma rápida de decisiones y mejora el nivel de conocimiento del equipo de desarrollo.

(4) **Entrega de resultados y retroalimentación:** una vez finalizado el sprint, se realiza una reunión con el propietario del producto en la que el equipo de desarrollo demuestra que los objetivos planteados se han cumplido con éxito mediante la revisión del sprint. En esta reunión se hace un proceso de retroalimentación, donde el cliente puede solicitar adaptaciones al proyecto según como haya cambiado el contexto del mismo. De no haber cambio alguno, el incremento del sprint es aceptado por el cliente y el equipo continuará con el desarrollo de los siguientes objetivos en la lista.

2.7 Java

Java es un lenguaje de programación simple. Orientado a objetos, basado en clases, distribuido, multiplataforma, interpretado, sólido, seguro, de arquitectura neutral, portable, de alto desempeño, de multihilos y dinámico. Gracias a Java se pueden crear todo tipo de aplicaciones, sin embargo destaca en la construcción de aplicaciones web (Gosling, Joy, Steele, Bracha, & Buckley, 2015).

Una plataforma Java es un ambiente particular en el cual las aplicaciones Java se ejecutan. Todas las plataformas se componen de los siguientes elementos:

- *Máquina virtual (JVM):* es un programa para una plataforma particular de hardware y software que ejecuta aplicaciones Java.
- *API (Application Programming Interface):* es una colección de componentes de software que se pueden usar para crear otros componentes de software y aplicaciones.

Cada plataforma provee de una máquina virtual y una API, esto permite a las aplicaciones escritas en Java ejecutarse en cualquier sistema compatible con todas las ventajas del lenguaje.

2.7.1 Evolución de Java

1992- se muestra una demostración acerca de Java mediante una consola y un muñeco electrónico.

1994- cambió el nombre de Oak a Java y la plataforma Java 1.0, estaba disponible para ser descargada en la Web

1995- Netscape anunció que incluiría soporte para Java en sus navegadores

1996- se lanza la versión JDK 1.0

1997- se lanza versión JDK 1.1 clases internas, JavaBeans.

1998- se lanza la versión J2SE 1.2 además la API gráfica Swing fue integrada en las clases básicas, la máquina virtual (JVM) de Sun fue equipada con un compilador JIT (Just in Time) por primera vez

2000- se lanza la versión J2SE 1.3, RMI fue cambiado para que se basara en CORBA se integró JavaSound2002 fue lanzada la versión J2SE 1.4, Logging API (Specified in JSR 47.), API I/O para la lectura y escritura de imágenes en formatos como JPEG o PNG

2004- se lanza la versión J2SE 5.0C; bucle for mejorado, Metadatos

2006- se lanza la versión Java SE6; Incluye un nuevo marco de trabajo y APIs que hacen posible la combinación de Java con lenguajes dinámicos como PHP, Python, Ruby y JavaScript incluye el motor Rhino, de Mozilla, una implementación de Javascript en Java.

2011- se lanza la versión Java SE7; Soporte para XML dentro del propio lenguaje,

2014- se lanza la versión Java SE8; Soporte para XML dentro del propio lenguaje.

2.7.2 Características de Java

En esta sección listaremos las características más importantes del lenguaje de programación orientado a objetos Java.

- **Orientado al objetos**

  A diferencia de otros lenguajes de programación que con el tiempo adoptaron características para dar soporte a la programación orientada a objetos (POO), Java fue ideado como un lenguaje propiamente de objetos.

  Desde el punto de vista de la programación, un objeto es un elemento creado en la memoria de un computador que posee dos elementos: atributos y métodos. Los atributos son todas las características del objeto que lo definen como tal y los métodos son todas aquellas operaciones que definen su comportamiento, qué es lo que hace y cómo lo hace (Rodríguez, Sosa y Prieto, 2004).

- **Distribuido**

  Java se ha diseñado para trabajar en ambiente de redes y contienen una gran biblioteca de clases para la utilización de protocolos como TCP/IP, SOAP, RMI, CORBA. Todos estos protocolos permiten diferentes tipos de comunicación entre las aplicaciones Java.

- **Sólido**

  En Java no es posible escribir en áreas arbitrarias de memoria ni realizar operaciones que corrompan el código.

- **Seguro**

  Al trabajar sobre ambientes de redes el tema de seguridad es muy importante. Java posee librerías y métodos integrados para evitar la manipulación de datos en sus aplicaciones.

- **Alto desempeño**

  Java al ser compilado genera porciones de código conocidas como bytecode, lo que se traduce en un mejor desempeño en tiempo de ejecución y rendimiento ya

que las aplicaciones programadas en Java deben ser compiladas una sola vez y cada acceso subsiguiente se lo realiza al código ya compilado (Oracle 2015).

- **Multihilos**

  Java puede realizar aplicaciones en las que ocurre más de una cosa a la vez, apoyándose en un sistema de gestión de eventos permite este tipo de conducta en tiempo real en sus programas.

2.7.3 Arquitectura Java

Para establecer Java como parte integral de la red, el compilador Java compila su código a un fichero objeto, de formato independiente de la arquitectura de la máquina en que se ejecutará. Cualquier máquina que tenga el sistema de ejecución (run-time) puede ejecutar ese código objeto, sin importar en modo alguno el entorno en que ha sido generado.

Diagrama sobre la arquitectura de Java

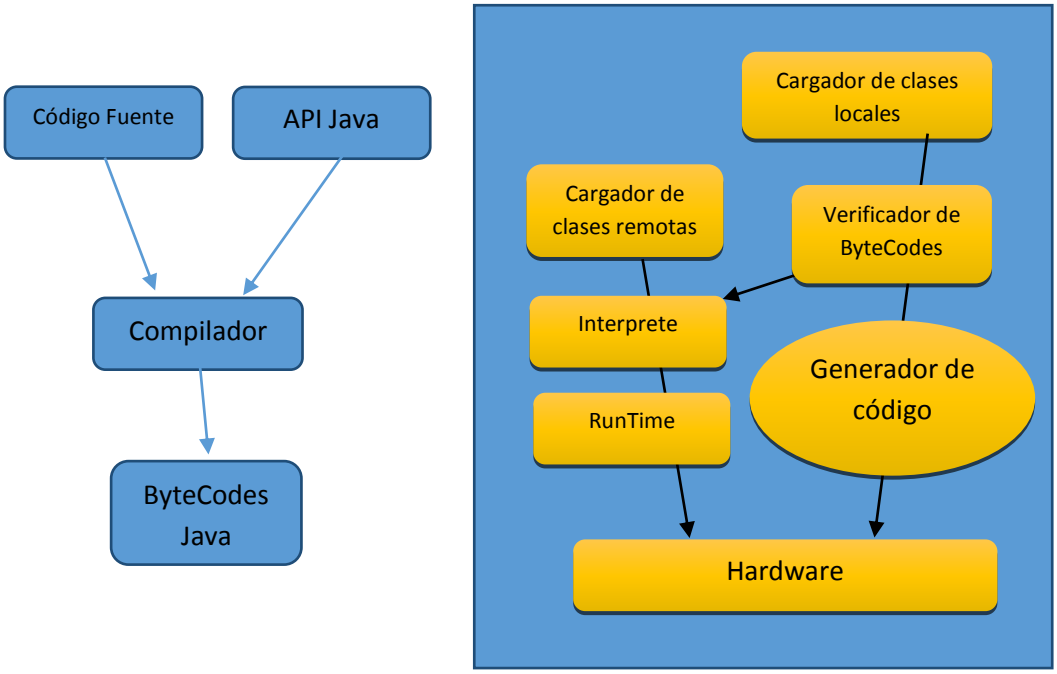


Figura 5. Arquitectura neutral de Java. Elaborado por Myrian Guijarro & Jorge Jaramillo

La arquitectura neutral de java como se muestra en la Figura 5, tiene tiempos de compilación y tiempo de ejecución. El código java y las API Java ingresan al compilador y se obtienen los ByteCodes, que es código compilado, el que se envía al cargador de clases locales. Si el proceso es exitoso pasa al verificador de ByteCodes,

y luego donde el intérprete, este proceso se ejecuta en la máquina virtual, independiente de la maquina física

El código fuente Java se compila en un código de alto nivel independiente de la máquina. Este código (ByteCode) está diseñado para ejecutarse en una máquina supuesta sobre un sistema denominado run-time, dependiente de la máquina física.

HotSpot, es un componente de java que interpreta los ByteCodes de manera más eficiente en relación al tiempo de ejecución de un programa que sea compilado, la Figura 6 nos muestra cómo actúa el sistema runtime en los diversos casos que pueden darse.

Diagrama de arquitectura de Hotspot

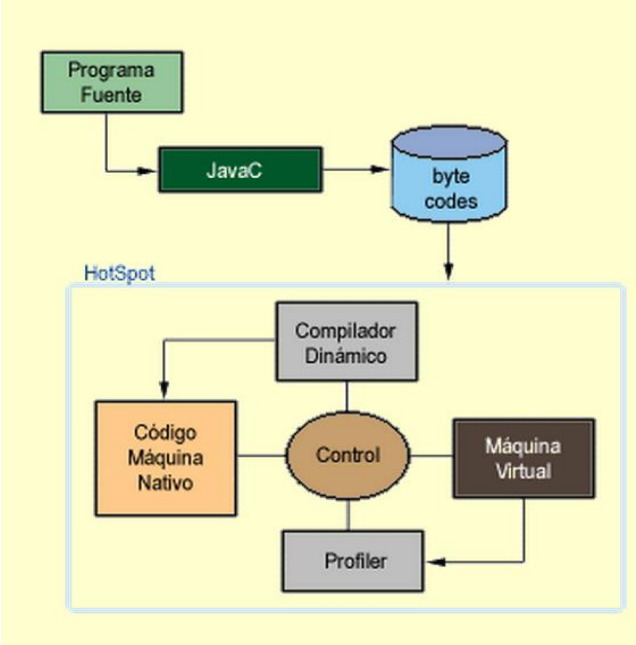


Figura 6. Arquitectura de hotspot. Fuente: Departamento de Informática y Sistemas de la Universidad de Murcia.

Modelo de carga

La carga de las clases tiene lugar dinámicamente durante la ejecución del programa, esto desequilibra la dinámica tradicional de la compilación, enlazado, carga y ejecución que se requiere para crear y ejecutar programas en otros lenguajes de programación.

Pasos para la carga de clases:

- Lectura de los bloques.

- Validación de los bloques que deben definir una clase bien estructurada conteniendo código Java bien estructurado.
- Construcción de la estructura de clase global.
- Resolución de referencias.
- Control de acceso, de forma que una aplicación o entorno decide las reglas de acceso para la carga de la clase.

La carga de clases se genera de acuerdo a la demanda. Este concepto se encuentra en librerías dinámicas, modelos de objeto (CORBA, COM y otros) así como la capacidad plug-in de algunos productos, la integración total de la carga de clases con el lenguaje y el entorno es una nueva característica de la carga de clases. La misma que desde fuentes dispares (disco local, web, dispositivos de red y otros) es un avance en la programación orientada a objetos, debido a que permite tratar a los objetos de código ejecutable con facilidad como antes se manejaban los objetos de datos. En la Figura 7 se ve un esquema del funcionamiento del modelo de carga.

Diagrama de arquitectura del modelo de carga Java

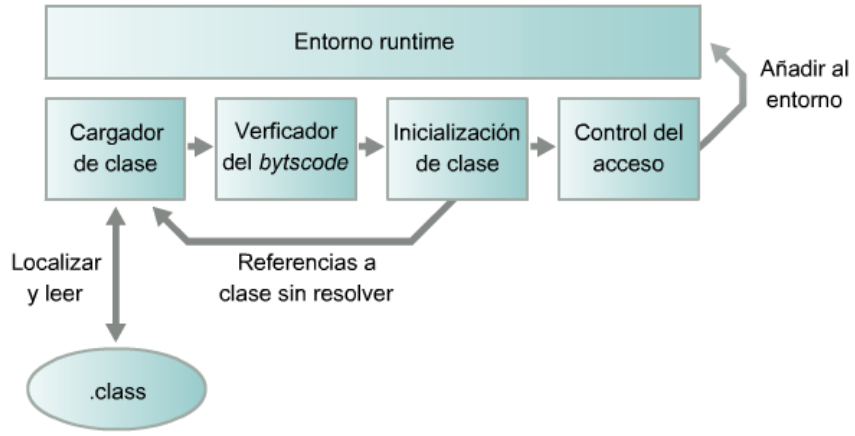


Figura 7. Arquitectura del modelo de carga. Fuente: EduKanda. Junta de Andalucía.

2.7.4 Java Enterprise Edition (JEE)

Java EE es una plataforma que provee una API para aplicaciones multicapa, escalables, confiables y distribuidas. La documentación de esta API es bastante amplia y describe el uso correcto de cada uno de sus componentes así como las dependencias y librerías adicionales disponibles, se la puede encontrar en la página web de documentación de Oracle Corporation.

Java EE permite el desarrollo de aplicaciones empresariales, se denominan de esta manera porque generalmente son diseñadas para resolver problemas de carácter empresarial. Estas aplicaciones son separadas en partes o capas de acuerdo a su funcionalidad según el modelo multicapa, El modelo más simple de este tipo de aplicaciones cuenta con tres capas, una capa de cliente, una capa media y una capa de datos (Gosling, Joy, Steele, Bracha, & Buckley, 2015).

La capa cliente consta de un programa que interactúa con el usuario. Este se encarga de hacer peticiones hacia la capa media donde están almacenadas todas las funciones o métodos de negocio. Una vez que se atienden las peticiones y se procesa la información se la almacena en la capa de datos.

2.7.5 Arquitectura de Java EE

Una de las ventajas de la arquitectura JEE es que los servidores de aplicación brindan un conjunto de servicios en forma de contenedor para cada uno de los componentes de las aplicaciones.

Los contenedores Java EE proporcionan acceso a los servicios subyacentes que poseen dentro del entorno del servidor de aplicaciones y en el entorno del cliente permitiendo que el equipo de desarrollo pueda concentrarse en resolver los problemas de negocio.

Los contenedores son una interface entre un componente y la funcionalidad de bajo nivel específica de la plataforma que soporta el componente. Como se muestra en la Figura 8, Java EE se maneja en función de tres contenedores: contenedor de aplicación, contenedor EJB y contenedor web. Todos estos componentes a su vez están construidos sobre las librerías de Java Standard Edition (Jendrock, Cervera Navarro, Evans, Haase, & Markito, 2014).

Los contenedores de Java EE son los encargados de gestionar las peticiones del cliente gracias a una serie de APIs repartidos entre ellos. En este documento se describe de manera breve la funcionalidad de los APIs nuevos, añadidos a cada contenedor en la versión 7 de Java EE, para mayor información referente a los otros APIs de Java se recomienda recurrir a la documentación oficial del lenguaje disponible en la web de Oracle.

A continuación se describe la arquitectura Java EE.

Diagrama de arquitectura de Java EE

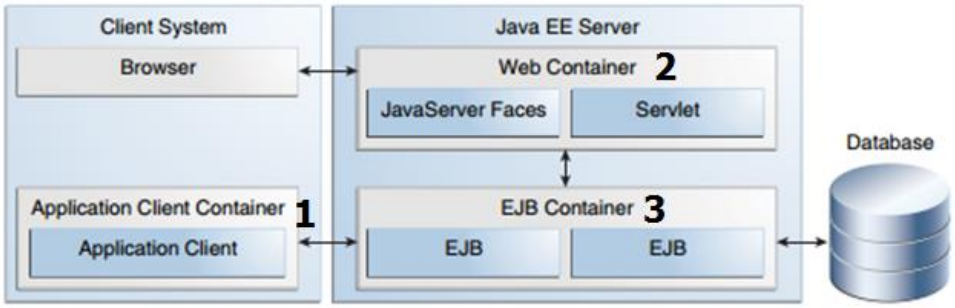


Figura 8. Arquitectura de Java EE. Fuente: Oracle Corporation

(1) Contenedor de aplicaciones cliente: controla la ejecución de los componentes de la aplicación cliente.

(2) Contenedor Web: controla la ejecución de páginas web, servlets y algunos componentes EJB en aplicaciones Java EE desplegadas en el servidor de aplicaciones.

(3) Contenedor EJB: es un programa Java que se ejecuta en el servidor y contiene todas las clases y objetos necesarios para el correcto funcionamiento de los Enterprise beans.

Enterprise JavaBeans (EJB)

EJB es una arquitectura de componentes de servidor parte del estándar de Java EE, que simplifica el proceso de construcción de aplicaciones empresariales en Java. Permiten desarrollar aplicaciones portables entre distintas plataformas y servidores de aplicaciones que cumplan la especificación estándar de Java.

Según la función que cumpla el EJB en la aplicación, pueden ser de tres tipos:

- **EJB de Entidad**: representan un objeto concreto que existe en la base de datos. Una instancia de un bean de entidad representa una fila en una tabla de la base de datos.
- **EJB de Sesión:** gestionan el flujo de la información en el servidor. Representa un proceso o una acción de negocio. Cualquier llamada a un servicio del servidor debe comenzar con una llamada a un bean de sesión. Este tipo de Bean, a su vez, se divide en dos tipos.
    - ***Bean de sesión sin estado:*** son componentes que no conservan ninguna información del cliente. Se limitan a responder las peticiones independientemente del proceso interno que posean.

- ***Bean de sesión con estado:*** son componentes que conservan la información del cliente mientras exista la instancia del Bean de sesión.

- **EJB dirigidos por mensajes:** son los únicos beans con funcionamiento asíncrono. Usando el Java Messaging System, se suscriben a un tema o a una cola y se activan al recibir un mensaje dirigido a dicho tema o cola. No necesitan objetos EJBObject porque los clientes no se comunican con ellos directamente.

2.7.6 Funcionamiento de los componentes EJB

El *cliente* que realiza peticiones y el servidor, que contiene el EJB, se ejecutan en máquinas virtuales Java distintas. El *contenedor EJB* en el servidor, al recibir una petición del *cliente*, asigna un objeto EJB al cliente. Cualquier petición del cliente se hace a través del *EJBObject*, el cual solicita al contenedor una serie de servicios y se comunica con el *EnterpriseBean* del servidor. Por último el Bean del servidor realiza las peticiones a la base de datos. Este proceso se encuentra esquematizado en la Figura 9 para un mejor entendimiento.

Diagrama de funcionamiento de EJB

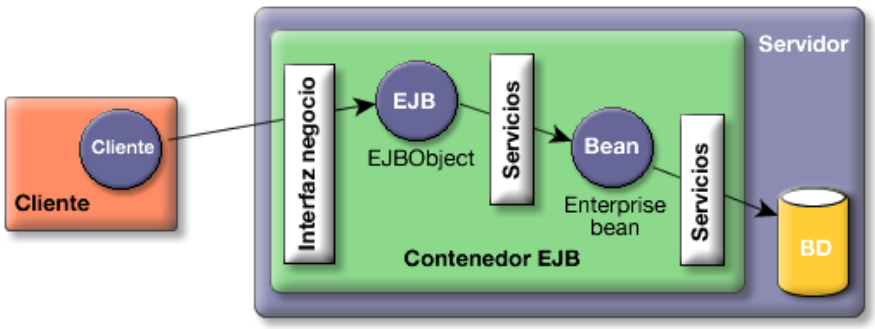


Figura 9. Funcionamiento de alto nivel de EJB. Fuente: Universidad de Alicante.

El propio contenedor EJB es quien se encarga de comprobar los parámetros de acceso, permisos, abrir y cerrar transacciones o refrescar los datos del Bean.

2.7.7 Java Server Faces - JSF

Java Server Faces (JSF) es un marco de trabajo para crear aplicaciones en el lenguaje Java basadas en el patrón modelo, vista y controlador (Jacobi & Fallows, 2006). JSF tiene como características principales:

- Utiliza páginas JSP para generar las vistas.
- Asocia a cada vista con formularios un conjunto de objetos java manejados por el controlador, estos controladores toman el nombre de managed beans, que

facilitan la toma, manipulación y visualización de los valores mostrados en los diferentes elementos de los formularios.

- Es extensible, es decir que se puede crear nuevos elementos de la interfaz o modificar los ya existentes, además, cuenta con una serie de librerías y componentes reutilizables y fáciles de usar.
- Forma parte del estándar Java EE que incluye a JSF para el diseño de la capa de presentación.

2.7.8 Evolución JSF

- 2004 -JSF 1.0- Lanzamiento inicial de las especificaciones de JSF.
- 2004 -JSF 1.1- Lanzamiento que solucionaba errores. Sin cambios en las especificaciones ni en el renderkit de HTML.
- 2006 -JSF 1.2- Lanzamiento con mejoras y corrección de errores.
- 2009 -JSF 2.0.- Lanzamiento con mejoras de funcionalidad, rendimiento y facilidad de uso.
- 2010 -JSF 2.1- Lanzamiento de mantenimiento, con mínimos cambios.
- 2013 -JSF 2.2- Lanzamiento que introduce soporte a HTML 5, Faces Flow, Stateless views y Resource library contracts.

2.7.9 Ciclo de vida de JSF

Para los autores De la Cruz Diaz, Gómez Rodriguez, Lara Jerónimo y Pech May (2008), la especificación de JSF define seis fases distintas en su ciclo de vida. El diagrama de la Figura 10 muestra este ciclo de vida dentro del marco de arquitectura de JEE en una secuencia de pasos de cada petición realizada desde el cliente al servidor.

- (1) Restauración de la vista: crea un árbol de componentes en el servidor para representar la información de un cliente.
- (2) Aplicar valores de la petición: actualiza los valores del servidor con datos del cliente.
- (3) Proceso de validación: valida los datos del usuario y hace la conversión a tipos específicos que se añaden a los componentes.
- (4) Actualización de valores del modelo: actualiza el modelo del servidor con nuevos datos.
- (5) Invocar la aplicación: ejecutar cualquier lógica de aplicación para cumplir con la solicitud.

- (6) Procesar la respuesta: guarda un estado y da una respuesta al cliente JSF.

Ciclo de vida de JSF

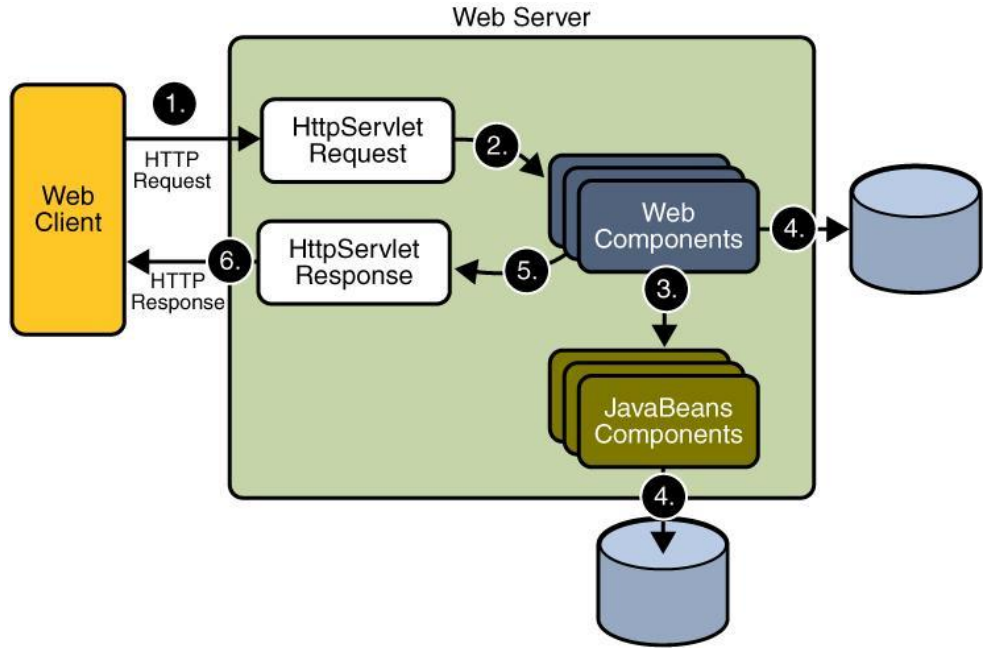


Figura 10. Ciclo de vida de una aplicación JSF. Fuente: Oracle Corporation.

2.8 Glassfish server

Es un servidor de aplicaciones Java compatible con la plataforma Java Enterprise Edition (Java EE) 6 desarrollada por Sun Microsystems, diseñada para desarrollar e implementar aplicaciones de Java EE y servicios web, GlassFish Sun Enterprise Server es gratuito si se utiliza para el desarrollo, no presenta coste por su uso, se distribuye bajo un licenciamiento dual a través de la licencia comercial CDDL y una licencia libre GNU.

La plataforma Java EE 6 mejora significativamente la productividad del desarrollador, ya que presenta un perfil ligero para aplicaciones web e incluye las últimas versiones de tecnologías como JAX-RS 1.1, JavaServer Faces(JSF) 2.0, Enterprise JavaBeans (EJB) 3.1, Java Persistence (App) 2.0, Context and Dependency Injection (CDI) 1.0 (Oracle Corporation, 2013).

2.8.1 Evolución de Glassfish

2005- Sun Microsystems anunció por primera vez el Proyecto GlassFish.

2006- El Proyecto GlassFish libera la primera versión que soporta la especificación de Java EE 5.

2007- La comunidad GlassFish liberó la versión 2 (también conocida como Sun Java System Application Server 9.1) con el soporte de total capacidad para clustering empresarial y además de Servicios Web interoperables con tecnologías Microsoft.
2009- Sun Microsystems y la comunidad liberaron la versión GlassFish 2.1 (también conocida como Sun GlassFish Enterprise Server 2.1.
En diciembre del mismo año GlassFish v3 es liberado. En esta versión GlassFish adiciona nuevas características para facilitar la migración desde Apache Tomcat a GlassFish. La modularidad era otra característica,
2010- Oracle emitió una hoja de ruta para las versiones 3.0.1, 3.1, 3.2 y 4.0 con temas girando en torno a Clustering, virtualización e integración con Coherence y otras tecnologías de Oracle.
2011- Oracle Corporation liberó GlassFish v3.1. Esta versión introdujo el soporte de aprovisionamiento basado en ssh, administración centralizada, clustering y balance de carga, en julio del mismo año, Oracle Corporation liberó GlassFish v3.1.1.
2012- Oracle Corporation liberó GlassFish v3.1.2.2. Esta fue una "micro" versión para solucionar cuestiones excepcionales relativas a GlassFish.
2013- Oracle Corporation liberó GlassFish Server Open Source Edition 4.0. Esta es una implementación de producción compatible de la especificación de la plataforma Java EE 7 construida usando una licencia de código abierto. Similar a Java EE 5 y Java EE 6, la Implementación de Referencia (IR) está derivada del Proyecto GlassFish. Como la IR, GlassFish Server está siempre al día con las últimas especificaciones de Java EE.
2013- Oracle anunció la futura hoja de ruta para Java EE y GlassFish Server, con una planeada edición de código abierto de GlassFish 4.1 y contínuas actualizaciones para GlassFish pero finalizando el fin del apoyo comercial de Oracle (Oracle Corporation, 2013).

2.8.2 Características Glassfish

Entre las características más importantes del servidor de aplicaciones Glassfish se pude citar:

- Compatibilidad con perfiles de Java EE
- Diseño modular
- Compatibilidad para la ampliación de Enterprise Server
- Integración de Herramienta de actualización

- Compatibilidad con lenguajes de secuencia de comandos
- Compatibilidad con las tecnologías de interoperabilidad de servicios web (WSíT)
- Mejoras de la utilidad appclient
- Integración de EclipseLink
- Coloca la configuración de servicio HTTP en servicios de red

2.8.3 **Arquitectura de Glassfish**.

Glassfish dispone de una arquitectura modular, basada en una estructura de clústeres flexibles que brindan soporte para la gestión remota multi-máquina y multi-dominio, lo que mejora el grado de disponibilidad, escalabilidad y rendimiento del servidor de aplicaciones para sistemas de alto volumen y servicios web. De esta manera s optimiza el tiempo de inicio, consumo de memoria y espacio en disco. Basándose en el modelo dinámico de componentes para Java OSGi (Open Services Gateway Initiative)

Arquitectura de Glassfish

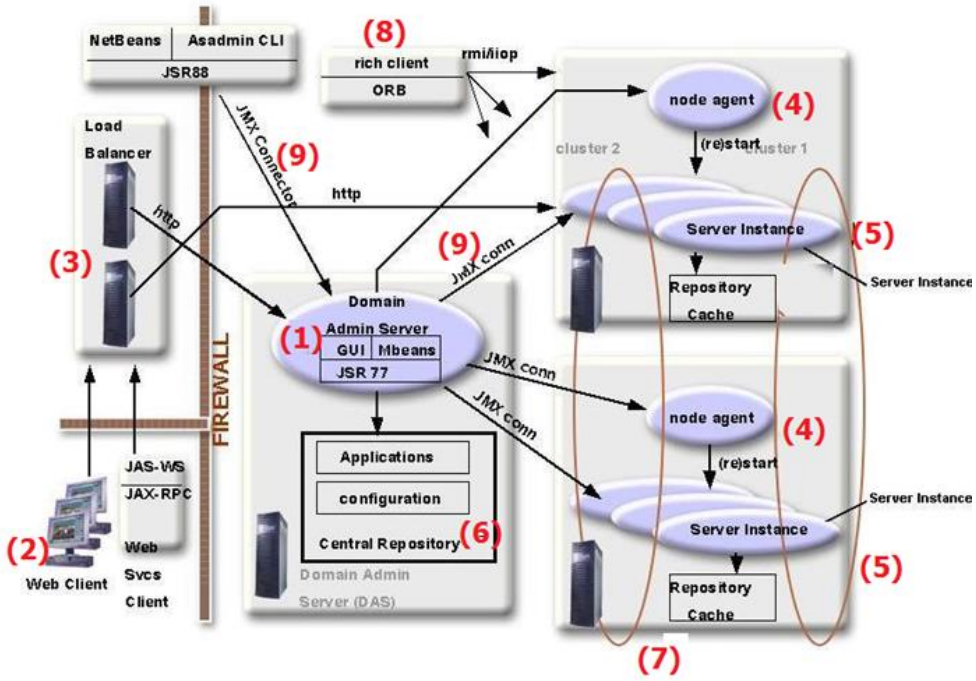


Figura 11. Arquitectura del servidor Glassfish. Fuente: Oracle Corporation.

Como se puede apreciar en la Figura 11, la arquitectura de Glassfish consta de los siguientes componentes (Oracle Corporation, 2010):

(1) Servidor de administración de dominio (DAS): DAS, facilita la administración del dominio. Todas las operaciones administrativas se dirigen a la DAS en lugar de las instancias de servidor directamente. El DAS esa conformado implícitamente por:

- ***Aplicaciones cliente administrativo (CLI, JSR 77, AMX):*** estas aplicaciones residen dentro o fuera de la Firewall y se comunican exclusivamente con el servidor del administrador de dominio (y no las instancias directamente). Estos clientes usan conectores de JMX JSR160 a la DAS. Para cruzar el cortafuego, También existe una interfaz MBEan llamado AMX que expone toda la funcionalidad a través del API.

- ***Navegador basado en interfaz gráfica de usuario administrativo:*** la aplicación administrativa de GUI (interfaz gráfica de usuario) basado en un navegador es desplegada en el caso de Java EE 5 que aloja el servidor de administrador de dominio.
- ***Administración de API:*** un subconjunto de la administración de Mbeans JMX en el DAS será expuesto y a disposición de los clientes. Estas interfaces se comunicarán con el DAS usando el conector estándar JSR 160 RMI.

(2) Aplicaciones de cliente web (clientes web, clientes de servicios web): un cliente web se comunica con la instancia del servidor mediante http a través de un navegador web o una invocación del servicio web.

(3) Balanceadores (LB) de carga: el equilibrador de carga es responsable de dirigir nuevas solicitudes entrantes a la máquina host menos cargada, reconociendo los nodos fracasados, reintentar las operaciones si es necesario y mantener la afinidad (adherencia) con una serie particular en que se ha establecido una sesión.

(4) Agente de nodo (NA): su objetivo principal será iniciar, detener y crear instancias de servidor como se indica por el DAS. El agente de nodo también actuará como un perro guardián y reiniciar procesos fallidos (por ejemplo, en la ausencia del núcleo de la web). Como el DAS, el agente de nodo debe ser sólo para ciertas operaciones administrativas y no debe esperarse que sea altamente disponible.

(5) Instancia del servidor (Sí): comunicación bidireccional entre las instancias de servidor y el DAS se basa en invocaciones de método de MBean remotas vía JSR160. El conector estándar JSR 160 RMI se utiliza para todos los servidores de comunicación DAS.

(6) Repositorio central: hay 2 repositorios principales de almacenamiento de información compartida por todas las instancias en el dominio. El repositorio de configuración contiene información de configuración del dominio y el repositorio de la aplicación aplicaciones J2EE desplegados. El repositorio central será normalmente basado bajo un solo directorio como tal puede fácilmente ser respaldado y restaurado.

(7) Caché del repositorio local: para evitar forzar restricciones de alta disponibilidad en el DAS y arranque más rápido, cada instancia mantiene en su sistema de archivos local, una memoria caché de la configuración del repositorio Central. La memoria caché se sincronizarán con el repositorio Central cada vez que la instancia se reinicia.

(8) Aplicaciones dinámicas de cliente (RMI/IIOP): un Rich client se ejecuta en un proceso alejado de las instancias de servidor y se comunica con las instancias de servidor a través de un stub cliente generado sobre el protocolo RMI/IIOP. El Rich client puede alojarse en el servidor de aplicación cliente (ACC) en el que tiene acceso a todos los recursos JNDI definido en el servidor, o puede ejecutarse como un cliente RMI.

(9) Tiempo de ejecución de JMX: la infraestructura JMX proporciona la infraestructura administrativa para realizar las operaciones de administración y configuración. El tiempo de ejecución JMX incluye los servicios de agente (servidor MBean, servicio de temporizador, control de servicio, etc.). Es el medio de comunicación entre todos los elementos de la arquitectura Glassfish.

2.9 PostgreSQL

Es un sistema de gestión de bases de datos de código abierto objeto-relacional, distribuido bajo licencia BSD con características semejantes que las bases de datos comerciales.

PostgreSQL utiliza un modelo cliente/servidor y usa multiprocesos en vez de multihilos para garantizar la estabilidad del sistema.

2.9.1 Evolución de PostgreSQL

1977- PostgreSQL comenzó como un proyecto denominado Ingres en la Universidad Berkeley de California.

1986- otro equipo dirigido por Michael Stonebraker de Berkeley continuó el desarrollo del código de Ingres para crear un sistema de bases de datos con objetos relacionales llamados Postgres.

1996- por el incremento de la funcionalidad del software, Postgres fue renombrado a PostgreSQL y se creó el Global Development Team.

1997- después de haber cambiado su nombre de Postgres95 a PostgreSQL fue lanzada al mercado la v6.0, con las siguientes novedades Se hicieron mejoras de velocidad importantes., hubo un aumento significativo en el rendimiento.

2000- fue lanzada la v7.0, y presenta las siguientes mejoras, el bloqueo a nivel de tabla fue reemplazado por MVCC.

2001- fue lanzada la v 7.1. Con las siguientes mejoras, confiabilidad e integridad en los datos priorizada históricamente.

2002- fue lanzada la v.2 y 1 se liberaron el mismo año la v7.3, con la novedad de que tenía un registro de escritura adelantada

2003- fue lanzada la v7.4 presentando actualizaciones como el uso de OUTER JOINs y Toast

2005- la versión v8.0 y la v8.1 se liberaron el mismo año, con grandes mejoras por ejemplo la partición de tablas mediante el uso de "exclusión de restricciones"

2006- fue lanzada la v8.2, La versión 8.2 fue notable por ser la versión usada por Greenplum como base para su producto de base de datos propietario, dirigido al mercado de almacenamiento de datos

2008- fue lanzada la v8.3, presentando estas mejoras en relación a la versión anterior, commit asíncrono, una característica introducida por Simon Riggs, CTO de 2ndQuadrant para permitirle a las transacciones hacer commit asíncronamente por razones de rendimiento, que pude ser usado por aplicaciones donde esto sea conveniente.

2009- se lanza al mercado la versión v8.4, presentando mejoras como la restauración paralela: restaurar la base de datos de un respaldo lógico en paralelo, el mapa de visualización, que reduce notablemente la sobrecarga de vacuum para tablas que no cambian frecuentemente.

2010- fue lanzada la v9.0 donde una característica trascendente es la replicación en el core.

2011- fue lanzada la v9.1 permite la construcción de aplicaciones que se ejecuten en la nube, presenta actualizaciones como replicación síncrona, tabla de invitados mejorando el rendimiento.
2012- fue lanzada la v9.2, esta versión fue mejorada en la optimización de índices ya que permite acceder más rápido a la información.
2014- fue lanzada la v9.3esta versión permite el intercambio de datos bidireccional, y la implementación de características que incrementan la fiabilidad, disponibilidad de PosgreSQL.
2015- fue lanzada la versión 9.4.4, en esta versión la API fue actualizada permitiendo filtrar, manipular el flujo de aplicaciones PosgreSQL.
Existe una gran comunidad de usuarios, programadores, administradores y empresas que colaboran activamente en numerosos aspectos relacionadas con el proyecto como aportes económicos y personal capacitado para mejorar PostgreSQL. Muchos desarrolladores y nuevas características están patrocinados por empresas privadas. En los últimos años los trabajos de desarrollo se han concentrado mucho en la velocidad de procesamiento y en características requeridas en el mundo empresarial. El proyecto PostgreSQL sigue actualmente activo mediante un proceso de desarrollo a nivel mundial gracias a un equipo de desarrolladores y contribuyentes de código abierto. (Padrón Ortiz, F. M. (2013).

2.9.2 Características de PostgreSQL

Está considerado como la base de datos de código abierto más avanzada del mundo. PostgreSQL proporciona un gran número de características que normalmente sólo se encontraban en las bases de datos comerciales tales como DB2 u Oracle. La siguiente es una breve lista de algunas de esas características, a partir de PostgreSQL 7.1.x.

- **DBMS Objeto-Relacional**

  PostgreSQL aproxima los datos a un modelo objeto-relacional, y es capaz de manejar complejas rutinas y reglas. Ejemplos de su avanzada funcionalidad son consultas SQLzz declarativas, control de concurrencia multi-versión, soporte multi-usuario, transactions, optimización de consultas, herencia, y arrays
- **Altamente_Extensible**

  PostgreSQL soporta operadores, funcionaes métodos de acceso y tipos de datos definidos por el usuario.
- **Soporte_SQL_Comprensivo**

PostgreSQL soporta la especificación SQL99 e incluye características avanzadas tales como las uniones (joins) SQL92.

- **Integridad Referencial**

  Soporta integridad referencial, la cual es utilizada para garantizar la validez de los datos de la base de datos.

- **API Flexible**

  La flexibilidad del API de PostgreSQL ha permitido a los vendedores proporcionar soporte al desarrollo fácilmente para el RDBMS PostgreSQL. Estas interfacaes incluyen Object Pascal, Python, Perl, PHP, ODBC, Java/JDBC, Ruby, TCL, C/C++, y Pike.

- **Lenguajes orientados a procedimientos**

  PostgreSQL tiene soporte para lenguajes procedurales internos, incluyendo un lenguaje nativo denominado PL/pgSQL. Este lenguaje es comparable al lenguaje procedural de Oracle, PL/SQL. Otra ventaja de PostgreSQL es su habilidad para usar Perl, Python, o TCL como lenguaje procedural embebido.

- **Cliente/Servidor**

  PostgreSQL usa una arquitectura proceso-por-usuario cliente/servidor. Esta es similar al método del Apache 1.3.x para manejar procesos. Hay un proceso maestro que se ramifica para proporcionar conexiones adicionales para cada cliente que intente conectar a PostgreSQL. (Solana, 2013)

2.9.3 Arquitectura de PostgreSQL

En la Figura 12 se ilustra de manera detallada los componentes de la arquitectura del motor de base de datos PostgreSQL.

(1) Aplicación cliente: esta es la aplicación cliente que utiliza PostgreSQL como administrador de bases de datos. La conexión puede ocurrir vía TCP/IP o sockets locales.

(2) Demonio postmaster: este es el proceso principal de PostgreSQL. Es el encargado de escuchar por un puerto/socket por conexiones entrantes de clientes. También es el encargado de crear los procesos hijos que se encargaran de autentificar estas peticiones, gestionar las consultas y mandar los resultados a las aplicaciones clientes

(3) Ficheros de configuración: los 3 ficheros principales de configuración utilizados por PostgreSQL, postgresql.conf, pg_hba.conf y pg_ident.conf

(4) Procesos hijos postgres: procesos hijos que se encargan de autentificar a los clientes, de gestionar las consultas y mandar los resultados a las aplicaciones clientes

Arquitectura de PostgreSQL

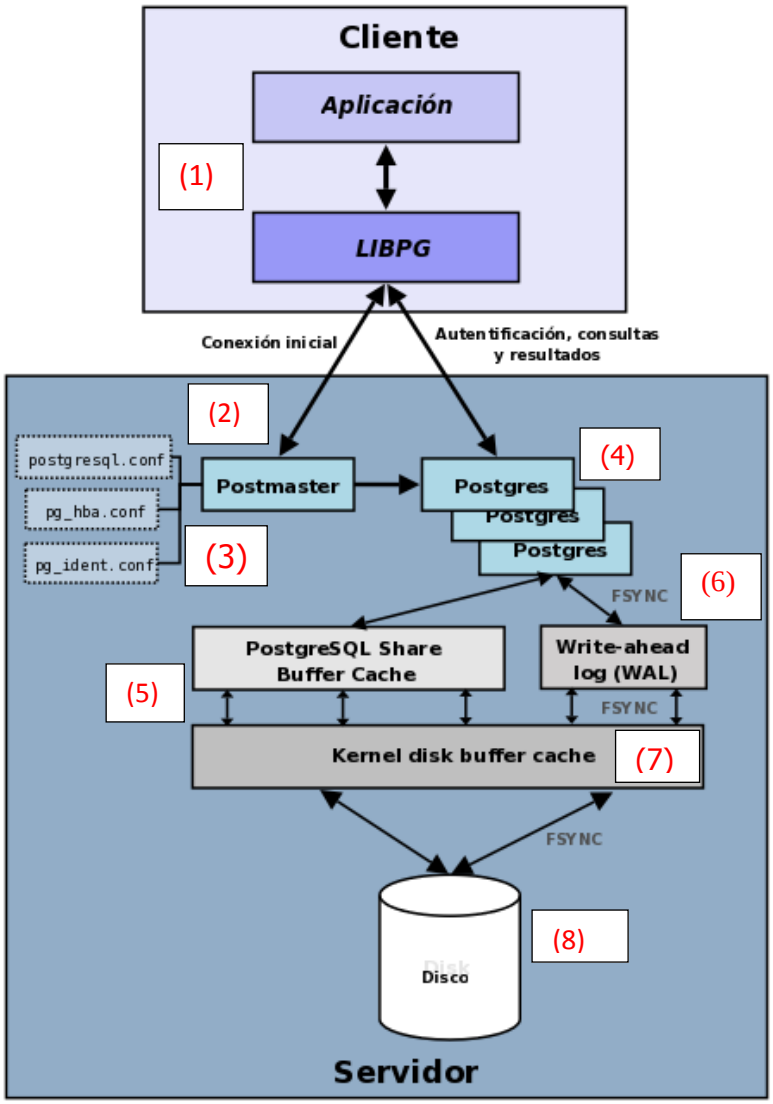


Figura 12. Componentes de la arquitectura PostgreSQL. Fuente: The PostgreSQL Global Development Group.

(5) PostgreSQL share buffer cache: memoria compartida usada por PostgreSQL para almacenar datos en cache.

(6) Write-Ahead Log (WAL): componente del sistema encargado de asegurar la integridad de los datos (recuperación de tipo REDO).

(7) Kernel disk buffer cache: caché de disco del sistema operativo.

(8) Disco: disco físico donde se almacenan los datos y toda la información necesaria para que PostgreSQL funcione.

CAPÍTULO 3

ANÁLISIS Y DISEÑO

Este capítulo presenta la etapa inicial del desarrollo del presente proyecto. Una vez claro el panorama expuesto en el capítulo anterior; sobre las herramientas con las que se trabajará, se realiza un análisis de los requerimientos del cliente en cuanto a funcionalidad del producto. Con este análisis se procede al diseño de los modelos de datos que se adaptan a los requerimientos para de esta forma dar inicio a la construcción del aplicativo final.

3.1 Descripción del producto

Esquema general de la aplicación

Figura 13. Descripción grafica del producto. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

A través de la Figura 13 se muestra el funcionamiento integral del sistema para gestionar las actividades internas del laboratorio de física, se ha dividido el sistema en dos módulos, gestión académica y gestión administrativa donde manejaremos,

inventario, evaluaciones, material didáctico, prácticas de laboratorio detallados a continuación:

(1) Inventario: maneja todos los bines que posee el laboratorio de física siendo estos materiales, equipos, los cuales serán gestionados a través de eventos de inventario como préstamos, mantenimiento, devoluciones, extraviados.

(2) Prácticas de Laboratorio: consiste en crear las prácticas de acuerdo al contenido, de la práctica.

(3) Evaluaciones: serán de tres tipos diferentes, coloquio, defensa, informe, las cuales serán asignadas por el docente y cada una de las evaluaciones serán bajo una nota y desarrollo diferente.

(4) Contenido didáctico: constará de una guía donde se detalla todo el procedimiento a realizarse en la práctica de laboratorio la cual estará en formato pdf, pudiendo ser descargada, de la misma manera estará disponible un video interactivo que muestra el desarrollo de una práctica, fragmentado en tres partes introducción, desarrollo, resultados lo que ayudara a los estudiantes ah desarrollar rápidamente sus prácticas.

A continuación se describe el sistema de manera técnica.

Con el fin de proporcionar una solución y la base para un mejor manejo de las actividades de los laboratorios de física, se plantea la implementación de un sistema informático orientado a la web que se desarrollará en la plataforma Java Server Faces (JSF), un marco de trabajo que permite la creación de aplicaciones web utilizando el lenguaje de programación Java. Así también, este proyecto se construirá sobre el motor de base de datos PostgreSQL, que es un sistema de código abierto para la gestión de bases de datos relacionales.

En cuanto a la metodología de desarrollo del proyecto, se seleccionó Scrum. Ésta es una metodología ágil para el desarrollo de sistemas informáticos y permite la construcción del producto final gracias a la elaboración de partes más pequeñas lo que facilita y hace más manejable el volumen de trabajo.

Este sistema funcionará de forma independiente a las plataformas virtuales con que cuenta la Universidad Politécnica Salesiana y constará, como se puede ver en la Figura 14, de dos grandes módulos desde un punto de vista funcional: gestión académica y gestión administrativa.

(1) Módulo de gestión académica: está enfocado en desarrollo de las actividades del estudiante antes, durante y después de la elaboración de las respectivas prácticas de laboratorio; dentro de este módulo se incluyen los siguientes componentes:

a) Asignación de estudiantes a grupos de trabajo.

b) Consulta a repositorio de documentos y material de apoyo; propios del laboratorio de física, relacionados al trabajo de los estudiantes, éste material debe incluir un video sobre el proceso de la práctica, dividida en tres partes: introducción, desarrollo y resultados.

c) Evaluación, previa y posterior al desarrollo de las actividades de las prácticas de laboratorio, siguiendo los modelos de preguntas propuestos por el CEAACES; entidad reguladora de la educación superior en el Ecuador, que plantea el uso de evaluaciones en base a reactivos.

d) Asignación de tareas y evaluaciones sobre las prácticas de laboratorio.

e) Informes de laboratorio en línea, que permitan al estudiante desarrollar su trabajo y cargarlo al sistema en la web mediante un archivo.

(2) Módulo de gestión administrativa: está dirigido a los docentes y ayudantes del laboratorio de física, con el fin de facilitar su trabajo y optimizar tiempo y recursos, brindándoles la oportunidad de enfocarse más en el desempeño académico de sus estudiantes, éste módulo se compone de los siguientes elementos:

a) Administración de perfiles y usuarios.

b) Envío de notificaciones a los estudiantes vía correo electrónico.

c) Gestión de inventario de equipos y enseres, el sistema permitirá ingresar información de los elementos del laboratorio para llevar un mejor control de los mismos además de un registro de los préstamos de equipos y materiales de laboratorio.

d) Planificación de mantenimiento y renovación, mediante un registro de los posibles daños y de las revisiones preventivas de los equipos del laboratorio.

Diagrama funcional del producto

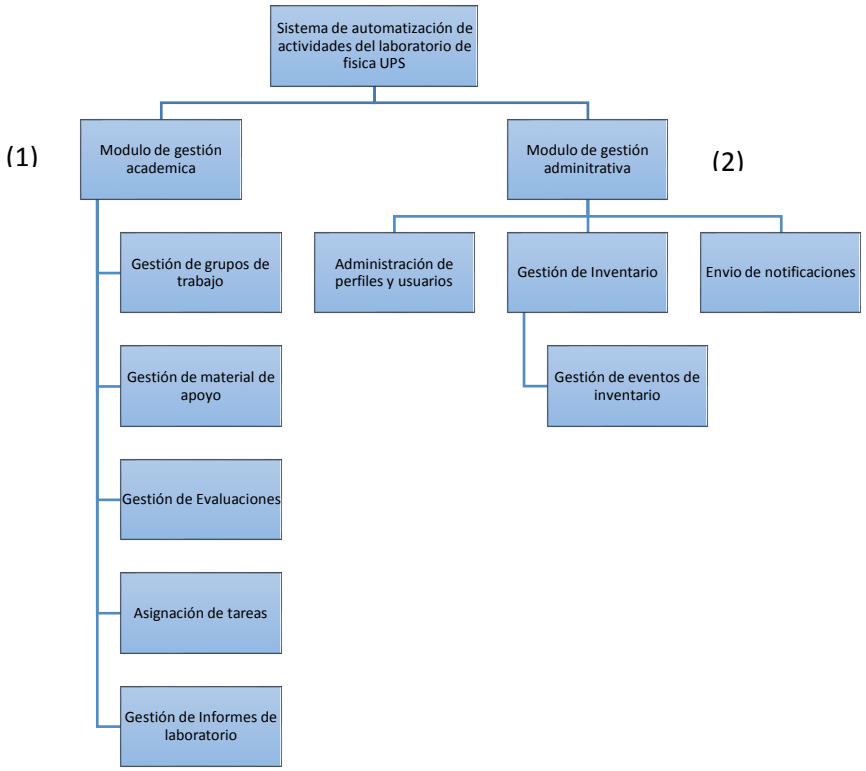


Figura 14. Estructura funcional del producto. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

3.2 Análisis de perfiles de usuario

A continuación se realiza una descripción detallada de todas las funciones en cada perfil de usuario. Se ha dividido dentro de la explicación de cada perfil, lo requisitos de acuerdo a su prioridad.

3.2.1 Administrador

El usuario del perfil administrador es aquel que realiza todas las actividades del módulo de gestión administrativa y sus subprocesos, los que se listan a continuación.

- Administración de usuarios. Ver Tabla 7.
- Registro de eventos de inventario. Ver Tabla 8.
- Administración de materias. Ver Tabla 9.
- Administración de docentes. Ver Tabla 10.
- Administración de inventario. Ver Tabla 11.
- Reportes.

Tabla 7. Administración de usuarios

ADMINISTRACIÓN DE USUARIOS	
OPCIONES	**DESCRIPCIÓN**
Ingresar manualmente los usuarios	Los datos del usuario se ingresarán de forma manual mediante un formulario. Los datos serán: nombres, apellidos, cedula de identidad, correo electrónico, teléfono, usuario y contraseña
Cargar estudiante mediante archivo csv	Los usuarios seran ingresados a través de un archivo CSV separado por comas, archivo que tendrá los siguientes datos: nombres, apellidos, cedula de identidad, correo electrónico, tipo de usuario, materia, nivel y paralelo
Mover estudiante	Seleccionamos el estudiante, y mediante un formulario se escoge el periodo, la carrera, materia y el docente a donde será movido

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

Tabla 8. Registro de eventos de inventario

REGISTRO DE EVENTOS DE INVENTARIO	
OPCIONES	**DESCRIPCIÓN**
Ingresar nuevo bien	Se ingresarán los bienes mediante un formulario, para lo cual se necesita la siguiente información: nombre del bien, cantidad, fecha de adquisición, vida útil, tipo de bien, marca, serie, observaciones, estado
Registrar evento	Existen cuatro eventos posibles: préstamo, devolución, mantenimiento, extraviado
Registrar Prestamos	Mediante este formulario se realizaran los préstamos de los bienes del laboratorio, ingresando lo siguiente: cantidad, cedula del responsable, nombres, apellidos, email, teléfono, fecha de préstamo, fecha de entrega, observaciones
Registrar Devolución	Mediante este formulario se realizaran las devoluciones de los bienes del laboratorio, ingresando lo siguiente: cedula responsable la misma que será validada y se cargaran automáticamente los datos del responsable, luego se ingresara la fecha de devolución, y las observaciones

Registrar mantenimiento	Mediante este formulario se registraran los bienes del laboratorio que ingresaran a mantenimiento, se debe ingresar: fecha de inicio del mantenimiento y fecha de fin del mantenimiento
Registrar bien extraviado	Mediante este formulario se registraran los bienes extraviados del laboratorio se debe ingresar: fecha extraviado, cedula del responsable, nombres, apellidos, email, teléfono, observaciones, fecha de reposición

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

Tabla 9. Administración de materias

ADMINISTRACIÓN DE MATERIAS	
OPCIONES	**DESCRIPCIÓN**
Ingresar nueva materia	Se ingresaran las materias mediante una interfaz ingresando la siguiente información: materia, nivel, periodo
Editar materia	La materia podrá ser editada, al modificar cualquier campo del formulario
Asignar materia por carrera	En la interfaz se seleccionará la materia creada, la carrera, así como la hora de entrada y la hora de salida de la materia
Modificar materia	La materia podrá ser editada, al modificar cualquier campo del formulario

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 10. Administración de docentes

ADMINISTRACIÓN DE DOCENTES	
OPCIONES	**DESCRIPCIÓN**
Asignar docente a la materia	En la interfaz se seleccionará el docente y la materia que dictara el docente
Modificar docente	Se podrá volver a seleccionar otra materia para asignarle al docente
Eliminar docente	Se eliminaran los datos del docente y la materia asignada

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 11. Administración de inventario

ADMINISTRACIÓN DE INVENTARIO	
OPCIONES	**DESCRIPCIÓN**
Editar el bien	Se podrá editar la cantidad de bienes de acuerdo a su estado en: buenos, malos y regulares

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.2.2 Docente

El usuario del perfil docente, es aquel que preparara todas las actividades que realizara el estudiante. Estas actividades se listan a continuación:

- Creación de grupos. Ver Tabla 12.
- Registro de preguntas. Ver Tabla 14.
- Prácticas de laboratorio. Ver Tabla 13.
- Lista de Evaluaciones. Ver Tabla 15.
- Asignar líder de grupo. Ver Tabla 16.
- Lista de informes. Ver Tabla 17.
- Reportes. Ver Tabla 21.

Tabla 12. Creación de grupos

CREACIÓN DE GRUPOS	
OPCIONES	**DESCRIPCIÓN**
Crear grupos	El docente creara los grupos de trabajo, donde los estudiantes trabajaran durante el semestre, a través de un formulario, el mismo que se debe llenar con los siguientes datos, nombre del grupo, cantidad de alumnos, horario y seleccionar el laboratorio, seleccionar periodo, seleccione materia
Editar grupo	El grupo podrá ser editado, al modificar cualquier campo del formulario

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 13. Prácticas de laboratorio

PRÁCTICAS DE LABORATORIO	
OPCIONES	**DESCRIPCIÓN**
Crear prácticas de laboratorio	Se crearan las prácticas de laboratorio, para lo cual es necesario ingresar nombre de la práctica y la descripción
Asignar material de apoyo	Se asignara el material de apoyo para cada práctica, se requiere seleccionar la práctica, material de apoyo que podrá ser guía y video
Ver material de poyo	Se podrá visualizar el material de poyo asignado

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 14. Gestión de cuestionario

GESTIÓN DE CUESTIONARIO	
OPCIONES	**DESCRIPCIÓN**
Crear preguntas	Se crearan las preguntas, de cada práctica de laboratorio, se requieren los siguientes datos para crear el cuestionario: Nivel de dificultad de la pregunta: se manejaran 3 niveles: básica, media y avanzada, las cuales tienen pesos correspondientes 1, 0.6 y 0.4 Enunciado: aquí se redactara la pregunta, y tendrá la posibilidad de subir un gráfico para complementar el enunciado Opciones de respuesta: se manejara cuatro opciones de respuesta, pero solo una respuesta es la correcta
Ver cuestionario	Se podrán ver el cuestionario creado

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 15. Lista de evaluaciones

LISTA DE EVALUACIONES	
OPCIONES	**DESCRIPCIÓN**
Listar evaluaciones	Aquí se listaran las evaluaciones creadas por cada práctica

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 16. Asignar líder de grupo

ASIGNAR LÍDER DE GRUPO	
OPCIONES	**DESCRIPCIÓN**
Asignar líder de Grupo	Aquí se asignara un estudiante como líder de grupo, se requiere el grupo de trabajo, para saber cuáles son los estudiantes del grupo

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 17. Lista de informes

LISTA DE INFORMES	
OPCIONES	**DESCRIPCIÓN**
Listar informes	Aquí podrá ver y calificar los informes que los estudiantes subirán

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.2.3 Estudiante

El usuario del perfil estudiante, es aquel que desarrolla todas las actividades que involucra el módulo de gestión académica, actividades solicitadas por el docente. Estas actividades son:

- Registrarse en un grupo de trabajo. Ver Tabla 18.
- Acceder a guías de evaluación. Ver Tabla 19.
- Realizar tareas asignadas. Ver Tabla 20.

Tabla 18. Registrarse en un grupo

REGISTRARSE EN UN GRUPO	
OPCIONES	**DESCRIPCIÓN**
Escoger un grupo	Aquí el estudiante podrá seleccionar el grupo donde va trabajar y ver los alumnos inscritos en el grupo

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 19. Acceder a guías de evaluación

GUÍA DE EVALUACIÓN	
OPCIONES	**DESCRIPCIÓN**
Descargar guía de evaluación	En esta área el estudiante podrá ver la lista de prácticas con su respectivo material de apoyo, el cual podrá ser descargado

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 20. Realizar tareas asignadas

TAREAS ASIGNADAS	
OPCIONES	**DESCRIPCIÓN**
Revisar tareas asignadas	En esta área el estudiante podrá revisar sus evaluaciones asignadas: coloquio y defensa Si el estudiante es jefe de grupo tendrá una evaluación adicional: informe Evaluación que únicamente se le asignara a él, por ser el líder
Resolver evaluaciones	En esta área el estudiante podrá resolver sus evaluaciones, se desplegaran las preguntas y las opciones de respuesta se mostraran al seleccionar la pregunta, las respuestas no podrán ser modificas, se admitirá un solo intento En el caso del estudiante líder de grupo podrá: • Subir el informe de laboratorio • Evaluar el desempeño de sus compañeros

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.2.4 Tareas comunes a varios perfiles

Los usuarios con perfil administrador y docente tendrán la opción de imprimir reportes desde el sistema. Los reportes evidenciarán información relevante para el usuario como por ejemplo las notas obtenidas por los estudiantes o información del inventario.

Tabla 21. Reportes

REPORTES	
OPCIONES	**DESCRIPCIÓN**
Alumno por materia	Este reporte mostrará la lista de alumnos de una materia específica, por lo tanto se debe seleccionar la materia y el periodo
Lista original por materia	Este reporte mostrará la lista de los alumnos de una materia incluyendo los alumnos que fueron movidos, es necesario seleccionar el periodo la carrera y la materia
Notas por informe	Este reporte mostrará los informe con su nota respectiva, será necesario seleccionar: periodo, carrera, materia, práctica
Inventario de laboratorio	Este reporte mostrará los bienes que hay en el laboratorio, es necesario seleccionar
Nota final	Este reporte mostrará las lista de alumnos con su nota final, se requiere seleccionar el periodo, carrera, materia, fecha de inicio, fecha de fin para tomar en ese rango las notas

Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.3 Diseño de la base de datos

Luego del análisis realizado se generó el diseño de la base de datos según el modelo entidad –relación, obteniéndose un total de 37 tablas con sus respectivas relaciones. En la Figura 15 se encuentra el modelo físico de la base de datos, para apreciar de mejor forma se ha separado de manera visual las tablas de cada módulo.

Módulo de gestión académica formado por las siguientes tablas.

justificacion	pregunta	matricula
preguntasxevaluacion	niveldepregunta	laboratorio
respuesta	informe	grupo
pregunta	persona	perfil
tipoevaluacion	practicaxmateria	permiso
evaluacion	materiaxcarrera	carrera
evaluacionxlumno	docente	alumnoxgrupo
materialdeapoyo	movimientoestudiante	practicaxalumno
practica	periodo	alumno

Módulo de gestión administrativa formado por las siguientes tablas.

tipobien	extraviado	mantenimiento
estado	prestamo	permisoxperfil
bien	usuario	variablesdelsistema

Modelo físico de la base de datos

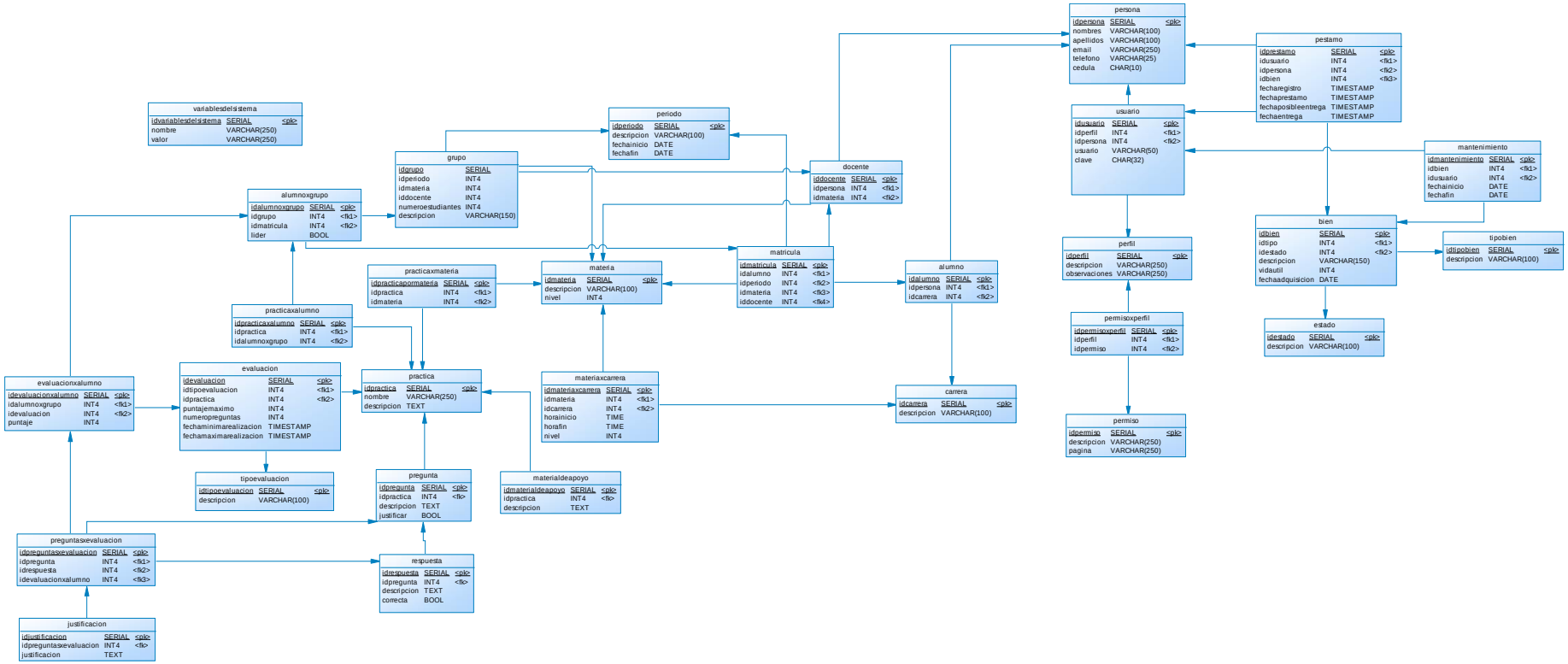


Figura 15. Modelo físico de la base de datos. Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.4 Planificación del proyecto

Para la Tabla 22, la prioridad se establecerá con los siguientes valores en un rango de 1-5, donde 1 es el valor de prioridad más alto y 5 el valor más bajo. Los objetivos que de acuerdo a su prioridad en el desarrollo del proyecto, se irán realizando paulatinamente.

Tabla 22. Lista de objetivos priorizados

Objetivos	Prioridad
Crear la base de datos	1
Programar los perfiles de usuario de Administrador, Docente, Alumno	1
Validar las cuentas de usuario	1
Construir el módulo de gestión administrativa	1
Crear Nuevo usuario	1
Subir estudiantes mediante el archivo csv	1
Mover estudiantes	2
Administración de inventarios	3
Administrar eventos de inventario	3
Construir el módulo de gestión académica	2
Asignar a los estudiantes grupos de laboratorio	2
Asignar líder de grupo	3
Crear prácticas de laboratorio	3
Crear cuestionarios sobre los temas de las prácticas	3
Crear evaluaciones de tipo (coloquio)	2
Crear evaluaciones (informe)	2
Calificar informes de laboratorio	3

Construir el módulo de gestión académica, para el perfil de usuario Alumno	3
Reporte de alumnos inscritos	3
Reporte de notas finales	3
Reporte de notas finales, detalladas por prácticas	3

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

3.4.1 Sprints

Los sprints son cada una de las iteraciones o repeticiones que se deben realizar para obtener el producto terminado. Para el desarrollo de este proyecto se lo ha dividido en cinco fases detalladas en las tablas a continuación.

- Primera fase. Construcción de la estructura general de la aplicación. Ver Tabla 23.
- Segunda fase. Construcción del módulo de gestión administrativa. Ver Tabla 24.
- Tercera fase. Construcción del módulo de gestión académica para el docente. Ver Tabla 25.
- Cuarta fase. Construcción del módulo de gestión académica para el estudiante. Ver
- Tabla 26.
- Quinta fase. Gestión de reportes. Ver Tabla 27.

Tabla 23. Fase 1 de construcción. Construcción de esquema general

SPRINT N°1	
Objetivo:	Crear la base de datos Programar los perfiles de usuario de Administrador, Docente, Alumno Validar las cuentas de usuario
Que se hará:	Realizar el modelo conceptual de la base de datos General el modelo físico de la base de datos

	General el script de la base de datos Generar modelo de persistencia (JPA) Crear plantilla maestra para las paginas jsf Realizar modelo navegacional Programar métodos de acceso para los perfiles de usuario: • Administrador • Docente • Alumno Asignar privilegios Validar las cuentas de usuario
Duración:	3 semanas

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 24. Fase 2 de construcción. Gestión administrativa

SPRINT N°2	
Objetivo:	Construir el módulo de gestión administrativa, formado por 2 subgrupos Administración de usuarios: Crear usuarios Subir estudiantes mediante el archivo csv Mover estudiantes Administración de inventarios Administrar eventos de inventario
Que se hará:	Crear las clases java para nuevo usuario Crear los paquetes

	Investigar el funcionamiento de manejo de archivos csv Crear método de programación para subir los alumnos mediante archivo
Duración:	4 semanas

Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 25. Fase 3 de construcción. Gestión académica del docente

SPRINT N°3	
Objetivo:	Construir el módulo de gestión académica, para el perfil de usuario del Docente Crear grupos de laboratorio Asignar a los estudiantes grupos de laboratorio Asignar líder de grupo Crear prácticas de laboratorio Crear cuestionarios sobre los temas de las prácticas Crear evaluaciones (coloquio, defensa, informe) Calificar informes de laboratorio
Que se hará:	Crear las clases java para nuevo usuario Crear los paquetes Creación de modelos de persistencia (JPA) Investigar el funcionamiento de manejo de archivos csv Crear método de programación para subir los alumnos mediante archivo
Duración:	6 semanas

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 26. Fase 4 de construcción. Gestión académica del estudiante

SPRINT N°4	
Objetivo:	Construir el módulo de gestión académica, para el perfil de usuario Alumno
Que se hará:	Seleccionar grupo de laboratorio Realizar tareas asignadas por el docente, coloquio, defensa informe Revisar le material de apoyo de cada practica Calificar miembros del grupo Subir informe de la práctica del laboratorio
Duración:	4 semanas

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 27. Fase 5 de construcción. Gestión de reportes

SPRINT N°5	
Objetivo:	Obtener los reportes de acuerdo a lo que requiere el docente, administrador: Reporte de alumnos inscritos Reporte de notas finales Reporte de notas finales, detalladas por prácticas
Que se hará:	Implementar jasper para los reportes Diseñar el formato del reporte Realizar consultas sql para obtener los datos del reporte Mostrar los reportes mediante la aplicación Exportar los reportes a formato pdf
Duración:	3 semanas

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

CAPÍTULO 4
CONSTRUCCIÓN Y PRUEBAS

Este capítulo se divide en dos secciones, la primera tratará sobre la obtención del producto mediante el desarrollo de cinco fases de construcción. Cada fase añade una funcionalidad al sistema según los incrementos previamente planificados. La segunda parte aborda el tema de las pruebas realizadas sobre el producto terminado en relación a rendimiento y pruebas internas realizadas durante el proceso de construcción.

4.1. Construcción de la estructura general de la aplicación

Gracias al uso de JSF, el esfuerzo requerido en la creación de clases se reduce considerablemente. Luego de diseñar y generar el esquema de la base de datos, se procedió a generar la conexión entre el motor de base de datos y el servidor de aplicaciones mediante un JDBC, configurado en el servidor de aplicaciones.

Con esta conexión se puede generar el módulo de persistencia, que como ya se mencionó en capítulos anteriores, es el encargado de gestionar las conexiones entre la aplicación Java EE y la base de datos.

Diagrama de clases

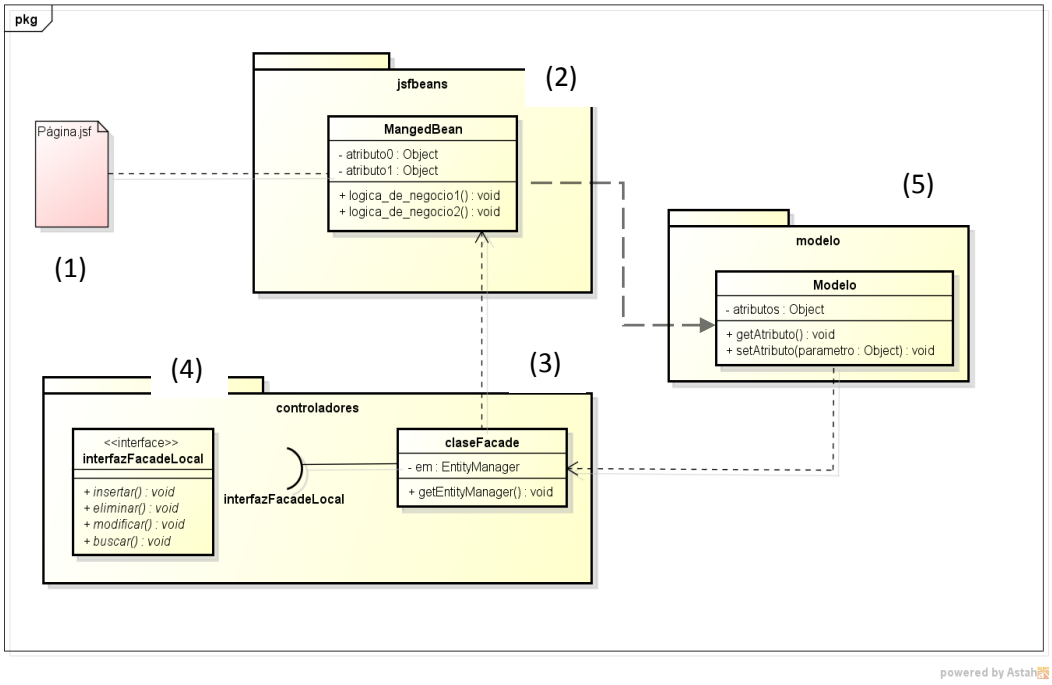


Figura 16. Diagrama general de clases. Elaborado por: Myrian Guijarro & Jorge Jaramillo

La Figura 16 muestra mediante un diagrama de clases general, la estructura de todo el sistema, el que trabaja de la siguiente manera:

(1) El usuario ingresa al sistema por medio de una página web, pagina.jsf.

(2) Realiza peticiones al servidor desde la pagina.jsf. Existe una clase de tipo ManagedBean, asociada a la función requerida por el usuario, la que contiene la lógica de negocio.

(3) El ManagedBean se comunica a su vez con los controladores en caso de requerir operaciones de la base de datos o con el modelo para recuperar datos de la capa de persistencia.

(4) Los controladores implementan una Interfaz que maneja las operaciones generales de la base de datos: insertar, eliminar, modificar, buscar.

(5) El modelo representa a la capa de persistencia entre la aplicación y la base de datos. Cada clase en el modelo es igual a su tabla respectiva en la base de datos.

4.2. Primera fase de construcción. Construcción de esquema general

- Crear la base de datos
- Programar los perfiles de usuario de Administrador, Docente, Alumno
- Validar las cuentas de usuario

4.2.1 Tablas relacionadas a los objetivos de la fase

En el proceso de inicio de sesión en el sistema intervienen cuatro tablas relacionadas como se muestra en la Figura 17. La descripción de la información que se almacena en las mismas se puede ver en la Tabla 28.

Diagrama de base de datos

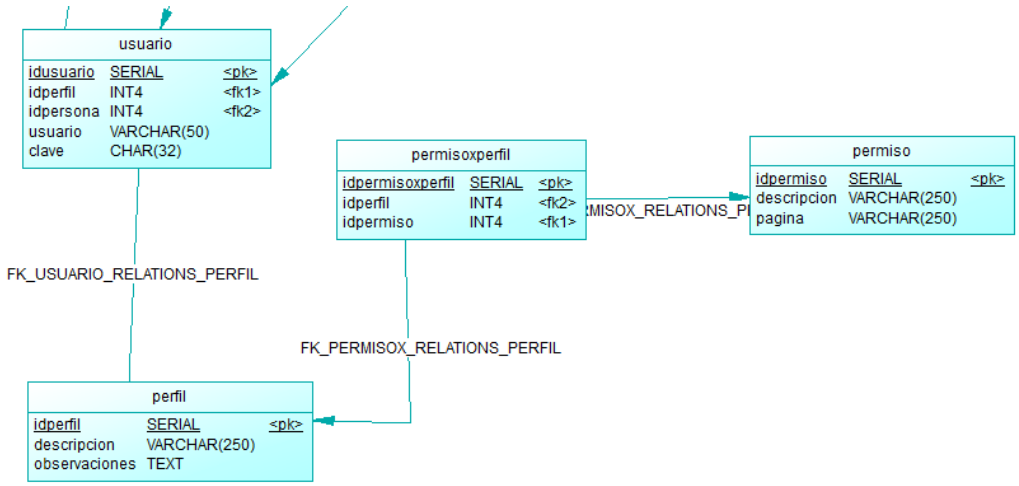


Figura 17. Tablas para inicio de sesión Elaborado por: Myrian Guijarro & Jorge Jaramillo.

Tabla 28. Descripción de tablas involucradas en inicio de sesión

Nombre de la tabla	**Descripción**
usuario	Almacena los datos de inicio de sesión de los usuarios
perfil	Almacena los datos que identifican a los tipos de usuario del sistema
permiso	Contiene los nombres de las distintas páginas web y sus rutas de acceso
permisoxperfil	Almacena los permisos de acceso que tiene cada perfil de usuario

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.2.2 Diagramas de Secuencia

Al finalizar la fase 1 se obtuvo la estructura general de la aplicación y el esquema básico de inicio de sesión para cada perfil de usuario.

Inicio de sesión en el sistema (Figura 18): se inicia en la página index.jsf, el usuario ingresa sus datos (nombre de usuario y contraseña) y envía la petición al Bean correspondiente, este realiza la validación de los datos proporcionados consultando al modelo. Si los datos son correctos, el Bean envía su respuesta y direcciona al usuario a la pantalla principal.jsf donde se construye el menú de usuario de acuerdo al perfil que posea (Administrador, Docente o Estudiante).

4.2.3 Diagrama de clases

La Figura 19 muestra el diagrama de clases para el proceso de inicio de sesión de usuarios en el sistema. Se puede apreciar claramente que sigue el patrón explicado con anterioridad en la Figura 16. La clase loginBean controla el proceso de ingreso al sistema comunicándose con el controlador.

Diagrama de Secuencia

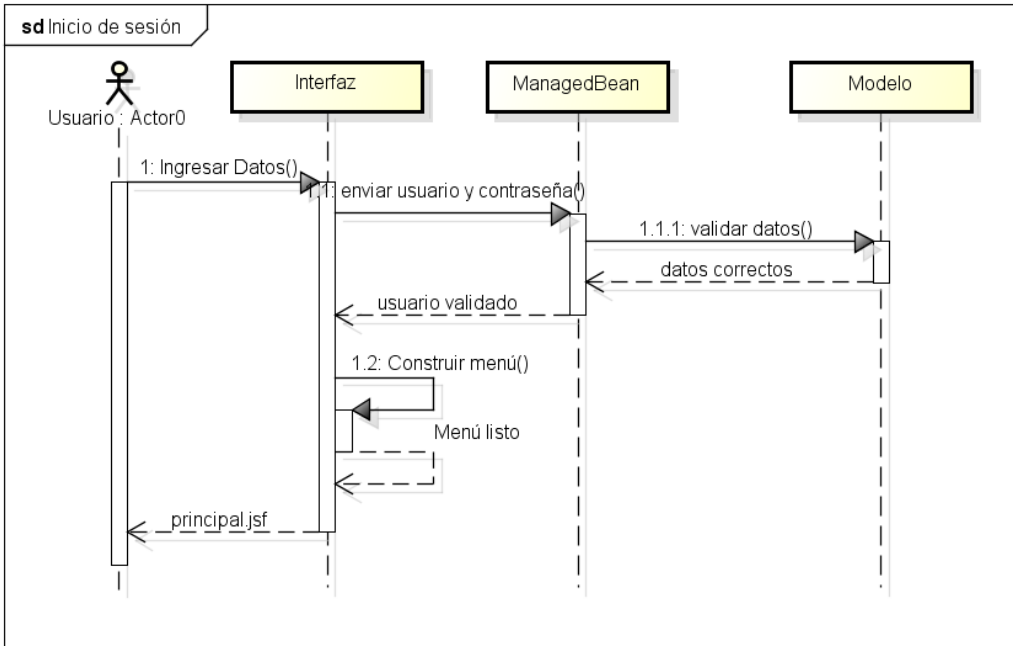


Figura 18. Proceso de inicio de sesión. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de secuencia

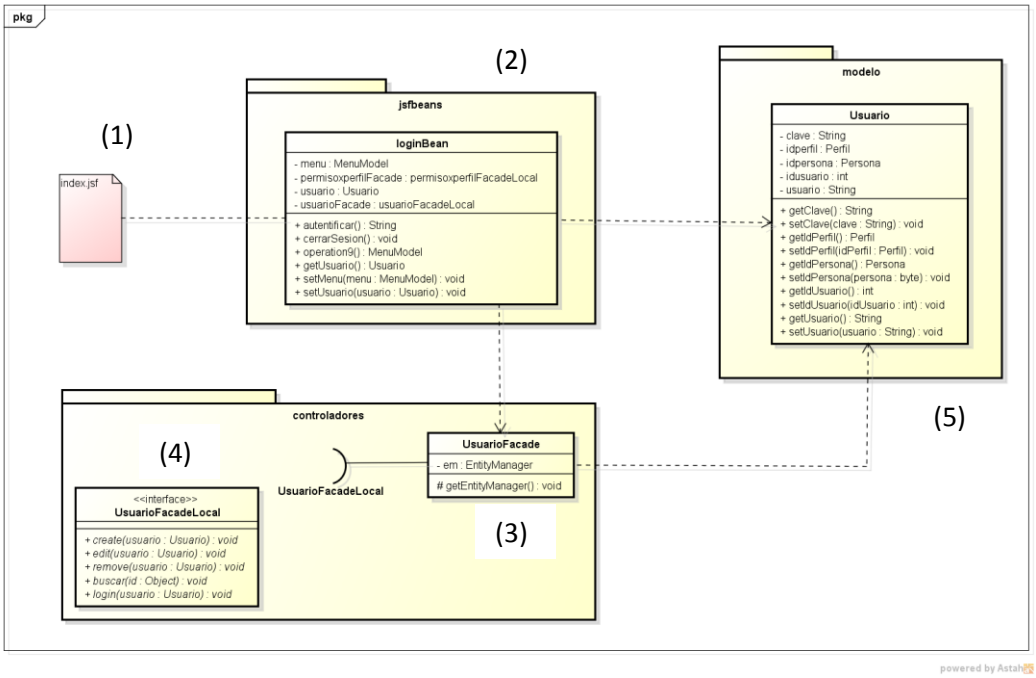


Figura 19. Ejemplo de diagrama de clases. Elaborado por: Myrian Guijarro & Jorge Jaramillo

(1) El usuario inicia desde la página index.jsf.

(2) La página jsf se comunica con la clase loginBean para realizar el proceso de autenticación.

(3) El bean se comunica con el controlador UsuarioFacade para validar la información ingresada por el usuario.

(4) El controlador implementa los métodos de su interfaz correspondiente, UsuarioFacadeLocal.

(5) La clase Usuario del paquete modelo contiene la representación de la tabla respectiva en la base de datos.

4.2.4 Vistas

La pantalla de inicio de sesión, Figura 20, muestra el formulario de usuario y contraseña.

Interfaz de usuario

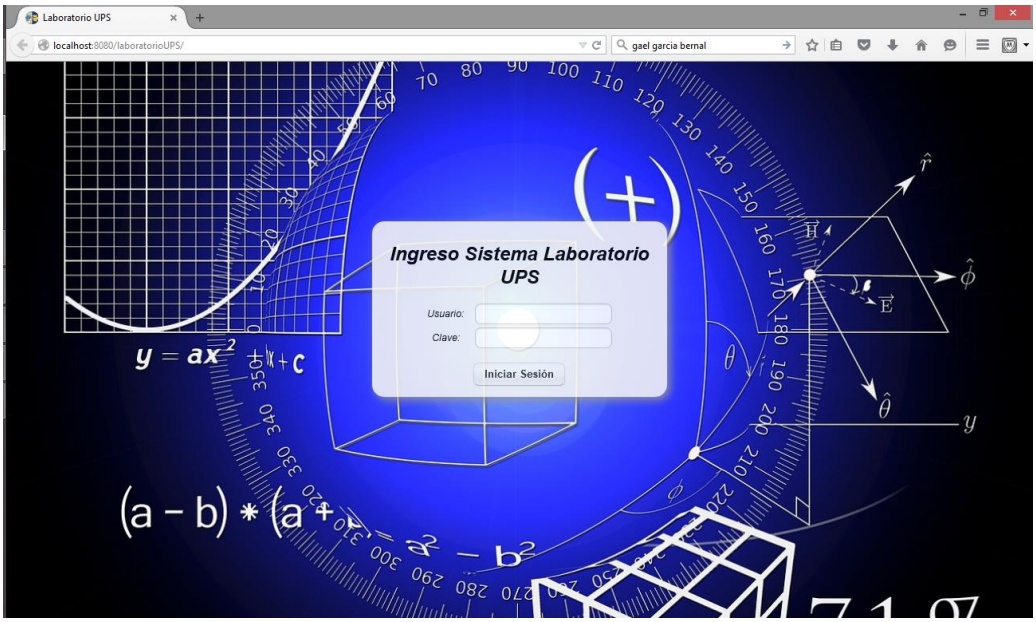


Figura 20. Pantalla de inicio de sesión. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Pantalla principal del sistema, Figura 21, contiene tres elementos principales: encabezado, menú y área de contenido. El encabezado será común a todas las páginas del sistema, el menú de cada usuario se muestra de acuerdo a su perfil y el área de contenido cambiará de acuerdo a la opción seleccionada en el menú por el usuario.

Interfaz de usuario

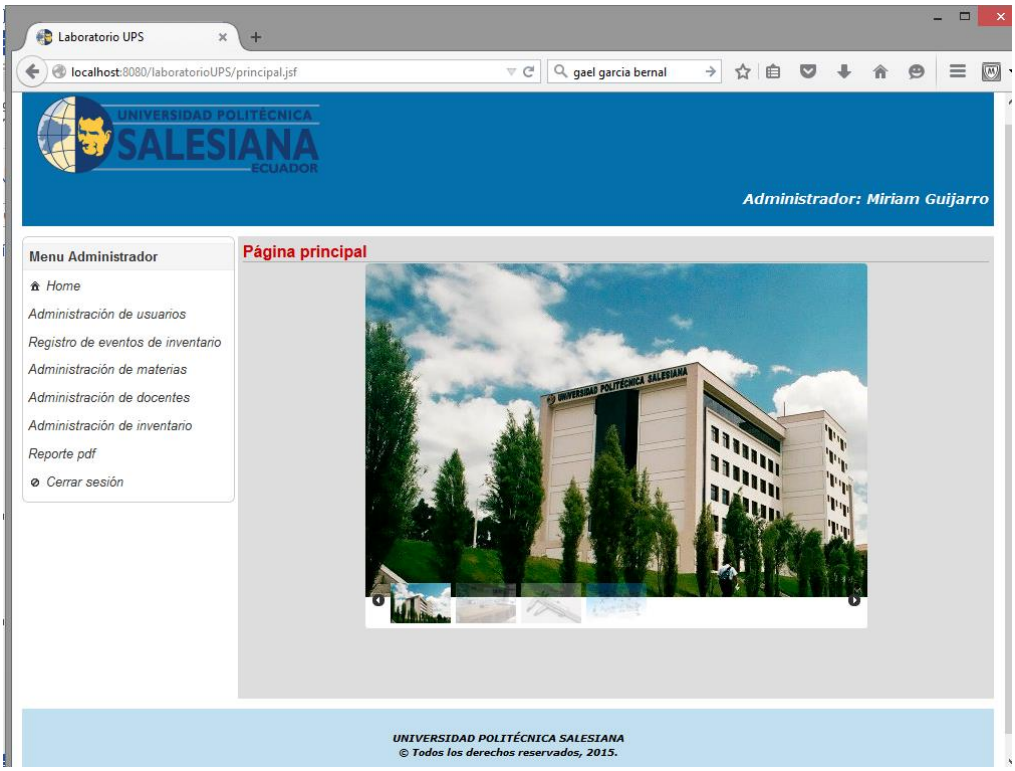


Figura 21. Pantalla principal. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.3. Segunda fase de construcción. Gestión administrativa

Se construye el módulo de gestión administrativa, en donde se incrementan funcionalidades al sistema para el control de las siguientes tareas:

- Administración de usuarios.
  - Crear usuarios.
  - Cargar estudiantes mediante archivo csv.
  - Mover estudiantes entre materias.
- Administración de inventario.
  - Administrar eventos de inventario (préstamo, pérdida, mantenimiento y devolución).
  - Administrar.

4.3.1 Tablas relacionadas a los objetivos de la fase

En la Tabla 29 se describen las tablas utilizadas para los procesos de Administración de usuarios con una breve descripción de los datos que se almacenan en ellas; las relaciones entre las tablas respectivas se pueden ver de manera gráfica en la Figura 22.

Diagrama de base de datos

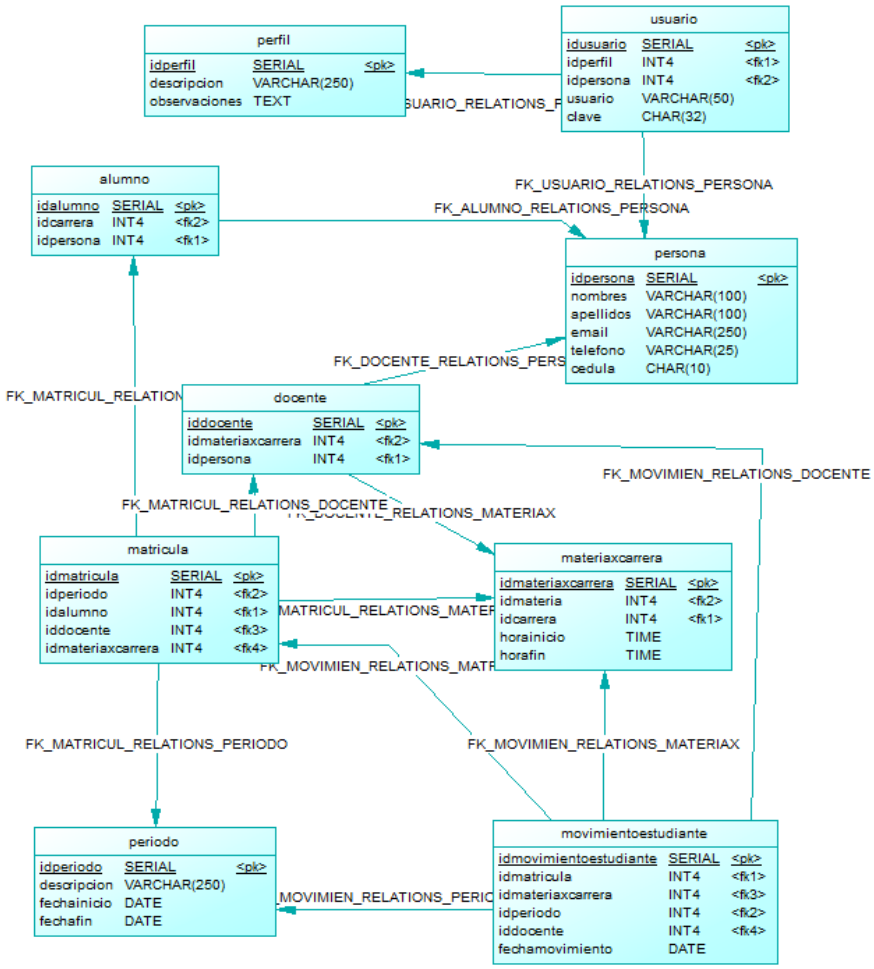


Figura 22. Tablas del módulo de gestión de usuarios. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Tabla 29. Tablas involucradas en administración de usuarios

Nombre de la tabla	Función
Perfil	Almacena datos del tipo de usuario
Usuario	Almacena los datos del usuario
Persona	Almacena datos de la persona
Alumno	Se crea un registro solo si el usuario que se está ingresando tiene el perfil alumno
Docente	Se crea un registro solo si el usuario que se está ingresando tiene perfil docente
materiaxcarrera	Almacena datos de las materias de laboratorio por cada carrera

Periodo	Almacena datos sobre el periodo académico actual
matricula	Almacena los datos relacionados al estudiante, la materia que toma, en qué periodo y quien es su docente
movimientoestudiante	Se crea un registro solo si se mueve a un estudiante. Almacena los datos de la matrícula original del estudiante que se mueve a otro curso, materia o nivel

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

La Tabla 30 muestra la descripción de los datos necesarios para la administración de inventarios y las tablas involucradas. Las respectivas relaciones se pueden ver en la Figura 23.

Tabla 30.Detalle de tablas involucradas en gestión de inventario

Nombre de la tabla	**Descripción**
Tipobien	Almacena los nombres de los tipos de bienes, identifica si es un mueble, equipo, material, etc
Estado	Almacena los nombres de los estados del bien. Ejemplo: activo, inactivo, obsoleto, etc
Bien	Almacena todos los datos del bien, material o equipo del inventario
persona/usuario	No almacenan ninguna información de la gestión de inventario, pero se relacionan con las tablas de los tipos de eventos del inventario: préstamo, pérdida, mantenimiento
Extraviado	Almacena los datos relacionados al bien cuando se lo reporta como extraviado. Se relaciona con la persona responsable de reponer el bien extraviado
Préstamo	Almacena los datos realacionados al bien cuando se lo presta a un usuario registrado o a una persona no registrada como usuario del sistema
Mantenimiento	Almacena los registros de cuando se realiza mantenimiento a los equipos del inventario

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de base de datos

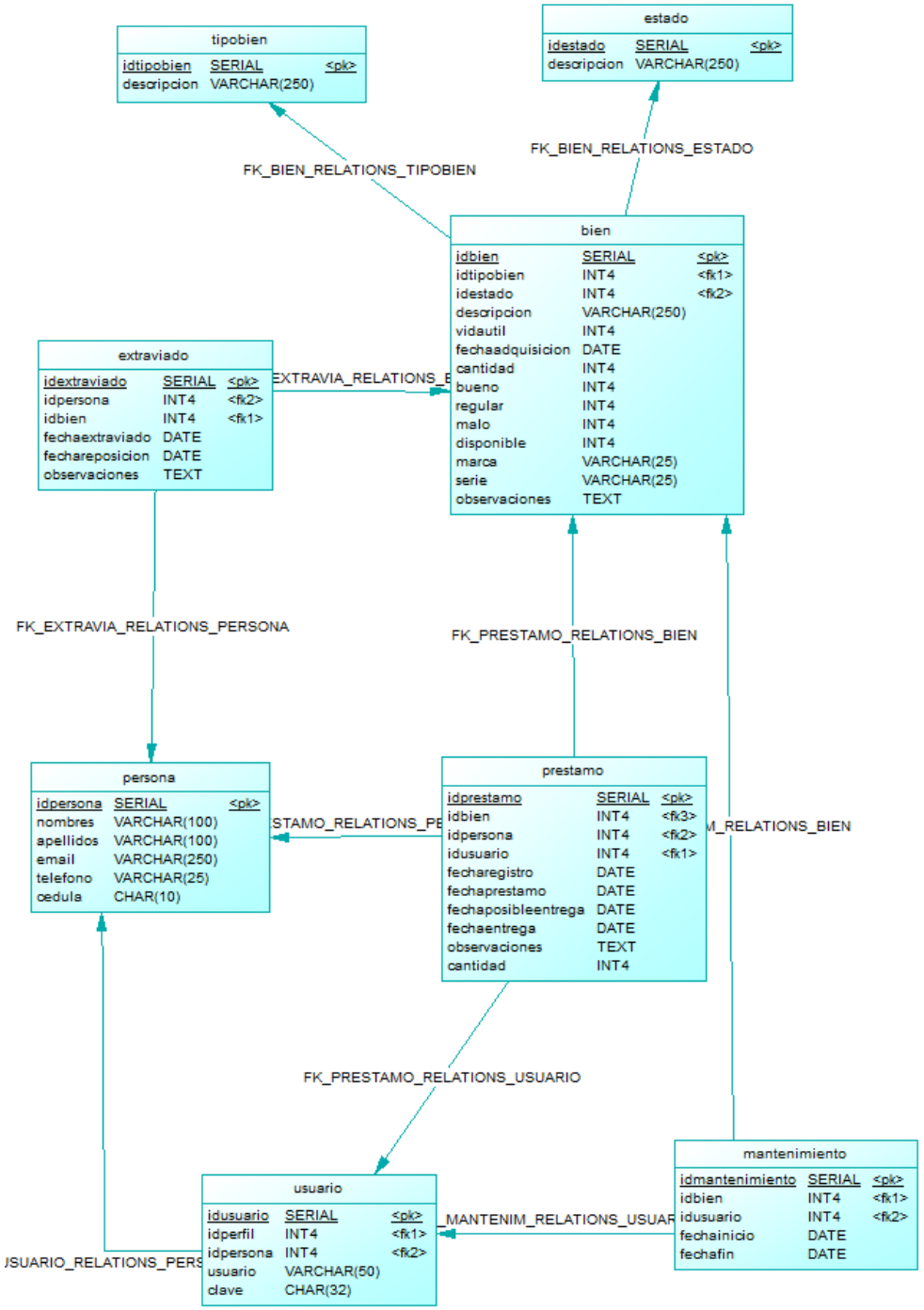


Figura 23. Tablas del módulo de gestión de inventario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.3.2 Diagramas de secuencia

Durante esta fase se añadieron dos grandes funcionalidades nuevas al sistema: la gestión de usuarios y la gestión de inventarios. A continuación se describen los procesos realizados en cada uno de estos módulos.

- Creación de usuario (Figura 24): el proceso de creación manual de usuarios inicia en la pantalla "Administración de usuarios". El actor presiona el botón "nuevo usuario" y el sistema muestra un diálogo con el formulario para ingresar los datos del nuevo usuario a crear. Una vez ingresados los datos necesarios, el actor envía la petición para guardar la información, el Managed Bean recibe los datos y se comunica con el controlador, que es en encargado de actualizar el modelo y la base de datos a través de la capa de persistencia.

Diagrama de secuencia

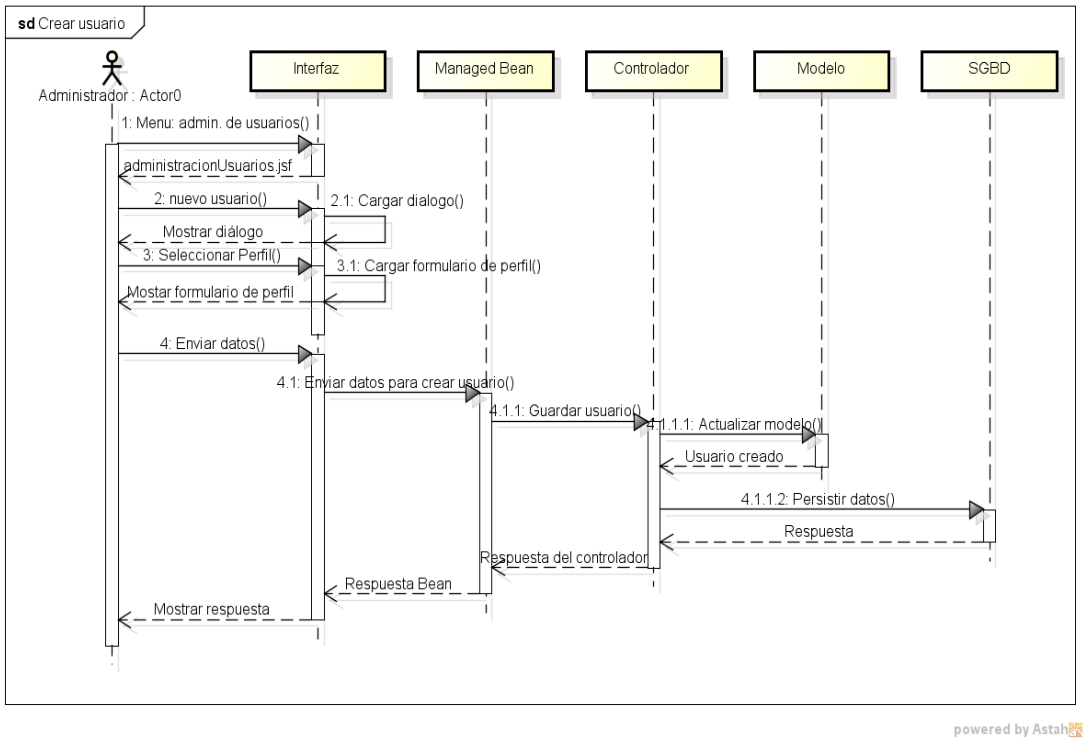


Figura 24.Creación de usuario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Cargar estudiantes con archivo (*Figura* 25): en la pantalla "Administración de usuarios", se presiona el botón "Cargar estudiantes desde CSV". El sistema muestra un diálogo para seleccionar el archivo que contiene la información necesaria para subir los alumnos. Se valida que el formato del archivo sea "csv".

El sistema lee línea por línea el archivo enviado y actualiza los datos en el modelo y la base de datos.

Diagrama de secuencia

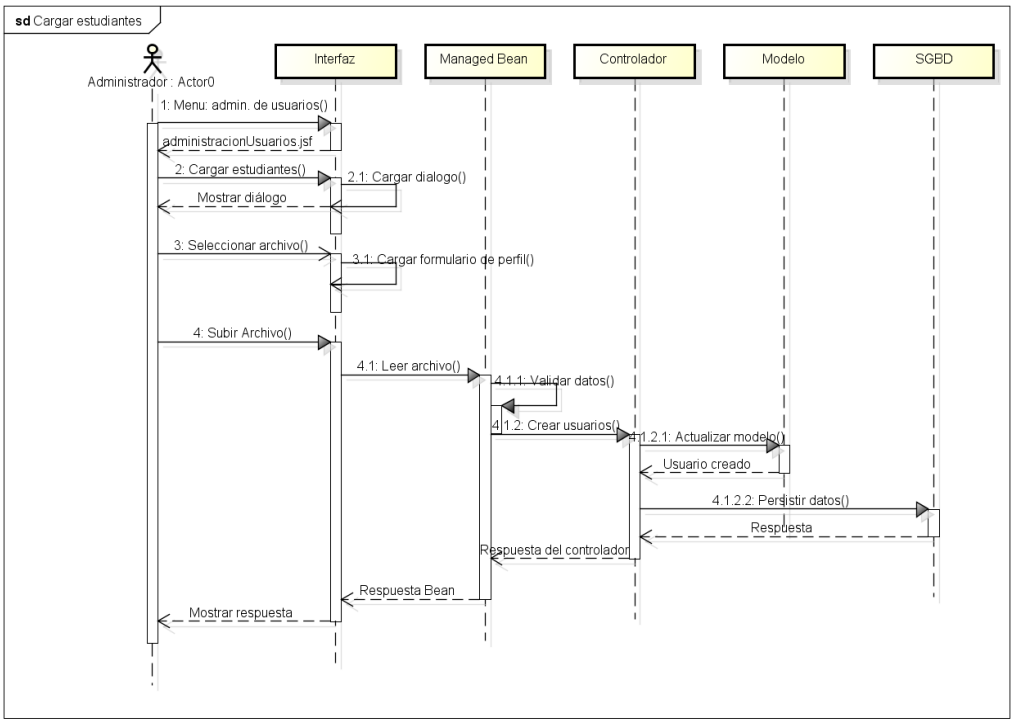


Figura 25. Cargar estudiantes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Mover estudiantes (Figura 26): el administrador puede mover a un estudiante a otro curso, nivel o materia de laboratorio; sin perder la información original de donde estaba inscrito el estudiante. Para realizar el "movimiento" del estudiante, se ingresa a la pantalla de administración de usuarios y se selecciona el botón "mover estudiante". El sistema direcciona hacia la pantalla "moverEstudiante.jsf". En esta pantalla se escoge un estudiante y el sistema busca sus datos en el modelo, una vez que los encuentra, muestra el diálogo al usuario para que complete la información.

  Según los datos seleccionados, el diálogo se actualiza dos veces para mostrar las materias, grupos y docentes disponibles para mover al estudiante. Una vez que se completa el formulario, el usuario envía la solicitud de guardar al bean que inicia el proceso de guardado.

Diagrama de secuencia

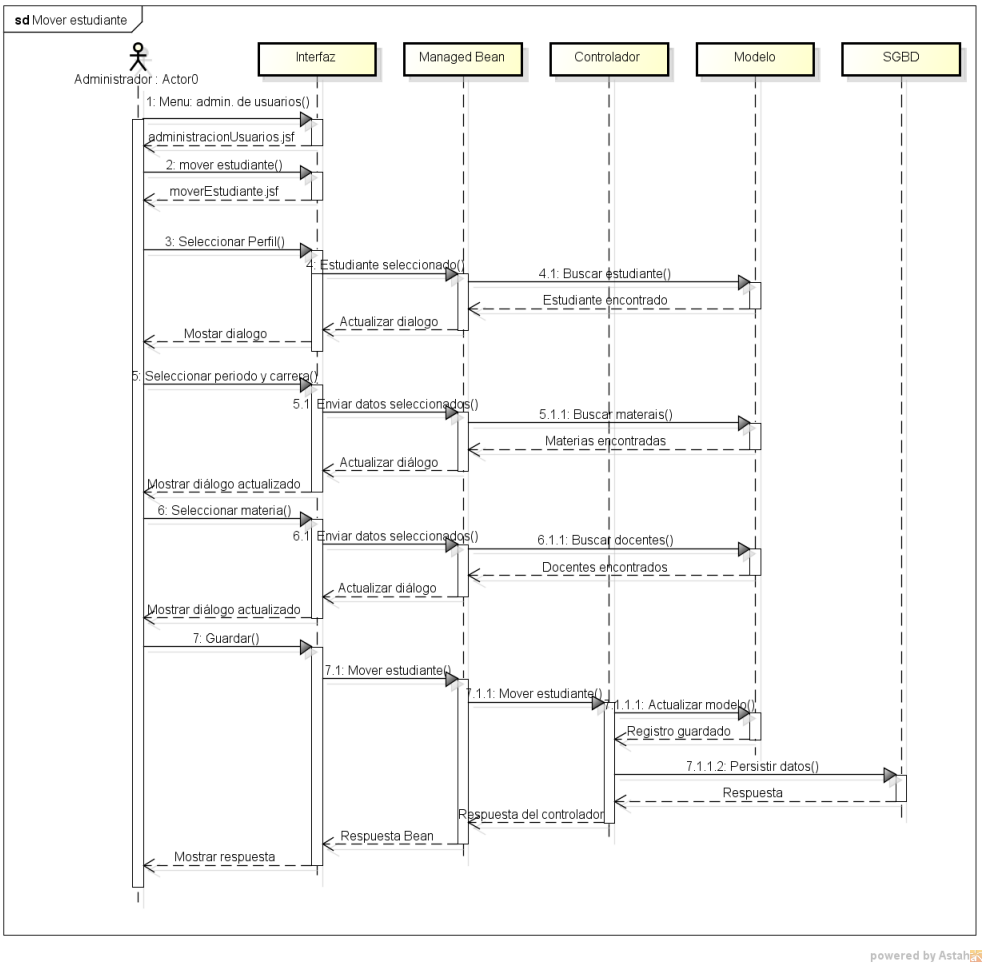


Figura 26. Mover estudiantes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Administración de inventario (Figura 27): el usuario ingresa al menú de administración de inventario, el sistema envía al bean respectivo la solicitud para recuperar todos los ítems del inventario. Al encontrarlos se envía la respuesta y se actualiza la interfaz mostrando la tabla que contiene todos los elementos encontrados.

  Al seleccionar cualquier ítem, se envía una solicitud para recuperar información sobre el estado del ítem y esta información se despliega mediante un mensaje sobre la tabla.

  Al presionar el botón para modificar un bien, el sistema recupera del modelo toda la información del bien actual y la muestra en un formulario dentro de un diálogo, para que el usuario edite la información que necesite, en especial las

cantidades que indican el estado de los ítems del inventario (bueno, regulare y malo).

Una vez editada la información, el usuario presiona el botón de guardar y se inicia el proceso de actualización del modelo y guardado de los datos, finalizando con un mensaje en pantalla de los datos actualizados.

Diagrama de secuencia

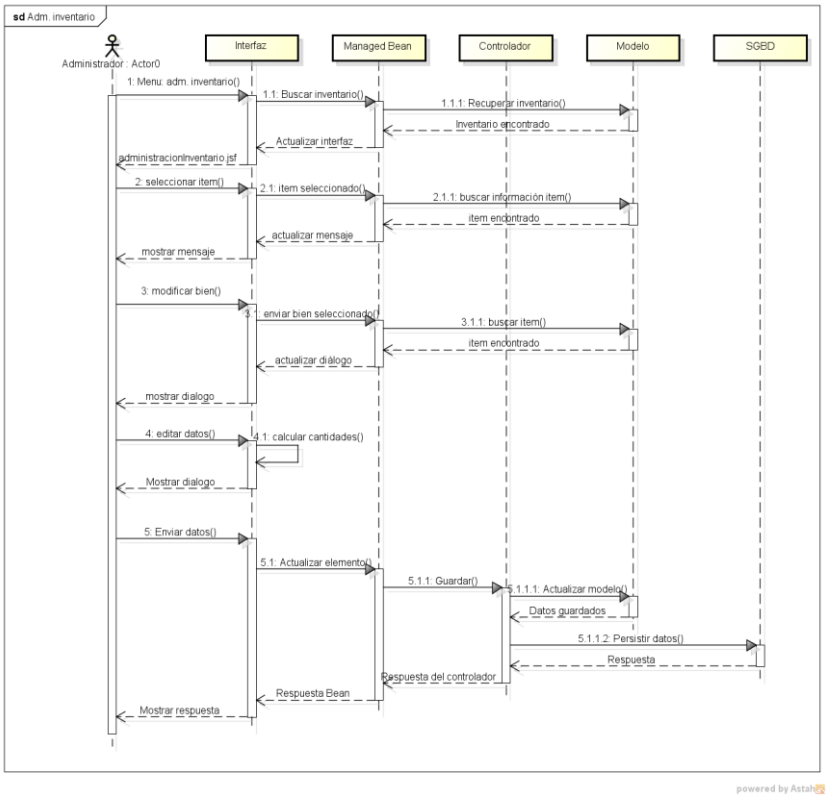


Figura 27. Administración de inventario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Administrar eventos de inventario: los eventos de inventario corresponden a cinco actividades; creación, préstamo, pérdida, devolución y mantenimiento de los ítems o bienes del inventario.

  Los eventos de creación (Figura 28) y mantenimiento (Figura 29) se manejan de manera similar en el sistema. El usuario ingresa a la pantalla de administración de eventos de inventario, el sistema busca todos los ítems registrados, actualiza el componente correspondiente y muestra la pantalla "administracionBienes.jsf". Tanto si el usuario presiona el botón "nuevo bien" como si presiona el botón "mantenimiento", aparece un diálogo que solicita la información necesaria para registrar el evento. El usuario completa la información y envía la solicitud de

guardado al Bean respectivo. Al finalizar la actualización del modelo y la base, se muestra un mensaje de confirmación.

Diagrama de secuencia

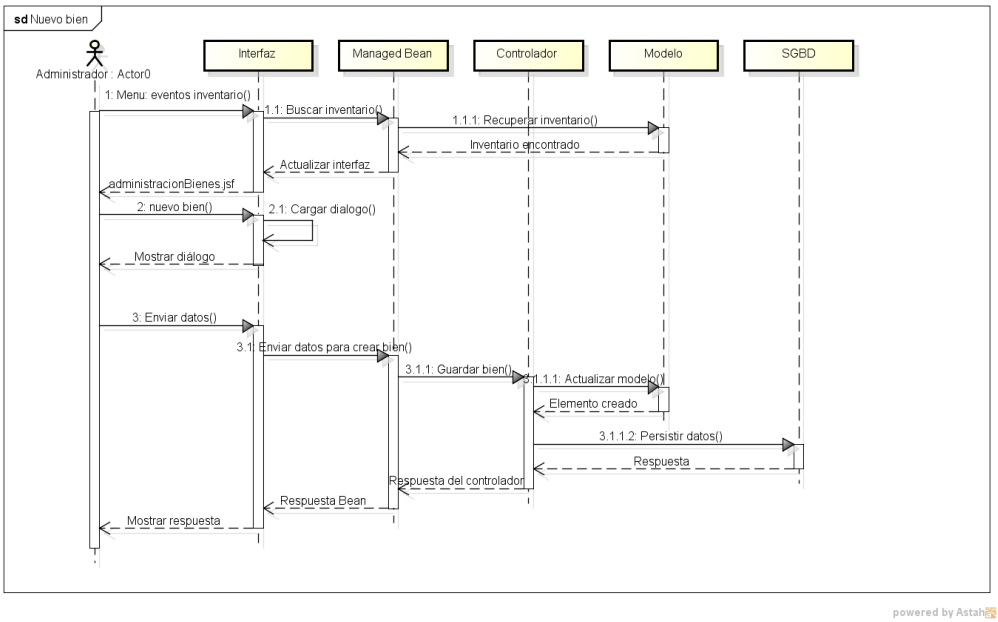


Figura 28. Nuevo bien de inventario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de secuencia

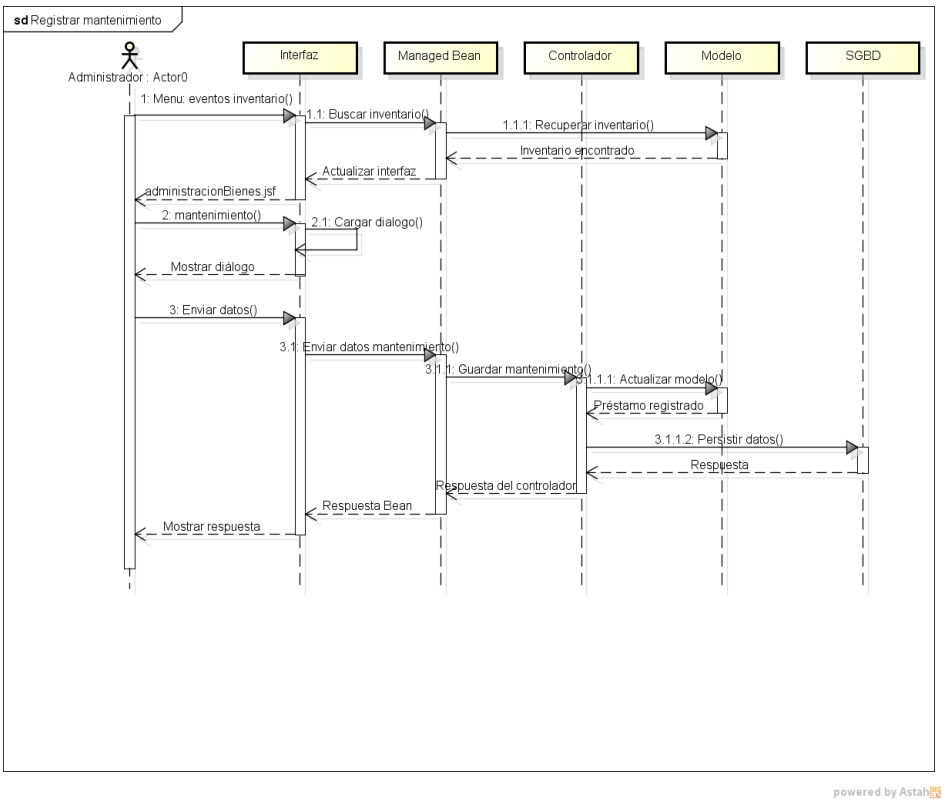


Figura 29. Registrar mantenimiento. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Los eventos préstamo, pérdida y devolución (Figura 30), funcionan de la misma manera por lo que se los puede explicar en un solo diagrama de secuencia. El usuario ingresa a la pantalla de administración de eventos la que se actualiza desde el modelo respectivo y muestra todos los ítems del inventario.

  El usuario presiona el botón de cualquiera de los eventos mencionados y el sistema envía una solicitud al Bean para recuperar los datos del ítem o bien seleccionado y mostrarlos en pantalla mediante un diálogo.

  En el diálogo se ingresa la cédula de la persona responsable sobre el evento del ítem, el Bean busca en el modelo los datos de la persona para mostrarlos en pantalla si no se encuentran los datos, el usuario debe ingresarlos manualmente. Una vez ingresada la información, se inicia el proceso de guardado del modelo y la base de datos.

Diagrama de secuencia

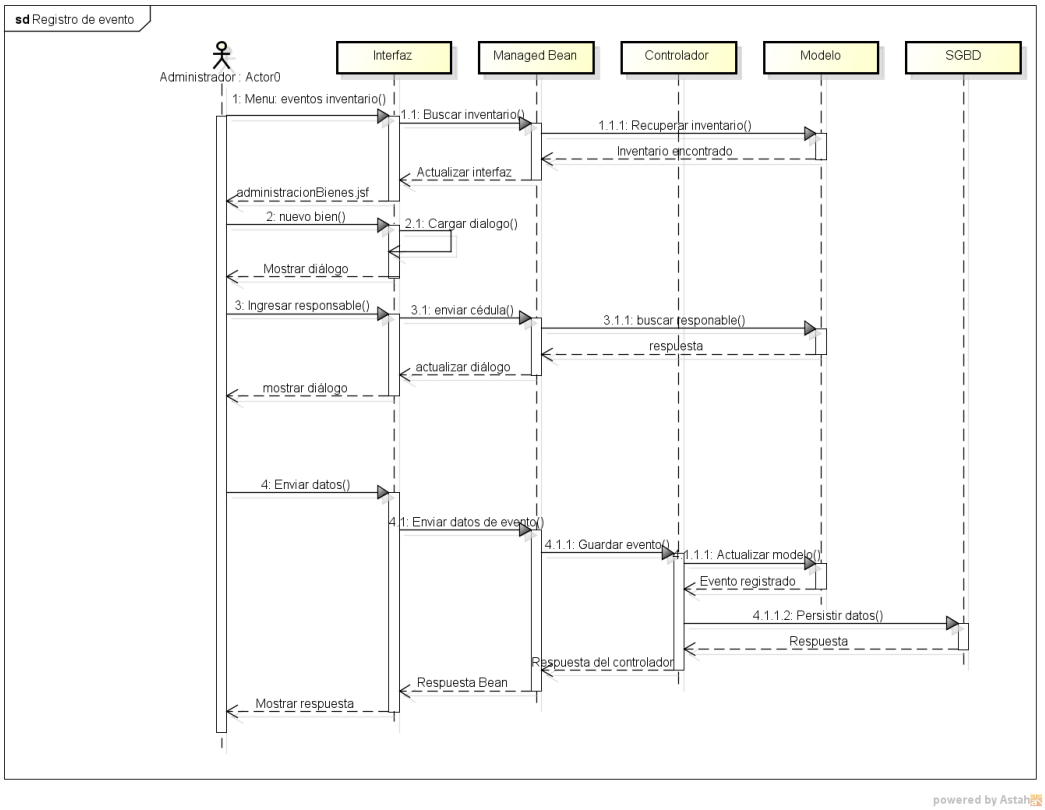


Figura 30. Registro de eventos de inventario.Elaborado por Myrian Guijarro & Jorge Jaramillo

4.3.3 Diagramas de clases

El proceso de gestión de usuarios se maneja desde dos páginas JSF a las que tiene acceso el perfil de Administrador. La Figura 31 muestra el esquema general de las clases que intervienen en el proceso de gestión de usuarios. Existe solo un controlador ya que todas las funciones relacionadas se administran desde la clase Usuario.

Diagrama de clases

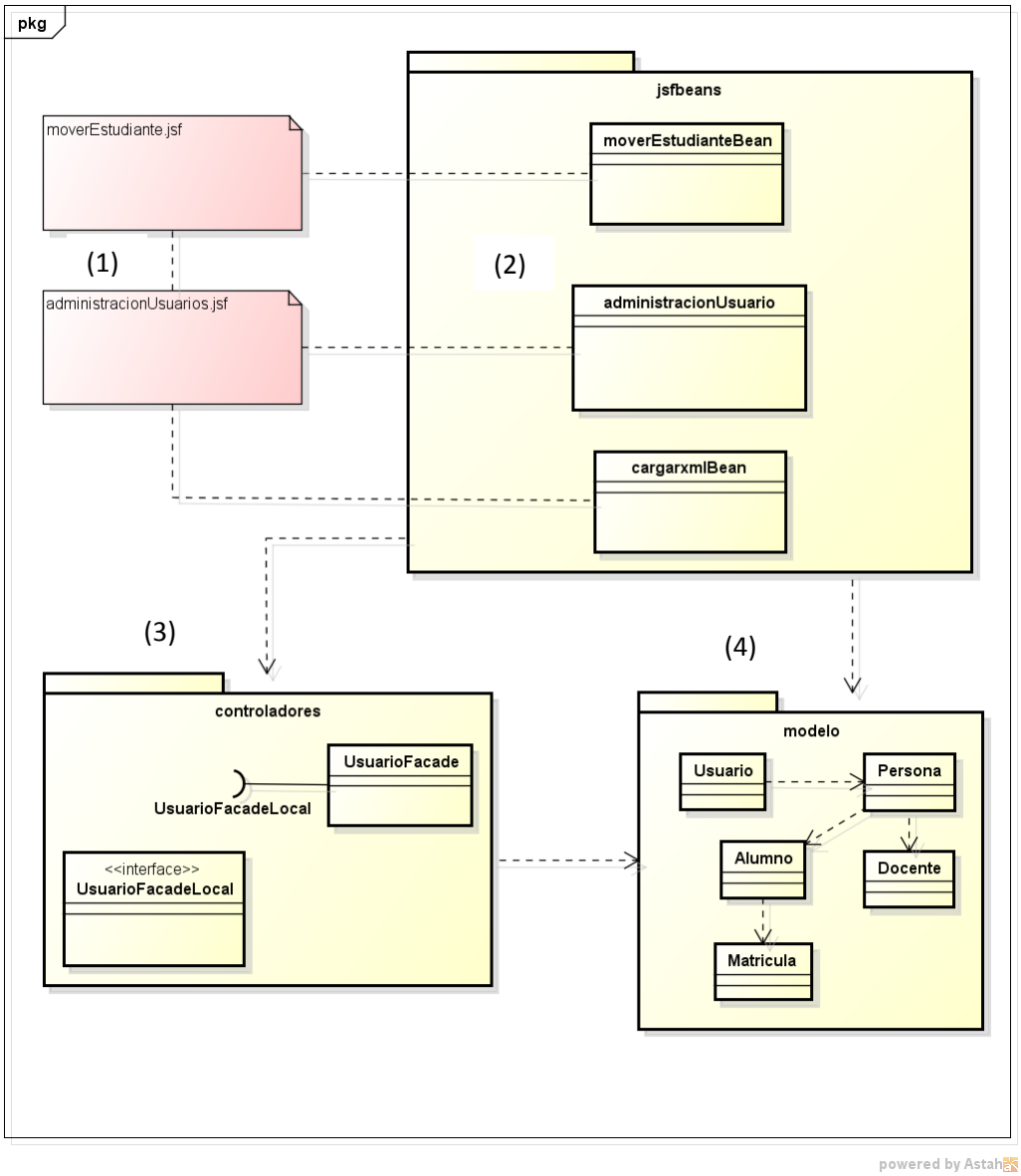


Figura 31. Administración de usuarios. Elaborado por: Myrian Guijarro & Jorge Jaramillo

(1) Las páginas administracionUsuarios.jsf y moverEstudiante.jsf son las que engloban las funciones de la gestión de usuarios.

(2) Existen 3 beans que se comunican con las páginas JSF, de los cuales la página moverEstudiante.jsf tiene asociado uno propio, los otros dos beans son parte de la funcionalidad de la página administracionUsuarios.jsf.

(3) El controlador UsuarioFacade es el encargado de la gestión de las funciones centrales de las páginas de administración de usuarios.

En cuanto al modelo, las clases principales que intervienen en este módulo son: usuario, persona, alumno, docente y matrícula.En cuanto a la gestión de inventario, los beans de las páginas JSF necesarias para la gestión de bienes y eventos del inventario se relacionan con el controlador BienFacade que es el encargado de comunicarse con las demás clases del sistema para recuperar los diferentes datos sobre un ítem registrado en el inventario.

Diagrama de clases

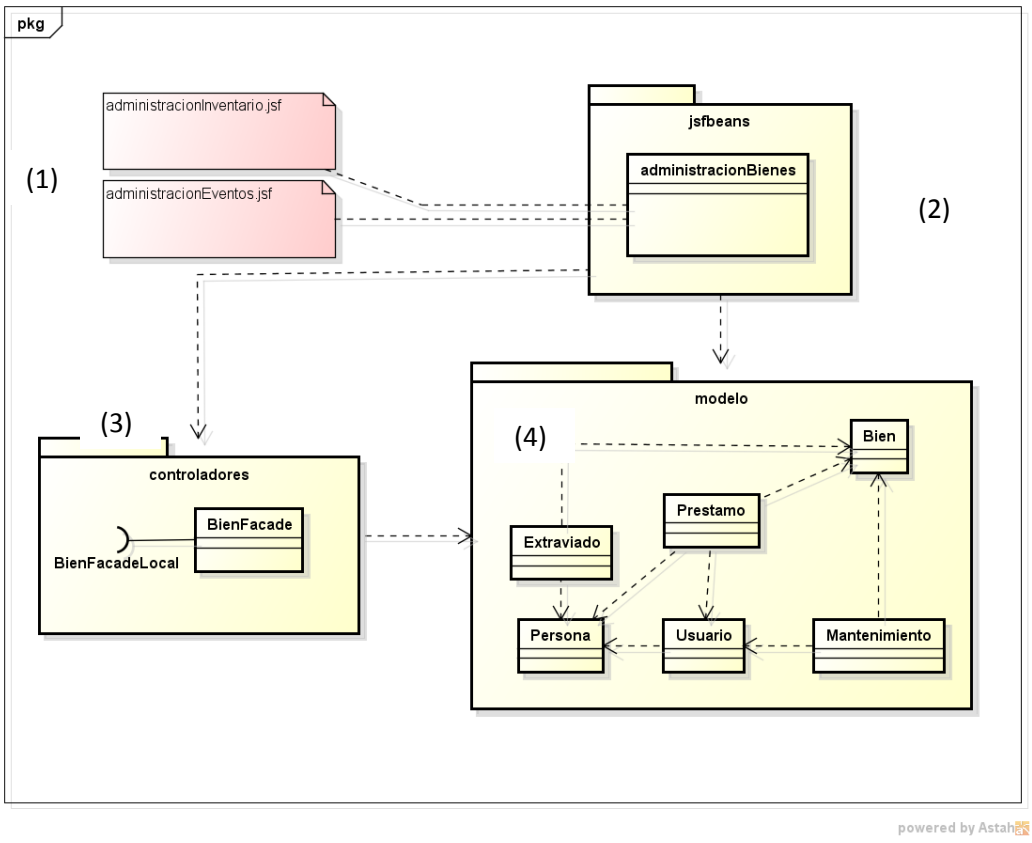


Figura 32. Administración de inventario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

La Figura 32 muestra cómo interactúan las clases que intervienen en los procesos de gestión de bienes y eventos de inventario junto con las páginas JSF que representan la interfaz con el usuario.

(1) La gestión de inventario se maneja desde dos páginas JSF: administracionInventario.jsf y administracionEventos.jsf.
(2) El bean administracionBienes recibe las peticiones de las dos páginas.
(3) El controlador BienFacade valida los datos enviados por el bean y se comunica con la capa de persistencia.
(4) En modelo, las clases principales que representan a sus equivalentes en la base de datos son: bien, préstamo, extraviado, mantenimiento, persona y usuario.

4.3.4 Vistas

Pantalla de Administración de usuarios (Figura 33), muestra el dialogo para creación de nuevo usuario.

Interfaz de usuario

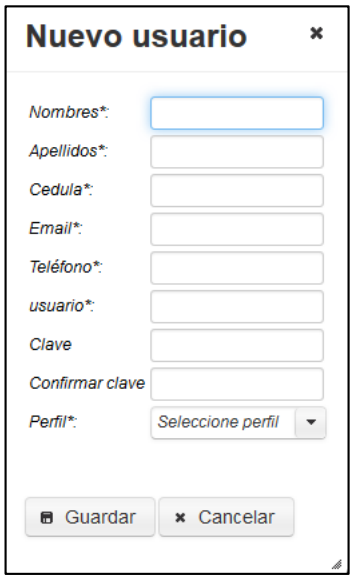


Figura 33. Diálogo de creación de usuarios. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Pantalla de administración de usuarios (Figura 34), muestra el diálogo para cargar archivo csv que contiene los datos para ingresar estudiantes en lote. Si el formato del archivo no corresponde a un csv se muestra un mensaje de error (Figura 35).

Interfaz de usuario

Figura 34. Cargar archivo .csv. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Interfaz de usuario

Figura 35. Error de formato de archivo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Pantalla para mover estudiante (Figura 36), muestra el diálogo para mover la matrícula del estudiante a otra carrera, materia o docente.

Interfaz de usuario

Figura 36. Mover estudiante. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

Administración de inventario (Figura 37): se muestra la pantalla principal para la gestión de elementos de inventario.

Interfaz de usuario

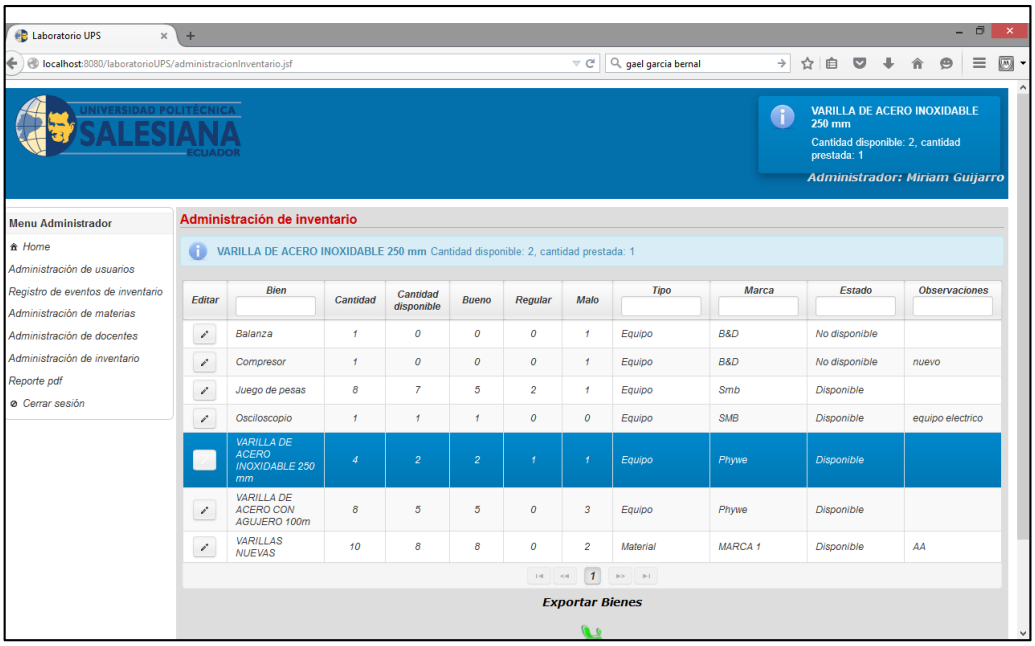


Figura 37. Administración de inventario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Administración de eventos de inventario (Figura 38): pantalla principal desde donde se gestionan los eventos sobre los elementos del inventario.

Interfaz de usuario

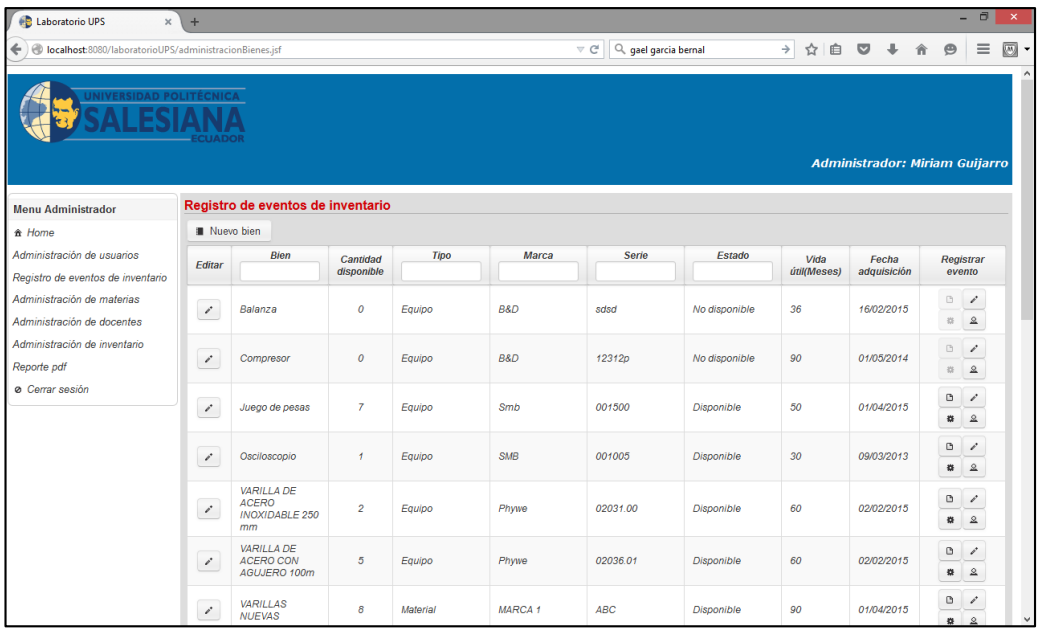


Figura 38. Administración de eventos. Elaborado por Myrian Guijarro & Jorge Jaramillo

4.4. Tercera fase de construcción. Gestión académica para el docente

Al concluir la tercera fase de construcción, el sistema cuenta con las funciones necesarias para las actividades del Docente dentro del módulo de gestión académica. Estas actividades incluyen lo siguiente:

- Crear grupos de trabajo para las materias de laboratorio de física.
- Asignar un líder de grupo entre los estudiantes de cada grupo de trabajo.
- Crear prácticas de laboratorio, con una descripción y el material de apoyo necesario para su desarrollo.
- Crear cuestionarios sobre los temas de las prácticas de laboratorio.
- Crear evaluaciones (coloquio, defensa, informe), para poder asignarlas a grupos de estudiantes.
- Visualizar y calificar informes de laboratorio cargados al sistema por los estudiantes.

4.4.1 Tablas relacionadas a los objetivos de la fase

La Tabla 31 describe de manera breve la función de cada una de las tablas involucradas con módulo de gestión académica y las actividades del docente.

Tabla 31.Tablas relacionadas con la gestión académica (perfil Docente)

Nombre de la tabla	Descripción
tipoevaluacion	Almacena los nombres de los tipos de evaluaciones, identifica si es coloquio, defensa, informe
respuesta	Almacena las respuestas ingresadas por pregunta
evaluación	Almacena todos los datos de las evaluaciones creadas
pregunta	Almacena los datos de las preguntas por práctica
nivelpregunta	Almacena el nivel de complejidad de las preguntas: básica, media, avanzada
materialdeapoyo	Almacena los datos del material de apoyo
practica	Almacena los datos de la practica
alumnoxinforme	Almacena la nota final del informe de cada alumno del grupo, luego de haberse promediado la nota del líder, la nota del docente
practicaxmateria	Almacena los datos de las practicas por materia
informe	Almacena los datos del informe, subido por el líder de

	cada grupo
alumnoxgrupo	Almacena los datos de los alumnos en cada grupo
materia	Almacena los datos de las materias
grupo	Almacena los datos de los grupos, por carrera, materia
docente	Almacena los datos del docente
laboratorio	Almacena los datos del laboratorio en relación al lugar donde se desarrollan las prácticas.

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de base de datos

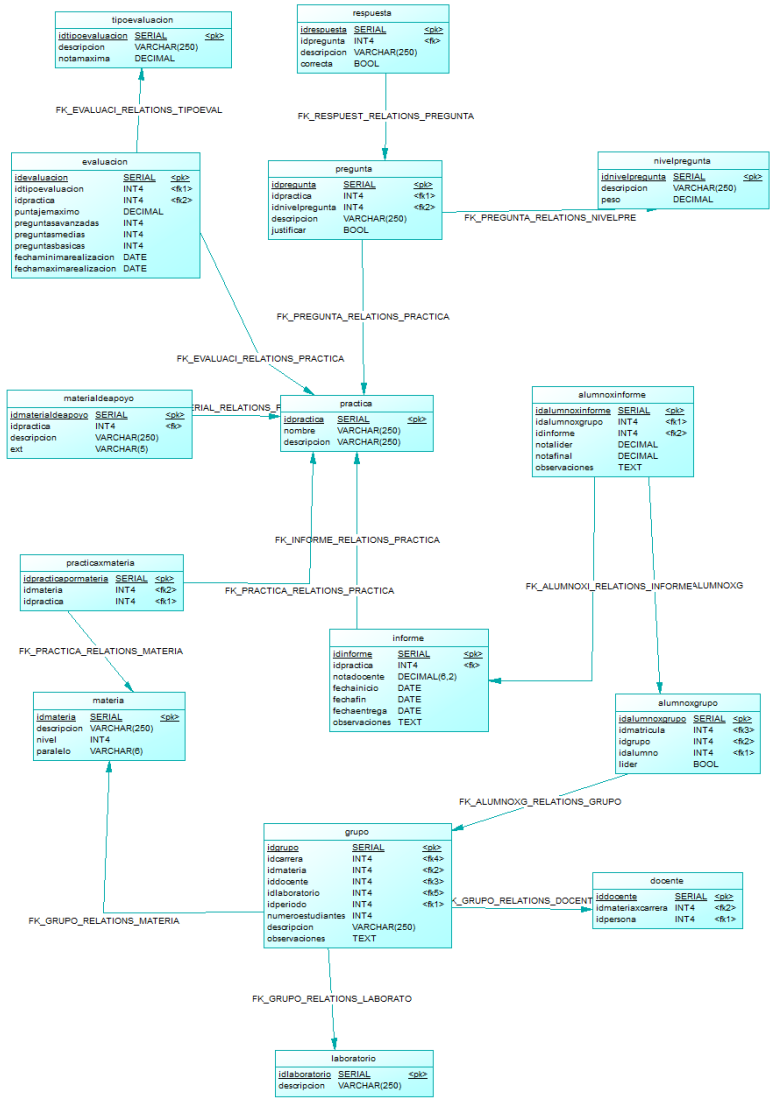


Figura 39. Tablas involucradas en objetivos de la fase 3. Elaborado por: Myrian Guijarro & Jorge Jaramillo

En la Figura 39 se muestran las tablas principales que se involucran en los procesos del módulo de gestión académica, tomando en cuenta que las funciones añadidas en esta fase tienen que ver exclusivamente con el perfil docente.

4.4.2 Diagramas de Secuencia

Las funcionalidades añadidas al sistema al finalizar la fase 3 de desarrollo se detallan a continuación, junto con los diagramas de secuencia respectivos.

- Crear grupos de laboratorio (Figura 40): el usuario ingresa al menú de creación de grupos. El sistema envía las credenciales del usuario actual al Bean respectivo para recuperar en el modelo las materias que dicta el docente. Se actualiza la vista y se muestra al usuario. El usuario ingresa los datos en el formulario y envía la petición para iniciar el proceso de guardar el grupo nuevo.

Diagrama de secuencia

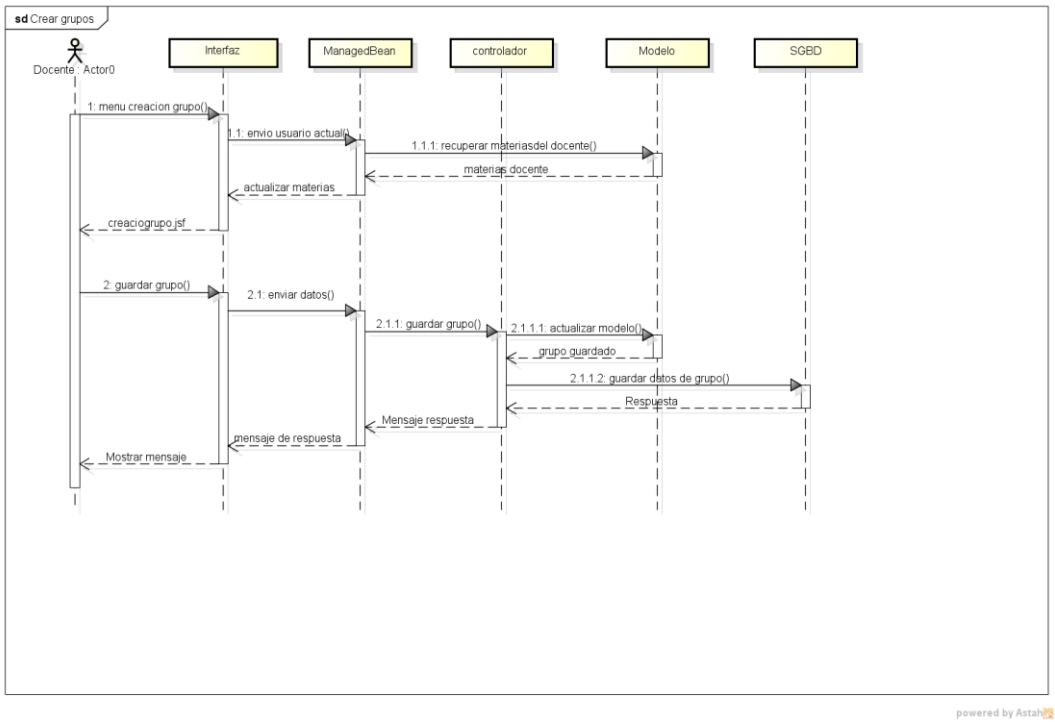


Figura 40. Crear grupo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Asignar líder de grupo (Figura 41): el usuario ingresa en el menú asignar líder, el sistema envía una solicitud al Bean para recuperar desde el modelo los datos de los líderes ya asignados. Se actualiza y se muestra el modelo.

El docente selecciona la materia y el periodo para visualizar los grupos de trabajo registrados en la materia y periodo actual. Luego selecciona el grupo para visualizar el listado de alumnos registrados en el grupo. Estos dos procesos se realizan solicitando al Bean respectivo que busque en el modelo la información requerida antes de actualizar la interfaz con los datos recuperados.

Finalmente, el docente presiona el botón guardar para iniciar el proceso de actualización en el modelo y la base de datos gracias al controlador.

Diagrama de secuencia

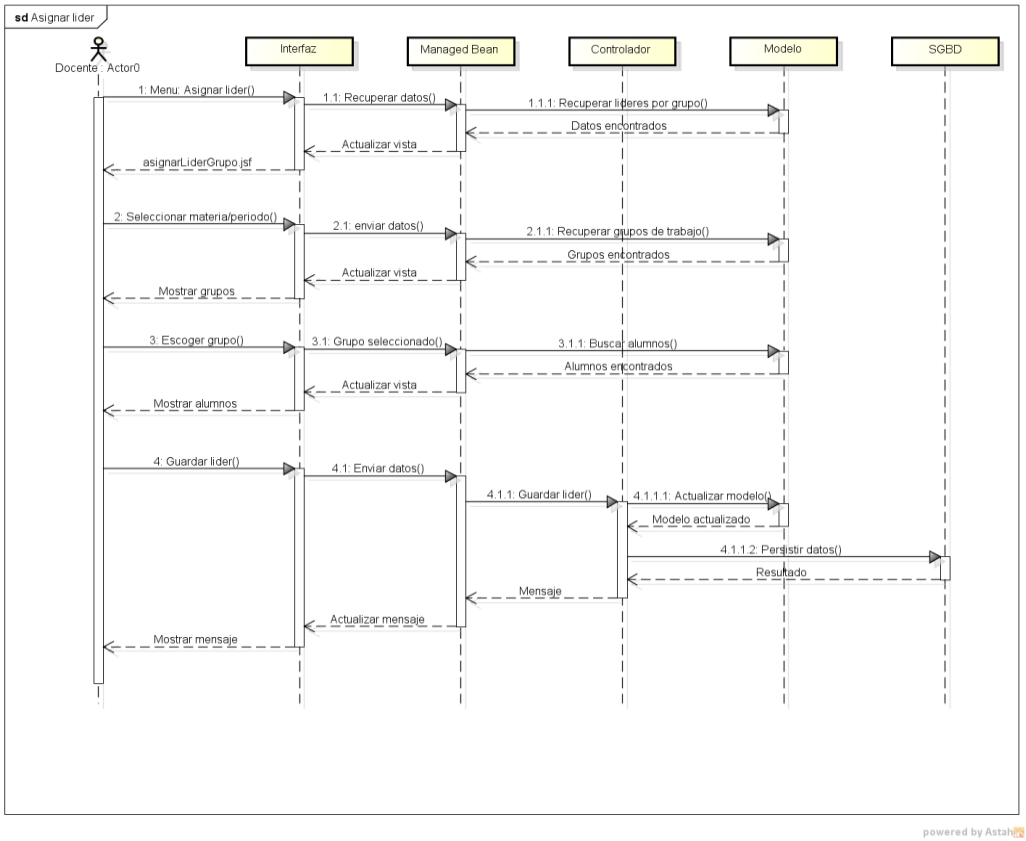


Figura 41. Asignar líder de grupo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Crear prácticas de laboratorio (Figura 42): el usuario ingresa al menú prácticas de laboratorio, el sistema envía una solicitud al Bean respectivo para recuperar el listado de prácticas existentes. Se actualiza la vista y se muestra al usuario.

  El usuario llena el formulario respectivo e inicia el proceso de guardado en el modelo y base de datos.

Diagrama de secuencia

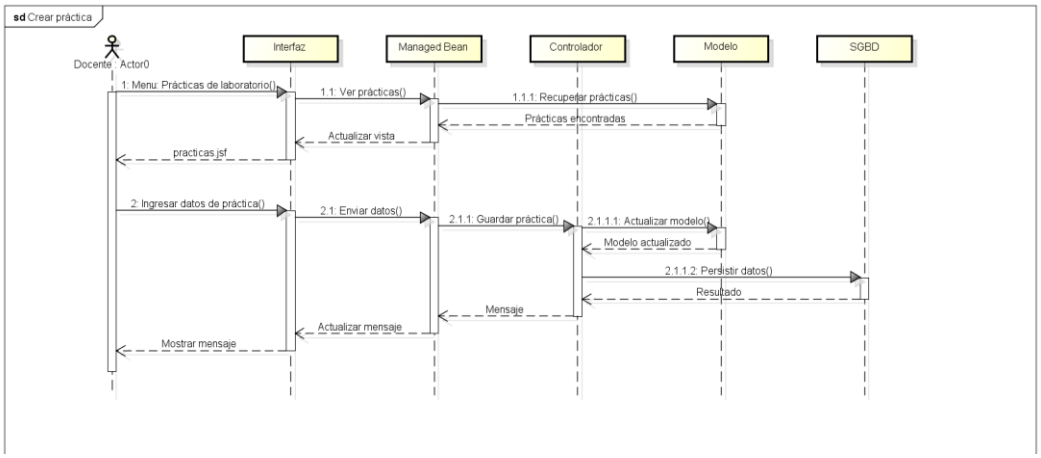


Figura 42. Crear práctica. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Crear cuestionarios de las prácticas (Figura 43): el usuario ingresa al menú registro de preguntas, el sistema realiza el proceso de recuperación de datos del modelo para mostrar en pantalla las preguntas creadas con anterioridad.

  El usuario selecciona la práctica donde se van a crear las preguntas y el sistema muestra un mensaje informativo de confirmación sobre la práctica seleccionada. El registro de preguntas trabaja como un formulario normal, se ingresan los datos de la práctica y se envía la petición de guardado para iniciar el proceso de actualización.

- Asignar evaluaciones: coloquio, defensa e informe (Figura 44): el usuario ingresa al menú asignar evaluación. El sistema solicita al Bean respectivo los datos para mostrar los filtros en la interfaz. El usuario selecciona el tipo de evaluación que quiere asignar, nuevamente, el Bean se encarga de consultar al modelo el valor de la nota máxima para el tipo de evaluación seleccionado y actualiza el campo.

  Al seleccionar materia y periodo se actualizan los grupos disponibles y según el grupo seleccionado, se actualizan los estudiantes del grupo. El usuario selecciona los estudiantes a quienes se asignará la evaluación y envía la solicitud para guardar los datos, iniciando el proceso en el modelo y tablas respetivas de la base de datos.

Diagrama de secuencia

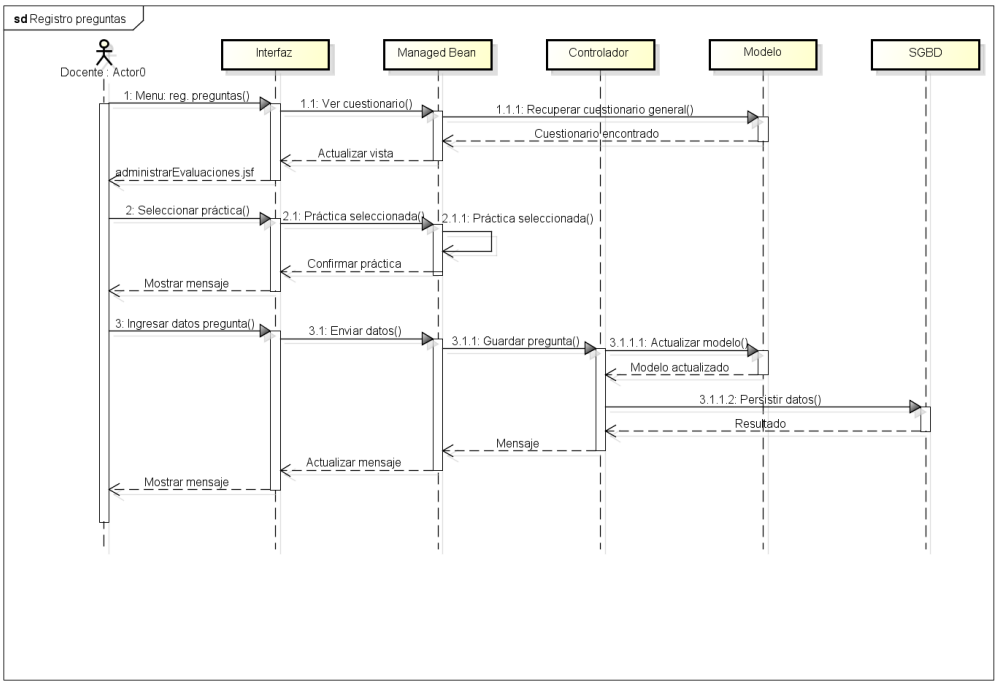


Figura 43. Crear cuestionario. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de secuencia

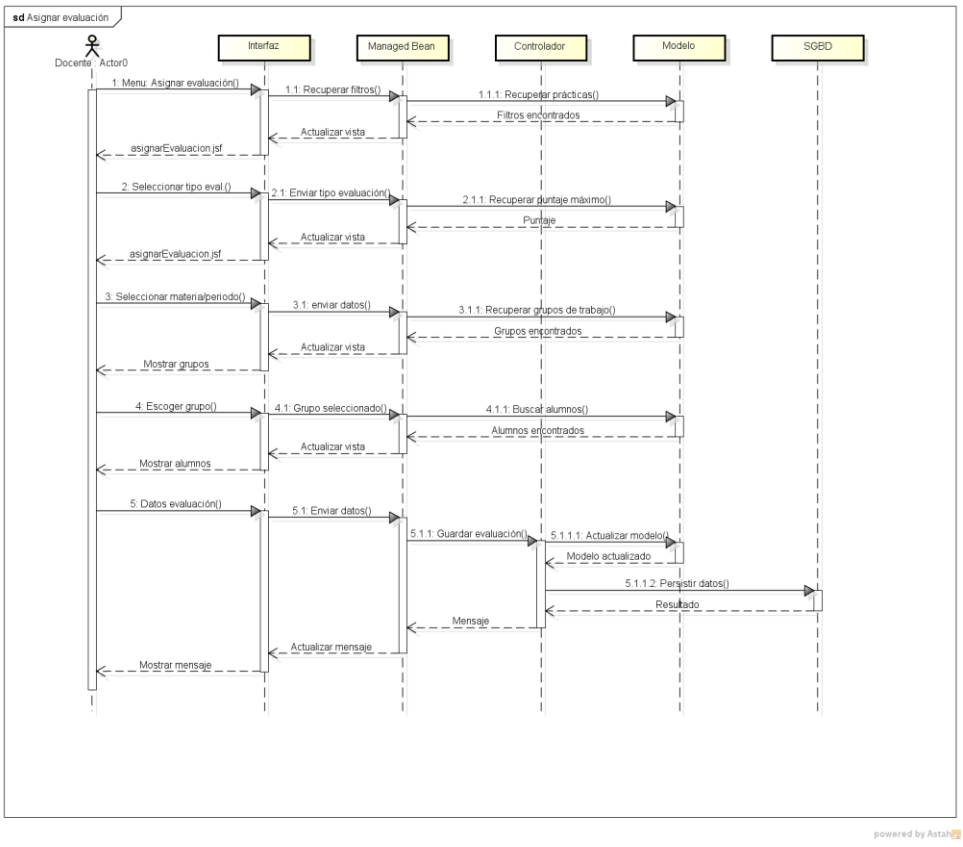


Figura 44. Asignar evaluación. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Lista de informes de laboratorio (Figura 45): el usuario ingresa al menú lista de informes, el sistema solicita al modelo el listado de informes asignados y actualiza la vista "verInformes.jsf". El usuario pude realizar dos tareas en esta pantalla: ver informe y calificar informe.

  Al seleccionar ver informe, el Bean recibe los datos del informe seleccionado y busca en el modelo el nombre del archivo para actualizar la vista y mostrar el archivo.

  La opción calificar informe, abre un diálogo para ingresar la nota del informe seleccionado. Una vez ingresados los campos requeridos, el usuario debe iniciar el proceso para guardar la información.
- Registro de alumnos a grupos de trabajo (Figura 46): el usuario con el perfil Alumno puede seleccionar su grupo de trabajo de laboratorio entre los grupos disponibles de la materia del docente. Para hacerlo, ingresa en el menú registrarse en un grupo donde se envía al Bean una solicitud para buscar los grupos disponibles en la materia del alumno y actualizar la vista. Cada vez que el usuario escoge un grupo, el bean busca en el modelo los alumnos inscritos en el grupo seleccionado y actualiza la interfaz para mostrar los datos al estudiante. Luego el estudiante debe enviar iniciar el proceso de guardado de los datos.

Diagrama de secuencia

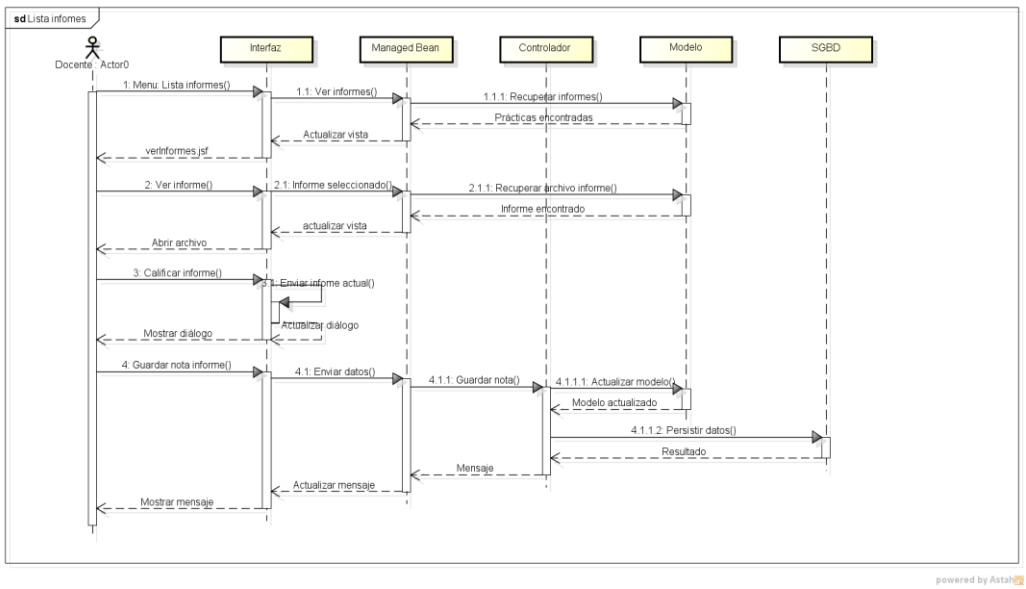


Figura 45. Ver lista de informes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de secuencia

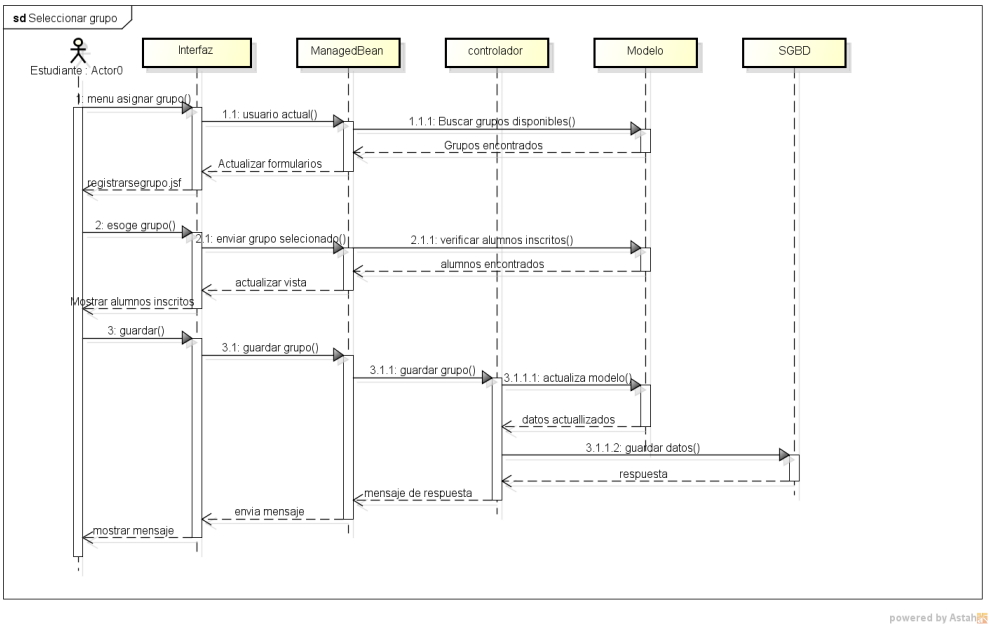


Figura 46. Selección de grupo de trabajo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.4.3 Diagrama de clases

El usuario con perfil Docente tiene acceso a seis páginas JSF desde las que controla todas sus actividades. La Figura 47 muestra un diagrama de clases general para el módulo de gestión académica del Docente, se han incluido solo las clases principales que tienen que ver con el módulo.

(1) Páginas JSF principales para la gestión académica del docente.

(2) Clases tipo bean que intervienen en la gestión académica del docente, reciben la información de las páginas JSF.

(3) Controladores principales que reciben las peticiones de los beans para los procesos de gestión académica.

(4) Modelo de persistencia, las tablas principales que intervienen en la gestión académica desde el punto de vista del docente: práctica, pregunta, respuesta, evaluación, materialapoyo, materia, practicaxmateria, grupo y docente.

Diagrama de clases

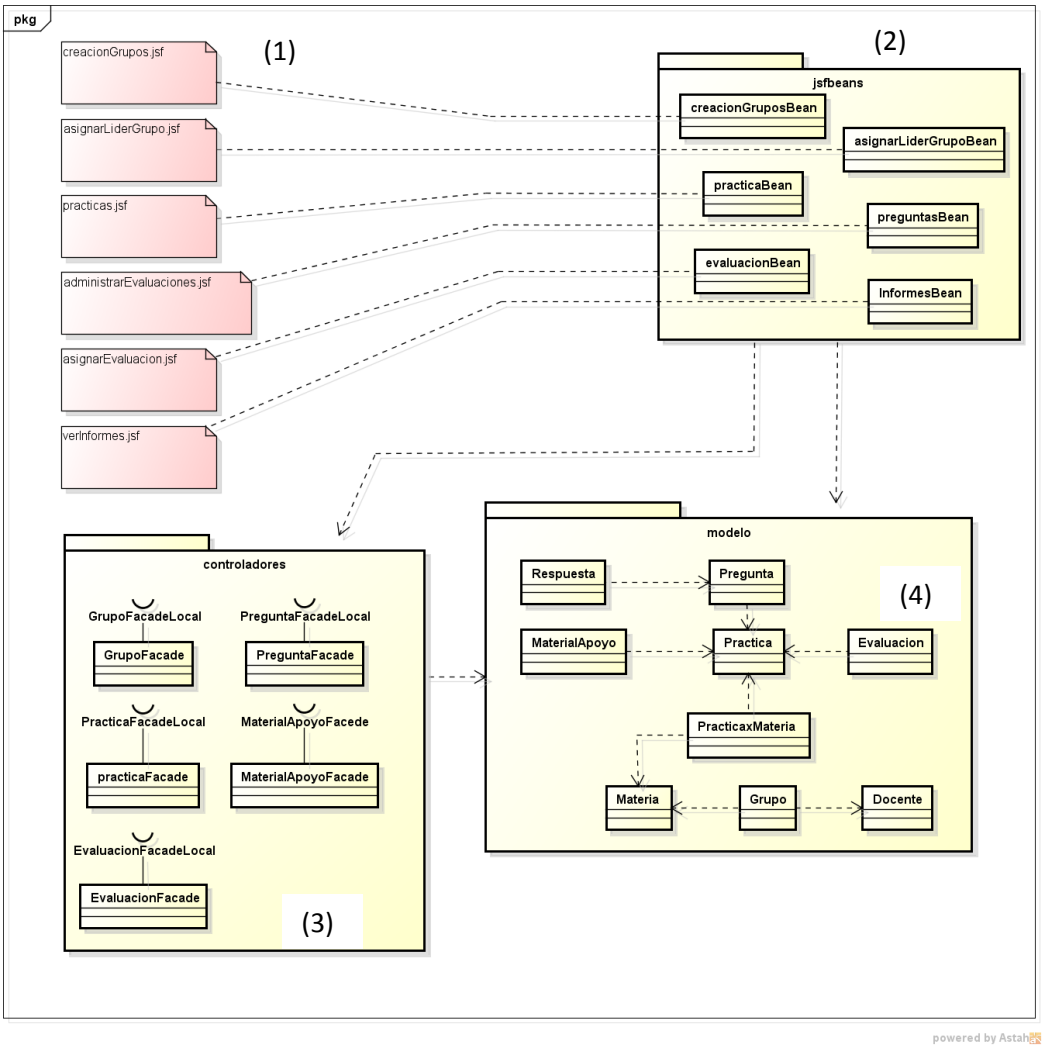


Figura 47. Gestión académica del docente. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

4.4.4 Vistas

A continuación se muestran los resultados obtenidos luego de finalizada la tercera fase de construcción del sistema. Pantalla principal del Docente, muestra el menú de acceso a todas las funciones propias del perfil de usuario.

La Figura 49 representa la pantalla para la creación de las prácticas de laboratorio y el ingreso del material de apoyo de las mismas. En la Figura 50 se muestra el registro de preguntas y respuestas del cuestionario.

Interfaz de usuario

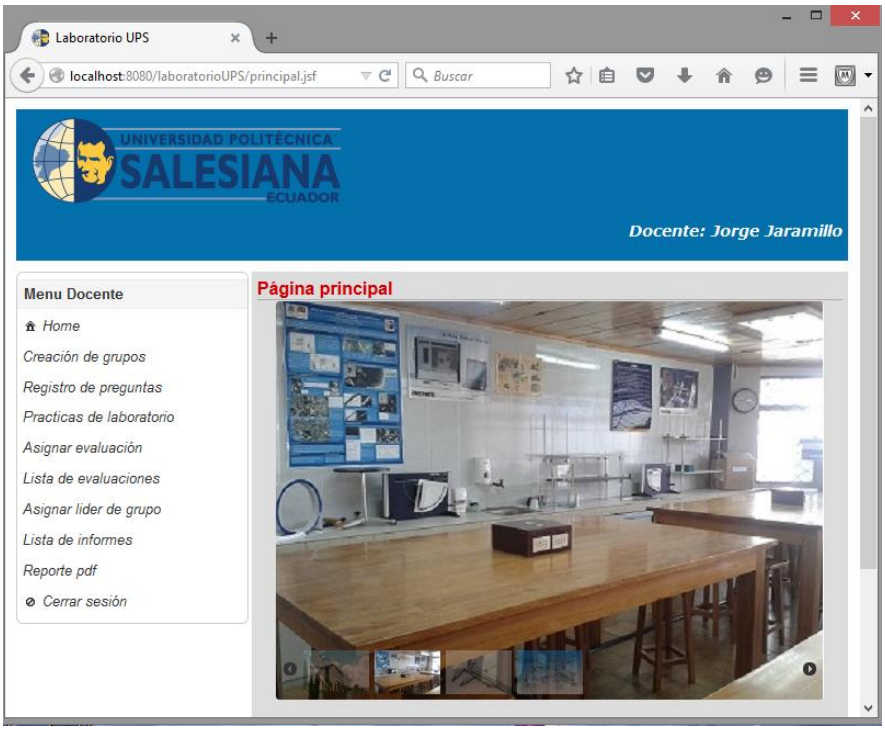


Figura 48. Pantalla principal para el perfil de usuario docente. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Interfaz de usuario

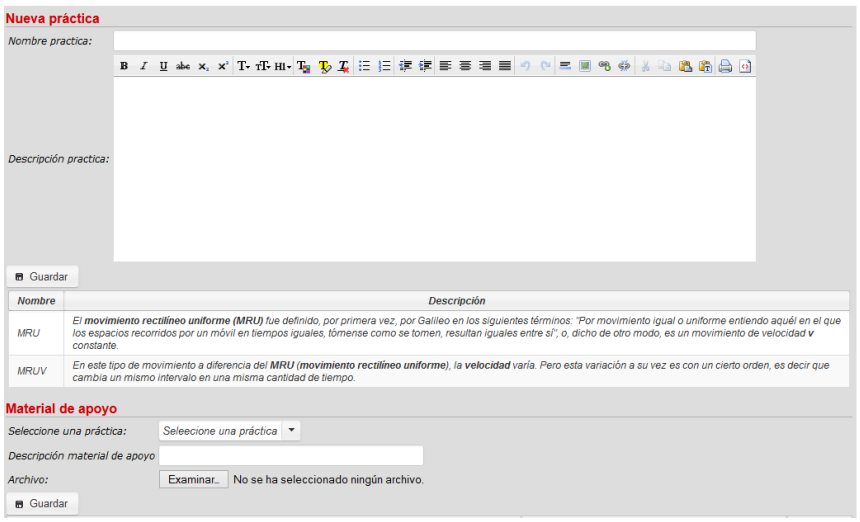


Figura 49. Pantalla para creación de prácticas. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Interfaz de usuario

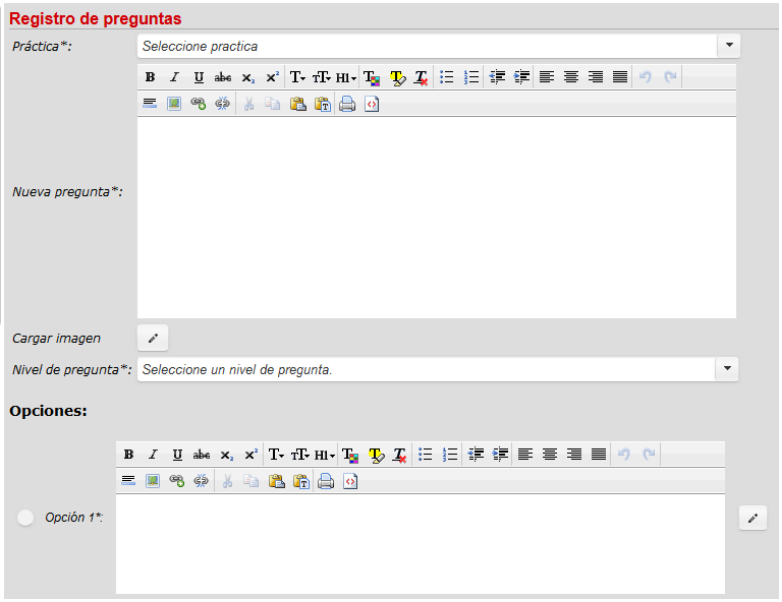


Figura 50. Registro de preguntas. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

La Figura 51 muestra el formulario para asignar evaluaciones a estudiantes. Se puede seleccionar cualquier tipo de evaluación: coloquio, defensa o informe.

Intefaz de usuario

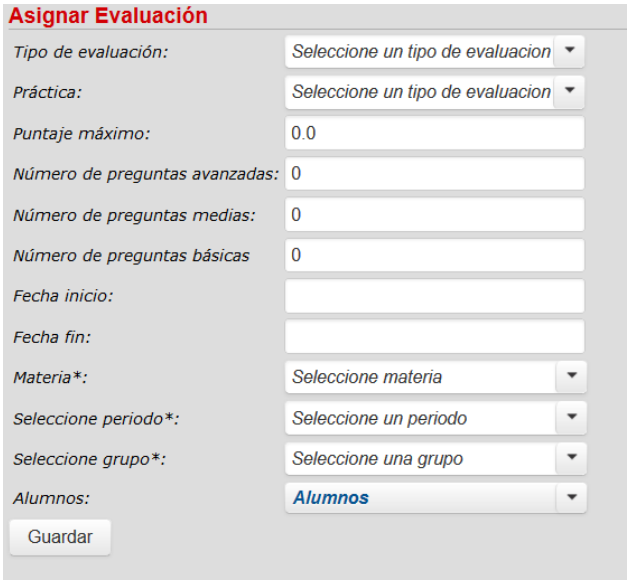


Figura 51. Asignar evaluaciones. Elaborado por: Myrian Guijarro & Jorge Jaramillo

La lista de informes se puede ver en la Figura 52, permite al docente revisar los archivos cargados por los estudiantes y colocar la calificación.

Interfaz de usuario

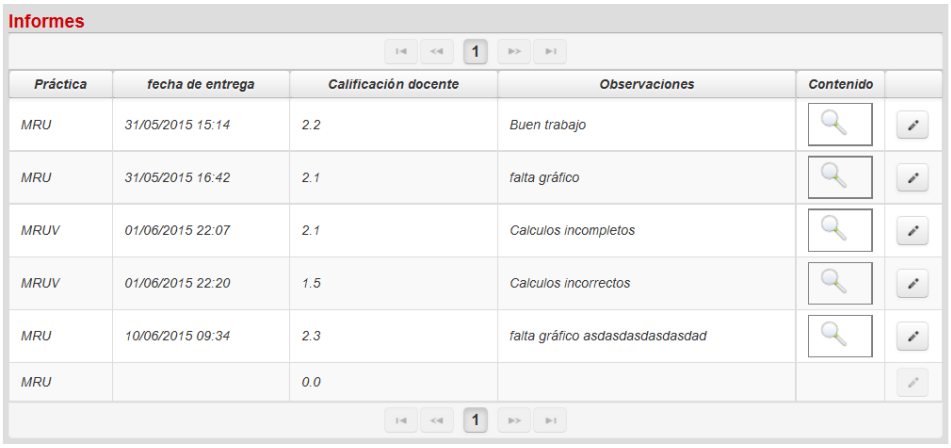


Figura 52. Revisar informes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

En la pantalla de la Figura 53, el estudiante puede registrarse en un grupo trabajo de laboratorio.

Interfaz de usuario

Figura 53. Registro en grupo de trabajo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.5. Cuarta fase de construcción. Gestión académica para el estudiante

La cuarta fase de construcción se centra en las actividades del estudiante en el módulo de gestión académica. Sus actividades, además de escoger el grupo de trabajo, incluyen:

- Calificar miembros del grupo.
- Subir informe de la práctica del laboratorio.
- Revisar le material de apoyo de cada práctica.
- Realizar tareas asignadas por el docente, coloquio, defensa informe.

4.5.3 Tablas relacionadas a los objetivos de la fase

La Figura 54 muestra las relaciones entre las tablas del módulo de gestión académica que están involucradas con las actividades del estudiante. Luego, en la

se describe brevemente la función de cada una de las tablas en la base de datos.

Tabla 32.Tablas relacionadas con el módulo de gestión académica (Estudiante)

Nombre de la tabla	Descripción
tipoevaluacion	Almacena los nombres de los tipos de evaluaciones, identifica si es coloquio, defensa, informe
respuesta	Almacena las respuestas ingresadas por pregunta
evaluación	Almacena todos los datos de las evaluaciones creadas
pregunta	Almacena los datos de las preguntas por práctica
nivelpregunta	Almacena el nivel de complejidad de las preguntas: básica, media, avanzada
materialdeapoyo	Almacena los datos del material de apoyo
practica	Almacena los datos de la practica
alumnoxinforme	Almacena la nota final del informe de cada alumno del grupo, luego de haberse promediado la nota del líder, la nota del docente
practicaxmateria	Almacena los datos de las practicas por materia
informe	Almacena los datos del informe, subido por el líder de cada grupo
alumnoxgrupo	Almacena los datos de los alumnos en cada grupo
materia	Almacena los datos de las materias
grupo	Almacena los datos de los grupos, por carrera, materia

docente	Almacena los datos del docente
laboratorio	Almacena los datos del laboratorio: laboratorio1,laboratorio2

Nota. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Diagrama de base de datos

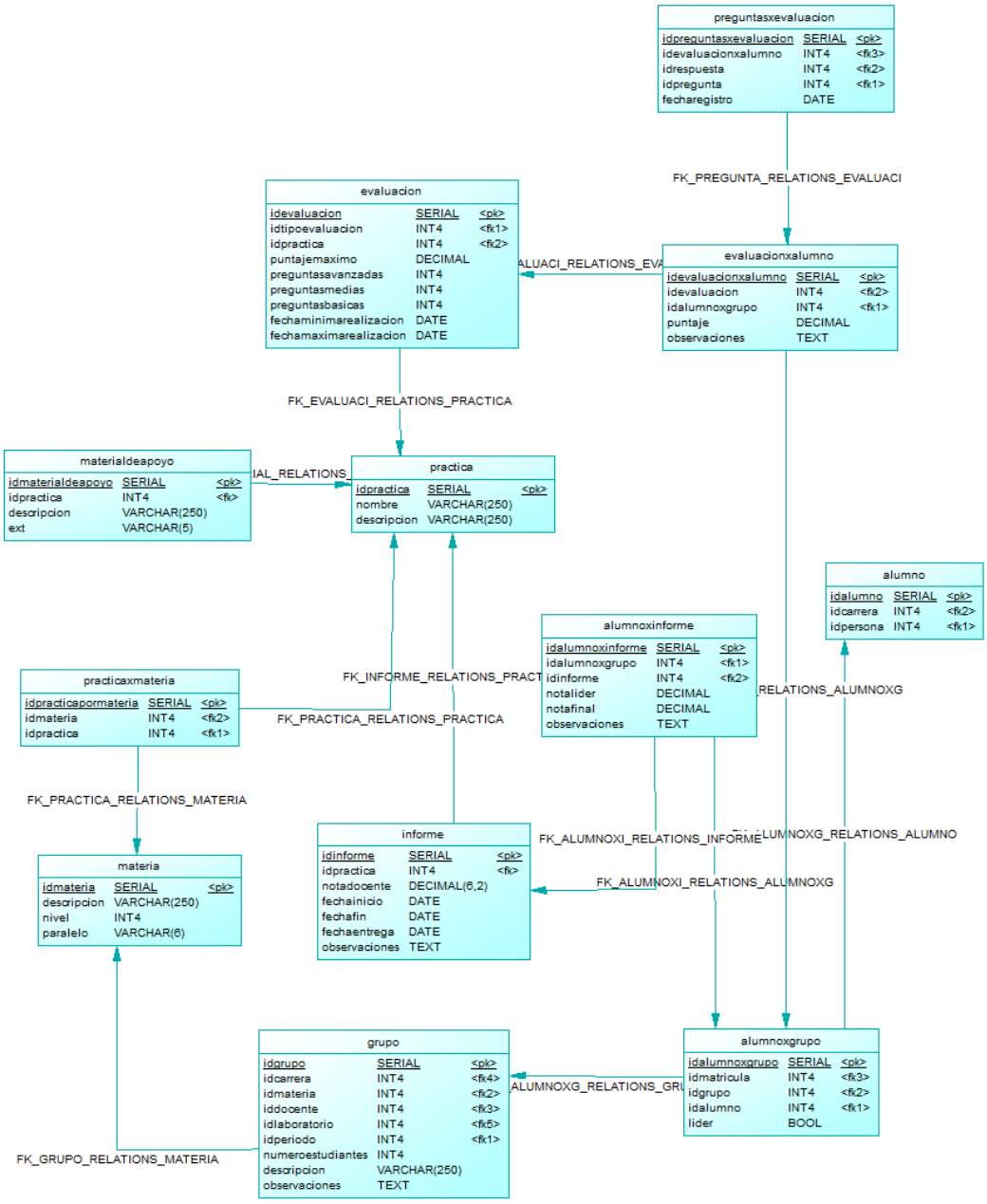


Figura 54. Tablas involucradas en incremento de fase de construcción 4. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.5.4 Diagramas de secuencia

Las funcionalidades añadidas al finalizar este sprint están detalladas en tres diagramas de secuencia detallados a continuación.

- Tareas asignadas (Figura 55): el usuario selecciona el menú tareas asignadas, el bean respectivo recupera desde el modelo todas las tareas asignadas y actualiza la interfaz. Cuando el usuario selecciona una tarea, el bean construye el cuestionario recuperando las preguntas según la especificación de la evaluación actual.

  El usuario selecciona una pregunta del cuestionario, el bean envía la pregunta al modelo para recuperar las opciones de respuesta y construir el diálogo en la interfaz. Una vez seleccionada la respuesta, el usuario selecciona el botón guardar para iniciar el proceso de actualización en el modelo y la base.

Diagrama de secuencia

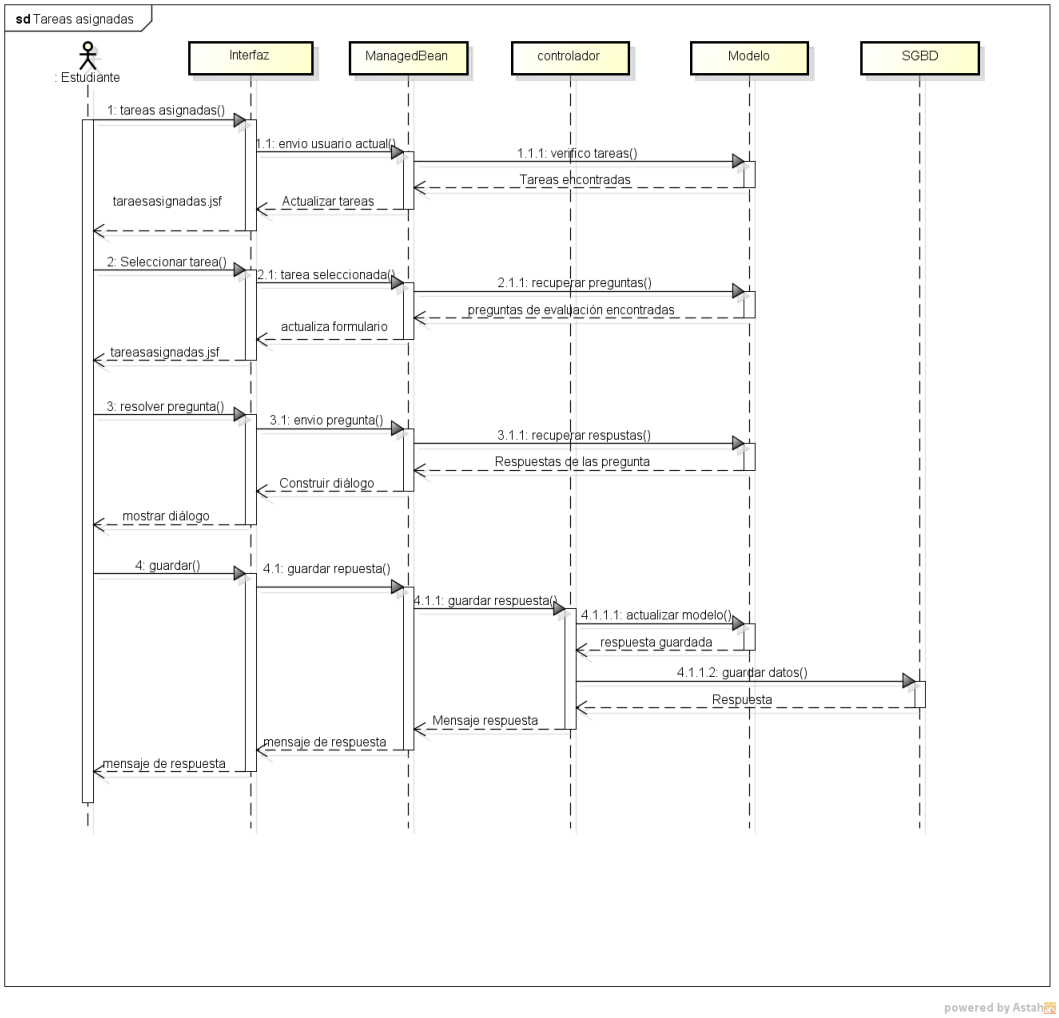


Figura 55. Tareas asignadas. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Informes de laboratorio por grupo (Figura 56): el usuario con perfil alumno que tiene la característica de líder de grupo puede subir el informe de la práctica y colocar la nota del líder en el informe del grupo. Primero, ingresa en el menú de tareas asignadas, el bean respectivo recupera las tareas, de tipo informe, asignadas y actualiza la interfaz para el usuario. Al seleccionar ver informe, el bean solicita al modelo que busque los datos del informe seleccionado y abre el archivo en pantalla.

  Para subir un informe, el usuario selecciona la opción subir archivo, se muestra el diálogo respectivo donde se valida el formato. Una vez seleccionado el archivo, el usuario inicia el proceso de guardado.

  Finalmente, el usuario puede colocar la nota del líder en el informe de cada uno de los integrantes del grupo de trabajo.

Diagrama de secuencia

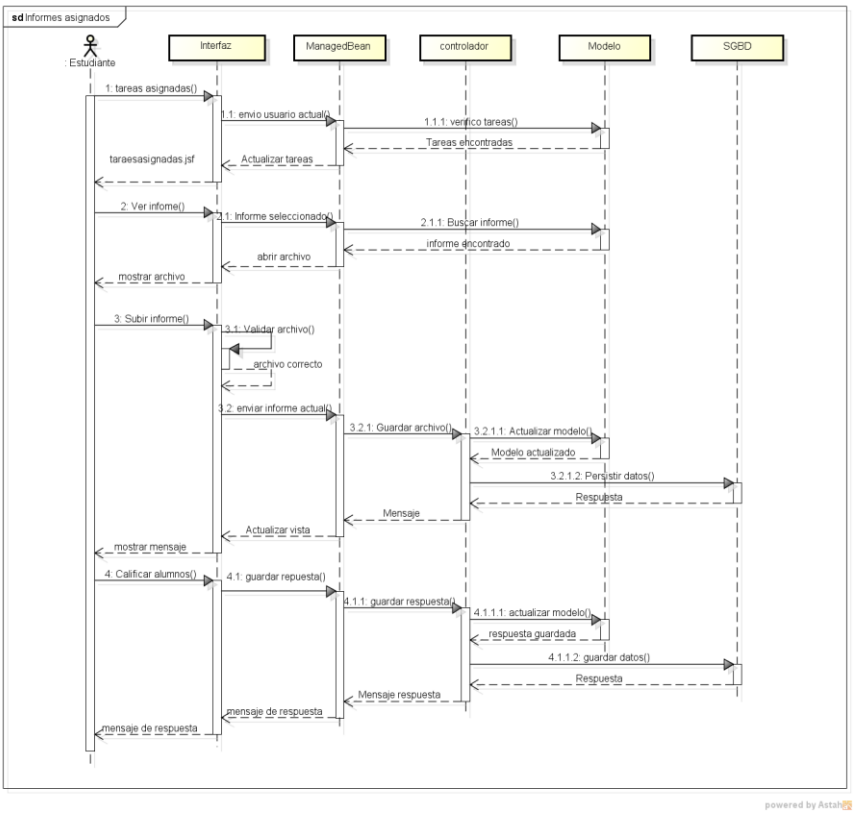


Figura 56. Informes de laboratorio por grupo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Revisar le material de apoyo (Figura 57): el usuario selecciona el menú guía de evaluación, el bean respectivo solicita al modelo el listado de todas las prácticas registradas para el usuario actual y actualiza la interfaz. Al seleccionar una

práctica, el bean solicita los materiales de apoyo asociados a la práctica actual y actualiza la interfaz donde el usuario puede visualizar cualquiera de los materiales de apoye que seleccione.

Diagrama de secuencia

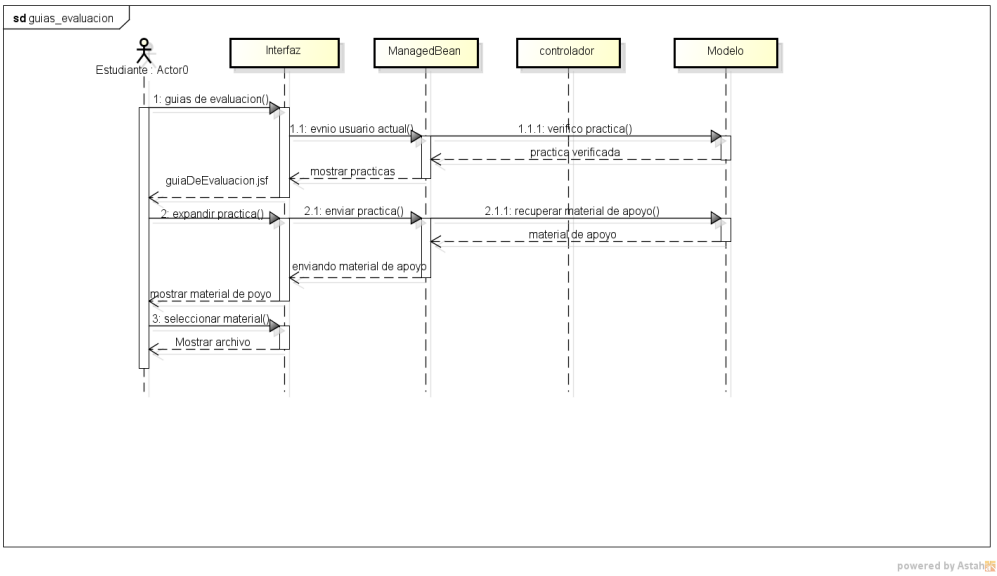


Figura 57. Revisar material de apoyo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.5.5 Diagrama de clases

La cuarta fase de construcción del sistema incrementa al módulo de gestión académica las funcionalidades necesarias para las actividades del perfil de usuario Estudiante. El estudiante tiene acceso a tres páginas JSF, cada una asociada a su propio bean. En la Figura 58, se representa las relaciones entre las diferentes clases, del módulo de gestión académica, que intervienen con las actividades del estudian.

(1) Las actividades del módulo de gestión académica para el alumno se manejan desde tres páginas JSF: registroGrupo.jsf, guiaDeEvaluacion.jsf y tareasAsignadas.jsf.
(2) Cada página mencionada tiene su propio bean que recibe las peticiones de la interfaz.
(3) Entre los controladores principales se incluye, además de los propios para cada clase principal, el de la clase Alumno.

(4) Las clases más importantes del modelo de persistencia relacionadas con el alumno en este módulo son: PreguntaxEvaluacion, EvaluacionxAlumno, Evaluacion, Practica, MaterialApoyo, Informe, AlumnoxInforme, AlumnoxGrupo, Alumno, Grupo.

Diagrama de clases

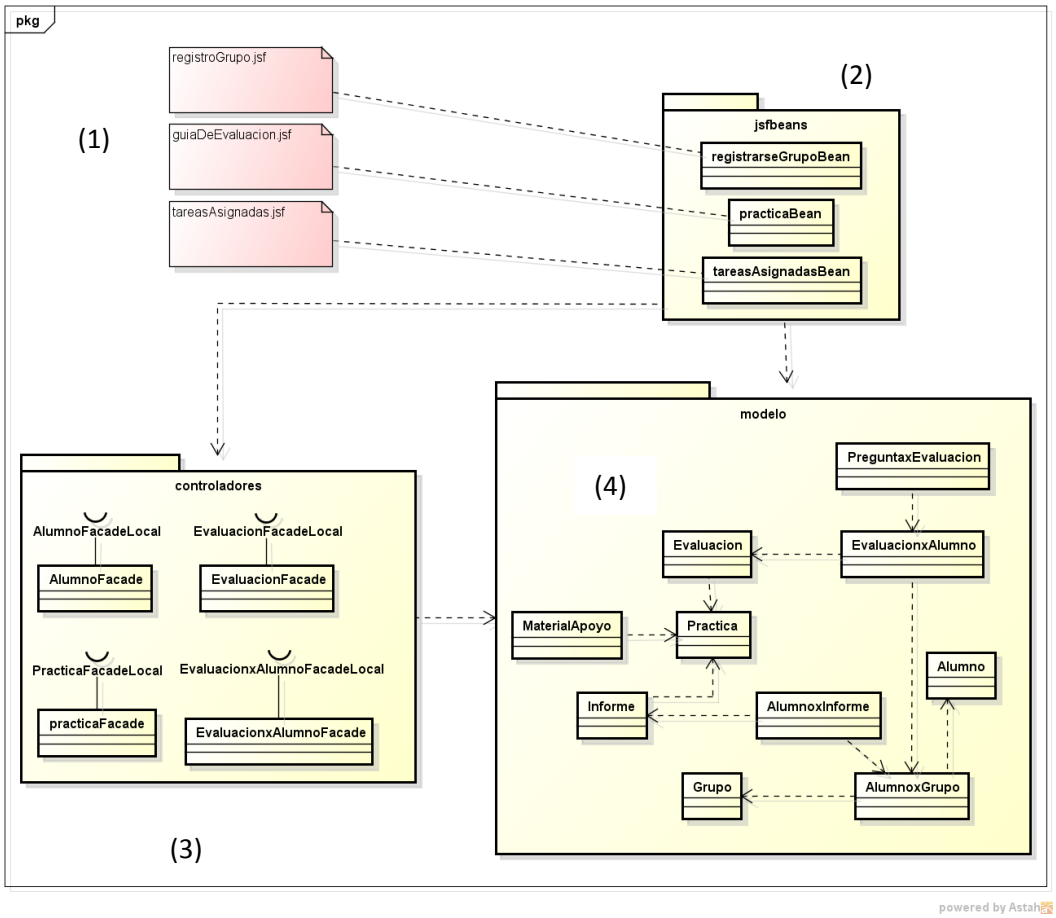


Figura 58. Gestión académica del alumno. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.5.6 Vistas

Ventana Guía de evaluación, permite revisar el material de apoyo de cada práctica asignada.

Interfaz de usuario

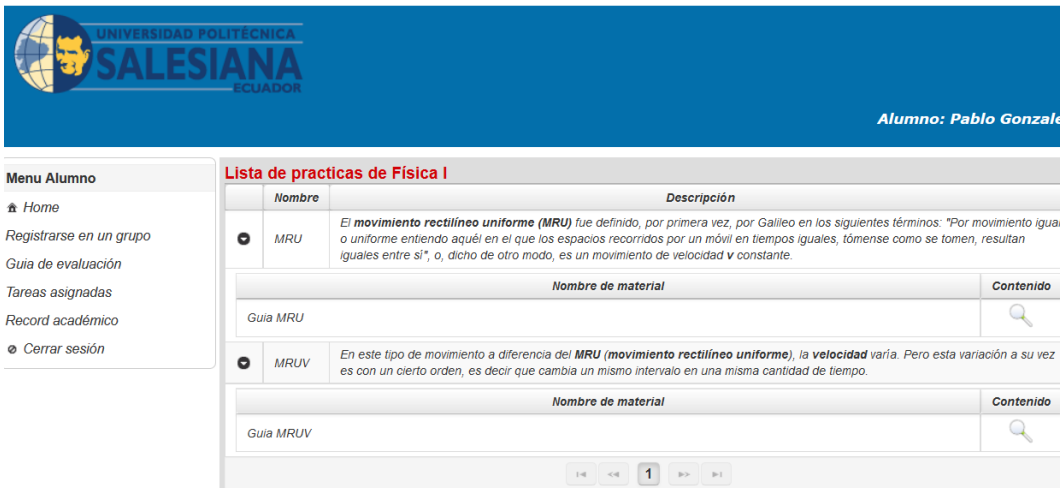


Figura 59. Lista de prácticas y material de apoyo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Ventana tareas asignadas, esta opción está habilitada únicamente para los usuarios líder de grupo, ya que en esta ventana podrán calificar al grupo como se muestra en la Figura 61 y subir informes.

Interfaz de usuario

Figura 60. Informes asignados. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Interfaz de usuario

Figura 61. Calificar desempeño del grupo de trabajo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

4.6. Quinta fase de construcción. Gestión de reportes

Una vez concluida la construcción de los objetivos de cada fase anterior, el sistema cuenta con el soporte necesario para las actividades principales de cada perfil de usuario: administrador, docente y estudiante. Para finalizar, la quinta fase añade la funcionalidad de reportes. Los reportes son uno de los elementos más importantes en este sistema ya que permiten obtener información precisa y resumida de interés para el usuario de cada módulo.

La obtención de reportes en el sistema se lo hace mediante una comunicación directa entre el bean y la base de datos, todos los reportes trabajan dela misma manera, con pequeñas variantes solo cuando el reporte no tiene parámetros de recuperación. Se ha omitido el detalle de las tablas involucradas con los objetivos de esta fase debido a que los reportes se pueden recuperar de cualquier tabla del sistema y el modelo completo de la base de datos ya se describió en el capítulo 3. El proceso de obtención de los reportes se muestra detalladamente a continuación.

4.6.1 Diagrama de secuencia

Diagrama de secuencia

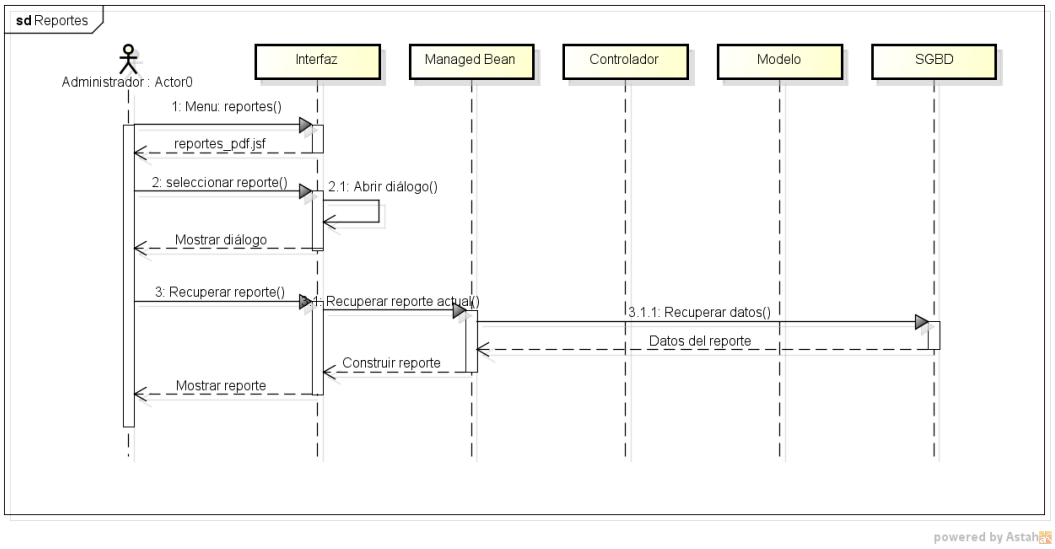


Figura 62. Visualizar reportes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

En la Figura 62, se representa el proceso necesario para la obtención de los reportes. El usuario ingresa en el menú de reportes y selecciona en el listado el reporte que desea ver. El sistema muestra un diálogo con los filtros específicos para el reporte seleccionado. Al presionar el botón para ver el reporte se envía una solicitud al bean para que realice el proceso necesario para construir el reporte. Terminado este proceso, se muestra al usuario el archivo final mediante un diálogo propio del browser.

4.6.2 Vistas

El menú de reportes presenta los botones como se muestra en la Figura 63.

Interfaz de usuario

Figura 63. Menú de reportes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

Formato de reporte

Reporte de notas por curso

Desde: '2015-01-01 12:00:00' Hasta: '2015-06-25 12:

Carrera: Ingeniería en Sistemas
Materia: Física I
Nivel: 1
Paralelo: 1
Docente: Patricio Nuñez

	MRU				MRUV				Nota Final
	Coloquio	Defensa	Informe	Total práctica	Coloquio	Defensa	Informe	Total práctica	
Alava Ramiro	1,67	3,53	2,10	7,30	1,60	3,75	2,25	7,60	7,45
Gavilanes Alejandro	1,50	3,53	2,60	7,63	2,00	3,25	2,25	7,50	7,56
Giron Leonel	0,00	2,35	2,55	4,90	2,00	3,25	2,55	7,80	6,35
Gonzales Pablo	2,00	2,94	2,60	7,54	1,60	4,00	2,25	7,85	7,70
Leon Guillermo	1,40	5,00	2,05	8,45	0,60	4,50	2,55	7,65	8,05
Loor Bruno	1,00	4,12	2,55	7,67	0,40	3,25	2,55	6,20	6,93
Ramirez Washington	1,40	3,53	0,00	4,93	0,00	0,00	0,00	0,00	4,93

Figura 64. Reporte de alumnos. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

4.7 Pruebas

Durante el proceso de desarrollo se llevaron a cabo las pruebas necesarias para garantizar el correcto funcionamiento del aplicativo y la satisfacción de los requerimientos del propietario del producto.

En primer lugar se realizaron pruebas funcionales de cada uno de los requerimientos solicitados. Luego de estas pruebas, se llevaron a cabo las respectivas reuniones de

trabajo con los involucrados para la presentación de los avances en el desarrollo y resultados de las pruebas. Como resultado de estas pruebas, en el anexo A se puede ver la carta de satisfacción del cliente con el producto final así como el registro de los temas tratados en las reuniones.

4.7.1 Pruebas de rendimiento

Para las pruebas de rendimiento se comprobó el comportamiento del servidor de aplicaciones durante la ejecución del sistema y el tiempo de respuesta de la aplicación durante las transacciones y procesos más complejos como el registro de usuarios y la obtención de reportes.

Estas pruebas se realizaron utilizando la herramienta Java VisualVM que forma parte del JDK. Es una interfaz gráfica de usuario que provee información detallada sobre aplicaciones basadas en Java mientras se están ejecutando en una Máquina Virtual Java. Con esta herramienta se puede evaluar el rendimiento de las aplicaciones y visualizar cómo responde la Máquina Virtual de Java durante la ejecución.

A continuación se detalla los resultados obtenidos en el monitoreo durante la ejecución de la aplicación.

- Uso de Java Heap: Java Heap es un espacio en memoria que se crea al iniciar la ejecución de una aplicación Java y almacena las instancias de todos los objetos y clases creados durante la ejecución de la aplicación. En la Figura 65 se observa que el valor máximo de memoria que la aplicación utiliza durante su ejecución es del 50% del total del espacio disponible asignado a la máquina virtual.

Gráfico de medición de uso de memoria

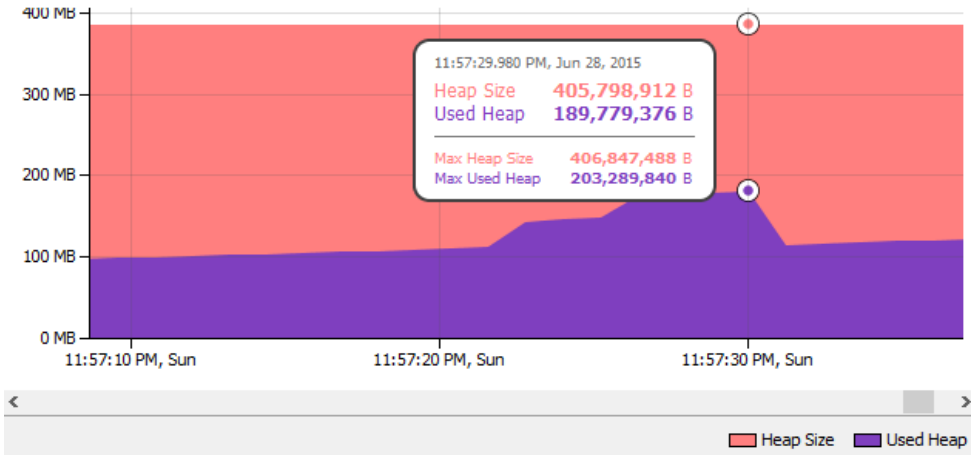


Figura 65. Máximo uso de Java Heap durante la ejecución. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

Las siguientes mediciones muestran el porcentaje del tiempo de ejecución de un determinado proceso y el tiempo que ocupa en ejecutarse cada método que interviene en el mismo proceso.

- Inicio de sesión: durante el proceso de inicio de sesión, como se puede ver en la Figura 66, el método que más tiempo de procesamiento consume en el método *login* del controlador UsuarioFacade, este es el método que se encarga de la validación de los datos y toma 173 ms en ejecutarse.

Medición de tiempos de ejecución

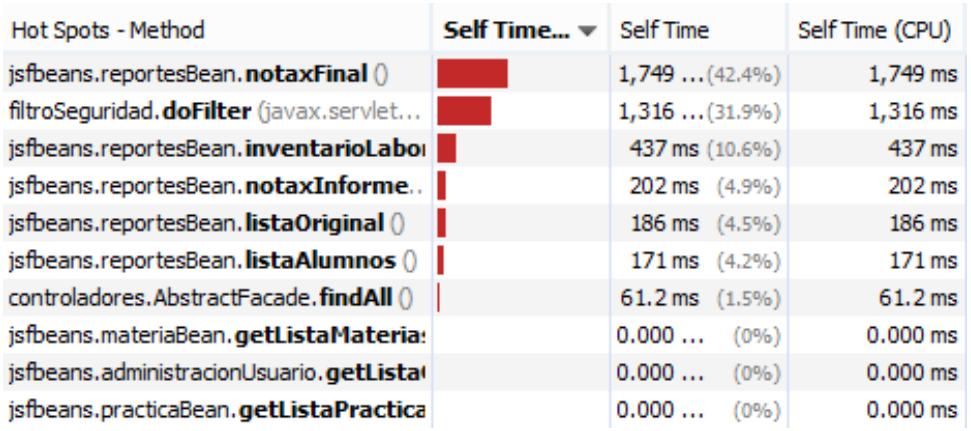
Hot Spots - Method	Self Tim...	Self Time	Self Time (...
controladores.UsuarioFacade.**login** (modelo.U...		173 ms (60.9%)	173 ms
filtroSeguridad.**doFilter** (javax.servlet.Servle...		95.8 ... (33.7%)	95.8 ms
controladores.PermisoxperfilFacade.**getPermis**		15.1 ... (5.3%)	15.1 ms
jsfbeans.loginBean.**autentificar** ()		0.00... (0%)	0.000 ms
jsfbeans.loginBean.**getModelo** ()		0.00... (0%)	0.000 ms

Figura 66. Tiempo utilizado por método al iniciar sesión. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Reportes: para esta prueba, se generó todos los reportes disponibles en el sistema. El resultado muestra en la Figura 67 que el reporte de notas finales; que es el que

utiliza el método notaxFinal mostrado en la imagen, es el que consume más tiempo de ejecución.

Medición de tiempos de ejecución

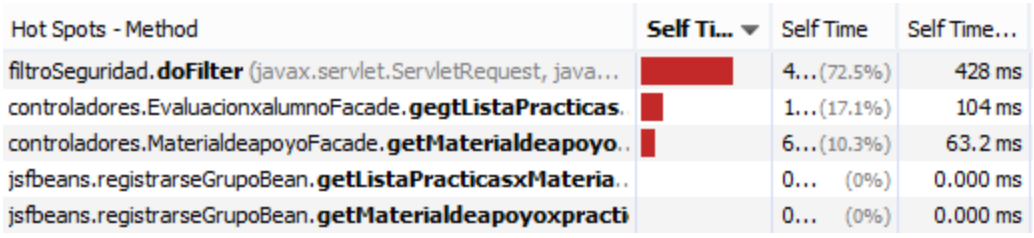


Hot Spots - Method	Self Time...	Self Time	Self Time (CPU)
jsfbeans.reportesBean.**notaxFinal** ()		1,749 ... (42.4%)	1,749 ms
filtroSeguridad.**doFilter** (javax.servlet...		1,316 ... (31.9%)	1,316 ms
jsfbeans.reportesBean.**inventarioLabo**		437 ms (10.6%)	437 ms
jsfbeans.reportesBean.**notaxInforme**..		202 ms (4.9%)	202 ms
jsfbeans.reportesBean.**listaOriginal** ()		186 ms (4.5%)	186 ms
jsfbeans.reportesBean.**listaAlumnos** ()		171 ms (4.2%)	171 ms
controladores.AbstractFacade.**findAll** ()		61.2 ms (1.5%)	61.2 ms
jsfbeans.materiaBean.**getListaMateria**		0.000 ... (0%)	0.000 ms
jsfbeans.administracionUsuario.**getLista**		0.000 ... (0%)	0.000 ms
jsfbeans.practicaBean.**getListaPractica**		0.000 ... (0%)	0.000 ms

Figura 67. Tiempo utilizado por método al obtener reportes. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Material apoyo: en cuanto al proceso de carga de materiales de apoyo, no representa un gran trabajo para el servidor, ya que como se aprecia en la Figura 68 la mayor proporción del tiempo consumido al ejecutar esta actividad corresponde al método de validación del sistema. Este método verifica constantemente los permisos de acceso y si la sesión del usuario actual está activa.

Medición de tiempos de ejecución



Hot Spots - Method	Self Ti...	Self Time	Self Time...
filtroSeguridad.**doFilter** (javax.servlet.ServletRequest, java...		4... (72.5%)	428 ms
controladores.EvaluacionxalumnoFacade.**gegtListaPracticas**		1... (17.1%)	104 ms
controladores.MaterialdeapoyoFacade.**getMaterialdeapoyo**..		6... (10.3%)	63.2 ms
jsfbeans.registrarseGrupoBean.**getListaPracticasxMateria**..		0... (0%)	0.000 ms
jsfbeans.registrarseGrupoBean.**getMaterialdeapoyoxpracti**		0... (0%)	0.000 ms

Figura 68. Tiempo utilizado por método al recuperar material de apoyo. Elaborado por: Myrian Guijarro & Jorge Jaramillo.

- Creación de usuarios: la Figura 69 muestra el tiempo requerido al realizar la creación de usuarios al importar desde un archivo csv, la prueba se realizó con un archivo de 1000 usuarios de manera exitosa lo que garantiza que no existirán problemas al realizar el trabajo regular del laboratorio.

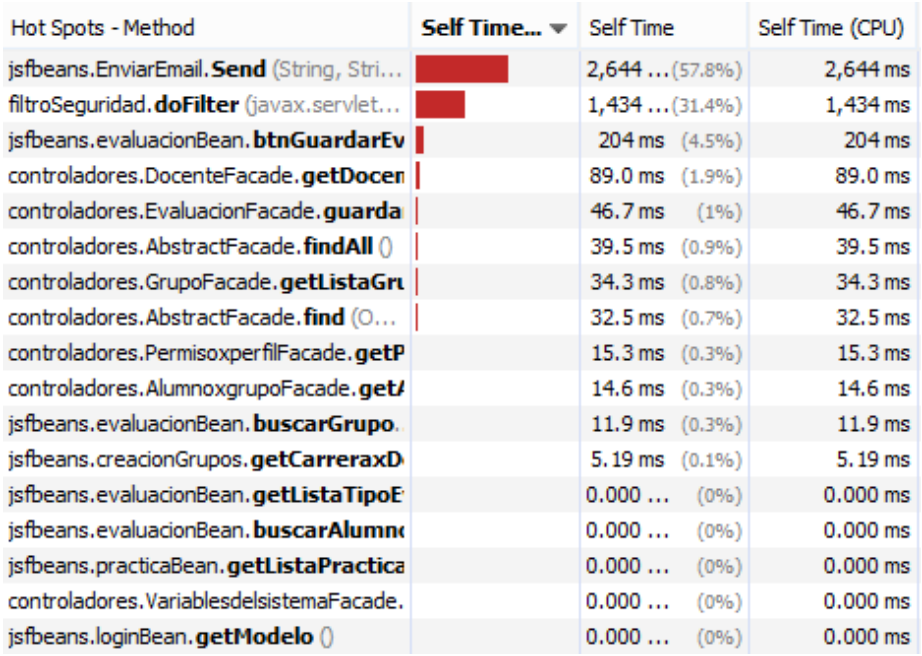
Medición de tiempos de ejecución

Hot Spots - Method	Self Tim...	Self Time	Self Time (C...
jsfbeans.cargarxmlBean.**subirArchivo** (or...		19,5... (66%)	19,533 ms
controladores.PersonaFacade.**buscarPerson**		5,83... (19.7%)	5,831 ms
controladores.AbstractFacade.**findAll** ()		1,68... (5.7%)	1,688 ms
filtroSeguridad.**doFilter** (javax.servlet.Ser...		962 ms (3.3%)	957 ms
controladores.PersonaFacade.**guardarPeson**		544 ms (1.8%)	544 ms
controladores.UsuarioFacade.**guardarUsuari**		328 ms (1.1%)	328 ms
controladores.AbstractFacade.**find** (Object)		193 ms (0.7%)	193 ms
controladores.UsuarioFacade.**getUsuarioxCe**		131 ms (0.4%)	131 ms
controladores.AlumnoFacade.**getAlumno** (...		88.9 ... (0.3%)	88.9 ms
modelo.Persona.**_persistence_shallow_clo**		55.1 ... (0.2%)	55.1 ms
controladores.AlumnoFacade.**guardarAlumn**		54.2 ... (0.2%)	54.2 ms
modelo.Usuario.**_persistence_shallow_clo**		43.9 ... (0.1%)	43.9 ms
controladores.MatriculaFacade.**guardarMatr**		35.3 ... (0.1%)	35.3 ms
modelo.Usuario.**setIdperfil** (modelo.Perfil)		33.2 ... (0.1%)	33.2 ms
modelo.Matricula.**_persistence_shallow_cl**		21.7 ... (0.1%)	21.7 ms
modelo.Usuario.**_persistence_new** (org....		15.9 ... (0.1%)	15.9 ms
modelo.Usuario.**_persistence_checkFetche**		12.6 ... (0%)	12.6 ms
modelo.Periodo.**_persistence_setId** (Obj...		10.9 ... (0%)	10.9 ms
modelo.Matricula.**<init>** ()		4.32 ... (0%)	4.32 ms
modelo.Alumno.**_persistence_isAttributeF**		1.63 ... (0%)	1.63 ms
jsfbeans.administracionUsuario.**getListaUsua**		0.00... (0%)	0.000 ms
controladores.PerfilFacade.**find** (Object)		0.00... (0%)	0.000 ms
controladores.MateriaxcarreraFacade.**find** ...		0.00... (0%)	0.000 ms
controladores.PeriodoFacade.**find** (Object)		0.00... (0%)	0.000 ms
modelo.Usuario.**_persistence_set_idperfil**		0.00... (0%)	0.000 ms
modelo.Alumno.**_persistence_get_idcarrer**		0.00... (0%)	0.000 ms
modelo.Alumno.**_persistence_checkFetche**		0.00... (0%)	0.000 ms
modelo.Alumno.**getIdcarrera** ()		0.00... (0%)	0.000 ms

Figura 69. Tiempo utilizado por método al importar usuarios desde archivo. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Asignar evaluación: durante la asignación de evaluaciones a estudiantes, la Figura 70 muestra que el proceso que lleva, relativamente, mayor tiempo es el envío de la notificación por correo electrónico.

Medición de tiempos de ejecución

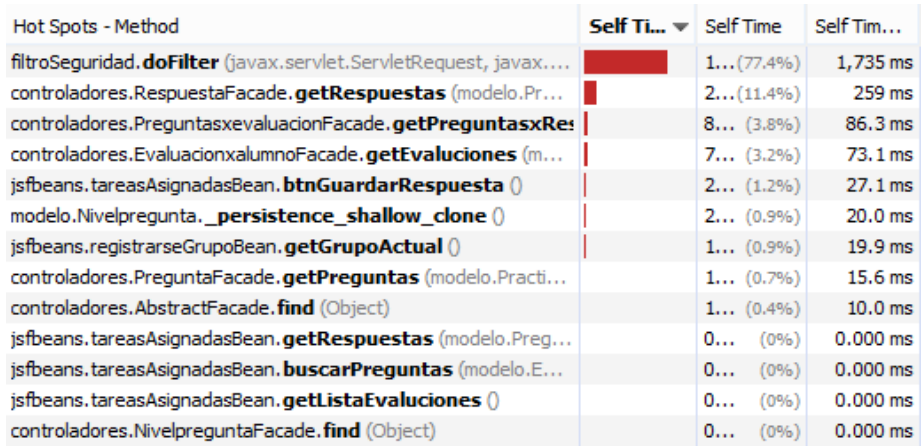


Hot Spots - Method	Self Time...	Self Time	Self Time (CPU)
jsfbeans.EnviarEmail.**Send** (String, Stri...		2,644 ... (57.8%)	2,644 ms
filtroSeguridad.**doFilter** (javax.servlet...		1,434 ... (31.4%)	1,434 ms
jsfbeans.evaluacionBean.**btnGuardarEv**		204 ms (4.5%)	204 ms
controladores.DocenteFacade.**getDocen**		89.0 ms (1.9%)	89.0 ms
controladores.EvaluacionFacade.**guarda**		46.7 ms (1%)	46.7 ms
controladores.AbstractFacade.**findAll** ()		39.5 ms (0.9%)	39.5 ms
controladores.GrupoFacade.**getListaGr**		34.3 ms (0.8%)	34.3 ms
controladores.AbstractFacade.**find** (O...		32.5 ms (0.7%)	32.5 ms
controladores.PermisoxperfilFacade.**getP**		15.3 ms (0.3%)	15.3 ms
controladores.AlumnoxgrupoFacade.**get**		14.6 ms (0.3%)	14.6 ms
jsfbeans.evaluacionBean.**buscarGrupo**		11.9 ms (0.3%)	11.9 ms
jsfbeans.creacionGrupos.**getCarreraxD**		5.19 ms (0.1%)	5.19 ms
jsfbeans.evaluacionBean.**getListaTipoE**		0.000 ... (0%)	0.000 ms
jsfbeans.evaluacionBean.**buscarAlumn**		0.000 ... (0%)	0.000 ms
jsfbeans.practicaBean.**getListaPractica**		0.000 ... (0%)	0.000 ms
controladores.VariablesdelsistemaFacade.		0.000 ... (0%)	0.000 ms
jsfbeans.loginBean.**getModelo** ()		0.000 ... (0%)	0.000 ms

Figura 70. Tiempo utilizado por método al asignar evaluaciones. Elaborado por: Myrian Guijarro & Jorge Jaramillo

- Resolver evaluación asignada: durante la resolución de la evaluación asignada por el estudiante, de igual manera que en un caso anterior, el tiempo invertido en el método de validación de usuarios es más alto que el tiempo que les toma a los demás métodos la realización de sus procesos.

Medición de tiempos de ejecución



Hot Spots - Method	Self Ti...	Self Time	Self Tim...
filtroSeguridad.**doFilter** (javax.servlet.ServletRequest, javax....		1... (77.4%)	1,735 ms
controladores.RespuestaFacade.**getRespuestas** (modelo.Pr...		2... (11.4%)	259 ms
controladores.PreguntasxevaluacionFacade.**getPreguntasxRes**		8... (3.8%)	86.3 ms
controladores.EvaluacionxalumnoFacade.**getEvaluciones** (m...		7... (3.2%)	73.1 ms
jsfbeans.tareasAsignadasBean.**btnGuardarRespuesta** ()		2... (1.2%)	27.1 ms
modelo.Nivelpregunta.**_persistence_shallow_clone** ()		2... (0.9%)	20.0 ms
jsfbeans.registrarseGrupoBean.**getGrupoActual** ()		1... (0.9%)	19.9 ms
controladores.PreguntaFacade.**getPreguntas** (modelo.Practi...		1... (0.7%)	15.6 ms
controladores.AbstractFacade.**find** (Object)		1... (0.4%)	10.0 ms
jsfbeans.tareasAsignadasBean.**getRespuestas** (modelo.Preg...		0... (0%)	0.000 ms
jsfbeans.tareasAsignadasBean.**buscarPreguntas** (modelo.E...		0... (0%)	0.000 ms
jsfbeans.tareasAsignadasBean.**getListaEvaluciones** ()		0... (0%)	0.000 ms
controladores.NivelpreguntaFacade.**find** (Object)		0... (0%)	0.000 ms

Figura 71. Tiempo utilizado por método al resolver evaluaciones. Elaborado por: Myrian Guijarro & Jorge Jaramillo

CONCLUSIONES

- El personal del Laboratorio de Física de la Universidad Politécnica Salesiana manejará de mejor manera sus actividades internas gracias a que el producto entregado agilita los procesos académicos y administrativos.
- En este proyecto, los reportes sirvieron para confirmar que todos los procesos del sistema funcionan como el cliente espera, es decir que el resultado total cumple con las expectativas y objetivos planteados por el cliente.
- Una buena estructura de datos permitió que los reportes mantengan la consistencia e integridad entre la información ingresada y la que se recupera del sistema.
- La implementación de videos tutoriales como material de apoyo permitirá que los estudiantes desarrollen de mejor manera sus actividades en el laboratorio.
- Debido a las funcionalidades de inventario integradas en el módulo de gestión administrativa, se elimina la necesidad de llevar registros manuales sobre el inventario de equipos y materiales del Laboratorio de Física, estos registros ahora se pueden manejar internamente en el sistema.
- El proceso de evaluación a los estudiantes es la principal actividad que se realiza en el sistema, gracias a la implementación del módulo de gestión académica, se elimina la necesidad de evaluar a los estudiantes en un medio físico como el papel y se puede crear cuestionarios mediante el uso de reactivos de acuerdo a la normativa del CEAACES.
- Scrum es una metodología ágil para el desarrollo de sistemas informáticos que permite la construcción del producto final en base a elaboración de partes más pequeñas durante varias iteraciones que facilitan y hacen más manejable el volumen de trabajo, esto a su vez ayuda a detectar de manera rápida cualquier posible error en cada etapa de desarrollo.
- Una buena comunicación entre el equipo de desarrollo y el cliente es de vital importancia para el éxito de los proyectos de desarrollo de software; esto se evidenció durante las reuniones de cada fase en las que se iban adaptando detalles al producto, lo que se traduce en un alto nivel de satisfacción del cliente con el resultado final.

- Gracias al uso de tecnologías como JSF, se facilita la comunicación entre las aplicaciones y los sistemas de almacenamiento de datos mediante un diseño en capas y basado en modelos de persistencia de datos.

RECOMENDACIONES

- Se recomienda generar políticas de respaldo automático de la base de datos para evitar posibles pérdidas de información debido a causas externas no previstas.
- Al momento el laboratorio de física cuenta con un equipo en donde se implementó el sistema y funciona como servidor de aplicaciones y de bases de datos. Es recomendable configurar un servidor independiente para cada una de estas funciones.
- Si bien se desarrolló un módulo de gestión académica para la realización de evaluaciones y asignación de tareas, al momento, el sistema no gestiona los horarios. Se recomienda integrar un módulo para la gestión de horarios de cada materia.
- El repositorio de materiales de apoyo organiza la información que el docente carga en el sistema clasificándola por práctica de laboratorio. Se sugiere la implementación de un sistema más completo para la gestión de materiales a modo de una biblioteca virtual.
- Se recomienda realizar una aplicación móvil, la cual permita a los estudiantes revisar tareas asignadas, calificaciones, notificaciones relacionadas con el laboratorio de física.

LISTA DE REFERENCIAS

Agile Alliance. (10 de 06 de 2015). *The Agile Manifesto.* Obtenido de http://www.agilemanifesto.org/iso/es/

Álvarez, R. B., & Mayo, C. I. (10 de noviembre de 2009). Las tecnologías de la información y la comunicación en la educación superior. Estudio descriptivo y de revisión. *Revista Iberoamericana de Educación*, 7.

Andrade, X. (2013). GUÍA PARA LA ELABORACIÓN DE PRUEBAS DE BASE .

Canos, J. H., Letelier, P., & Penadés, M. C. (2003). *Metodologías Ágiles en el desarrollo de Software.* Universidad Politécnica de Valencia.

CEAACES. (2012). Construcción de ítems.

de la Cruz Diaz, L. A., Gómez Rodriguez, M. A., Lara Jerónimo, S. U., & Pech May, F. (s.f.). *Desarrollo de Aplicaciones web con JPA, EJB, JSF y PrimeFaces.* Instituto Tecnológico Superior de los Ríos, Tabasco, México.

Domínguez, M., Reguera, P., & Fuertes, J. J. (2005). Laboratorio remoto para la enseñanza de la automatica en la universidad de Leon. (España). *Revista Iberoamericana de Automática e Informática Industrial*, 36-45.

Eguiluz, J. (2014). *Introducción a AJAX*. LibrosWeb.es. Obtenido de http://librosweb.es/ajax/

Figueroa, R. G., Solís, C. J., & Cabrera, A. A. (2013). *Metodologías Tradicionales vs. Metodologías Ágiles.* Universidad Técnica Particular de Loja, Escuela de Ciencias en Computación. Loja: Universidad Técnica Particular de Loja. Obtenido de http://adonisnet.files.wordpress.com/2008/06/articulo-metodologia-de-sw-formato.doc

Garret, J. J. (18 de Febrero de 2005). Ajax: A New Approach to Web Applications. *adaptive path*. Obtenido de https://courses.cs.washington.edu/courses/cse490h/07sp/readings/ajax_adaptive_path.pdf

Gosling, J., Joy, B., Steele, G., Bracha, G., & Buckley, A. (2015). *The Java Language Specification.* California: Oracle Corporation.

Guijarro, M. (2015). *postgres*.

Hilbert, M., Bustos, S., & Ferraz, J. C. (marzo de 2005). Estrategias nacionales para la sociedad de la información en América Latina y el CAribe. Santiago de Chile: CEPAL.

IBM. (2005). Rational Unified Process: Best practices for software development teams.

Jacobi, J., & Fallows, J. R. (2006). *Pro JSF and Ajax: Building Rich Internet Components.* New York: Apress.

Jendrock, E., Cervera Navarro, R., Evans, I., Haase, K., & Markito, W. (2014). *Java Platform, Enterprise Edition The Java EE Tutorial, Release 7.* Oracle Corporation.

Kruchten, P. (2004). *The Rational Unified Process. An Introduction.* Boston: Pearson Education, Inc.

Martín Gutiérrez, A. (2007). Integración curricular de las TIC y educación para los medios en la sociedad del conocimiento. *Revista Iberoamericana de educación*, 141-156.

Microsoft. (06 de 2015). *SQL Server 2014*. Obtenido de http://www.microsoft.com/es-es/server-cloud/products/sql-server/

Monge Nájera, J., Rivas Rossi, M., & Méndez Estrada, V. H. (2002). *La evolucion de los laboratorios virtuales durante una experiencia de cuatro años con estudiantes a distancia.* San José: Centro para el Mejoramiento de los Procesos Académicos (CEMPA), UNED.

Moral Pajares, E., & Rodríguez Cohard, J. C. (2007). Las TIC en la estrategia de desarrollo e integraciónde América Latina y el Caribe. *BOLETÍN ECONÓMICO DE ICE N° 2903*.

Oracle Corporation. (07 de 2010). *GlassFishV2Architecture*. Obtenido de https://wikis.oracle.com/display/GlassFish/GlassFishV2Architecture

Oracle Corporation. (03 de 2013). *GlassFish Server Open Source Edition.* . Obtenido de https://glassfish.java.net/docs/4.0/quick-start-guide.pdf

Oracle Corporation. (2013). *Notas de la version de Sun GlassFish Communications Server 4.0.* Oracle.

Oracle Corporation. (05 de 2015). *¿Qué es la tecnología Java y para qué la necesito?* Obtenido de https://www.java.com/es/download/faq/whatis_java.xml

Oracle Corporation. (06 de 2015). *GlassFish v3 Get Started*. Obtenido de https://glassfish.java.net/es/public/getstarted.html

Oracle Corporation. (06 de 2015). *JavaServer Faces Technology.* Obtenido de http://www.oracle.com/technetwork/java/javaee/javaserverfaces-139869.html

Oracle Corporation. (06 de 2015). *MySQL*. Obtenido de http://www.oracle.com/es/products/mysql/index.html

PostgreSQL. (10 de 2010). *Sobre PostgreSQL*. Obtenido de PostgreSQL-es: http://www.postgresql.org.es/sobre_postgresql

Pressman, R. S. (2010). *Ingeniería del software. Un enfoque práctico.* New York: McGraw-Hill.

Project Grizzly. (03 de 2015). *NIO Event Development Simplified*. Obtenido de Project Grizzly: https://grizzly.java.net/

Python Software Foundation. (2015). *The Python Tutorial.* Obtenido de https://docs.python.org/3/tutorial/index.html

Red Hat, Inc. (2007). Plataforma de Aplicaciones JBoss Enterprise.

Reenskaug, T. (12 de 06 de 2010). *Model-View-Controller*. Obtenido de http://folk.uio.no/trygver/themes/mvc/mvc-index.html

Rodriguez Cobos, E. M. (2009). Ventajas e inconvenientes de las TICs en el aula. *Cuadernos de Educación y desarrollo*.

Rosado, L., & Herreros, J. R. (2005). *Nuevas aportaciones didacticas de los laboratorios virtuales y remotos en la enseñanza de la Fisica.* Madrid: UNED.

Rothamel, L. (2014). La tecnología y su incorporación en las aulas para propiciar la construcción de conocimiento. En *Reflexión Académica en Diseño y Comunicación NºXXV* (págs. 134-135). Buenos Aires: Facultad de Diseño y Comunicación - Universidad de Palermo.

Schwaber, K., & Sutherland, J. (01 de 05 de 2015). *La Guía de Scrum.* Obtenido de Scrum.org: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-ES.pdf#zoom=100

Solana, F. A. (2013). *Index Adviser para PostgreSQL 9.2.* HABANA.

Solis Murillo, M. d. (2013). *Metodología para la autoevaluación institucional de la PUCESE basada en el modelo y estándares del CEAACES con fines de acreditación.* Esmeraldas: PUCESE.

Sunkel, G. (2010). TIC para la educación en América Latina. *Congreso Iberoamericano de Educación* . Buenos Aires.

The Apache Software Foundation. (06 de 2015). *Apache TomEE*. Obtenido de http://tomee.apache.org/apache-tomee.html

The PHP Documentation Group. (2015). *Manual de PHP*. Obtenido de php.net: http://php.net/manual/es/

UNESCO. (01 de 08 de 2014). *La Educación y las TIC*. Obtenido de http://www.unesco.org/es/higher-education/higher-education-and-icts/

Universidad ORT Uruguay. (2003). *Metodología XP*. Facultad de Ingeniería.

Wells, D. (8 de 10 de 2013). *Extreme Programming: A gentle introduction.* Obtenido de http://www.extremeprogramming.org/