Universidad de las Ciencias Informáticas

Facultad 4

"Integración entre el Juez en Línea Caribeño y su foro de discusión"

Trabajo de diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autor:

Humberto Esteban González Leyva

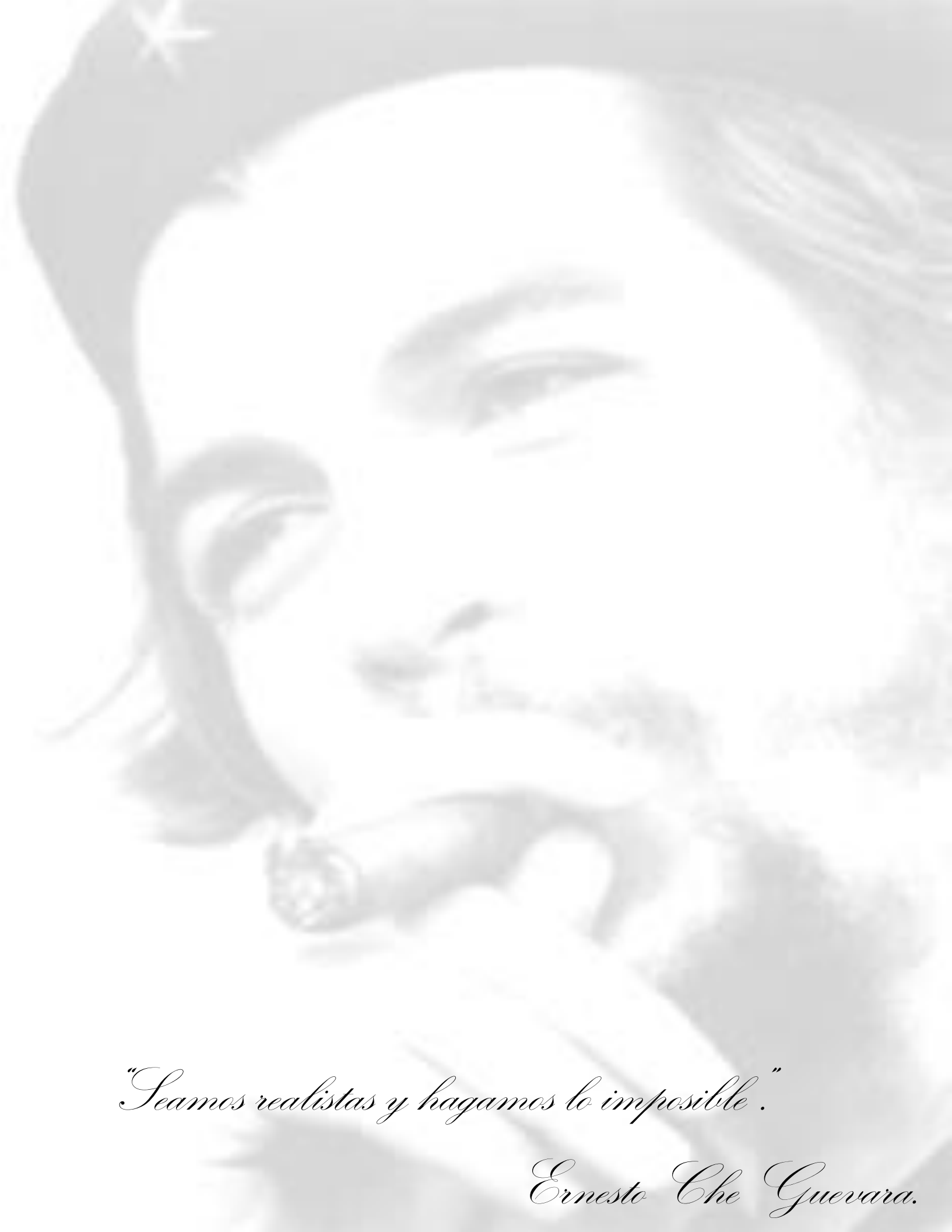
Tutores:

Ing. Yonny Mondelo Hernández

Ing. María Antonia Montesino Menéndez

La Habana, 2016

Año 58 de la Revolución



"Seamos realistas y hagamos lo imposible."

Ernesto Che Guevara.

Declaración de autoría

Declaro que soy el único autor de este trabajo y autorizo a la Universidad de las Ciencias Informáticas (UCI), para que hagan el uso que estimen pertinente con este trabajo.

Para que así conste firmo la presente a los ___días del mes de ____ del año ____.

_____

Firma del Autor

Humberto Esteban González Leyva

_____ _____

Firma del Tutor Firma de la Tutora

Ing. Yonny Mondelo Hernández Ing. María Antonia Montesino Menéndez

Datos de contacto

Humberto Esteban González Leyva

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: hegonzalez@estudiantes.uci.cu

Ing. Yonny Mondelo Hernández

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: ymondelo@uci.cu

Ing. María Antonia Montesino Menéndez

Universidad de las Ciencias Informáticas, La Habana, Cuba

Correo electrónico: marian@uci.cu

Dedicatoria

A mi familia y amigos por confiar

en mí y apoyarme siempre.

Agradezco:

A todo aquel que de una forma u otra me apoyó para que este día se hiciera realidad.

Resumen

El uso de los jueces en línea como plataforma de entrenamiento para los programadores se encuentra estrechamente ligado a la programación competitiva, término informático que se refiere a un tipo de competencia donde se ponen a prueba los conocimientos de programación. Debido a las necesidades de tener un propio sistema para el entrenamiento de la programación en la región del Caribe, surge el Juez en Línea Caribeño el cual ha alcanzado desde su surgimiento resultados más allá de las fronteras y brinda entre sus muchos servicios un foro de discusión virtual para el intercambio de ideas y opiniones. Dicha aplicación web no posee en la actualidad una relación automática con respecto al COJ, cada funcionalidad referente a las cuentas de usuarios de ambas interfaces y a las discusiones de los problemas del COJ se realiza manualmente, proceso tedioso y que tiene como consecuencias problemas de normalización de la información. Para dar solución a la problemática existente se desarrolló un módulo para automatizar las funcionalidades en ambas plataformas; haciendo uso de servicios web REST y obteniendo como resultado una completa normalización de la información y la automatización de procesos. Además, se garantizó la calidad del módulo y la veracidad de sus resultados con varios ciclos de prueba y el análisis de los criterios de aceptación de los administradores.

Palabras clave: juez en línea, foro, plataforma, aplicación web, servicios web, discusión virtual

Índice

Índice de ilustraciones

Índice de tablas

Introducción

La programación como concepto en la informática, es el proceso de escribir, analizar, probar, depurar y mantener el código fuente de un software con el objetivo de resolver un problema determinado mediante el uso de la computación (1). A su vez, la programación competitiva es un concurso donde varios programadores se someten a un examen en un tiempo limitado para resolver cierta cantidad de ejercicios mayormente matemáticos, lo cual genera un interés particular para las grandes empresas por la capacidad de resolver problemas reales. Esto conlleva a especialistas de la informática a dedicar una considerable cantidad de tiempo en la práctica y aumento de sus conocimientos en determinado lenguaje de programación y sus habilidades para escribir algoritmos (2).

Los Jueces en Línea son sistemas, por lo general web, que permiten evaluar de forma automática programas de computación que intenten solucionar tareas propuestas. La salida de cada código enviado por un usuario es capturada por el sistema y comparada contra la solución que se tiene de la tarea en cuestión o será evaluada por un evaluador externo en el caso que así sea requerido (3). Existen distintos tipos de jueces en línea, los que se dedican al entrenamiento de los lenguajes de acceso a bases de datos como SQL (Lenguaje de Consulta Estructurado, por sus siglas en inglés: *Structured Query Language*) y los que se especializan en la programación de algoritmos en diferentes lenguajes como C#, *C*++, *Java*[1], entre otros. Este último tipo de plataforma viene ganando prestigio desde el año 1997 con la publicación del Juez en Línea de la Universidad de Valladolid (UVA) en España (4).

En el 2006, programadores de la Universidad de las Ciencias Informáticas (UCI) y de la comunidad caribeña del movimiento Internacional de Competencias de Programación Inter-Colegios de la Asociación para Máquinas Computadoras (por sus siglas en inglés: ACM-ICPC) comenzaron el desarrollo del Juez en Línea Caribeño (COJ por sus siglas en inglés: *Caribbean Online Judge*) convirtiéndose en la plataforma principal de entrenamiento para los programadores. Los resultados de este proyecto trascienden las fronteras cubanas, incluso de la región caribeña (5).

El Juez en Línea Caribeño cuenta con varios servicios como mensajería interna para los usuarios registrados, recomendación de problemas, interfaz por correo electrónico para enviar soluciones y

[1] Lenguaje de programación de algoritmos

consultar el estado y foro de discusión, entre otros. El foro es un espacio de discusión virtual que se utiliza para el intercambio de conocimientos y opiniones en torno a los diferentes temas tratados en el COJ, leguajes de programación, competencias y concursos nacionales e internacionales, y el conjunto de ejercicios (6). Cada ejercicio del COJ posee su hilo de discusión en el foro, pero no existe un enlace automático entre ambas interfaces, funcionan como dos plataformas separadas, cuando se adiciona un ejercicio a la base de datos del COJ se debe crear después manualmente su respectiva discusión en el foro. Al no existir dicha relación automática existen problemas de actualización de las discusiones del foro, lo que genera errores de concordancia que conllevan a confundir a los usuarios y a un mal funcionamiento de los sitios. Por otra parte, cada plataforma tiene su propio sistema de control de acceso individual, cuando un usuario desea formular un comentario debe identificarse también en el foro de discusión, proceso que resulta tedioso si el usuario no posee un registro previo.

Atendiendo a la anterior problemática se identifica el siguiente **problema a resolver**: ¿Cómo lograr la integración entre el Juez en Línea Caribeño y su foro de discusión?

Siendo identificado como **objeto de estudio** de la investigación: el proceso de integración entre plataformas web.

Delimitándose como **campo de acción:** mecanismos para la integración entre aplicaciones web y tecnologías de gestión de foros de discusión.

Para darle solución al problema anteriormente planteado, se define como **objetivo general** de la investigación: desarrollar un módulo para lograr la integración entre el Juez en Línea Caribeño y su foro de discusión.

Para dar cumplimiento al **objetivo general** de la investigación se trazan los siguientes **objetivos específicos**:

1. Fundamentar los conceptos y características relacionadas con el Juez en Línea Caribeño y su foro de discusión.

2. Identificar los requisitos para determinar las funcionalidades y características del sistema.

3. Analizar y diseñar una propuesta de solución.
4. Implementar el módulo que logre la integración del Juez en Línea Caribeño y su foro de discusión.
5. Validar la solución propuesta.

Los **resultados esperados** tras la implementación son:

1. La normalización de la información del foro de discusión y el COJ.
2. Contar con un módulo que integre el Juez en Línea Caribeño y su foro de discusión.
3. Obtener la documentación asociada al desarrollo de la solución.
4. Tener a disposición el documento sobre la investigación.

Para el desarrollo de la investigación se utilizaron los siguientes métodos científicos:

Métodos teóricos:

- ✓ **Analítico-Sintético:** su objetivo en la investigación es analizar las teorías, documentos, etc., permitiendo la extracción de los elementos más importantes que se relacionan con el objeto de estudio.
- ✓ **Histórico-lógico:** su objetivo en la investigación es constatar teóricamente cómo ha evolucionado un determinado fenómeno en un período de tiempo, en toda su trayectoria o en un fragmento temporal de la lógica de su desarrollo.

Métodos empíricos:

- ✓ **Observación:** es el instrumento universal del científico, se realiza de forma consciente y orientada a un objetivo determinado.

Se caracteriza por:

- ✓ Puede utilizarse en distintos momentos de la investigación.

- ✓ Permite la recogida de la información de cada uno de los conceptos o variables definidas.
- ✓ El documento guía de la observación debe ser lo suficientemente preciso y claro para garantizar que diferentes observadores lo apliquen de igual forma

El trabajo de diploma está estructurado en los siguientes capítulos:

Capítulo I. Fundamentación teórica

Se documentan los principales conceptos relacionados con el Juez en Línea Caribeño y su foro de discusión, además de presentar otros sistemas similares a la solución deseada. Se realiza un estudio de las metodologías, lenguajes y herramientas necesarias para el desarrollo de la solución.

Capítulo II. Análisis y diseño

Este capítulo hace alusión a las fases de la metodología de desarrollo seleccionada, incluyendo las funcionalidades del sistema y las características del sistema presentes en el software. Se construyen los artefactos necesarios para la interpretación del producto requerido.

Capítulo III. Implementación y pruebas

En este capítulo se describe cómo fue desarrollada la solución y se describen las pruebas realizadas para lograr un producto con calidad que satisfaga las necesidades de los usuarios.

Capítulo I: Fundamentación teórica

Introducción

En este capítulo se abordan los elementos fundamentales para explicar por qué es necesaria la elaboración de la solución propuesta teniendo en cuenta la problemática existente. Además, se hace referencia al estudio de otros sistemas similares a la solución esperada.

Se expone la metodología de desarrollo de software a usar teniendo en cuenta el ciclo de vida de la producción en la uci, los lenguajes definidos para la implementación y las herramientas elegidas para elaborar la solución.

1.1 Definiciones de interés

Aplicación web: es cualquier aplicación a la que se accede vía web mediante una red como internet o una intranet. En general, el término también se utiliza para designar aquellos programas informáticos que son ejecutados en el entorno del navegador o codificado con algún lenguaje soportado por el navegador; confiándose en el navegador web para que reproduzca la aplicación.

Una de las ventajas de las aplicaciones web cargadas desde internet (u otra red) es la facilidad de mantener y actualizar dichas aplicaciones sin la necesidad de distribuir e instalar un software en, potencialmente, miles de clientes. También constituye una ventaja la posibilidad de ser ejecutadas en múltiples plataformas (7).

HTML: (por sus siglas en inglés: *Hyper Text Markup Language*) es el elemento de construcción más básico de una página web y se usa para crear y representar visualmente una página web. Determina el contenido de la página web, pero no su funcionalidad (8).

JavaScript: es un lenguaje ligero e interpretado, orientado a objetos con funciones de primera clase, más conocido como el lenguaje de script para páginas web. Es un lenguaje script multi-paradigma, basado en prototipos, dinámico, soporta estilos de programación funcional, orientado a objetos e imperativo (8).

1.2 Estudio de sistemas homólogos

UVa Online Judge

UVa Online Judge es un juez en línea automatizado para problemas de programación, organizado por la Universidad de Valladolid. Su archivo de problemas tiene más de 4300 ejercicios y los registros de usuarios están abiertos a todo el mundo. Posee un foro de discusión creado bajo la tecnología phpBB el cual cuenta con más de 8775 temas y 28835 usuarios registrados. Actualmente hay más de 100.000 usuarios registrados. Los usuarios pueden presentar las soluciones en diferentes lenguajes como son: C, C ++, Pascal y Java o C ++. Fue abierto al público en abril de 1997. En noviembre de 1999 y 2000, *Uva Online Judge* fue sede del concurso de programación ACM ICPC (4).

Timus Online Judge

Timus Online Judge un juez en línea ruso que contiene el archivo más grande de problemas de programación con un sistema de evaluación automática. Posee entre sus servicios un foro de discusión soportado por la tecnología phpBB. Los problemas se recogen en su mayoría de los concursos celebrados en la Universidad Federal de los Urales, Campeonato del Ural, concursos subregionales Ural ACM ICPC y campos de entrenamiento Petrozavodsk. *Timus Online Judge* ofrece la oportunidad de participar en versiones en línea de muchos concursos celebrados en la Universidad Federal de los Urales, abiertos para todo el mundo (9).

Sphere Online Judge

SPOJ es un juez en línea con más de 200.000 usuarios registrados y más de 20.000 problemas. Las tareas se crean por su comunidad o se toman de concursos de programación anteriores. SPOJ permite a los usuarios avanzados organizar concursos bajo sus propias reglas e incluye un foro donde los programadores pueden discutir cómo resolver un problema particular, dicho foro creado por la tecnología phpBB. Además del idioma inglés, SPOJ también ofrece su contenido en los idiomas polaco, portugués y vietnamita. La solución a los problemas puede ser presentada en más de 40 lenguajes de programación. También permite a los usuarios comparar paradigmas y enfoques con una amplia variedad de lenguajes (10).

A partir del estudio de los anteriores Jueces en Línea, se puede constar que todos poseen un foro de discusión utilizando la tecnología phpBB que no es más que un sistema de foros gratuito basado en un conjunto de paquetes de código programados en el lenguaje de programación web PHP, cuya intención es la de proporcionar fácilmente, y con amplia posibilidad de personalización, una herramienta para crear comunidades. Su nombre es por la abreviación de PHP *Bulletin Board*[2], también es conocido como phpBB3 por su última versión (11). Todos los ejemplos constan de este sistema de foros al igual que el COJ, pero el enlace automático requerido no existe en ellos.

1.3 Metodología de desarrollo de software

Las metodologías de desarrollo de software son un conjunto de procedimientos, técnicas y ayudas a la documentación para el desarrollo de productos de software. Se van detallando todas las actividades a realizar para lograr el producto informático deseado, indicando además qué personas deben participar en el desarrollo de las actividades y qué papel deben desempeñar. Además, detallan la información que se debe producir como resultado de una actividad y la necesaria para comenzarla (12). Existe una gran variedad de metodologías para la creación del software que se clasifican en dos grandes grupos: las metodologías tradicionales o pesadas y las metodologías ligeras o ágiles.

Metodologías pesadas

Son las más tradicionales, se centran en la definición detallada de los procesos y tareas a realizar, herramientas a utilizar, y requiere una extensa documentación, ya que pretende prever todo de antemano. Este tipo de metodologías son más eficaces y necesarias cuanto mayor es el proyecto que se pretende realizar respecto a tiempo y recursos que son necesarios emplear, donde una gran organización es requerida (12).

Algunas de las metodologías pesadas existentes son:

- RUP *(Rational Unified Process)*
- MSF *(Microsoft Solution Framework)*

[2] Sistema para la creación de foros virtuales

Metodologías ligeras/ágiles

Surgen como una alternativa a las metodologías tradicionales. Su filosofía se centra en el individuo, la colaboración con el cliente, que este sea partícipe del proceso de desarrollo del software, y el desarrollo incremental del producto software en sí con iteraciones cortas. Se ajustan a proyectos de corta duración y cuya planificación no debe ser estricta, de modo que responda a los cambios (requisitos, tecnologías) que pueden ocurrir a lo largo del proyecto (12).

Principales ideas de la metodología ágil:

- Se encarga de valorar al individuo y las iteraciones del equipo más que a las herramientas o los procesos utilizados.
- Se hace mucho más importante crear un producto (software) que funcione que escribir mucha documentación.
- El cliente está en todo momento colaborando en el proyecto.
- Es más importante la capacidad de respuesta ante un cambio realizado que el seguimiento estricto de un plan.

Algunas de las metodologías ágiles existentes son:

- XP (*Extreme Programming*)
- AUP *(Agile Unified Process)*
- SCRUM

Atendiendo a los rasgos fundamentales de la problemática se decide seleccionar una metodología ágil ya que las mismas permiten una respuesta rápida a cambios de requisitos a lo largo del desarrollo del software, permiten la entrega continua y en plazos cortos de un software funcional y que cada componente del producto final ha sido probado y satisface los requerimientos.

De las metodologías ágiles estudiadas se escoge AUP-UCI, una variación de la metodología AUP, que se adapta al ciclo de vida definido para la actividad productiva de la UCI.

AUP-UCI

Al no existir una metodología de software universal, ya que toda metodología debe ser adaptada a las características de cada proyecto (equipo de desarrollo, recursos, etc.) exigiéndose así que el proceso sea configurable, se decide hacer una variación de la metodología AUP, de forma talque se adapte al ciclo de vida definido para la actividad productiva de la UCI.

AUP-UCI cuenta con 7 disciplinas al igual que AUP, pero a un nivel más atómico.

- ✓ Los flujos de trabajos: modelado de negocio, requisitos, análisis y diseño, en AUP están unidos en la disciplina Modelo, en la variación para la UCI se consideran a cada uno de ellos disciplinas.
- ✓ Se mantiene la disciplina implementación, en el caso de prueba se desagrega en 3 disciplinas: pruebas internas, de liberación y aceptación.
- ✓ Las restantes 3 disciplinas de AUP asociadas a la parte de gestión, para la variación UCI serían: gestión de la configuración, planeación de proyecto y monitoreo y control de proyecto.

Descripción de los 4 escenarios definidos por AUP-UCI:

1. Proyectos que modelen el negocio con CUN (Caso de uso del negocio) solo pueden modelar el sistema con CUS (Casos de uso del sistema).
2. Proyectos que modelen el negocio con MC (Modelo conceptual) solo pueden modelar el sistema con CUS (Casos de uso del sistema).
3. Proyectos que modelen el negocio con DPN (Descripción de proceso de negocio) solo pueden modelar el sistema con DRP (Descripción de Requisitos por Proceso).
4. Proyectos que no modelen negocio solo pueden modelar el sistema con HU (Historias de usuario) (13).

Debido a las características del proyecto y a que no se modela negocio, solo se puede modelar la solución con HU, lo que representa el escenario 4.

1.4 Lenguajes de programación definidos

Los lenguajes que se utilizarán para el desarrollo de la solución serán los definidos en ambas plataformas. En el Juez en Línea Caribeño se utilizarán como lenguajes de desarrollo Java, JavaScript y CSS[3]; y en el foro de discusión php[4]. Además se usa como lenguaje de acceso a base de datos el *SQL*[5]en ambos casos.

PHP

Es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en *HTML*[6].No necesita ser compilado para ejecutarse. Puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo. Se considera uno de los lenguajes más flexibles, potentes y de alto rendimiento conocido (14).

Java

Java es un lenguaje de programación utilizado para realizar cualquier tipo de programa. En la actualidad es un lenguaje muy extendido y cada vez cobra más importancia tanto en el ámbito de Internet como en la informática en general. Fue desarrollado por la compañía *Sun Microsystems* (15).

Una de las principales características por las que Java se ha hecho muy famoso es que es un lenguaje independiente de la plataforma. Eso quiere decir que, al crear un programa en Java, este podrá funcionar en cualquier ordenador del mercado. Es una ventaja significativa para los desarrolladores de software, pues antes tenían que hacer un programa para cada sistema operativo, por ejemplo, Windows, Linux, Apple, etc. Esto lo consigue porque se ha creado una Máquina de Java para cada sistema que hace de puente entre el sistema operativo y el programa de Java y posibilita que este último se entienda perfectamente.

CSS3

[3] Lenguajes de programación
[4] Lenguaje de programación
[5] Lenguaje de acceso a base de datos
[6] Lenguaje de programación

Hoja de estilo en cascada o CSS (por sus siglas en inglés: *Cascading Style Sheets*) es un lenguaje usado para definir y crear la presentación de un documento estructurado escrito en HTML o XML2.

Algunas de las funcionalidades que ofrece son:

- ✓ Propiedades de las fuentes como tipo, tamaño, énfasis...
- ✓ Color de texto, fondos, bordes u otros elementos.
- ✓ Atributos del texto, como espaciado entre palabras, letras, líneas, etcétera.
- ✓ Alineación de textos, imágenes, tablas u otros.
- ✓ Propiedades de caja, como margen, borde, relleno o espaciado.
- ✓ Propiedades de identificación y presentación de listas (16).

JavaScript

JavaScript es un lenguaje de scripting multiplataforma y orientado a objetos. Es un lenguaje pequeño y liviano. Dentro de un ambiente de host, JavaScript puede conectarse a los objetos de su ambiente y proporcionar el control programático sobre ellos. La herencia se modela mediante prototipos de mecanismo, y es posible añadir propiedades y métodos a cualquier objeto de forma dinámica (17).

SQL

SQL es el lenguaje estándar ANSI/ISO de definición, manipulación y control de bases de datos relacionales. Es un lenguaje declarativo: sólo hay que indicar qué se quiere hacer. En cambio, en los lenguajes procedimentales es necesario especificar cómo hay que hacer cualquier acción sobre la base de datos. El SQL es un lenguaje muy parecido al lenguaje natural; concretamente, se parece al inglés, y es muy expresivo. Por estas razones, y como lenguaje estándar, el SQL es un lenguaje con el que se puede acceder a todos los sistemas relacionales comerciales (18).

1.5 Tecnologías definidas para la solución

Partiendo de los requerimientos definidos para la implementación del sistema y las características de los mismos, se realizó un estudio de las tendencias y tecnologías existentes en la actualidad para el desarrollo de la solución.

1.5.1 Framework de desarrollo

Un *framework*[7] es una estructura de soporte definida, en la cual otro proyecto de software puede ser organizado y desarrollado.

Suelen incluir:

- ✓ Soporte de programas.
- ✓ Bibliotecas.
- ✓ Lenguaje de scripting.
- ✓ Software para desarrollar y unir diferentes componentes de un proyecto de desarrollo de programas.

Permiten:

- ✓ Facilitar el desarrollo de software.
- ✓ Evitar los detalles de bajo nivel, permitiendo concentrar más esfuerzo y tiempo en identificar los requerimientos de software (19).

Spring

Es un *framework* para el desarrollo de aplicaciones y contenedor de inversión de control, de código abierto para la plataforma Java. Es el más popular de los *frameworks Java*. Proporciona varios módulos los cuales abarcan la mayor parte de las funcionalidades a utilizar en cualquiera de las capas de la aplicación (20).

Bootstrapv3.0

[7] Marco de trabajo

Es un *framework* que simplifica el proceso de creación de diseños web combinando *CSS* y *JavaScript*. Ha sido desarrollado por *Twitter*[8]. Posee como mayor ventaja, poder crear interfaces que se adapten a los distintos navegadores apoyándose en un *framework* potente con numerosos componentes webs que permiten el ahorro de esfuerzo y tiempo.

- ✓ Permite crear interfaces que se adapten a los diferentes navegadores, tanto de escritorio como *tablets*[9] y móviles a distintas escalas y resoluciones.
- ✓ Se integra perfectamente con las principales librerías Javascript, por ejemplo *JQuery*[10].
- ✓ Ofrece un diseño sólido usando estándares como CSS3/HTML5.

Funciona con todos los navegadores, incluido Internet Explorer (21).

1.5.2 Lenguaje unificado de modelado

UML es un popular lenguaje de modelado de sistemas de software. Se trata de un lenguaje gráfico para construir, documentar, visualizar y especificar un sistema de software. Entre otras palabras, UML se utiliza para definir un sistema de software.

Posee la riqueza suficiente como para crear un modelo del sistema, pudiendo modelar los procesos de negocios, funciones, esquemas de bases de datos, expresiones de lenguajes de programación, etc. (22)

1.5.2.1 Herramientas CASE

Las herramientas de ingeniería de software asistida por computadora (CASE por sus siglas en inglés: *Computer Aided Software Engineering*), son aplicaciones computacionales en conjunto que soportan y ayudan al proceso de análisis y desarrollo de software. Sirven a los analistas de sistemas, ingenieros de software y desarrolladores en el proceso de modelado, durante todos los pasos del Ciclo de Vida del desarrollo del software (23).

---

[8] Red social
[9] Dispositivo móvil
[10] Librería JavaScript

Rational Rose

IBM Rational Rose Enterprise proporciona un conjunto de prestaciones controladas por modelo para desarrollar muchas aplicaciones de software, incluidas aplicaciones Ada, ANSI C++, C++, CORBA, Java, Java EE, Visual C++ y Visual Basic. El software permite acelerar el desarrollo de estas aplicaciones con código generado a partir de modelos visuales mediante el lenguaje UML (Unified Modeling Language).

Rational Rose Enterprise ofrece una herramienta y un lenguaje de modelado común para simplificar el entorno de trabajo y permitir una creación más rápida de software de calidad.

- ✓ Modelado de las aplicaciones más habituales: proporciona prestaciones de modelado visual para desarrollar muchos tipos de aplicaciones de software.
- ✓ Desarrollo de aplicaciones para la web: contiene herramientas web y XML para el modelado de aplicaciones web.
- ✓ Integración del diseño de aplicaciones con el desarrollo: unifica el equipo del proyecto proporcionando una ejecución y una notación de modelos UML comunes (24).

Erwin

PLATINUM Erwin es una herramienta de diseño de base de datos. Brinda productividad en diseño, generación, y mantenimiento de aplicaciones. Desde un modelo lógico de los requerimientos de información, hasta el modelo físico perfeccionado para las características específicas de la base de datos diseñada (25).

EasyCASE

Esta herramienta permite automatizar las fases de análisis y diseño dentro del desarrollo de una aplicación, para poder crear las aplicaciones eficazmente desde procesamiento de transacciones a la aplicación de bases de datos de cliente/servidor, así como sistemas de tiempo real (25).

Oracle Designer

Oracle Designer es un juego de herramientas para guardar las definiciones que necesita el usuario y automatizar la construcción rápida de aplicaciones cliente/servidor flexibles y gráficas (25).

Power Designer

Power Designer es una suite de aplicaciones de Powersoft para la construcción, diseño y modelado de datos a través de diversas aplicaciones. Es la herramienta para el análisis, diseño inteligente y construcción sólida de una base de datos y un desarrollo orientado a modelos de datos a nivel físico y conceptual, que dan a los desarrolladores Cliente/Servidor la más firme base para aplicaciones de alto rendimiento.

System Architect

Esta herramienta posee un repositorio único que integra todas las herramientas, y metodologías usadas. En la elaboración de los diagramas, el System Architect conecta directamente al diccionario de datos, los elementos asociados, comentarios, reglas de validaciones, normalización, etc. (25)

Visual Paradigm 8.0

Visual Paradigm es una herramienta CASE profesional fácil de utilizar, que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientado a objetos, construcción, pruebas y despliegue. Es una herramienta de software libre, de probada utilidad para el análisis, está disponible para varias plataformas e idiomas y posee licencia libre y comercial. Su diseño es centrado en casos de uso y enfocado al negocio. Tiene la capacidad de ingeniería directa e inversa, usa un lenguaje estándar para la buena comunicación entre los desarrolladores, y sus modelos y códigos se sincronizan durante todo el ciclo de vida del proyecto. Genera códigos en lenguaje Java y exporta HTML (25).

Por todas estas características Visual Paradigm es la herramienta que se define para el desarrollo de la solución.

1.5.3 Sistemas gestores de base de datos

Un sistema gestor de base de datos (SGBD) es el software que permite la utilización y/o actualización de los datos almacenados en una o varias bases de datos, por uno o varios usuarios desde diferentes puntos de vista. El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le

permitan manipular los datos, de forma que no le sea necesario conocer el modo de almacenamiento ni el método de acceso empleado (26).

Tanto el Juez en Línea Caribeño y el foro de discusión utilizan:

PostgreSQL 9.3

PostgreSQL es un potente sistema de gestión de bases de datos relacional, multiusuario, centralizado y de propósito general, que está siendo desarrollado desde 1977 y está liberado bajo la licencia *Berkeley Software Distribution* (BSD). Es ampliamente considerado como el sistema gestor de base de datos de código abierto más avanzado del mundo.

El gestor de bases de datos *PostgreSQL* es robusto y por ende hoy en día es muy usado con respecto a otros gestores libres existentes como *SQLite, MySQL, FireBird*[11], ya que presenta una mayor posibilidad de escalado y rendimiento bajo grandes cargas de trabajo. A continuación, se relacionan algunas de las principales ventajas.

- ✓ Instalación ilimitada: no hay costo asociado a la licencia de software.
- ✓ Soporte: existe una gran comunidad de profesionales y empresas que ofrecen soporte a *PostgreSQL*, de la cual el Grupo Global de Desarrollo de *PostgreSQL* es la principal.
- ✓ Estabilidad y compatibilidad legendarias: debido a la cantidad de compañías que reportan su buen funcionamiento.
- ✓ Multiplataforma: soporta alrededor de 34 plataformas, incluyendo Linux y Unix en todas sus variantes y Windows (27).

1.5.4 Herramienta de administración de base de datos

El software que permite la utilización y/o la actualización de los datos almacenados en una (o varias) base(s) de datos por uno o varios usuarios desde diferentes puntos de vista y a la vez, se denomina sistema de gestión de bases de datos (SGBD). El objetivo fundamental de un SGBD consiste en suministrar al usuario las herramientas que le permitan manipular, en términos abstractos, los datos, o

[11] Gestores de base de datos

sea, de forma que no le sea necesario conocer el modo de almacenamiento de los datos en la computadora, ni el método de acceso empleado (28).

La herramienta que se utiliza en el Juez en Línea Caribeño y en el foro de discusión es PgAdmin3, además se utilizó en el proceso de desarrollo la herramienta MySQL Workbench 6.3.

PgAdmin 3

PgAdmin3 es una aplicación gráfica para gestionar las bases de datos *PostgreSQL*[12], siendo la más completa y popular de código abierto. Es capaz de gestionar versiones a partir de *PostgreSQL* 7.3 ejecutándose en cualquier plataforma. Está diseñado para responder a las necesidades de los usuarios, desde escribir consultas SQL simples hasta desarrollar bases de datos complejas. La interfaz gráfica soporta todas las características de *PostgreSQL* y facilita enormemente la administración. La aplicación también incluye un editor SQL, un editor de código de la parte del servidor y un agente para lanzar scripts programados (29).

MySQL Workbench 6.3

Es una herramienta visual de diseño de bases de datos que integra desarrollo de software, administración de bases de datos, diseño de bases de datos, creación y mantenimiento para el sistema de base de datos MySQL (30).

1.5.5 Servidor de aplicaciones web

Un servidor web es un programa que procesa una aplicación que realiza la conexión entre el servidor y los clientes, recibiendo y/o entregando información, esta información se puede ver en los navegadores web. Básicamente un servidor web consta de un intérprete HTTP el cual se mantiene a la espera de peticiones de clientes y le responde con el contenido según sea solicitado. El cliente, una vez recibido el código, lo interpreta y lo exhibe en pantalla (31).

El Juez en Línea Caribeño utiliza como servidor web Apache Tomcat7 y el foro de discusión utiliza Apache2.

[12] Sistema gestor de base de datos

Servidor Apache

El servidor Apache es un servidor web de código abierto, flexible, rápido y eficiente, continuamente actualizado y adaptado a nuevos protocolos. Puede ser adaptado a diferentes entornos y necesidades. Es extensible, gracias a ser modular se han desarrollado diversas extensiones entre las que destaca PHP, un lenguaje de programación del lado del servidor. Permite la configuración de mensajes de errores personalizados (32).

Apache Tomcat 7

Apache Tomcat (también llamado Jakarta Tomcat o simplemente Tomcat) funciona como un contenedor de *servlets*[13] (clase en el lenguaje de programación Java, utilizada para ampliar las capacidades de un servidor) desarrollado bajo el proyecto Jakarta en la *Apache Software Foundation*. Tomcat implementa las especificaciones de los *servlets* y de *JavaServer Pages* (JSP) de *Oracle Corporation* (aunque creado por *Sun Microsystems*). Los usuarios disponen de libre acceso a su código fuente y a su forma binaria en los términos establecidos en la *Apache Software License* (32).

Apache 2

Apache 2 es el servidor web más utilizado en los sistemas Linux. Es desarrollado y mantenido por una comunidad de usuarios bajo la supervisión de la *Apache Software Foundation* dentro del proyecto HTTP Server. Apache 2 presenta entre otras características altamente configurable, bases de datos de autenticación y negociado de contenido, pero fue criticado por la falta de una interfaz gráfica que ayude en su configuración (32).

1.5.6 Entorno de desarrollo integrado (IDE)

Un Entorno de Desarrollo Integrado (IDE), es una aplicación que facilita el desarrollo de aplicaciones. De manera general un IDE de desarrollo consiste en un editor de código, un compilador, un depurador y una interfaz gráfica, que ayudan a un desarrollador de software a crear aplicaciones de forma independiente o

[13] Clase del lenguaje de programación Java

que sean parte de aplicaciones ya existentes. Existen diferentes entornos de desarrollo entre los que se encuentran PHPStorm, Eclipse, Intellij IDEA y NetBeans[14] (33).

Para el desarrollo de la solución propuesta se seleccionó Intellij IDEA 15.0 para el trabajo en el COJ y PHPStorm 8.0 para el foro de discusión.

Intellij IDEA 15.0

IntelliJ IDEA se autodefine como un entorno inteligente para desarrollar aplicaciones Java, cliente y servidor. Es un completísimo IDE que también permite desarrollar aplicaciones para móviles. Posee un avanzado editor de código, compatible con multitud de tecnologías. Permite la escritura de código sin complicaciones. Practica un abordaje no intrusivo e intuitivo para ayudarle a escribir, depurar, probar y aprender su código. Gracias a su profunda comprensión de los lenguajes y tecnologías, IntelliJ IDEA proporciona un segundo par de manos para cuando se necesite. Crea un entorno adecuado en el que todos los miembros del equipo pueden trabajar juntos de manera eficiente. Integración transparente con una amplia variedad de sistemas de control de versiones lo que permite a los miembros del equipo permanecer en sincronía con los cambios de otros, asegurando que todas las contribuciones serán productivas (34).

PHPStorm 8.0

PHPStorm es un IDE de programación desarrollado por JetBrains[15]. Es uno de los entornos de programación más completos de la actualidad, permite editar código no sólo del lenguaje de programación php como lo indica su nombre. Actualmente es compatible con Sistemas Operativos Windows, Linux y Mac OS X.

Principales características:

- ✓ Permite la gestión de proyectos fácilmente.
- ✓ Proporciona un fácil autocompletado de código.
- ✓ Soporta el trabajo con PHP 5.5

[14] Entornos de desarrollo
[15] Compañía de desarrollo de software

- ✓ Sintaxis abreviada (35).

Conclusiones parciales:

Como parte del desarrollo de este capítulo se determinan las siguientes conclusiones parciales:

- ✓ El análisis de los referentes teóricos y metodológicos, así como la aplicación de los métodos y las técnicas durante la investigación permitió hacer una valoración respecto a la necesidad de mejorar la integración entre el Juez en Línea Caribeño y su foro de discusión.
- ✓ El estudio de sistemas homólogos posibilitó el entendimiento acerca del funcionamiento de estos sistemas, y la identificación de elementos comunes que pueden ser incorporados a la solución esperada.
- ✓ El estudio de las herramientas y tecnologías permitió seleccionar las más adecuadas para el desarrollo de la solución partiendo de los requerimientos definidos en el levantamiento de información.

Capítulo II: Análisis y diseño

Introducción

La metodología AUP-UCI define en cada una de sus fases un grupo de artefactos que brindan una comprensión clara del negocio y el sistema a desarrollar. Al tener la misma un enfoque ágil, permite que el equipo elabore la documentación necesaria que facilite la comunicación entre los desarrolladores y las partes interesadas. En este capítulo se presentarán los siguientes artefactos: modelo de dominio, la especificación de requisitos funcionales y no funcionales que tendrá la propuesta de solución, se modelarán las Historias de usuario y la interacción de los actores con el sistema.

2.1 Modelo de dominio

Un modelo del dominio o también denominado modelo conceptual es una representación visual de las clases conceptuales u objetos del mundo real en un dominio de interés. Este captura los tipos más importantes de objetos en el contexto del sistema. Muchos de los objetos del dominio o clases pueden obtenerse de una especificación de requisitos o mediante la entrevista con los expertos del medio. El objetivo del modelado del dominio es comprender y describir las clases más importantes dentro del medio (36).

Conceptos del dominio

Usuario: persona registrada en el COJ o en el foro de discusión, que tiene permisos concedidos para enviar soluciones al sistema y hacer comentarios en los hilos de discusión.

Foro: espacio virtual de discusión para el intercambio de ideas, comentarios y aclaración de dudas referentes a temas específicos.

Ejercicio: situación a resolver mediante la concepción de un algoritmo escrito en uno de los lenguajes de programación soportados por el Juez en Línea Caribeño.

Discusión: discurso o conversación en la que se intercambian puntos de vista, ponencias y críticas entre dos o más personas sobre un tema propuesto a debatir.

Cuenta: las cuentas de usuario sirven para identificar al actor de forma unívoca. Son personales e intransferibles.

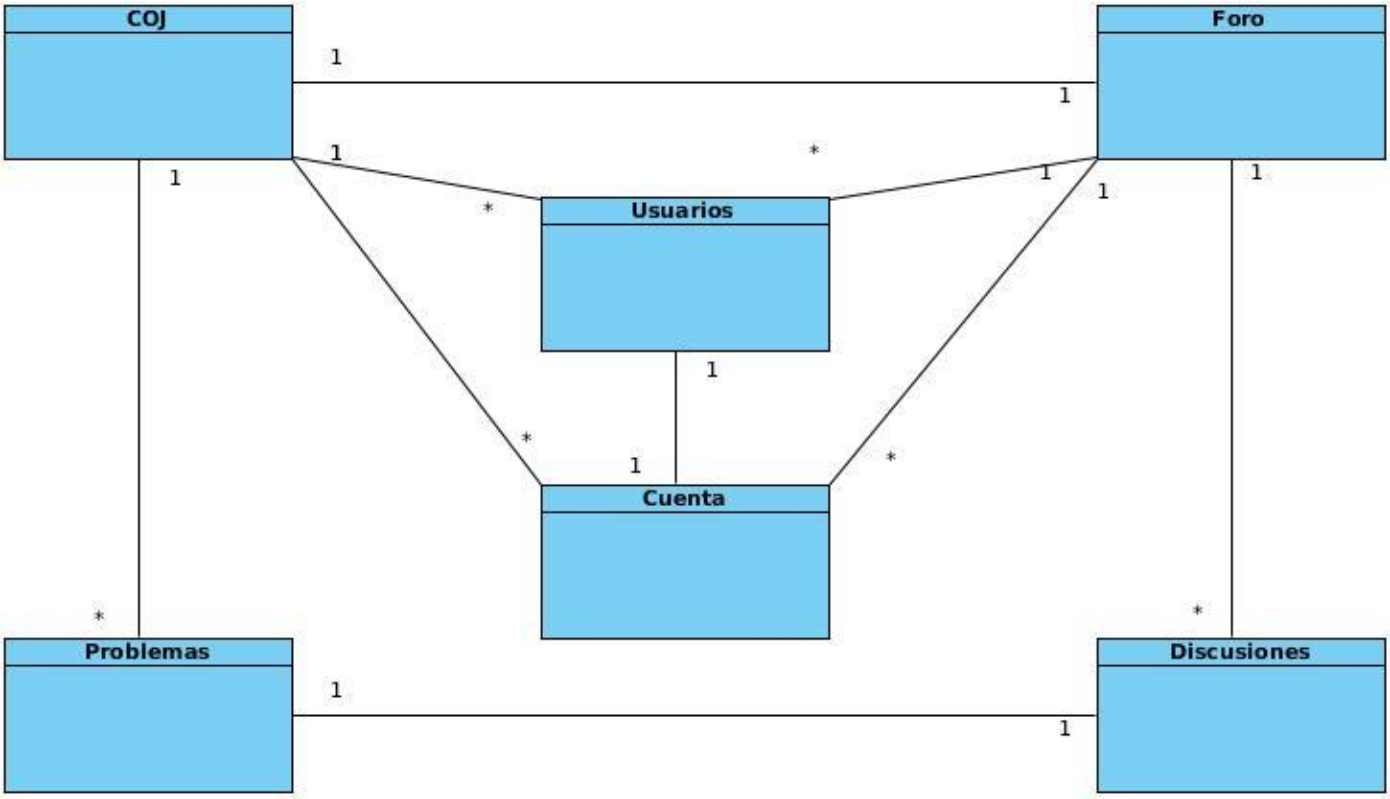


Ilustración 1 - Modelo de dominio

2.2 Propuesta de solución

Debido a la necesidad de interconectar funcionalidades e intercambiar datos e información entre ambas plataformas se realizó un estudio de las diferentes tecnologías diseñadas con dichos fines, escogiéndose para la solución el uso de servicios web para dar cumplimiento a los objetivos trazados.

Un servicio web es un conjunto de aplicaciones o de tecnologías con capacidad para operar en la web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la web.

Proporcionan mecanismos de comunicación estándares entre diferentes aplicaciones, que interactúan entre sí para presentar información dinámica al usuario. Para proporcionar interoperabilidad y extensibilidad entre estas aplicaciones, y que al mismo tiempo sea posible su combinación para realizar operaciones complejas (37).

Tecnologías de desarrollo de servicios web y comparación en cuanto a su uso:

- ✓ REST (*Representational State Transfer*)
- ✓ SOAP (*Simple Object Access Protocol*)
- ✓ XML-RPC (*Extensible Markup Language - Remote Procedure Call*)
- ✓ JavaScript

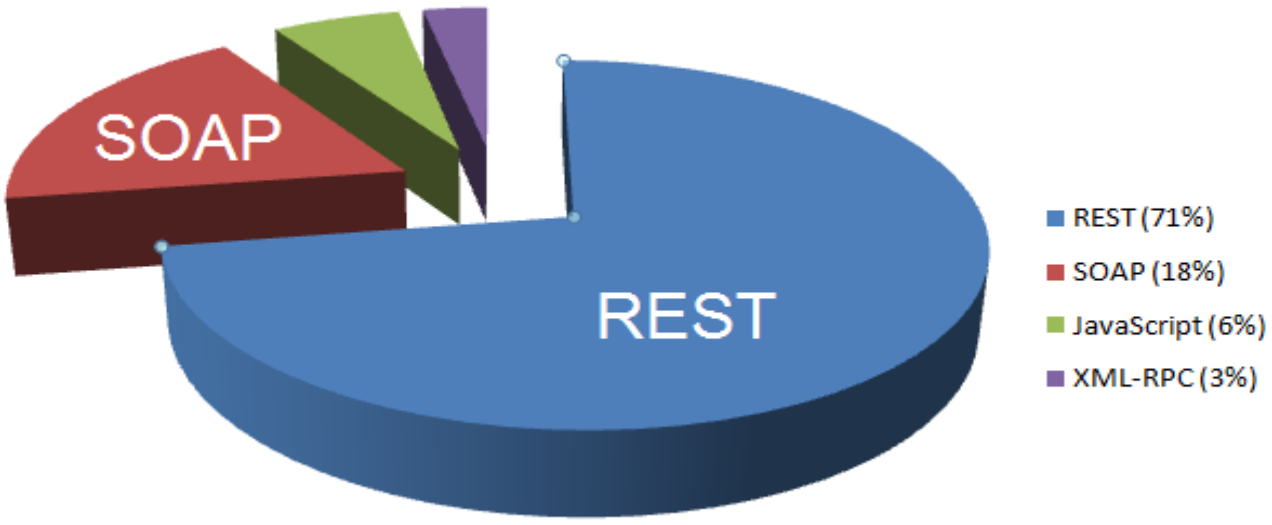


Ilustración 2 - Comparación entre las tecnologías de desarrollo de servicios web

Luego de realizado el estudio de las tecnologías de desarrollo de servicios web y atendiendo a la comparación entre las mismas, se decide utilizar REST.

REST:

REST define un set de principios arquitectónicos por los cuales se diseñan servicios web haciendo foco en los recursos del sistema, incluyendo cómo se accede al estado de dichos recursos y cómo se transfieren por *HTTP*[16] hacia clientes escritos en diversos lenguajes. REST emergió en los últimos años como el modelo predominante para el diseño de servicios. De hecho, REST logró un impacto tan grande en la web que prácticamente logró desplazar a SOAP y las interfaces basadas en WSDL[17] por tener un estilo bastante más simple de usar (38).

2.3 Requerimientos del sistema

La definición de requisitos del sistema permite establecer las capacidades de este y detallar las funciones, servicios y restricciones operativas que el sistema debe cumplir. Dichos requisitos son planteados con el fin de lograr un entendimiento claro entre el cliente y el equipo de desarrollo en aras de satisfacer sus verdaderas necesidades.

2.3.1 Requisitos funcionales del sistema

RF1.1 Registrar usuario en el foro a partir del registro en el COJ.

RF1.2 Actualizar perfil de usuario.

RF2 Normalizar cuentas de usuario.

RF2.1 Listar propuestas de normalización de cuentas de usuarios.

RF2.2 Ejecutar propuestas de normalización de cuentas de usuarios.

RF2.3 Listar usuarios del COJ sin cuentas en el foro.

RF2.4 Replicar usuarios del COJ sin cuentas en el foro.

RF2.5 Listar usuarios del foro sin cuentas en el COJ.

RF2.6 Borrar usuarios del foro sin cuentas en el COJ.

[16] Protocolo de comunicación
[17] Formato que se utiliza para describir servicios Web

RF3.1 Crear entrada en el foro a partir del registro de un problema en el COJ.

RF3.2 Actualizar entrada en el foro a partir de la actualización de un problema en el COJ.

RF4 Normalizar Problemas del COJ y entradas del foro.

RF4.1 Listar propuestas de normalización de problemas del COJ y entradas del foro.

RF4.2 Ejecutar propuestas de normalización de problemas del COJ y entradas del foro.

RF4.3 Listar problemas del COJ sin entradas en el foro.

RF4.4 Replicar problemas del COJ sin entradas en el foro.

RF4.5 Listar entradas del foro sin problemas en el COJ.

RF4.6 Borrar entradas del foro sin problemas en el COJ.

RF5.1 Listar últimas entradas en el foro de problemas del COJ.

RF6 Gestionar registros de acciones.

RF6.1 Crear registro.

RF6.2 Listar registros.

2.3.2 Requisitos no funcionales del sistema

Usabilidad

El sistema debe poseer una interfaz fácil de utilizar para cualquier tipo de usuario con conocimientos básicos de informática y en el manejo de ordenadores.

Seguridad

El acceso a la información debe estar restringido por usuario, contraseña y rol, siendo el rol Administrador, el que tendrá el control total sobre la información del sistema.

Cuando un usuario se autentique o registre en el sistema se cifrará la contraseña del usuario mediante el algoritmo MD5, tanto para almacenarla en la base de datos, cuando se produzca un registro nuevo como para compararla cuando se inicie sesión.

Rendimiento

El tiempo de respuesta del sistema a cada acción que ejecute el usuario no debe ser superior a 6 segundos.

Accesibilidad

El sistema debe estar disponible desde cualquier estación de trabajo conectada a la red y permitir su administración de forma remota.

2.4 Historias de usuario

Las historias de usuario son un instrumento para el levantamiento de requerimientos en el desarrollo de un software, que ha emergido con la aparición de los nuevos marcos de trabajo de desarrollo ágil. Son descripciones cortas de una necesidad del cliente del software. El tratamiento de las historias de usuario es muy dinámico y flexible, ya que en cualquier momento las historias de usuario pueden romperse, reemplazarse por otras más específicas o generales, añadirse nuevas o ser modificadas. Cada historia de usuario es lo suficientemente comprensible y delimitada para que los programadores puedan implementarla en unas semanas (39).

A continuación, se presentan 3 ejemplos de historias de usuario de varios requisitos funcionales**.** Las demás historias de usuario se encuentran en los anexos.

Tabla 1 HU – 2.1 Listar propuestas de normalización de cuentas de usuarios

Número: 3.1	**Nombre del requisito: Listar propuestas de normalización de cuentas de usuarios**	
Programador: Humberto Esteban González Leyva		**Iteración Asignada: 1ra**
Prioridad: Media		**Tiempo Estimado: 1h**
Riesgo en Desarrollo:		**Tiempo Real:**
Descripción: El sistema debe mostrar una lista con los usuarios existentes en ambas interfaces con igual dirección de correo electrónico pero con datos diferentes.		
Observaciones		
Prototipo de interfaz		

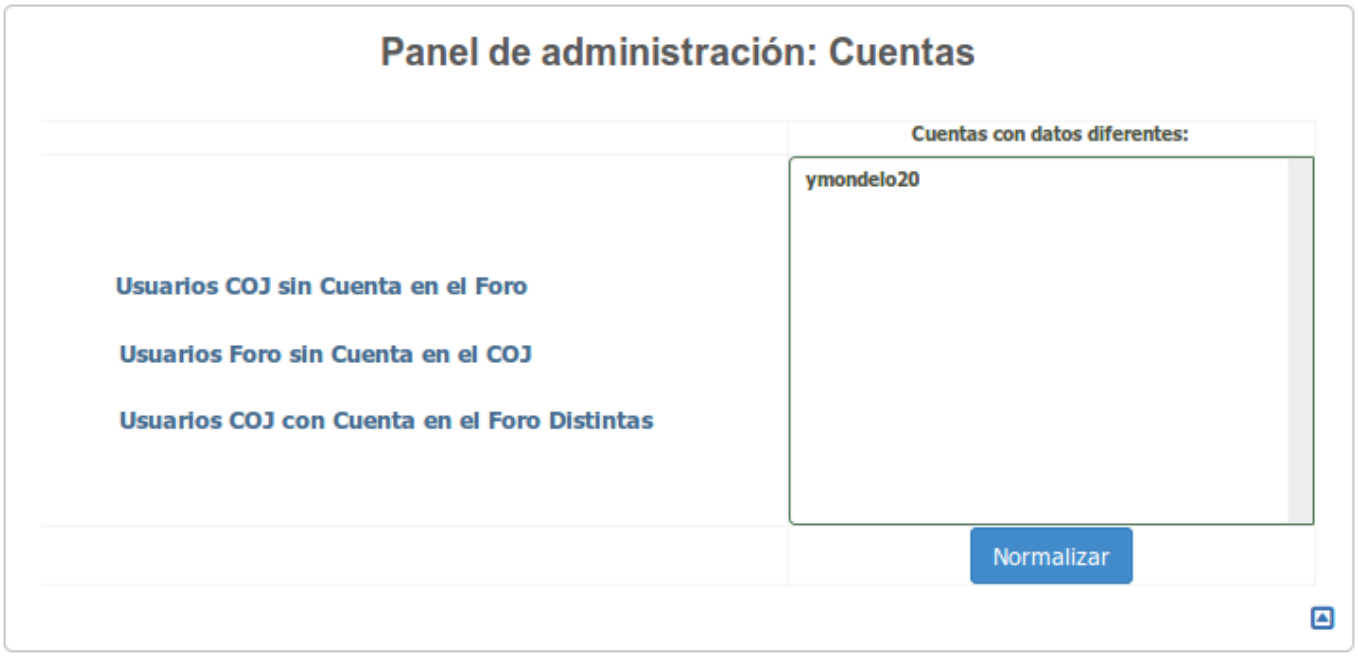


Tabla 2 HU – 2.3 Listar usuarios del COJ sin cuentas en el foro

Número: 2.3	**Nombre del requisito: Listar usuarios del COJ sin cuentas en el foro**
Programador: Humberto Esteban González Leyva	**Iteración Asignada: 1ra**
Prioridad: Media	**Tiempo Estimado: 1h**
Riesgo en Desarrollo:	**Tiempo Real:**

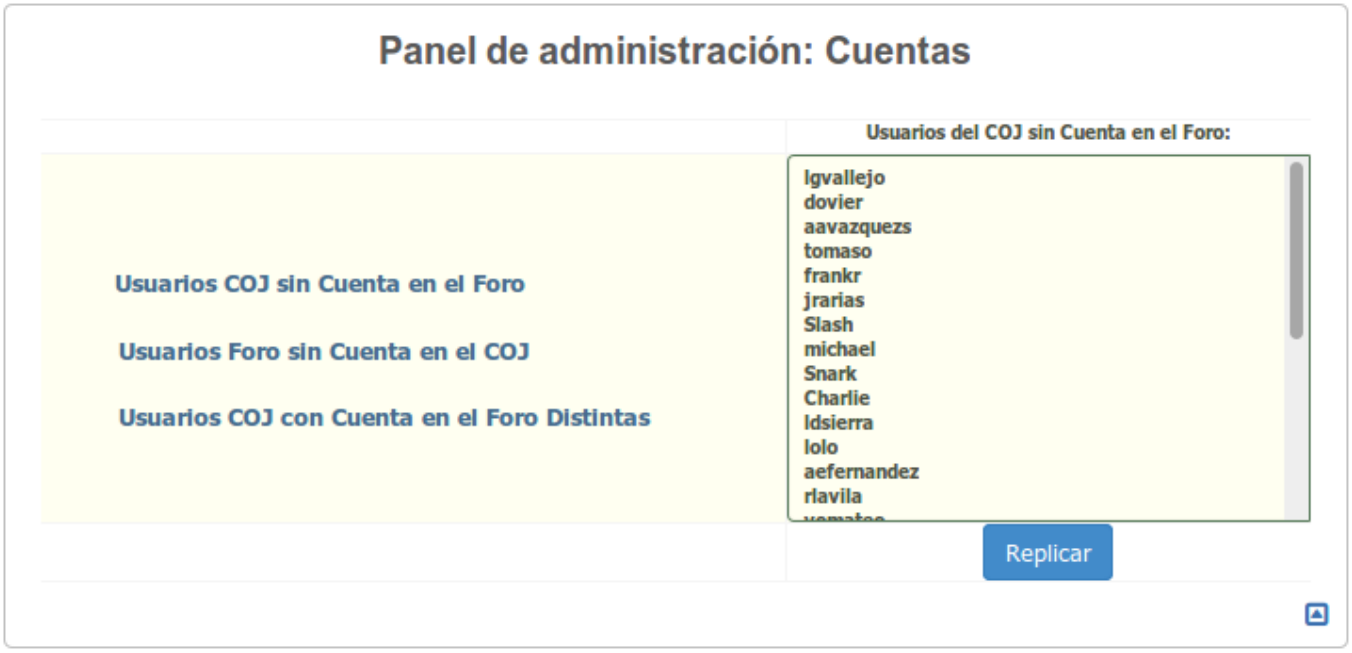
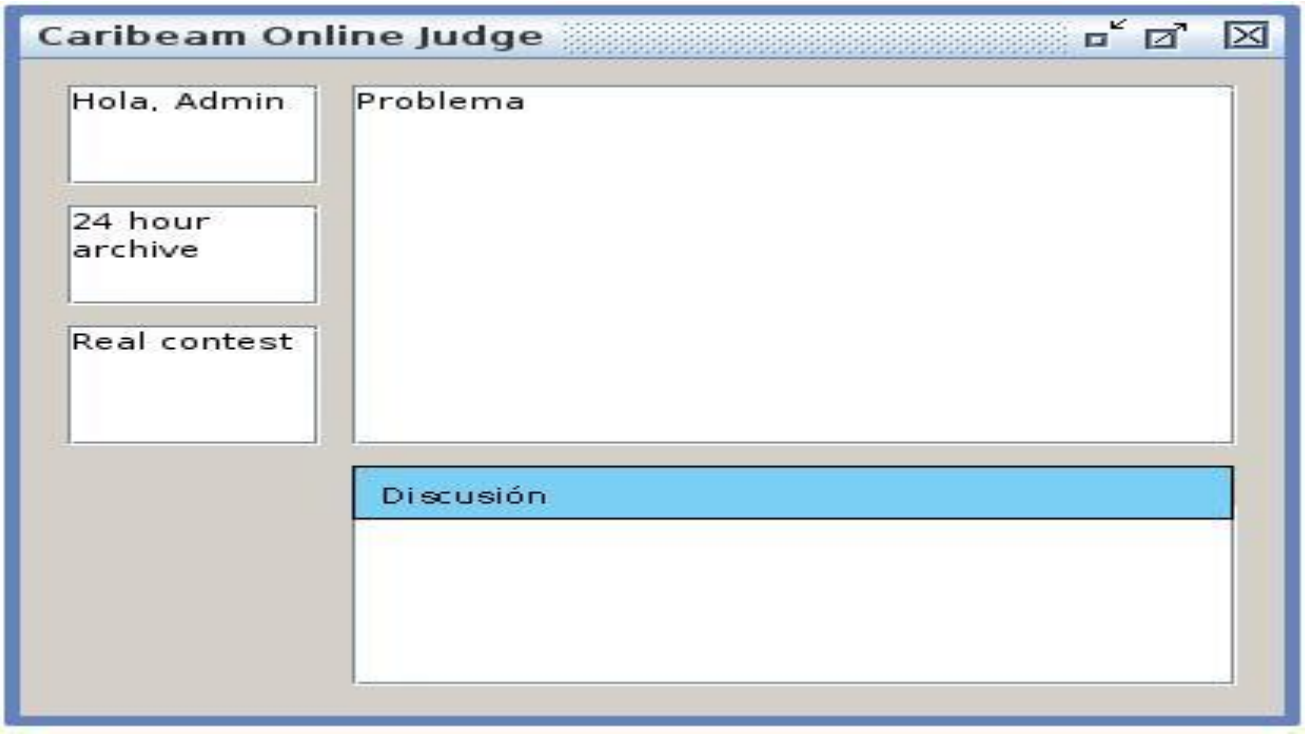
Descripción: El sistema debe listar los usuarios registrados en el COJ que no poseen una cuenta en el foro de discusión.
Observaciones
Prototipo de interfaz   

Tabla 3 HU – 5.1 Listar últimas entradas en el foro de problemas del COJ

Número: 5.1	**Nombre del requisito: Listar últimas entradas en el foro de problemas del**

	COJ
Programador: Humberto Esteban González Leyva	**Iteración Asignada: 2ra**
Prioridad: Media	**Tiempo Estimado: 1h**
Riesgo en Desarrollo:	**Tiempo Real:**
Descripción: El sistema debe listar las últimas entradas del foro de discusión de cada problema del COJ debajo de la descripción del problema.	
Observaciones	
Prototipo de interfaz	



2.5 Diseño de la solución propuesta

Es el primer paso en la fase de desarrollo de cualquier producto o sistema de ingeniería. Define como el proceso de aplicar distintas técnicas y principios con el propósito de definir un dispositivo, proceso o sistemas con los suficientes detalles como para permitir su realización física. El objetivo del diseño es que se produzca un modelo o representación de la entidad que será construida más adelante.

2.5.1 Diagrama de clases del análisis

Es un tipo de diagrama de estructura estática que describe la estructura de un sistema mostrando las clases del sistema, sus atributos, operaciones, y las relaciones entre los objetos.

A continuación, se muestra el diagrama de clases de análisis, referentes al requisito funcional **1.1 - Registrar usuario en el foro a partir del registro en el COJ**, los demás diagramas de clases de análisis se encuentran en los anexos:

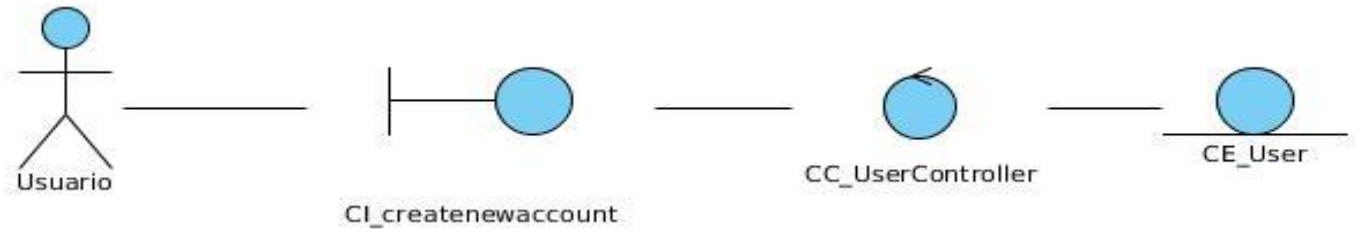


Ilustración 3 - Diagrama de clase de análisis

2.5.2 Diagrama de colaboración del análisis

Es esencialmente un diagrama que muestra interacciones organizadas alrededor de los roles. También llamados diagramas de comunicación, muestran explícitamente las relaciones de los roles. Muestra cómo las instancias específicas de las clases trabajan juntas para conseguir un objetivo común. Implementan las asociaciones del diagrama de clases mediante el paso de mensajes de un objeto a otro (40).

A continuación, se muestra el diagrama de colaboración del análisis del requisito funcional **5.1 - Listar últimas entradas en el foro de problemas del COJ,** los restantes diagramas correspondientes a los demás requisitos funcionales se encuentran en los anexos:

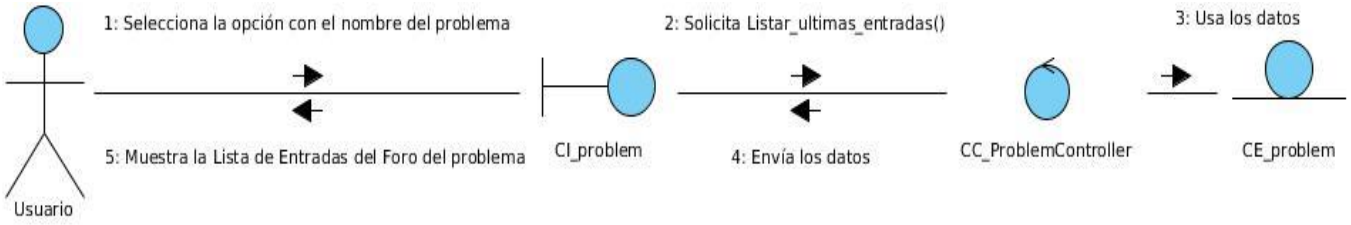


Ilustración 4 - Diagrama de colaboración de análisis

2.5.3 Arquitectura de software

La arquitectura de software es un conjunto de patrones que proporcionan un marco de referencia necesario para guiar la construcción de un software, permitiendo a los programadores, analistas y todo el conjunto de desarrolladores del software compartir una misma línea de trabajo y cubrir todos los objetivos

y restricciones de la aplicación. Es considerada el nivel más alto en el diseño de la arquitectura de un sistema puesto que establecen la estructura, funcionamiento e interacción entre las partes del software.

La arquitectura de software hace uso de un conjunto de patrones y abstracciones que conforman un hilo conductor que guía el desarrollo del software. Es por tal razón que se hace necesario realizar la selección de los patrones arquitectónicos y de diseños a emplear en el desarrollo de la aplicación (41).

2.5.3.1 Patrones arquitectónicos

Los patrones arquitectónicos se utilizan para expresar una estructura de organización base o esquema para un software. Proporcionando un conjunto de subsistemas predefinidos, especificando sus responsabilidades, reglas, directrices que determinan la organización, comunicación, interacción y relaciones entre ellos (42).

El Juez en Línea Caribeño está creado bajo el patrón arquitectónico Modelo-Vista-Controlador.

Arquitectura Modelo-Vista-Controlador

Modelo-Vista-Controlador es un patrón de diseño de software para programación que propone separar el código de los programas por sus diferentes responsabilidades. MVC es una propuesta de diseño de software utilizada para implementar sistemas donde se requiere el uso de interfaces de usuario. Surge de la necesidad de crear software más robusto con un ciclo de vida más adecuado, donde se potencie la facilidad de mantenimiento, reutilización del código y la separación de conceptos.

Descripción de la Arquitectura Modelo-Vista-Controlador

El **Modelo** es el objeto que representa los datos del programa. Maneja los datos y controla todas sus transformaciones. El modelo no tiene conocimiento específico de los controladores o de las vistas, ni siquiera contiene referencias a ellos. Es el propio sistema el que tiene encomendada la responsabilidad de mantener enlaces entre el modelo y sus vistas, y notificar a las vistas cuando cambia el modelo.

La **Vista** es el objeto que maneja la presentación visual de los datos representados por el Modelo. Genera una representación visual del Modelo y muestra los datos al usuario. Interactúa con el Modelo a través de una referencia al propio Modelo.

El **Controlador** es el objeto que proporciona significado a las órdenes del usuario, actuando sobre los datos representados por el Modelo. Cuando se realiza algún cambio, entra en acción, bien sea por cambios en la información del Modelo o por alteraciones de la Vista. Interactúa con el Modelo a través de una referencia al propio Modelo.

Ventajas de la Arquitectura Modelo-Vista-Controlador

Brinda una separación total entre lógica de negocio y presentación. Se le pueden aplicar opciones como el multilenguaje, distintos diseños de presentación, etc. sin alterar la lógica de negocio. La separación de capas como presentación, lógica de negocio, acceso a datos es fundamental para el desarrollo de arquitecturas consistentes, reutilizables y más fáciles de mantener, lo que al final resulta en un ahorro de tiempo en desarrollo en posteriores proyectos.

Al existir la separación de vistas, controladores y modelos es más sencillo realizar labores de mejora como:

- ✓ Agregar nuevas vistas.
- ✓ Agregar nuevas formas de recolectar las órdenes del usuario (interpretar sus modelos mentales).
- ✓ Las vistas también son susceptibles de modificación sin necesidad de provocar que todo el sistema se paralice. Adicionalmente el patrón MVC propende a la especialización de cada rol del equipo, por tanto en cada liberación de una nueva versión se verán los resultados (43).

2.5.3.2 Patrones de Diseño

Los patrones de diseño expresan esquemas para definir estructuras de diseño (o sus relaciones) con las que construir sistemas de software. Describen la estructura comúnmente recurrente de los componentes en comunicación, que resuelve un problema general de diseño en un contexto particular (44).

Patrones GRASP

Los patrones GRASP (por sus siglas en inglés: *General Responsibility Assignment Software Patterns*) describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones (45).

En el desarrollo de la solución se emplean los siguientes patrones GRASP:

El patrón **Experto**: se basa en que la responsabilidad de realizar una tarea es de la clase que posee los datos involucrados; una clase contiene toda la información necesaria para realizar la tarea que tiene encomendada (45).

El patrón **Controlador:** se asocia con operaciones del sistema y respuestas a sus eventos, tal como se relacionan los mensajes y los métodos. El controlador delega en otros objetos el trabajo que se necesita hacer pero coordina o controla la actividad (45).

El patrón **Alta Cohesión**: se manifiesta en la medida en que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos y auto identificable, una clase con baja cohesión hace muchas cosas no relacionadas o hace demasiado trabajo (45).

El patrón **Bajo Acoplamiento**: asigna la responsabilidad de controlar el flujo de eventos del sistema a clases específicas. Esto facilita la centralización de actividades como las validaciones y la seguridad de la aplicación (45).

```
ConexionForo cxn = new ConexionForo();
boolean b = cxn.CrearHiloDiscucionForo(title, enlace, word_text);
userDAO.insertLink(dir + cxn.Consumir_Servicioh_WSLink()[0].getTopic_id(), problem.getPid());
```

Ilustración 5 - Ejemplo del uso del patrón controlador

2.5.4 Diagrama de clases del diseño

El diagrama de clases del diseño describe gráficamente las especificaciones de las clases de software y de las interfaces en una aplicación.

A continuación, se adjunta el diagrama de clases del diseño referente a los requisitos funcionales **1.1 - Registrar usuario en el foro a partir del registro en el COJ** y **1.2 - Actualizar perfil de usuario del foro a partir de la actualización del usuario en el COJ,** los restantes diagramas de clases del diseño se encuentran en los anexos:

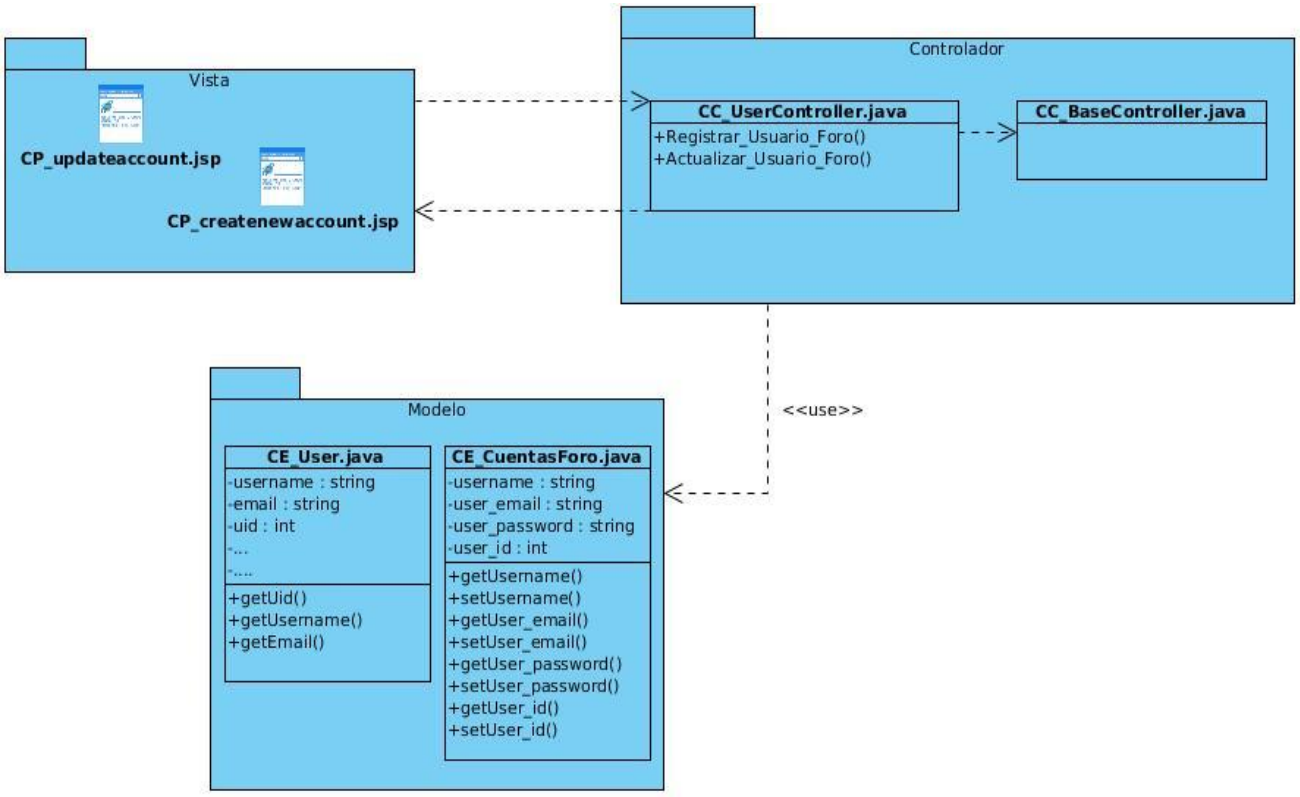


Ilustración 6 - Diagrama de clases del diseño

2.5.5 Diagrama de secuencia del diseño

Los diagramas de secuencia ilustran las interacciones entre objetos con el transcurso del tiempo en un tipo de formato con aspecto de una valla, en el que cada objeto nuevo se añade a la derecha. Estos muestran los objetos participes de la interacción y la secuencia de los mensajes intercambiados (46).

A continuación, se muestra el diagrama de secuencia de la historia de usuario **RF5.1 - Listar últimas entradas en el foro de problemas del COJ,** los restantes diagramas de secuencia se encuentran en los anexos:

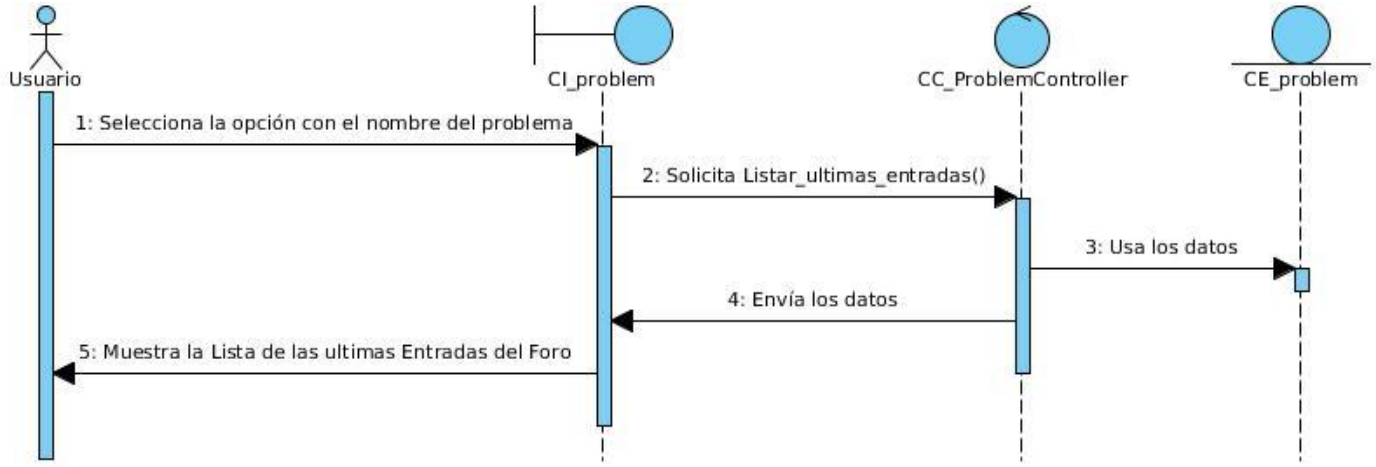


Ilustración 7 - Diagrama de secuencia

2.5.6 Modelo de despliegue

El modelo de despliegue es un modelo de objetos, que describe la distribución física del sistema en términos de cómo se distribuye la funcionalidad entre los nodos interconectados. Estos nodos representan elementos de hardware sobre los cuales pueden ejecutarse los elementos de software. Se utiliza como entrada fundamental en las actividades de diseño e implementación debido a que la distribución del sistema tiene una influencia principal en su diseño (47).

En la figura se muestra el diagrama de despliegue que corresponde a la solución propuesta.

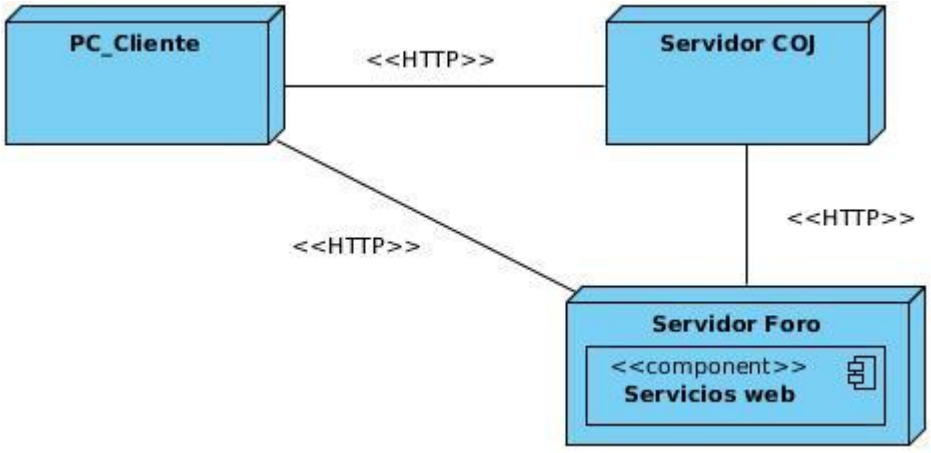


Ilustración 8 - Diagrama de despliegue

Conclusiones parciales

Como parte del desarrollo de este capítulo se determinan las siguientes conclusiones parciales:

- ✓ El modelado del dominio permitió conocer las características de los objetos del campo de acción y su relación entre ellos.
- ✓ El levantamiento de los requisitos funcionales y no funcionales permitió identificar las funcionalidades deseadas para la implementación.
- ✓ La propuesta de solución definió el curso del desarrollo de la solución.
- ✓ Las Historias de usuario y los artefactos definidos ayudaron a la comprensión de las funcionalidades a desarrollar.

Capítulo III: Implementación y prueba

Introducción

En el presente capítulo se procede a realizar la implementación y las pruebas de software para garantizar la obtención de un producto de calidad. Se crea el diagrama de componentes del sistema y se definen los estándares de codificación empleados para el desarrollo de la solución. La realización de pruebas a un producto asegura la satisfacción del cliente con el mismo, para lograr esto son definidos los niveles y técnicas de pruebas que le serán aplicadas. Se diseñan también casos de pruebas que servirán para llevar a cabo estas comprobaciones de la calidad.

3.1 Diagrama de componentes

Un diagrama de componentes muestra la división entre los distintos elementos de software que componen un sistema y las relaciones existentes entre los mismos. Permite visualizar la estructura de alto nivel del sistema y el comportamiento del servicio que estos componentes proporcionan y usan a través de interfaces (48).

Seguidamente se adjunta el diagrama de componentes:

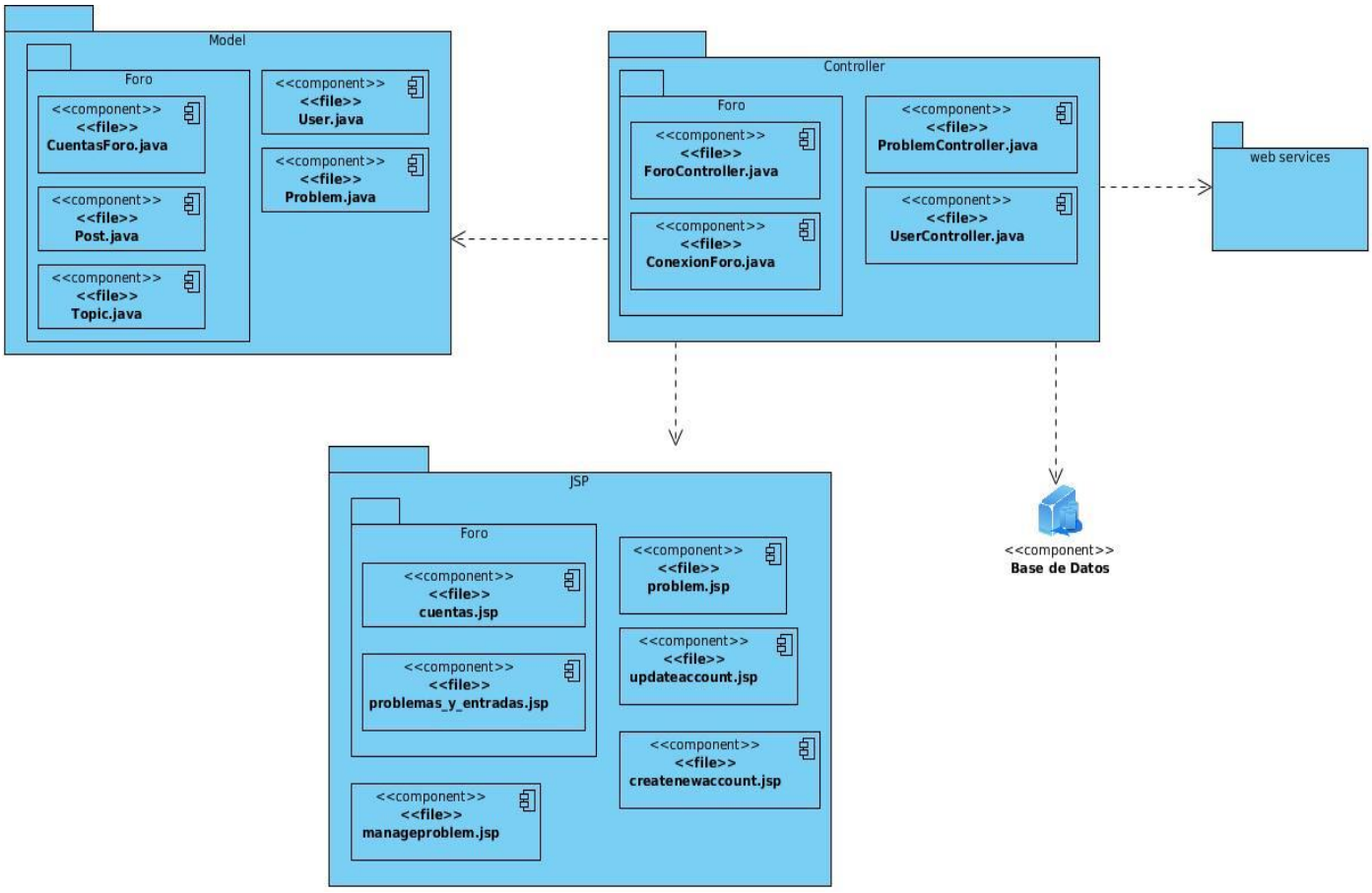


Ilustración 9 - Diagrama de componentes

3.2 Estándares de codificación

Un estándar de codificación es un conjunto de reglas que se utilizan para escribir archivos de código fuente con el objetivo de lograr estructuras de código mucho más comprensibles e identificables para otros programadores diferentes al autor (49).

Para el desarrollo de la aplicación se decide utilizar como estándar de codificación el mismo utilizado para el desarrollo del COJ.

CamelCase:

Es un tipo de escritura que se caracteriza por la unión de palabras sin espacios; con la peculiaridad de que la primera letra de cada término se encuentra en mayúscula para hacer más legible el conjunto.

El CamelCase admite dos posibles combinaciones entre mayúsculas y minúsculas: la primera palabra en mayúscula y el resto en minúscula (UpperCamelCase) o, por el contrario, cuando la primera está en minúscula y las demás están en mayúscula (LowerCamelCase) (50).

Ejemplo del uso de UpperCamelCase:

```
public int getTopic_id(){}
```

Ejemplo del uso de LowerCamelCase:

```
public boolean DeleteUserForo(int user_id) {}
```

3.3 Pruebas de software

Las pruebas de software consisten en la dinámica de la verificación del comportamiento de un programa en un conjunto finito de casos de prueba, debidamente seleccionados de por lo general infinitas ejecuciones de dominio, contra la del comportamiento esperado. Son una serie de actividades que se realizan con el propósito de encontrar los posibles fallos de implementación, calidad o usabilidad de un programa, probando el comportamiento del mismo (51).

Objetivos

1. Detectar defectos en el software.
2. Verificar la integración adecuada de los componentes.
3. Verificar que todos los requisitos se han implementado correctamente.
4. Identificar y asegurar que los defectos encontrados se han corregido antes de entregar el software al cliente.

5. Diseñar casos de prueba que sistemáticamente saquen a la luz diferentes clases de errores, haciéndolo con la menor cantidad de tiempo y esfuerzo.

3.3.1 Niveles de prueba

Cuando se le van a aplicar pruebas a un software, se tienen en cuenta una serie de objetivos en diferentes escenarios y niveles de trabajo, debido a que las pruebas son agrupadas por niveles que se encuentran en distintas etapas del proceso de desarrollo (51).

Pruebas unitarias

La prueba de unidad es la primera fase de las pruebas dinámicas y se realizan sobre cada módulo del software de manera independiente. El objetivo es comprobar que el módulo, entendido como una unidad funcional, está correctamente codificado.

Se enfocan en un programa o un componente que desempeña una función específica que puede ser probada y que se asegura que funcione tal y como lo define la especificación del programa. Los programadores siempre prueban el código durante el desarrollo, por lo que las pruebas unitarias son realizadas solamente después de que el programador considera que el componente está libre de errores.

Consiste en una prueba estructural enfocada a los elementos más pequeños del software. Esta prueba es aplicable a componentes representados en el modelo de implementación, para verificar que los flujos de control y de datos estén cubiertos y que ellos funcionen como se espera (51).

Pruebas de aceptación

Son realizadas principalmente por los usuarios con el apoyo del equipo del proyecto. El propósito es confirmar que el sistema está terminado, que desarrolla puntualmente las necesidades de la organización y que es aceptado por los usuarios finales.

La prueba de aceptación puede tener lugar a lo largo de semanas o meses, descubriendo así errores acumulados que pueden ir degradando el sistema. Es muy recomendable que las pruebas de aceptación se realicen en el entorno en que se va a explotar el sistema incluyendo el personal que lo va a manejar (51).

3.3.2 Técnicas de prueba

Las técnicas de prueba facilitan una guía sistemática para diseñar pruebas que comprueben la lógica interna de los componentes de software y verifiquen los dominios de entrada y salida del programa para descubrir errores en la funcionalidad, el comportamiento y el rendimiento (51). Seguidamente se describen las pruebas de caja blanca que será la técnica utilizada para comprobar la calidad de la solución.

Pruebas de caja blanca

Son un tipo de pruebas de software que se realiza sobre las funciones internas de un módulo. Así como las pruebas de caja negra ejercitan los requisitos funcionales desde el exterior del módulo, las de caja blanca están dirigidas a las funciones internas. Entre las técnicas usadas se encuentran; la cobertura de caminos (pruebas que hagan que se recorran todos los posibles caminos de ejecución), pruebas sobre las expresiones lógico-aritméticas, pruebas de camino de datos (definición-uso de variables), comprobación de bucles (se verifican los bucles para 0,1 e interacciones, y luego para las interacciones máximas, máximas menos uno y más uno). Las pruebas de caja blanca se llevan a cabo en primer lugar, sobre un módulo concreto, para luego realizar las de caja negra sobre varios subsistemas (51).

Pruebas de caja negra

Las pruebas de caja negra se centran en los requisitos funcionales. Son pruebas que se realizan sobre la interfaz del software, enfocadas en las entradas y salidas y no en el código fuente.

Muchos autores consideran que estas pruebas permiten encontrar:

1. Funciones incorrectas o ausentes.
2. Errores de interfaz.
3. Errores en estructuras de datos o en accesos a las Bases de Datos externas.
4. Errores de rendimiento.
5. Errores de inicialización y terminación (51).

3.3.3 Casos de prueba

Consiste en el diseño de pruebas que tengan la mayor probabilidad de encontrar el mayor número de errores con la mínima cantidad de esfuerzo y de tiempo. Un buen caso de prueba es aquel que tiene una alta probabilidad de mostrar un error no descubierto hasta entonces (51). Para probar el correcto funcionamiento de la aplicación se diseñaron un conjunto de casos de prueba que así lo confirman. A continuación, se muestran los casos de prueba de cada historia de usuario.

Tabla 4 CP – HU1.1 Registrar usuario en el foro a partir del registro en el COJ

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC1.1 Registrar Usuario.	Se selecciona la opción registrar usuario del COJ.	El sistema muestra una página de registro de cuentas donde el usuario debe rellenar los campos para crear la nueva cuenta.	Inicio/Registrar cuenta de usuario
EC1.2 Crear.	Se selecciona la opción Crear.	El sistema verifica que se hayan introducido los datos necesarios, que estén correctamente y que no existan otros usuarios ya registrados con las mismas credenciales. Luego envía un email al usuario pidiendo la activación de la cuenta.	Inicio/Registrar cuenta de usuario/Crear
EC1.3 Activar cuenta.	Se selecciona la opción	El sistema activa la cuenta	Email/Activar cuenta

	Activar cuenta en el email de activación de cuenta.	del COJ. Envía los datos necesarios al foro de discusión creando una réplica de la cuenta del COJ.	

Tabla 5 CP – HU1.2 Actualizar perfil de usuario del foro al actualizar usuario en el COJ

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC1.1 Editar cuenta.	Se selecciona la opción Editar cuenta del COJ.	El sistema muestra una página de actualización de cuentas donde el usuario debe rellenar los campos que considera debe actualizar.	Inicio/Editar cuenta
EC1.2 Editar.	Se selecciona la opción Editar.	El sistema verifica que se hayan introducido los datos a editar correctamente, actualiza ambas cuentas, en el COJ y la cuenta correspondiente en el foro de discusión, y muestra el perfil del usuario.	Inicio/Editar cuenta /Editar

Tabla 6 CP – HU2.1Listar propuestas de normalización de cuentas de usuarios

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar propuestas de normalización de cuentas de usuario.	Se selecciona la opción Cuentas.	El sistema muestra una página con la lista de los usuarios registrados en el COJ y en el foro de discusión con igual email pero diferentes credenciales.	Inicio/Administrador/Cuentas

Tabla 7 CP – HU2.2 Ejecutar propuestas de normalización de cuentas de usuarios

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Ejecutar propuestas de normalización de cuentas de usuarios	Se selecciona la opción Normalizar.	El sistema actualiza los datos del usuario en el foro de discusión con los datos de la cuenta registrada en COJ.	Inicio/Administrador/Cuentas/Normalizar

Tabla 8 CP – HU2.3 Listar usuarios del COJ sin cuentas en el foro

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar usuarios del COJ sin cuentas en el foro	Se selecciona la opción Cuentas.	El sistema muestra una página con la lista de los usuarios registrados en el COJ sin	Inicio/Administrador/Cuentas/

		registro en el foro de discusión.	

Tabla 9 CP – HU2.4 Replicar usuarios del COJ sin cuentas en el foro

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC1.1 Replicar usuarios del COJ sin cuentas en el foro	Se selecciona el usuario del COJ a replicar. Se selecciona la opción Replicar.	El sistema crea a partir de los datos de los usuarios del COJ sin cuentas en el foro de discusión seleccionado, una cuenta en el foro con las mismas credenciales.	Inicio/Administrador/Cuentas/Replicar

Tabla 10 CP – HU2.5 Listar usuarios del foro sin cuentas en el COJ

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**
EC1.1 Listar usuarios del foro sin cuentas en el COJ	Se selecciona la opción Cuentas.	El sistema muestra una página con la lista de los usuarios registrados en el foro de discusión sin registro en el COJ.	Inicio/Administrador/Cuentas/

Tabla 11 CP – HU2.6 Borrar usuarios del foro sin cuentas en el COJ

Escenario	**Descripción**	**Respuesta del sistema**	**Flujo central**

EC1.1 Borrar usuarios del foro sin cuentas en el COJ	Se selecciona el usuario a borrar. Se selecciona la opción Borrar.	El sistema elimina el usuario registrado en el foro de discusión sin registro en el COJ seleccionado.	Inicio/Administrador/Cuentas/Borrar

Tabla 12 CP – HU3.1 Crear entrada en el foro a partir del registro de un problema en el COJ

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Agregar problema.	Se selecciona la opción Agregar problema del COJ.	El sistema muestra una página de registro de problemas donde el administrador debe rellenar los campos para crear el problema.	Inicio/Administrador/Problemas/
EC1.2 Crear.	Se selecciona la opción Crear.	El sistema verifica que se hayan introducido los datos necesarios y que estén correctamente. Envía los datos necesarios al foro de discusión creando un hilo de entradas correspondiente al problema añadido.	Inicio/Administrador/Problemas//Crear

Tabla 13 CP – HU3.2 Actualizar entrada en el foro al actualizar un problema en el COJ

Escenario	Descripción	Respuesta del sistema	Flujo central

EC1.1 Actualizar problema.	Se selecciona la opción Actualizar problema del COJ.	El sistema muestra una página de actualización de problemas donde el administrador debe rellenar los campos que desea actualizar.	Inicio/Administrador/Problemas/Actualizar
EC1.2 Actualizar.	Se selecciona la opción Actualizar.	El sistema verifica que se hayan introducido los datos a actualizar correctamente. Envía los datos necesarios al foro de discusión actualizando el hilo de entradas correspondiente al problema añadido.	Inicio/Administrador/Problemas/Actualizar

Tabla 14 CP – HU4.1 Listar propuestas de normalización de problemas del COJ y entradas del foro

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar propuestas de normalización de problemas del COJ y entradas del foro	Se selecciona la opción Problemas y Entradas.	El sistema muestra una página con la lista de los problemas del COJ y con entradas en el foro de discusión con diferentes datos.	Inicio/Administrador/Problemas y Entradas

Tabla 15 CP – HU4.2 Ejecutar propuestas de normalización de problemas del COJ y entradas del foro

Escenario	Descripción	Respuesta del sistema	Flujo central

EC1.1 Ejecutar propuestas de normalización de problemas del COJ y entradas del foro	Se selecciona la entrada a normalizar. Se selecciona la opción Normalizar.	El sistema actualiza los datos de la entrada en el foro de discusión con los datos correctos del problema registrado en COJ.	Inicio/Administrador/Problemas y entradas/Normalizar

Tabla 16 CP – HU4.3 Listar problemas del COJ sin entradas en el foro

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar problemas del COJ sin entradas en el foro	Se selecciona la opción Cuentas.	El sistema muestra una página con la lista de los problemas del COJ sin entradas en el foro de discusión.	Inicio/Administrador/Problemas y Entradas/

Tabla 17 CP – HU4.3 Listar problemas del COJ sin entradas en el foro

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Replicar problemas del COJ sin entradas en el foro	Se selecciona el problema a replicar. Se selecciona la opción Replicar.	El sistema crea a partir de los datos del problema seleccionado del COJ, una entrada correspondiente en el foro de discusión.	Inicio/Administrador/Problemas y Entradas/Replicar

Tabla 18 CP – HU4.5 Listar entradas del foro sin problemas en el COJ

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar entradas del foro sin problemas en el COJ	Se selecciona la opción Problemas y entradas.	El sistema muestra una página con la lista de las entradas del foro de discusión sin problemas en el COJ.	Inicio/Administrador/Problemas y entradas/

Tabla 19 CP – HU4.6 Borrar entradas del foro sin problemas en el COJ

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Borrar entradas del foro sin problemas en el COJ	Se selecciona la entrada a borrar. Se selecciona la opción Borrar.	El sistema elimina la entrada seleccionada.	Inicio/Administrador/Problemas y entradas/Borrar

Tabla 20 CP – HU5.1 Listar últimas entradas en el foro de problemas del COJ

Escenario	Descripción	Respuesta del sistema	Flujo central
EC1.1 Listar últimas entradas en el foro de problemas del COJ	Se selecciona la opción Problemas. Se selecciona la opción del nombre de un problema.	El sistema muestra una página la descripción del problema y las últimas entradas correspondientes al mismo del foro de discusión.	Inicio/Problemas/Nombre del problema

3.3.4 Resultados de las pruebas al software

Se realizaron dos iteraciones de pruebas al sistema para las cuales se obtuvieron 7 no conformidades para la primera iteración; resueltas exitosamente, no encontrándose ninguna para la segunda iteración.

Tabla 21 Resultados de las pruebas al software

No. DC	Requisito Funcional	Descripción	Complejidad	Estado
1	1.1	Se crea la cuenta en el foro sin estar activada la cuenta del COJ.	Media	Resuelta
2	2.1	Salen usuarios repetidos.	Media	Resuelta
3	2.2	No se normalizan los nombres de usuarios	Media	Resuelta
4	2.5	Se listan todos los usuarios del foro, incluso los que tienen cuenta en el COJ.	Media	Resuelta
5	2.6	Borra completamente los usuarios del foro, solo debe inhabilitar la cuenta	Media	Resuelta

6	3.1	Se crea la entrada repetida	Alta	Resuelta
7	5.1	No se muestran el nombre del usuario que realizó el comentario	Alta	Resuelta

A continuación, se presenta una gráfica con la descripción de las iteraciones de prueba y sus resultados.

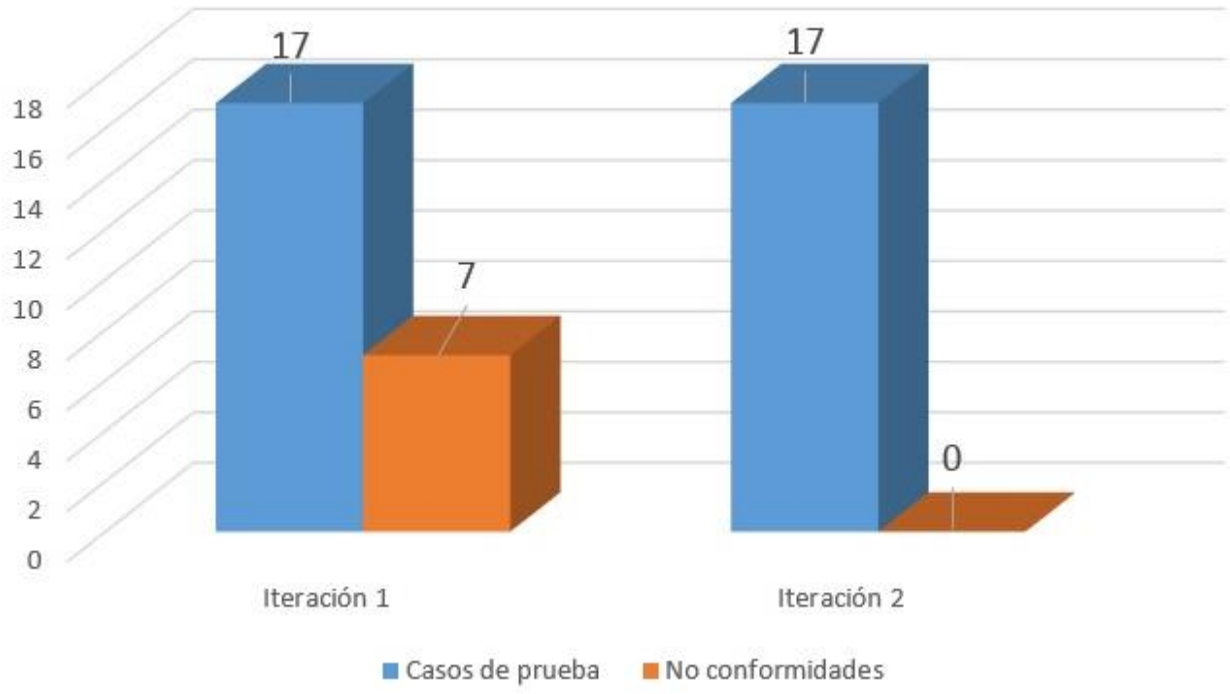


Ilustración 10 - Iteraciones de prueba y resultados

Conclusiones parciales

Como parte del desarrollo de este capítulo se determinan las siguientes conclusiones parciales:

- ✓ La creación de un diagrama de componentes permitió realizar un desglose de los diferentes elementos de software que componen la herramienta para un mayor entendimiento de cómo está conformada.
- ✓ Tras especificar los estándares de codificación se obtiene un código fuente limpio que asegura un fácil entendimiento y futuro mantenimiento del mismo.
- ✓ La realización de pruebas permitió la validación del software tras identificar y corregir las no conformidades resultando en un producto que cumple con los requisitos especificados.

Conclusiones generales

Con el cumplimiento del objetivo general definido en la investigación se concluye lo siguiente:

- El estudio del arte posibilitó la información base para la implementación de la solución.
- Con la implementación de la solución se logró la normalización de la información entre el COJ y su foro de discusión.
- El desarrollo del módulo de integración garantizó la automatización de los procesos entre el COJ y el foro de discusión.
- Con el presente documento se registró toda la información asociada a la investigación y al desarrollo de la solución.

Recomendaciones

Para futuro se recomienda:

- ✓ Implementar funcionalidades que automaticen procesos referentes a las competencias y sus respectivas discusiones.
- ✓ Agregar tecnologías que faciliten el proceso de autentificación en ambas plataformas.

Referencias bibliográficas

1. *Biblioteca práctica de la computación.* **Laboda, Javier, Josep Galimany, Rosa María Pena, Antoni Gual.** Barcelona : Ediciones Océano-Éxito, S.A., 1985.

2. **V. Martínez, F. Aguilar.** Hola Mundo. *Hola Mundo.* [En línea] 2013. [Citado el: 29 de 5 de 2016.] http://www.holamundo.mx/programacion-competitiva/.

3. **Yisel Quiñones Guerrero, Leandro González Vallejo.** Serie Científica - Universidad de las Ciencias Informáticas. *Serie Científica - Universidad de las Ciencias Informáticas.* [En línea] 2012. [Citado el: 29 de 5 de 2016.] http://publicaciones.uci.cu/index.php/SC/article/viewFile/1001/577.

4. **Revilla, Miguel A.** UVa Online Judge. *UVa Online Judge.* [En línea] 1995. [Citado el: 29 de 5 de 2016.] https://uva.onlinejudge.org/.

5. **Tomas Orlando Junco Vazquez, Enrique Jose Altuna Castillo, Jorge Amado Soria Ramirez, Jose Ernesto Lara Rodriguez, Raciel Yera Toledo, Leandro Gonzalez Vallejo.** Caribbean Online Judge. *Caribbean Online Judge.* [En línea] 5 de June de 2010. [Citado el: 29 de 5 de 2016.] http://coj.uci.cu/general/about.xhtml.

6. **LinkedIn Corporation.** SlideShare. *SlideShare.* [En línea] 2016. [Citado el: 29 de 5 de 2016.] http://es.slideshare.net/cruizb14/el-foro-virtual-25964815.

7. **ALEGSA.** DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA. *DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA.* [En línea] 1998. [Citado el: 29 de 5 de 2016.] http://www.alegsa.com.ar/Dic/aplicacion%20web.php.

8. **FLORENTINO, MARVIN.** Mozillla Developer Network. *Mozillla Developer Network.* [En línea] 24 de febrero de 2016. [Citado el: 29 de 5 de 2016.] https://developer.mozilla.org/es/docs/Web/HTML.

9. **Timus Online Judge. *Timus Online Judge.* [En línea] 2000. [Citado el: 29 de 5 de 2016.] http://acm.timus.ru/.**

10. Sphere Online Judge. *Sphere Online Judge.* [En línea] [Citado el: 29 de 5 de 2016.] http://www.spoj.com/.

11. phpBB. *phpBB.* [En línea] 2001. [Citado el: 29 de 5 de 2016.] https://www.phpbb.com/.

12. Ciencia y Tecnología Administrativa. *Ciencia y Tecnología Administrativa.* [En línea] 2001. [Citado el: 29 de 5 de 2016.] http://www.cyta.com.ar/ta0502/v5n2a1.htm.

13. Rodríguez, Tamara Sánchez. *Metodología de desarrollo para la Actividad productiva de la UCI.* La Habana. Cuba: s.n. : s.n.

14. php. *php.* [En línea] 2001. [Citado el: 29 de 5 de 2016.] http://php.net/manual/es/intro-whatis.php.

15. Alvarez, Miguel Angel. desarrolloweb.com. *desarrolloweb.com.* [En línea] 2001. [Citado el: 29 de 5 de 2016.] http://www.desarrolloweb.com/articulos/497.php.

16. Lazaro, Juliana Monteiro. desarrolloweb.com. *desarrolloweb.com.* [En línea] 1 de 1 de 2001. [Citado el: 29 de 5 de 2016.] http://www.desarrolloweb.com/articulos/26.php.

17. Mozilla Developer Network. *Mozilla Developer Network.* [En línea] 2005. [Citado el: 29 de 5 de 2016.] https://developer.mozilla.org/es/docs/Web/JavaScript/Guide/Introducci%C3%B3n.

18. Chapple, Mike. about-tech. *about-tech.* [En línea] 16 de Octubre de 2015. [Citado el: 29 de 5 de 2016.] http://databases.about.com/od/sql/a/sqlfundamentals.htm.

19. Alegsa, Leandro. ALEGSA.com.ar. *ALEGSA.com.ar.* [En línea] 12 de mayo de 2010. [Citado el: 29 de 5 de 2016.] http://www.alegsa.com.ar/Dic/framework.php.

20. Tutoriales de Programación Web. *Tutoriales de Programación Web.* [En línea] 2 de septiembre de 2010. [Citado el: 29 de 5 de 2016.] http://www.javatutoriales.com/2010/09/spring-parte-1-introduccion.html.

21. Rodríguez, Txema. GENBETA. *GENBETA.* [En línea] 16 de junio de 2012. [Citado el: 29 de 5 de 2016.] http://www.genbetadev.com/frameworks/bootstrap.

22. Alegsa, Leandro. DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA. *DICCIONARIO DE INFORMÁTICA Y TECNOLOGÍA.* [En línea] 26 de noviembre de 2008. [Citado el: 29 de 5 de 2016.] http://www.alegsa.com.ar/Dic/uml.php.

23. DANIEL LÓPEZ ORTEGA, JESSICA ANDREA SANTA VILLA. [En línea] 2012. [Citado el: 29 de 5 de 2016.] http://recursosbiblioteca.utp.edu.co/tesisd/textoyanexos/0053L864e.pdf.

24. IBM. *IBM.* [En línea] [Citado el: 8 de 6 de 2016.] http://www-03.ibm.com/software/products/es/enterprise.

25. Meza, Mirna. Herramientas CASE. *Herramientas CASE.* [En línea] 2 de 4 de 2015. [Citado el: 8 de 6 de 2016.] http://fds-herramientascase.blogspot.com/.

26. PÉREZ, TERESA GARZÓN. [En línea] 1988. [Citado el: 29 de 5 de 2016.] http://www.csi-csif.es/andalucia/modules/mod_ense/revista/pdf/Numero_30/TERESA_GARZON_1.pdf.

27. Ma, Rafael. PostgreSQL-Es. *PostgreSQL-Es.* [En línea] 2 de octubre de 2010. [Citado el: 29 de 5 de 2016.] http://www.postgresql.org.es/sobre_postgresql.

28. Bravo, Indira Martinez. Informatica. *Informatica.* [En línea] [Citado el: 29 de 5 de 2016.] http://indira-informatica.blogspot.com/2007/09/qu-es-un-sistema-de-gestin-de-base-de.html.

29. Guia-ubuntu. *Guia-ubuntu.* [En línea] 2008. [Citado el: 29 de 5 de 2016.] http://www.guia-ubuntu.com/index.php/PgAdmin_III.

30. Lischke, Mike. The MySQL Workbench Developer Central Site. *The MySQL Workbench Developer Central Site.* [En línea] 19 de diciembre de 2015. [Citado el: 8 de 6 de 2016.] http://mysqlworkbench.org/.

31. Gregorio, Jose. SlideShare. *SlideShare.* [En línea] 28 de junio de 2011. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/josegregoriob/servidor-web-8451426.

32. Apache. *Apache.* [En línea] 1997. [Citado el: 30 de 5 de 2016.] http://httpd.apache.org/docs/.

33. SlideShare. *SlideShare.* [En línea] 2009. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/GhaBiithahh/entornos-de-desarrollo-integrados.

34. Software.com.ar. *Software.com.ar.* [En línea] 2007. [Citado el: 30 de 5 de 2016.] http://www.software.com.ar/p/intellij-idea.

35. Schinkel, Mike. Jetbrains. *Jetbrains.* [En línea] 2016. [Citado el: 8 de 6 de 2016.] https://www.jetbrains.com/phpstorm/.

36. Thames, Juan Pablo Bustos. SlideShare. *SlideShare.* [En línea] 27 de mayo de 2011. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/jpbthames/modelos-de-dominio-8082870.

37. W3C. *W3C.* [En línea] 2016. [Citado el: 30 de 5 de 2016.] http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb.

38. Deseta, Leonardo. DosIdeas.com. *DosIdeas.com.* [En línea] 13 de Noviembre de 2008. [Citado el: 30 de 5 de 2016.] http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html.

39. Mora, Miquel. SlideShare. *SlideShare.* [En línea] 10 de diciembre de 2010. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/MiquelMora/historias-de-usuario.

40. SladeShare. *SladeShare.* [En línea] 21 de mayo de 2011. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/d-draem/diagramas-de-colaboracion-8052167.

41. ALEGSA. *ALEGSA.* [En línea] 12 de mayo de 2010. [Citado el: 30 de 5 de 2016.] http://www.alegsa.com.ar/Dic/arquitectura%20de%20software.php.

42. IngedioDS. *IngedioDS.* [En línea] 16 de septiembre de 2013. [Citado el: 30 de 5 de 2016.] https://ingeniods.wordpress.com/2013/09/16/patrones-arquitectonicos/.

43. Alvarez, Miguel Angel. DesarrolloWeb.com. *DesarrolloWeb.com.* [En línea] 2 de enero de 2014. [Citado el: 30 de 5 de 2016.] http://www.desarrolloweb.com/articulos/que-es-mvc.html.

44. GENBETA. *GENBETA.* [En línea] 14 de julio de 2014. [Citado el: 30 de 5 de 2016.] http://www.genbetadev.com/metodologias-de-programacion/patrones-de-diseno-que-son-y-por-que-debes-usarlos.

45. Kord, Maia. TuxNots. *TuxNots.* [En línea] 3 de diciembre de 2011. [Citado el: 30 de 5 de 2016.] https://sites.google.com/site/tuxnots/materias-de-la-facu/metodologia-de-sistemas/patronesgrasppatronesgofdiferenciaentregraspygof.

46. Garcia, Fabian. SlideShare. *SlideShare.* [En línea] 26 de octubre de 2008. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/FABIANGARCIA/diagramas-de-secuencia-presentation.

47. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. *El Proceso Unificado de desarrollo de software.*

48. SlideShare. *SlideShare.* [En línea] 7 de abril de 2011. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/uitron/diagrama-de-componentes-7551535.

49. Alejandro Villa Betancur, Jorge E. Giraldo Plaza. ACADEMIA. *ACADEMIA.* [En línea] 2011. [Citado el: 30 de 5 de 2016.] http://www.academia.edu/18122205/Estrategia_de_dise%C3%B1o_para_la_automatizaci%C3%B3n_de_pruebas_unitarias_de_c%C3%B3digos_php_utilizando_el_framework_phpunit.

50. MANUALES.COM. *MANUALES.COM.* [En línea] [Citado el: 6 de 6 de 2016.] http://www.manuales.com/manual-de/que-es-el-camelcase.

51. PRESSMAN, Roger. *Ingenieria del Software. Un enfoque práctico.* 5ta.

52. Mesa, Mirna. Herramientas CASE. *Herramientas CASE.* [En línea] 2 de 4 de 2011. [Citado el: 29 de 5 de 2016.] http://fds-herramientascase.blogspot.com/.

53. Dosideas.com. *Dosideas.com.* [En línea] 13 de Noviembre de 2008. [Citado el: 30 de 5 de 2016.] http://www.dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.html.

54. Kord, Maia. TuxNots. *TuxNots.* [En línea] 3 de diciembre de 2011. [Citado el: 30 de 5 de 2016.] https://sites.google.com/site/tuxnots/materias-de-la-facu/metodologia-de-sistemas/patronesgrasppatronesgofdiferenciaentregraspygof.

55. S., Javier Zapata. Pruebas de Software. *Pruebas de Software.* [En línea] 21 de enero de 2013. [Citado el: 30 de 5 de 2016.] https://pruebasdelsoftware.wordpress.com/.

56. Sebastian, Andrés José. SlideShare. *SlideShare.* [En línea] 19 de octubre de 2012. [Citado el: 30 de 5 de 2016.] http://es.slideshare.net/rinconsete/pruebas-de-caja-blanca-y-negra.