Universidad de las Ciencias Informáticas

Facultad 4

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

Título: Sistema de pruebas automáticas para el sitio Juez en Línea Caribeño.

Autores: Yudel López Pérez

Yordano Sanabia Padrón

Tutor: Ing. Jorge Amado Soria Ramírez

La Habana, junio 2015
"Año 57 de la Revolución"

Quien tiene paciencia,
obtendrá lo que desea.

Benjamín Franklin

Declaración de Autoría

Declaramos que somos los únicos autores del trabajo “Sistema de pruebas automáticas para el sitio Juez en Línea Caribeño” y autorizamos a la Facultad 4 de la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.
Para que así conste firmamos la presente a los ____ días del mes de _____ del año _____.

_____ _____

Autor: Yudel López Pérez **Autor:** Yordano Sanabia Padrón

_____

Tutor: Ing. Jorge Amado Soria

DEDICATORIA

De Yudel

Este trabajo se lo quiero dedicar a las personas más importantes de mi vida como son:

Mi novia Leydis: por ser mi amiga, mi novia, mi vida, mi amor y mi hermanita. Por estar todo este tiempo a mi lado y siempre apoyarme a lo largo de este año, el más importante de mis estudios. Te amo con mi Vida.

A mi hermano Yuniel: para que me veas como un ejemplo y por su increíble habilidad de hacerme enojar y un minuto más tarde estar al borde de llorar de riza.

A mi hermana Yesika por inspirarme siempre la confianza necesaria para culminar mis estudios.

A mi mamá Benita y mi papá Israel: por sacrificarse siempre para que el yuni y yo tengamos lo mejor que nos pueden ofrecer. Por construir la familia más hermosa que una se puedan imaginar. Por ser mis ídolos más grandes y ayudarme a cumplir mi sueño de ser Ingeniero al igual que ustedes. Espero que estén orgullosos de mí.

De Yordano

Este trabajo se lo quiero dedicar a varias personas que son muy importantes para mí:

Primero que todo a mi novia Yisel, que me ha apoyado todo este año con la tesis y siempre ha estado a mi lado cada vez que me ha hecho falta.

A mis tres hermanos, que pese a nuestras diferencias nos queremos muchos los unos a los otros y a los cuales siempre he tratado de ser un ejemplo para ellos.

A mi papá, que ya no está entre nosotros pero sé que estuviera muy orgulloso de mí y sabe que siempre lo tendré conmigo.

Por último y sobre todo, a mi mamá, que se ha sacrificado por mantener todos estos años a sus cuatro hijos, que gracias a ella, hoy estoy donde estoy y quiero que sepa, que para mí es la mejor madre del mundo y la quiero mucho.

AGRADECIMIENTOS

Agradecemos a la Revolución por darnos la oportunidad y los medios para convertirnos en ingenieros y en especial a Fidel y Raúl por crear este centro de altos estudios donde nos formamos como profesionales.

A nuestro tutor Jorge Amado Soria, por darnos su apoyo incondicional, por estar ahí cuando lo hemos necesitado y por confiar en nosotros.

Al tribunal conformado por Maritza, Yusdel y Jorge Luis, a nuestro oponente William por apoyarnos en todo momento y ayudarnos a graduarnos con resultados satisfactorios.

A los profesores: que nos han dado clases en cualquier momento, ya que nos han ayudado poniendo su granito de arena, en la formación como hombres de bien guiándonos por el buen saber. Por educarnos de la forma más amable posible, logrando desarrollar en nosotros los valores de un digno profesional de la Universidad de las Ciencias Informáticas.

De Yudel

Sencillamente, 80 páginas no bastarían para agradecer a todas las personas que de una forma u otra me han apoyado, guiado y aconsejado a lo largo de toda mi vida, a todos ellos les agradezco por ayudarme a convertirme en la persona que soy hoy.

A toda mi familia por parte de madre "los muchos" (el familión de Pinar de Río).

A toda la familia por parte de padre también: que tanto me han apoyado y siempre confiaron en mí.

A mi mamá Benita: por darme su apoyo incondicional, amarme y siempre estar a mi lado a pesar de todas las adversidades siendo el modelo de mujer, amiga y madre más perfecto que haya podido conocer a lo largo de mi vida.

A mi papá Israel: por confiar en mí más que mí mismo ggg. Por guiarme por el buen camino con sus concejos y por ser mi ídolo de cada día para convertirme en un excelente profesional.

A mi novia, esposa, mujer y amiga leydita linda: que en tan poco tiempo se ha convertido en la mujer más importante en mi vida y la cual siempre está a mi lado apoyándome en los momentos difíciles, ayudándome a salir de los baches de la vida, acompañándome en mi andar y cuidándome cuando más lo necesitaba, convirtiéndose en mi

diosa, mi musa y mi amor. Si me dejaran pedir dos deseos estos serían: salud para todas las personas que me han ayudado en mi vida y casarme contigo para compartir una vida plena a tu lado.

A mi hermanito Yuniel: por ser mi inspiración a convertirme cada día en una mejor persona y en alguien en quien él pueda apoyarse.

A mi hermana Yesika: por ser la persona más noble que he conocido en mi vida, con sentimientos especiales y estar pendiente de mí y de mis resultados, siempre aconsejándome y apoyándome con la tecnología necesaria para lograr realizar esta tesis.

A mi tía Milagros por ayudarme con las comunicaciones con mi hermana y apoyarla a ella en todo convirtiéndola en la persona tan especial que es.

A mi abuelita Lidia: por siempre confiar en mí aunque pregunte las cosas 100 veces.

A mi primo Dennis por siempre estar preocupado y ser más que primo un hermano mayor, en el cual me puedo apoyar ante cualquier duda, inquietud o adversidad y que siempre está ahí en las buenas y en las malas.

A mi primo Ernesto: que aunque no se encuentra entre nosotros, él fue la persona que siempre me ayudo en esta universidad.

A los compañeros de aula, a los que terminaron y los que no también, por explicarme las dudas y ser buenos compañeros ante todo.

Al piquete del apartamento, Luciano, Manuel, Kaio por ser los amigos más grandes que he tenido. Al Yankee el cual le deseo suerte y éxito en su tesis. A Yasiel, Arturo, Yadián por ser mis compañeros de cuarto desde 1er año y convertirse en lo mejor que me llevo de la universidad (Hermanos).

A los vecinos del barrio de artemisa como son pepito y mi otro papá pepe, Daniel, Daviamnit, Guille, Víctor, Sexy, Lázaro, Marlen, Emily, Grisel, Cheo que siempre me apoya, el rafa, danger, el negro, Alejandro, Rene, Magalis y al único Yosuani

A mi compañero de tesis Yordano: por ser la persona que desde 1er año también ha compartido momentos inolvidables conmigo y siempre ha estado presente aunque sea para hacerme reír, por haber confiado en mí para ser su compañero de tesis, siendo parte del grupo de hermanos que me llevo de mis paso por la universidad, convirtiéndose en el pilar

fundamental para juntos lograr el preciado título de Ingeniero en Ciencias Informáticas.

Muchas gracias a todos por formar parte de mi vida.

De Yordano

A mi mamá: por siempre estar pendiente, preocupada y presente en cada momento de mi vida, por quererme y siempre darme la confianza para lograr mis propósitos.

A mi madrina Maribel: que es una de las personas que más me han apoyado y siempre se ha preocupado mucho por mí y a la cual le estoy muy agradecido.

A mis hermanos: que siempre hemos estado juntos, en las buenas y en las malas, y que saben que siempre estaré ahí para lo que necesiten.

A mi novia Yisel: por darme ánimos siempre que los necesité y ayudarme en todo momento.

A todo mi familia: mis tíos, tías y primos que son muchos para mencionarlos, que siempre se han preocupado por mi tesis y siempre hemos sido una familia unida.

A mis amigos de la casa: Luisito, Jean Carlos, El Papo y Jorgito, que siempre se han preocupado por mi tesis.

A mis amigos de la escuela: Yasiel, Yadian, Arturo, Mario y Ernest que siempre nos hemos apoyado mutuamente en la escuela.

A mi compañero de tesis Yudel: que siempre hemos trabajado juntos no solo en la tesis, sino en todos estos años en la realización de muchas actividades docentes y no docentes.

A todos: los que de una forma u otra me ayudaron cuando las cosas se ponían difíciles, y en los momentos en que todo parecía perdido, supieron apoyarme y me ayudaron a continuar hacia adelante, a todas estas personas…

Gracias.

Resumen

Los jueces en línea de programación son soluciones creadas con el objetivo de potenciar la práctica y el desarrollo de la programación. Estos están siendo utilizados actualmente para la creación de concursos de programación, los cuales, poseen un alto nivel de aceptación en Cuba y en el resto del mundo. Este auge aún no ha dado todos los frutos esperados debido a la falta de preparación por parte de los estudiantes y profesores, ya que no son capaces de lidiar con todo el flujo de información que sobreviene hacia ellos. Actualmente en la Universidad de las Ciencias Informáticas se encuentra desplegado el sitio web Juez en Línea Caribeño(COJ), el cual está perdiendo usuarios en todo el mundo por diferentes errores de regresión, que ocurren cuando se le inserta un nuevo módulo o funcionalidad al sistema. Este sitio no cuenta con un orden y estructura de pruebas bien definidas para asegurar la calidad del mismo. En el presente trabajo se concibe un sistema de pruebas que, integrado al juez en línea de programación, intenta solucionar esta problemática al generar automáticamente pruebas funcionales al COJ, basados en su rendimiento y progreso actual, para darle solución desde el enfoque del problema general de la investigación.

Índice

Contenido

Índice de tablas

Índice de figuras

Introducción

En un mundo donde la ciencia y la tecnología desempeñan un papel importante, surge y se desarrolla de manera vertiginosa un nuevo concepto conocido como las Tecnologías de la Información y las Comunicaciones (TIC). El uso de las TIC se evidencia en todos los sectores de la sociedad. Las mismas son el conjunto de tecnologías tanto de hardware como de software, que permiten el acceso, estudio, desarrollo, tratamiento, almacenamiento y distribución de la información.

En la década del 70 surgen los concursos de programación gestados por William B. Poucher quien se desempeña como Director General del ACM International Collegiate Programming Contest (ACM-ICPC), evento organizado anualmente por la Association for Computing Machinery (ACM, por sus siglas en inglés). Estas competencias han favorecido el desarrollo de habilidades en las técnicas de análisis y diseño de algoritmos, la cooperación en un ambiente de equipo, la productividad y el pensamiento algorítmico, para contribuir a la formación de mejores profesionales. [1]

Para apoyar la preparación de estos concursos y la enseñanza de la programación en sentido general, surgen los jueces en línea. Estos básicamente son aplicaciones web, disponibles todo el tiempo que automatizan la evaluación de las respuestas a los problemas de programación de competencia. En los últimos años se ha incrementado el acceso a los juez en línea de programación desde diversas instituciones cubanas (Universidades, Joven Clubs, Centros de trabajo e Institutos Preuniversitarios) fruto del despliegue y uso del Juez Caribeño en Línea COJ, por sus siglas en inglés) (http://coj.uci.cu), que está disponible para usuarios del país y el mundo a través de Internet [1]. Además, este sitio cuenta con un solo desarrollador, el cual se encarga de corregir los errores detectados y a su vez extender el sistema con nuevas funcionalidades. La incorporación de nuevas funcionalidades y/o modificaciones a funcionalidades ya desarrolladas en el COJ propician la ocurrencia de problemas de regresión, fallos en módulos y funcionalidades anteriores, que antes no tenían errores, lo cual atenta contra la calidad de este sistema. Estos errores en gran medida son detectados por los usuarios finales del COJ, lo cual provoca descontento en la comunidad de usuarios y en el peor de los casos la reducción en el número de usuarios con que cuenta el sistema, elementos que repercuten negativamente en el prestigio del sitio a nivel internacional.

Actualmente el COJ no cuenta con un orden y estructura de pruebas bien definidas para asegurar la calidad del sistema, lo que provoca constantes despliegues de nuevas

versiones como respuesta a los errores detectados por los usuarios, proceso bien engorroso y contrario a los principios que rigen el proceso de desarrollo de un software, y más para un sistema de tal importancia y relevancia tanto nacional como internacionalmente.

Para darle solución a este problema y teniendo en cuenta sobre todo el limitado número de personas con que cuenta el equipo de desarrollo es necesario realizar un conjunto de pruebas automáticas para detectar los posibles fallos del sistema y corregirlos con anterioridad a ser ejecutados por los usuarios finales.

A partir de los elementos anteriormente expuestos, se deriva como **problema a resolver** ¿Cómo contribuir a la calidad del sitio Juez en Línea Caribeño?

El **objeto de estudio** de la investigación son los sistemas de pruebas automáticas para sitios web en línea, por lo que se identifica como **campo de acción** al desarrollo de un sistema de pruebas automáticas para el sitio Juez en Línea Caribeño.

El **objetivo general** de este trabajo es desarrollar un sistema de pruebas automáticas al sitio Juez en Línea Caribeño que contribuya a la calidad del sistema, planteándose de esta manera las siguientes **Tareas de investigación:**

- Analizar los conceptos y tecnologías más utilizadas para conformar la base teórica metodológica de la investigación.
- Analizar el funcionamiento y desarrollo del sitio web Juez en Línea Caribeño:
    - Analizar el funcionamiento del Módulo Concursos reales.
    - Analizar el funcionamiento del Módulo Problemas.
- Analizar el funcionamiento de las herramientas a utilizar para realizar el conjunto de casos de pruebas.
- Elaborar pruebas automáticas para el sistema Juez en Línea Caribeño:
    - Elaborar pruebas automáticas al módulo Concursos reales.
    - Elaborar pruebas automáticas al módulo Problemas.

Durante el trabajo se sustenta la siguiente **hipótesis:** con la aplicación del sistema de pruebas desarrollado para el sitio Juez en Línea Caribeño, se contribuirá al aumento de su calidad.

Para realizar la investigación en curso se siguió una estrategia descriptiva, donde se reflejaron los aspectos esenciales y más significativos del objeto de investigación. Se emplearon varios métodos científicos clasificados en teóricos y empíricos, así como una técnica para la recopilación de información. Como métodos teóricos se aplicaron los siguientes:

- **Histórico-Lógico:** este método se basa en examinar teóricamente cómo ha evolucionado un fenómeno en un período de tiempo determinado. En la actual investigación se utilizó en la búsqueda de los antecedentes relacionados con errores en el sitio web, donde se detectó, que al incorporar una nueva funcionalidad o un nuevo módulo se incrementan la cantidad de errores encontrados por los usuarios finales.
- **Analítico-Sintético:** en las consultas de las teorías y manejo de los diferentes conceptos fundamentales, por los que se regirá este proyecto. Además, de la recopilación de la información necesaria, que brindaron elementos justificados para el desarrollo de esta aplicación, y para darle cumplimiento a cada una de las tareas investigativas.

Como método empírico se utilizó:

- **La observación:** este método es el instrumento que permite estudiar más de cerca el objeto de la investigación, las acciones, causas, consecuencias, entre otros.

Se utilizó la siguiente técnica para la recopilación de la información:

- **La entrevista:** se realizó a un especialista del centro FORTES con el objetivo de obtener criterios para comprender los elementos necesarios para el desarrollo de la solución informática.

El documento de la presente investigación está estructurado de la siguiente manera:

Capítulo 1: Fundamentación Teórica:

Contiene un estudio del estado de los sistemas existentes a nivel mundial. Posee un grupo de características correspondientes a las tecnologías, metodología y herramientas a utilizar para el desarrollo de la solución informática.

Capítulo 2: Desarrollo de la solución propuesta:

Como parte de una propuesta de solución informática para resolver la problemática, se describen las principales características del sistema. Se realiza el modelo de dominio, la especificación de los requerimientos funcionales y no funcionales, así como la descripción de los casos de prueba del sistema y la ejecución de los mismos para la solución informática.

Capítulo 1: Fundamentación teórica

En el presente capítulo se ofrece una introducción a los términos asociados a la investigación para un mejor entendimiento de los temas a tratar. El origen del interés actual por la calidad se puede explicar al recurrir al estudio de la evolución en la comercialización de los productos. Actualmente no basta con producir y distribuir masivamente los productos o servicios sino que se debe lograr la aceptación absoluta por parte del cliente. De allí el esfuerzo que se ha desplegado para obtener soluciones de software con una alta calidad.

1.1 Fundamentación del tema

Esta sección trata de describir el entorno en el cual ha de desarrollarse el software, tratando de introducir conceptos y mostrar explicaciones acerca de los orígenes del tipo de trabajo a realizar, del estado actual del tema a nivel mundial y acerca de las causas que da origen al presente trabajo.

1.1.1 Proceso de Pruebas de Software

El único instrumento adecuado para determinar el estado de la calidad de un producto es el proceso de pruebas de software. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el mismo cumple con los requerimientos y reducir los errores que estos puedan presentar.

Dentro de los tipos de pruebas que se le pueden realizar al software se encuentran las pruebas de aceptación, rendimiento, sistema y las pruebas funcionales. Esta última es fundamental pues permite verificar que se apliquen apropiadamente cada regla de negocio, que los resultados esperados ocurran cuando se introduzcan datos válidos, así como que sean desplegados los mensajes apropiados de error y precaución cuando se introduzcan datos inválidos. Con esto, aumentaría la satisfacción de los usuarios finales ya que obtendrían un software que no solo cumple con sus expectativas en cuanto a funcionalidades, sino que presenta la menor cantidad de errores posibles. Para cumplir estos objetivos, muchas organizaciones realizan las pruebas funcionales manualmente y aunque ayuda, también puede convertirse en un problema cuando se necesite ejecutar muchas pruebas al mismo tiempo o el software sea muy grande.

En la actualidad, el uso de las herramientas se ha hecho imprescindible para automatizar muchos procesos. Con el fin de ayudar a aliviar la situación planteada, algunas compañías desarrollaron herramientas automatizadas destinadas a la realización de pruebas funcionales. Estas permiten a los probadores automatizar las pruebas para concentrarse en otros aspectos también importantes en el desarrollo del

software. De ahí la importancia de realizar un estudio acerca de estas herramientas para mejorar la calidad y eficiencia de dicho flujo. La presente investigación tiene como objetivo principal: elaborar una propuesta de una herramienta para la realización de pruebas funcionales de software, que permita la obtención de un producto final con la mayor calidad.

1.1.2 Pruebas de Software

El único instrumento adecuado para determinar el estado de la calidad de un producto software, es el proceso de pruebas. En este proceso se ejecutan pruebas dirigidas a componentes del software o al sistema de software en su totalidad, con el objetivo de medir el grado en que el software cumple con los requerimientos. En las pruebas se usan casos de prueba, especificados de forma estructurada mediante Técnicas de prueba[2]. Estas, al contrario de lo que muchas personas creen, no se deben dejar para el final de la etapa de construcción del software. Las pruebas se deben empezar a realizar desde la misma etapa de análisis de requerimientos, ya que desde un principio se puede incurrir en malas interpretaciones de las "reglas del negocio", lo que finalmente tendrá como consecuencia incongruencia entre lo que el cliente quiere y lo que se ha desarrollado [3].

Principalmente, las ventajas que trae la realización de pruebas, en un desarrollo de software son las siguientes:

- Reducen la posibilidad de agregar defectos al software. Si hay que realizar una adición de características requeridas por el cliente y se ve que ya no funcionan bien algunas de las cosas que anteriormente servían, se puede inferir la nueva funcionalidad es la que contiene defectos, por lo que no hay necesidad de realizar modificaciones a los componentes realizados anteriormente.
- Las pruebas aportan buena documentación. Es preferible ver unas pequeñas líneas de "Código de prueba", que son concisas y realmente son fáciles de entender, a revisar línea por línea de código para poder entender que hace determinado componente de software.
- Reducen el costo del cambio. Evitan que se descubran los defectos hasta el final del desarrollo de software.
- Permiten evaluar la flexibilidad del software construido, la eficiencia de la aplicación, la confiabilidad en caso de que no se ejecute adecuadamente.
- Permiten realizar re-implementación. Se puede llegar a necesitar re-implementar determinada funcionalidad en un sistema, debido a fallos de seguridad, rendimiento, o simplemente porque no reunía lo que el cliente esperaba de este,

entonces dada tal situación, las pruebas realizadas a dicha funcionalidad van a permitir que se vuelva a desarrollar de manera segura, ya que la prueba encierra el criterio de aceptación sobre la funcionalidad, permitiendo de esta forma que se vuelva a desarrollar algo acorde con la especificación.

- Hacen que el desarrollo sea más rápido, ya que a medida que se agregan características al software, las funcionalidades anteriores pueden fallar, pero si se han hecho las pruebas pertinentes a las funcionalidades anteriores, se puede descartar inmediatamente que existan defectos en estas, por lo que se puede concentrar tranquilamente en la funcionalidad nueva.

1.1.3 Tipos de pruebas de software [4]

Pruebas de aceptación: Pretende lograr una revisión final por parte de la organización que solicitó el sistema, lo cual, a menudo significa validación del sistema.

Pruebas de sistema: Verifica el sistema completo o su aplicación como tal. Se toma el punto de vista del usuario final y los casos de uso de pruebas.

Pruebas de rendimiento: Tiene como propósito medir la capacidad de procesamiento del sistema bajo diferentes cargas, incluyendo espacio de almacenamiento y utilización de la unidad de procesamiento.

Pruebas de integración: En ella se verifica que las unidades trabajen juntas correctamente.

Pruebas de unidad: Mediante esta prueba solo una unidad es probada como tal, como una clase, u otro paquete de servicio o un subsistema.

Pruebas funcionales: Tiene por objetivo probar que el sistema desarrollado cumple con las funciones específicas para el cual ha sido creado.

1.1.4 Pruebas Funcionales

Se denominan pruebas funcionales o *Functional Testing*, a las pruebas de software mediante la cual se prueba que los sistemas desarrollados cumplan con las funciones específicas para los cuales han sido creados. Es común que este tipo de pruebas sean desarrolladas por analistas de pruebas con apoyo de algunos usuarios finales, esta etapa suele ser la última etapa de pruebas y al dar conformidad sobre esta, el paso siguiente es el pase a piloto. A este tipo de pruebas se les denomina también pruebas de comportamiento o pruebas de caja negra.

Al realizar pruebas funcionales lo que se pretende es ponerse en los pies del usuario, usar el sistema como él lo usaría, sin embargo, el analista de pruebas debe ir más allá que cualquier usuario. Generalmente se requiere apoyo de los usuarios finales ya que ellos pueden aportar mucho en el desarrollo de casos de prueba complejos, enfocados básicamente al negocio, posibles particularidades que no se hayan contemplado adecuadamente en el diseño funcional. El analista de pruebas debería dar fuerza a las pruebas funcionales y más aún a las de robustez, generalmente los usuarios realizan las pruebas con la idea que todo debería funcionar, a diferencia del analista de pruebas que tiene más bien una misión destructiva. Su objetivo será encontrar alguna posible debilidad y si la llega a ubicar se esforzará por que deje de ser pequeña y posiblemente se convertirá en un gran error, cada error encontrado por el analista de pruebas es un éxito [5].

Con el objetivo de hacer más rápido y eficiente este proceso de pruebas, se realiza el estudio sobre las herramientas más usadas para realizar pruebas funcionales de software, las cuales se estarán caracterizando a continuación:

1.2 Estudio de las herramientas existentes

Para la realización de este trabajo, se realiza un estudio de los sistemas que pudieran utilizarse para la realización de las pruebas de software, los sistemas que pueden servir de documentación y de referencia para comprender el contexto donde se desarrollan los procesos de pruebas de software, se abordan cada uno de los que se consideraron representativos, el resto de la documentación que se encuentra disponible se anexa al documento.

1.2.1 HTTPUnit

Esta herramienta es una extensión de JUnit para aumentar sus capacidades, ya sea para probar componentes especializados, para facilitar las pruebas de clases individuales, realizar las pruebas dentro del contenedor de aplicaciones (pruebas de integración), o escribir pruebas funcionales capaces de interactuar con interfaces web.

Se pueden realizar pruebas funcionales antes de que estén generadas las páginas web. Esencialmente es un cliente web programable. Permite escribir *tests* que simulen a un usuario accediendo a una aplicación basada en web mediante un navegador. No se basa en los controles que tenga la página, si no que se basa en los valores de entrada que el usuario pueda ingresar. Es muy adecuado para ser utilizado en combinación con JUnit, con el fin de escribir fácilmente las pruebas que verifican el comportamiento adecuado de un sitio web [6].

1.2.2 JMeter

Es una herramienta de escritorio puro Java desarrollado por *The Apache Software Foundation* (Comunidad descentralizada de desarrolladores que trabajan cada uno en sus propios proyectos de código abierto). JMeter permite realizar pruebas de rendimiento y pruebas funcionales sobre aplicaciones web. Existen un gran número de herramientas para realizar pruebas, gratuitas (JUnit, JWebUnit) y de pago (*LoadRunner*), pero se destaca por su versatilidad, estabilidad y por ser de uso gratuito. Permite realizar test de FTP, JDBC, LDAP, SOAP y *Web Service* (en Beta) y la ejecución de pruebas distribuidas entre distintos ordenadores para realizar pruebas de rendimiento. Pero en particular, no ejecuta el código Java Script que se encuentran en las páginas HTML. Aunque se pueden realizar pruebas funcionales con ella, está más enfocada a las pruebas de rendimiento. Muestra los resultados de las pruebas en una amplia variedad de informes y gráficas [7].

1.2.3 Sahi

Es una herramienta libre de código abierto desarrollada por "*Tyto Software*" (Empresa que crea herramientas y soluciones innovadoras para simplificar la automatización de aplicaciones web). Sahi permite grabar y ejecutar scripts de automatización utilizando Internet Explorer de por medio. Soporte para HTTP, HTTPS y Ajax y es desarrollada en Java y Java script. Puede ser instalado y ejecutado en cualquiera de los tres sistemas operativos principales actuales: Windows, Mac OS y Linux, soportando prácticamente cualquier navegador web que soporte Java script. La instalación tiene como limitación que requiere de la máquina virtual de Java 1.4 o superior y de 32 bits, con la versión de 64 bits no es posible correr Sahi [8].

1.2.4 Selenium IDE

Es una herramienta diseñada por la empresa Software *Quality Systems* (Dedicada al aseguramiento de la calidad del software, soporte y creación de herramientas resultado de sus investigaciones). Selenium IDE permite realizar de manera sencilla pruebas en aplicaciones web, está orientado a la ejecución de pruebas funcionales a nivel de usuario directamente desde el navegador. Selenium IDE es una extensión para Firefox que registra nuestra actividad en el navegador durante un período determinado; esta actividad se traduce en una serie de comandos que se podrán repetir y repetir, incluyendo aserciones y advertencias para realizar el test propiamente dicho. Esta herramienta trabaja con aplicaciones en línea y es multilenguaje (HTML, PHP, Java, C# y algunos más). Admite funcionalidad para Java Script así como llamadas AJAX, permite también un fácil y rápido testeo de aplicaciones complejas Web 2.0 [9].

1.2.5 Fundamentación de la selección de la herramienta

Luego de consultar las diferentes bibliografías referentes a las herramientas planteadas y que fueron referenciadas anteriormente, conocer sus características y ventajas ha sido posible identificar los criterios más importantes para establecer una comparación entre ellas, los cuales han sido utilizados en comparaciones similares. Además, precisamente los criterios definidos para la comparación son de vital importancia para la toma de decisiones en cuanto a la selección de una herramienta. Tanto la licencia, como la plataforma y los lenguajes de programación que soporta, ya sea por cuestiones económicas como por la transferencia tecnológica que se impone en la actualidad, marcan una diferencia relevante con respecto a otros criterios.

Tabla 1. Comparación de las herramientas.

Herramientas / Criterios	Licencia	Multiplataforma	Lenguajes de programación	Navegadores
JUnit	CPL(Libre)	Sí	Java	Firefox, Internet Explorer, Safari
HTTPUnit	Propietaria	Sí	Java	Firefox, Internet Explorer, Safari
JMeter	Apache (Libre)	Sí	Java	Firefox, Internet Explorer
Sahi	Apache (Libre)	Sí	Java y Java script	Firefox, Internet Explorer
Selenium IDE	Apache (Libre)	Sí	C#, Java, PHP, Ruby,	Firefox, Chrome, Internet

			Groovy, Python y Perl	Explorer, Opera, Safari

Luego del análisis de cada una de las herramientas presentadas y de esta comparación, se considera, que las más apropiadas para realizar pruebas funcionales de software son: Sahi y Selenium IDE. Aunque en especial se usará esta última para la realización de las pruebas en cuestión. Esta propuesta está basada en que además de sus características de ser simple, flexible y ser *open source* con licencia libre, es posible ser controlado por muchos lenguajes de programación y frameworks de pruebas. También, a diferencia del resto de las herramientas, puede ejecutarse en varios navegadores y sistemas operativos como se puede observar en la tabla anterior. Aun cuando Sahi presenta características similares a Selenium, tiene ciertas limitaciones en estos criterios del lenguaje de programación y los navegadores, así como con la instalación que se torna más difícil y exigente, mientras que Selenium se instala fácilmente, demostrando ser una de las más completas y eficientes para este tipo de pruebas.

También, haciendo uso del Google Trends fue posible conocer las tendencias de estas dos herramientas en los últimos años, así como algunos de los países que más las utilizan. Esto es un elemento más que se considera para la selección de la propuesta antes mencionada. Como se muestra en la gráfica 1, la herramienta Sahi tuvo su mayor fuerza en el año 2010, pero luego tuvo un decremento en los años 2011 y 2012. Los 10 países y sus ciudades que más la utilizan se pueden observar en la Figura 1.

Gráfica 1: Tendencias de las herramientas Sahi y Selenium

En cambio como se puede observar en la gráfica 1, la herramienta Selenium IDE desde el año 2008 ha ido aumentado considerablemente su aplicación hasta la actualidad. Los 10 países y sus ciudades que más la utilizan se pueden observar en la Figura 2.

Regions		Cities	
1. Pakistan		1. Lahore, Pakistan	
2. India		2. Karachi, Pakistan	
3. Turkey		3. Islamabad, Pakistan	
4. United Arab Emirates		4. Mahape, India	
5. Saudi Arabia		5. New Delhi, India	
6. Finland		6. Pune, India	
7. Canada		7. Mumbai, India	
8. United Kingdom		8. Konya, Turkey	
9. Poland		9. Kayseri, Turkey	
10. Australia		10. Calcutta, India	

Figura 1: Ranking de países que utilizan Sahi

En cambio como se puede observar en la gráfica 1, la herramienta Selenium IDE desde el año 2008 ha ido aumentado considerablemente su aplicación hasta la actualidad. Los 10 países y sus ciudades que más la utilizan se pueden observar en la Figura 2.

Regions		Cities	
1. India		1. Noida, India	
2. Belarus		2. Hyderabad, India	
3. Ukraine		3. Bangalore, India	
4. Philippines		4. Pune, India	
5. Czech Republic		5. Chennai, India	
6. Ireland		6. Mumbai, India	
7. Russia		7. San Jose, CA, USA	
8. Japan		8. New Delhi, India	
9. United States		9. San Francisco, CA, USA	
10. Romania		10. Dublin, Ireland	

Figura 2: Ranking de países que utilizan Selenium

De todo lo anteriormente expuesto se llega a la conclusión de que Selenium es la mejor herramienta a utilizar, por lo cual la mayoría de las pruebas se realizarán en dicha herramienta, pero también surge la necesidad de realizar pruebas al código del COJ, por lo cual a continuación se explicará la selección del lenguaje a utilizar así como el Entorno de Desarrollo Integrado (IDE) en donde se trabajará.

1.3 Lenguaje de programación

Entre las posibles opciones para la elección del lenguaje de desarrollo se valoraron fundamentalmente tres, a continuación se exponen las principales características de PHP, C# y Java. Se escogen estos tres para estudiarlos teniendo en cuenta su empleo en la realización de las pruebas.

1.3.1 C#

Scott Wiltamuth, Anders Hejlsberg y la compañía Microsoft fueron los creadores de C#, lenguaje de programación orientado a objetos que ha deleitado a los programadores de todo el mundo y está incluido en la plataforma .NET. Entre sus principales características

se puede señalar la sencillez, modernidad, gestión automática de memoria, indexadores, eventos, delegados y tipos genéricos que lo convierten en una poderosa opción. [10]

1.3.2 PHP

En el año 1994 Rasmus Lerdorf crea PHP, acrónimo recursivo que significa PHP Hypertext Pre-processor un lenguaje interpretado ideal para el desarrollo de sitios web dinámicos. Se pueden resaltar entre sus principales ventajas su escaso consumo de recursos, su rapidez y amplia capacidad de conectividad. Lamentablemente requiere un alto nivel de experiencia para la realización de tareas complejas. [11]

1.3.3 Java

Sun Microsystems se gana el protagonismo al diseñar Java en el año 1991, la principal característica buscada con su creación se convierte en la principal ventaja, y está dada por ser un lenguaje multiplataforma, capaz de correr no solo en computadoras sino en celulares, hasta equipos electrodomésticos, cuyo único requisito es tener la máquina virtual de Java, concepto desarrollado por Sun con el objetivo de proveer independencia de la arquitectura sobre la que corre la aplicación. Es un lenguaje de programación no propietario.

Figura 3: Logo de Java

Sun describe a Java como simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico [12]. Su estrecha unión con internet está dada primeramente por el auge que alcanzaron ambos en fechas similares. Se puede señalar como desventaja la dependencia de la máquina virtual para que los programas funcionen, también al tener que pasar por este paso intermedio se ralentiza un poco el proceso total de ejecución.

1.3.4 Fundamentación de la selección

Se decide el uso de Java por las razones siguientes:

- Es un lenguaje muy utilizado por la UCI al no ser propietario.
- El COJ está desarrollado en este lenguaje.
- Los servidores actuales sobre los que está desplegado el COJ son servidores de aplicaciones Java.
- El ambiente de trabajo utilizado actualmente por el equipo de desarrollo del COJ es Java, otro lenguaje se saldría de los esquemas del proceso integral de desarrollo de la aplicación cliente.

1.4 Entorno de desarrollo

En inglés *Integrated Development Environment* (IDE) es un programa compuesto por un conjunto de herramientas para un programador [13]. Puede dedicarse exclusivamente para un lenguaje de programación o bien para varios.

1.4.1 NetBeans

NetBeans es de código abierto, soporta el desarrollo de todos los tipos de aplicación Java (J2SE, web, EJB y aplicaciones móviles). Existe un número importante de módulos para extenderlo. Es un producto libre y gratuito sin restricciones de uso. Entre sus características se encuentra un sistema de proyectos basado en control de versiones y refactorización. Todas las funciones del IDE son provistas por plugins[1].

NetBeans proporciona plantillas de proyectos y proyectos de ejemplo que le ayudan a crear aplicaciones Java SE, aplicaciones Java EE, Java ME aplicaciones, aplicaciones HTML5, las aplicaciones de la plataforma NetBeans, aplicaciones PHP y C / C ++.

NetBeans IDE 8.0.2 proporciona fuera de la caja de analizadores de código y editores para trabajar con las últimas tecnologías Java 8 - Java SE 8, Java SE Embedded 8 y Java ME Embedded 8. El IDE también tiene una gama de nuevo mejoras que mejoran aún más su apoyo a Maven y Java EE con PrimeFaces; nuevas herramientas para HTML5, en particular para AngularJS; y mejoras en la compatibilidad con PHP y C / C ++.

1.4.2 Eclipse

Eclipse es una plataforma de desarrollo, diseñada para ser extendida de forma indefinida a través de plugins2. Fue concebida desde sus orígenes para convertirse en una plataforma de integración de herramientas de desarrollo. No tiene en mente un lenguaje específico, sino que es un IDE genérico, aunque goza de mucha popularidad entre la comunidad de desarrolladores del lenguaje Java usando el plugins JDT que viene incluido en la distribución estándar del IDE.

Proporciona herramientas para la gestión de espacios de trabajo, escribir, desplegar, ejecutar y depurar aplicaciones.

Principales características de Eclipse [14]:

[1] **Plugin:** Complemento o aplicación que se relaciona con otra para aportar- le una función nueva y generalmente muy específica.

- ✓ Perspectivas, editores y vistas: en Eclipse el concepto de trabajo está basado en las perspectivas, que no es otra cosa que una pre-configuración de ventanas y editores, relacionadas entre sí, y que nos permiten trabajar en un determinado entorno de trabajo de forma óptima.
- ✓ Gestión de proyectos: el desarrollo sobre Eclipse se basa en los proyectos, que son el conjunto de recursos relacionados entre sí, como puede ser el código fuente, documentación, ficheros configuración, árbol de directorio, entre otros. El IDE nos proporcionará asistentes y ayudas para la creación de proyectos. Por ejemplo, cuando se crea uno, se abre la perspectiva adecuada al tipo de proyecto que se está creando, con la colección de vistas, editores y ventanas pre-configuradas por defecto.
- ✓ Depurador de código: se incluye un potente depurador, de uso fácil e intuitivo, y que visualmente nos ayuda a mejorar nuestro código. Para ello sólo se debe ejecutar el programa en modo depuración (con un simple botón). De nuevo, se tiene una perspectiva específica para la depuración de código, la perspectiva depuración, donde se muestra de forma ordenada toda la información necesaria para realizar dicha tarea.
- ✓ Extensa colección de plugins: están disponibles en una gran cantidad, unos publicados por Eclipse, otros por terceros. Al haber sido un estándar de facto durante tanto tiempo (no el único estándar, pero sí uno de ellos), la colección disponible es muy grande. Los hay gratuitos, de pago, bajo distintas licencias, pero casi para cualquier cosa que se imaginen existe el plugin adecuado.

1.4.3 Fundamentación de la selección del IDE

Aunque ambos IDE son similares en cuanto a las funcionalidades y ventajas que ofrecen para el programador, la dirección del proyecto decidió escoger Eclipse como IDE de desarrollo, debido a que es el IDE utilizado por los programadores del sitio COJ, lo cual permitiría una mejor adaptación a las pruebas automáticas que se realizaron.

Posteriormente a la selección del IDE, es necesario decidir la herramienta a utilizar para la realización de las pruebas que se realizarán al código del COJ, para ello se comparan dos posibles herramientas.

1.4.3.1 TestNG

TestNG (*Next Generation*) es un marco de pruebas que, inspirado en JUnit y NUnit, pero la introducción de muchas nuevas funcionalidades innovadoras como las pruebas de dependencia, que agrupa el concepto de hacer la prueba más potente y fácil de hacer.

Está diseñado para cubrir todas las categorías de pruebas: unidad, funcional, de extremo a extremo, la integración, entre otros [15].

TestNG es un framework para pruebas y *testing* que trabaja con Java y está basado en JUnit (para Java) y NUnit (para .NET), pero introduciendo nuevas funcionalidades que los hacen más poderosos y fáciles de usar, tales como:

- Anotaciones JDK 5.
- Configuración flexible de pruebas.
- Soporte para pruebas.
- Soporte de pasaje de parámetros.
- Soportado por herramientas y plugins importantes y variados como: (Eclipse, IDEA, Maven, entre otros.).
- Funciones JDK por defecto de *runtime* y *logging*.
- Métodos dependientes para pruebas sobre servidores de aplicación.

1.4.3.2 JUnit

Es un conjunto de bibliotecas creadas por Erich Gamma y Kent Beck en el año 2003, estas son utilizadas para automatizar las pruebas unitarias de aplicaciones Java. Se utiliza en la fase de desarrollo, su utilización por parte de los desarrolladores permite la creación de software de mayor calidad. JUnit es un conjunto de clases (framework) que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente. [16]

1.4.3.3 Selección del tipo de prueba

De estos dos tipos de pruebas o test se utilizará JUnit ya que es el requerido por el cliente y el más apropiado para lograr realizar dichas pruebas de una forma eficiente.

1.5 Proceso de desarrollo

El objetivo de un proceso de desarrollo de software es elevar la calidad de este, a través de la transparencia y control sobre su desarrollo, y lograr que dichas pautas sean reproducibles en cada proyecto, independientemente de su magnitud. Actualmente

existe una gran cantidad de procesos de desarrollo, agrupados en dos tendencias: las metodologías ágiles y las metodologías tradicionales.

1.5.1 Metodologías tradicionales

Las metodologías tradicionales se centran específicamente en el control del proceso. Son eficientes y se utilizan en numerosos proyectos, sobre todo los de gran tamaño, teniendo en cuenta los recursos y el tiempo. Estas no ofrecen un buen rendimiento en proyectos, donde no se tiene bien definido que es lo que se desea hacer. Por lo que el cliente se ve forzado a tomar la mayoría de las decisiones en un principio. Una mala decisión implica, retrasos en la planificación, sistemas deteriorados, requisitos mal comprendidos e incluso, cambios de negocio y personal. [17]

1.5.1.1 Proceso unificado de desarrollo (RUP)

Esta metodología, aparte de realizar el análisis y diseño orientado a objetos, para utilizar varias técnicas que ayudan al soporte del ciclo completo de desarrollo de software, resulta que es todo un proceso sobre componentes, dirigido por los casos de uso, centrado en la arquitectura, iterativo e incremental. Esta metodología permite realizar un estimado del esfuerzo requerido y analizar el costo para la ejecución del proyecto. El cliente es el encargado de determinar las funcionalidades del software completo, y el equipo de desarrollo planifica cada paso de manera detallada para elaborar la solución. RUP utiliza diagramas UML como lenguaje de modelado y está definido por cuatro fases esenciales; Inicio, Elaboración, Construcción y Transición. [18]

RUP es un proceso para el desarrollo de un proyecto de un software que define claramente quién, cómo, cuándo y qué debe hacerse en el proyecto.

Características principales de RUP:

• Se centra en los modelos.

• Guiado por los casos de uso.

• Centrado en la arquitectura.

• Iterativo e incremental.

• Verifica la calidad del software.

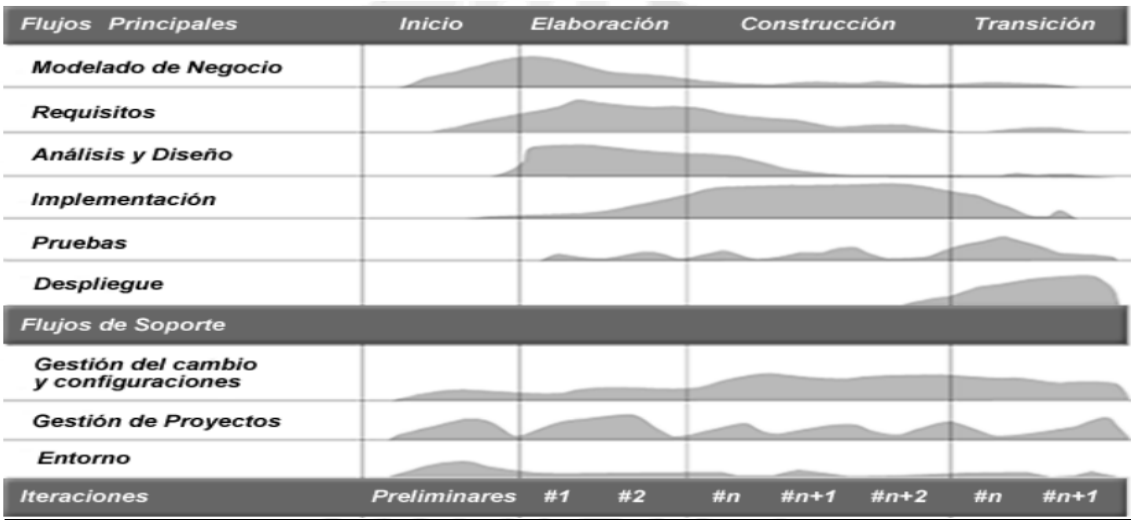


Figura 4: Metodología RUP

1.5.2 Metodologías ágiles

Las metodologías ágiles son muy utilizadas en varios mercados del mundo, buscando desarrollar aplicaciones de la forma más rápida posible y con características adaptables. La misma se basa en la flexibilidad en el proceso de un software, que puede estar sujeto a cambios que se den de manera impredecible, en dependencia de las ideas que pudieran ser generadas por los integrantes del equipo de desarrollo. Todo esto a partir del comienzo y construcción de una aplicación, la cual está dispuesta a sufrir cambios en cualquier momento, para ajustarse a medidas y modificaciones durante todo el desarrollo.

Por las características que estas presentan, los softwares que se desarrollan atendiendo a todo el procedimiento de desarrollo, se realiza de la forma iterativa e incremental. El objetivo principal de este método es entregarle pequeñas versiones del producto al cliente, cada cierto tiempo, y así realizarle pruebas, las cuales permitirán identificar fallos y posibles errores, además de las ideas que pudiera ir aportando el cliente, conforme sus necesidades sobre la aplicación que se está desarrollando. [19]

1.5.2.1 *Extreme Programming* (XP)

La Programación Extrema (XP) es una metodología ágil centrada en potenciar las relaciones interpersonales como clave para el éxito en el desarrollo de software, promoviendo el trabajo en equipo, preocupándose por el aprendizaje de los programadores y propiciando un buen clima de trabajo. Se basa en la retroalimentación continua entre el cliente y el equipo de desarrollo, comunicación fluida entre todos los participantes, simplicidad en las soluciones implementadas y buena actitud a la hora de enfrentar los cambios. XP se

Figura 5: Logo de Extreme Programming

define como especialmente adecuada para proyectos con requisitos imprecisos y muy cambiantes [20]. Se considera ligera, flexible, predecible y de bajo riesgo. Tiene pocos requerimientos de documentación y planificación. El cliente forma parte integral del proceso de desarrollo. [21]

Esta metodología genera pocos artefactos en el momento de explicar la lógica del negocio, enfocándose mucho más por el funcionamiento o desarrollo del sistema, que por toda la documentación que se pueda llevar a cabo durante el proceso de investigación para la creación del producto. Es una metodología que se basa en el desarrollo iterativo e incremental, a través del cual se le realizan una serie de pequeñas pruebas a la aplicación, lo que da la posibilidad de detectar errores que serán corregidos posteriormente, para que no existan consecuencias en todo el proceso de elaboración del producto, y el mismo tenga la mejor calidad posible al ser terminado. Todo este proceso es realizado en dependencia de las exigencias del cliente, las expectativas de los usuarios, y de las acciones que pueda realizar el grupo de trabajo sobre el software. [22]

XP básicamente utiliza las Historias de Usuario (HU), las que son descritas por el cliente, para definir los escenarios claves para el correcto funcionamiento del software. Estas HU son muy útiles ya que a partir de las mismas se les empieza a realizar una serie de pequeñas versiones del software entregadas al cliente antes de terminar el producto, y durante este proceso el cliente tiene la posibilidad de interactuar con el producto dándole la posibilidad de generar nuevas ideas para su mejor funcionamiento, según sus exigencias y las capacidades de desarrollo de todo el equipo. Esta metodología se basa en la realimentación de forma progresiva entre el cliente y el equipo de desarrolladores, la misma se define como adecuada en proyectos donde los requisitos no estén bien definidos y sufran cambios constantemente. [23]

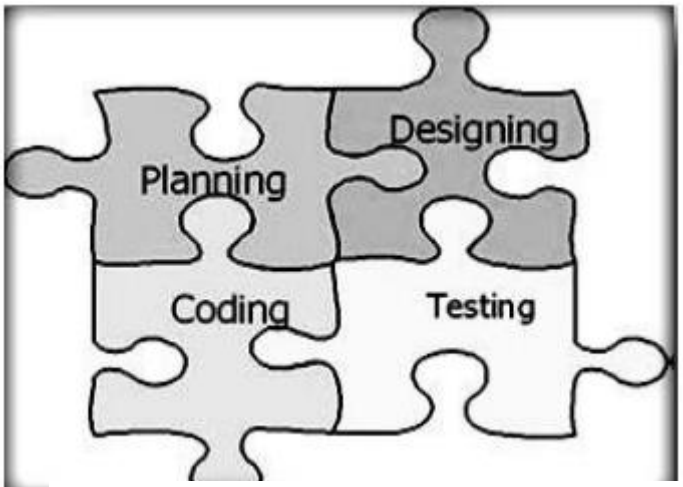


Figura 6: Fases de XP

La misma cuenta con cuatro fases importantes:

• Planificación del proyecto.

En esta primera fase se realiza la recopilación de todos los requerimientos del proyecto, también debe haber una interacción con el usuario, y se debe planificar bien entre los desarrolladores del proyecto que es lo que se quiere para así lograr los objetivos finales.

• Diseño.

Se sugiere realizar diseños simples y sencillos con el objetivo de hacer todo lo menos complicado posible para el usuario o cliente.

• Codificación.

Antes de codificar cada historia de usuario, el cliente debe especificar detalladamente lo que ésta hará; también tendrá que estar presente cuando se realicen las pruebas que verifiquen que la historia implementada cumple la funcionalidad especificada. En esta fase, los clientes y los desarrolladores del proyecto deben estar en comunicación para que los desarrolladores puedan codificar todo lo necesario para el proyecto.

• Prueba.

En esta fase se comprueba el correcto funcionamiento de los códigos que se vayan implementando. Aquí se realizarán también otro tipo de pruebas, entre las cuales están las de aceptación.

1.5.3 Fundamentación de la selección de la metodología

Debido a las características que presenta y la flexibilidad en la que es determinada cada una de las decisiones durante toda la etapa de implementación y creación del software, se determina el uso de XP como metodología de desarrollo para la realización de este proyecto. La misma se acomoda con la estructura del equipo de desarrollo y el buen intercambio que existe entre estos y el cliente. XP nos ha facilitado establecer comunicación directa y de forma periódica con el cliente, para de esta manera poder ejecutar las acciones pertinentes, en dependencia de las necesidades planteadas por el mismo.

Es una metodología ágil, no genera un gran número de artefactos, por lo que reduce de esta manera el tiempo que se pudiese invertir en estos, se genera específicamente el artefacto de las historias de usuario, para aligerar la carga por el gran contenido de trabajo que se encierra en este proyecto, y debido al poco tiempo disponible para su posterior desarrollo. Aunque brinda la posibilidad de hacer uso de los artefactos necesarios con el fin de poder explicar la lógica del negocio, lo mejor posible. Por otra parte se preocupa por el entorno de trabajo, y por la capacitación del personal a través de cursos o conferencias.

1.6 Lenguaje de modelado

El Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, *Unified Modeling Language*) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; aunque todavía no es un estándar oficial, sí lo es de facto [24]. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema de software. UML ofrece un estándar para describir un plano del sistema, incluyendo aspectos conceptuales, tales como procesos de negocios y funciones del sistema, y aspectos concretos, como expresiones de lenguajes de programación, esquemas de bases de datos y componentes de software reutilizables. En fin UML es el enlace entre quien tiene la idea y el desarrollador, la comunicación en su principal objetivo.

Es importante resaltar que UML es un lenguaje para especificar y no para describir métodos o procesos. Se utiliza para definir un sistema de software, para detallar los artefactos en el sistema y para documentar y construir. En otras palabras, es el lenguaje en el que está descrito el modelo. Se puede aplicar en una gran variedad de formas para dar soporte a una metodología de desarrollo de software (tal como RUP), pero no especifica en sí mismo qué metodología o proceso usar. El UML está compuesto por diversos elementos gráficos que se combinan para conformar diagramas. Debido a que el UML es un lenguaje, cuenta con reglas para combinar tales elementos. En UML 2.0, la última versión disponible de este estándar de facto, hay 13 tipos diferentes de diagramas. Para comprenderlos de manera concreta, a veces es útil categorizarlos jerárquicamente.

Los **Diagramas de Estructura** enfatizan en los elementos que deben existir en el sistema modelado; incluye, entre otros, Diagrama de clases, Diagrama de componentes, Diagrama de objetos y Diagrama de paquetes.

Los **Diagramas de Comportamiento** enfatizan en lo que debe suceder en el sistema modelado; ejemplo son el Diagrama de actividades y Diagrama de casos de uso.

Los **Diagramas de Interacción** son un subtipo de diagramas de comportamiento, que enfatiza sobre el flujo de control y de datos entre los elementos del sistema modelado; aquí se puede ver el Diagrama de secuencia y de Diagrama de colaboración.

El uso de UML y el trabajo continuo que se ha venido realizando sobre él lo convierten en una práctica que asegura la especificación de los procesos en el desarrollo del software, y es muy favorable su uso como vía de comunicación y documentación. [24]

1.7 Herramienta CASE

Ingeniería de Software Asistida por Computación. La misma propicia un conjunto de ayudas para el desarrollo de programas informáticos, desde la planificación, pasando por el análisis y el diseño, hasta la generación del código fuente de los programas y la documentación, como es el caso de Visual Paradigm [24].

Figura 7: Logo del Visual Paradigm

Según Michael Lucas Gibson [25]:

"CASE es una filosofía que se orienta a la mejor comprensión de los modelos de empresa, sus actividades y el desarrollo de los sistemas de información. Esta filosofía involucra además el uso de programas que permiten:

• Construir los modelos que describen la empresa.

• Describir el medio en el que se realizan las actividades.

• Llevar a cabo la planificación.

• El desarrollo del Sistema Informático, desde la planificación, pasando por el análisis y diseño de sistemas, hasta la generación del código de los programas y la documentación."

Sin duda alguna estas herramientas pueden ser de gran utilidad, en el ciclo de vida en el desarrollo del software. Las mismas hacen que sean más productivas las áreas de desarrollo y mantenimiento del sistema informático. Ayudan en la calidad del software elaborado, se mejora la gestión en el proyecto en cuanto a la planificación, control y ejecución. Su manejo es sencillo y reduce la dependencia de analistas y programadores.

Estas herramientas tienen el objetivo de llevar todo el ciclo de vida completo del proceso de desarrollo de software, utilizando todo tipo de diagramas para su representación. Visual Paradigm for UML Enterprise Edition en la versión 8.0 es una de ellas, siendo una variante de software libre y la que llevará a cabo todo el proceso de modelado del

software. Es una aplicación fiable, utilizada bajo el enfoque orientado a objetos, emplea un lenguaje estándar, lo que ayuda con la comunicación de todo el equipo de desarrollo, soporta aplicaciones web. Es compatible con otras versiones y entre otras características genera base de datos transformando el diagrama de Entidad – Relación en tablas de base de datos.

1.8 Sistema Gestor de Base de Datos

Un Sistema Gestor de Base de Datos (SGBD, en inglés DBMS: *Data Base Management System*) es un sistema de software que permite la definición de bases de datos; así como la elección de las estructuras de datos necesarios para el almacenamiento y búsqueda de los datos, ya sea de forma interactiva o a través de un lenguaje de programación. Un SGBD relacional es un modelo de datos que facilita a los usuarios describir los datos que serán almacenados en la base de datos junto con un grupo de operaciones para manejar los datos.

Los SGBD relacionales son una herramienta efectiva que permite a varios usuarios acceder a los datos al mismo tiempo. Brindan facilidades eficientes y un grupo de funciones con el objetivo de garantizar la confidencialidad, la calidad, la seguridad y la integridad de los datos que contienen, así como un acceso fácil y eficiente a los mismos.[26]

1.8.1PostgreSQL

Figura 8: Logo de PostgreSQL

PostgreSQL es un Sistema Gestor de Bases de Datos Relacionales Orientadas a Objetos, derivado de Postgres, desarrollado en la Universidad de California, en el Departamento de Ciencias de la Computación de Berkeley. Es un gestor de bases de datos de código abierto, brinda un control de concurrencia multi-versión (MVCC por sus siglas en inglés) que permite trabajar con grandes volúmenes de datos; soporta gran parte de la sintaxis SQL y cuenta con un extenso grupo de enlaces con lenguajes de programación.

Posee características significativas del motor de datos, entre las que se pueden incluir las subconsultas, los valores por defecto, las restricciones a valores en los campos (*constraints*) y los disparadores (*triggers*). Ofrece funcionalidades en línea con el estándar SQL92, incluyendo claves primarias, identificadores entrecomillados, conversión de tipos y entrada de enteros binarios y hexadecimales.

El código fuente se encuentra disponible para todos sin costo alguno. Está disponible para 34 plataformas con la última versión estable. Es totalmente compatible con ACID (acrónimo de *Atomicity*, *Consistency*, *Isolation* and *Durability*; en español: Atomicidad, Consistencia, Aislamiento y Durabilidad).

Posee una integridad referencial e interfaces nativas para lenguajes como ODBC, JDBC, C, C++, PHP, PERL, TCL, ECPG; PYTHON y RUBY. Funciona en todos los sistemas operativos Linux, UNIX (AIX, BSD, HP-UX, SGI IRIX, Mac OS X, Solaris, Tru64), y Windows.
Debido a la liberación de la licencia, PostgreSQL se puede usar, modificar y distribuir de forma gratuita para cualquier fin, ya sea privado, comercial o académico. [26].

1.8.2 MySQL

MySQL es un sistema gestor de bases de datos relacionales rápido, sólido y flexible. Es idóneo para la creación de bases de datos con acceso desde páginas web dinámicas, así como para la creación de cualquier otra solución que implique el almacenamiento de datos, posibilitando realizar múltiples y rápidas consultas. Está desarrollado en C y C++, facilitando su integración en otras aplicaciones desarrolladas también en esos lenguajes.

Es un sistema cliente/servidor, por lo que permite trabajar como servidor multiusuario y de subprocesamiento múltiple, o sea, cada vez que se crea una conexión con el servidor, el programa servidor establece un proceso para manejar la solicitud del cliente, controlando así el acceso simultáneo de un gran número de usuarios a los datos y asegurando el acceso a usuarios autorizados solamente. Es uno de los sistemas gestores de bases de datos más utilizado en la actualidad, utilizado por grandes corporaciones como Yahoo! Finance, Google, Motorola, entre otras. [26]

1.8.3 Selección del SGBD

A continuación se detallan las principales características de MySQL y PostgreSQL como posibles gestores de base datos a utilizarse para el desarrollo.

MySQL por su parte es más ligero que PostgreSQL y la resolución de consultas tiende a realizarse más rápido, son factores también positivos la amplia documentación existente y la calidad de las herramientas de administración. PostgreSQL por su parte provee bases de datos con altos niveles de integridad y consistencia, su comportamiento ante consultas concurrentes es mucho mejor que MySQL, es escalable y estable antes cargas excesivas [27].

Finalmente, se selecciona PostgreSQL para el desarrollo de la aplicación en correspondencia de la tecnología actualmente usada por el COJ y las líneas estratégicas de la UCI y atendiendo fundamentalmente a su superioridad en el manejo de la concurrencia. Además, las carencias de MySQL con respecto al uso de transacciones, trigger y el manejo de integridad referencial hacen de PostgreSQL un mejor candidato.

1.9 Conclusiones Parciales

Al realizar la debida investigación teórica del tema se realiza un resumen sobre los conceptos fundamentales que se abordan principalmente para la comprensión de las pruebas que se van a realizar. Se realizó un análisis de las aplicaciones existentes, documentando sus principales características: lenguajes que soportan, algoritmos en que basan su funcionamiento, forma de acceder al servicio, presentación de los resultados, rendimiento y tipo de patente. Atendiendo a las tendencias actuales y principalmente a la experiencia del sitio COJ, se eligieron las siguientes herramientas y se fundamentó la elección de las mismas para completar las necesidades del proceso de desarrollo, como metodología de desarrollo XP, como lenguaje de programación se seleccionó a Java complementando con el framework JUnit para la implementación de las pruebas, el gestor de bases de datos PostgreSQL, Eclipse como IDE de programación y el Visual Paradigm como herramienta CASE.

Capítulo 2: Desarrollo de la solución propuesta

2.1 Introducción

En el siguiente capítulo se realiza un análisis de las características del sistema como propuesta de solución, para que exista un mejor entendimiento de la aplicación que se quiere desarrollar, generando los artefactos necesarios en el modelamiento de la lógica del negocio del sistema. También se definirán los requisitos funcionales y no funcionales que el producto debe cumplir, además de su estructura.

2.2 Modelo de dominio

El modelo de dominio es un artefacto de análisis, que se basa en las reglas de UML en la fase de concepción. El mismo está formado por uno o varios diagramas de clases que contienen, no conceptos propios de un software sino de la propia realidad física. Este modelo se emplea fundamentalmente como medio para comprender el negocio al cual el sistema va a servir.

Utilizando la notación UML, un modelo de dominio se representa con un conjunto de diagramas de clases en los que no se define ninguna operación. Pueden mostrar [28]:

- Objetos del dominio o clases conceptuales.
- Asociaciones entre las clases conceptuales.
- Atributos de las clases conceptuales.

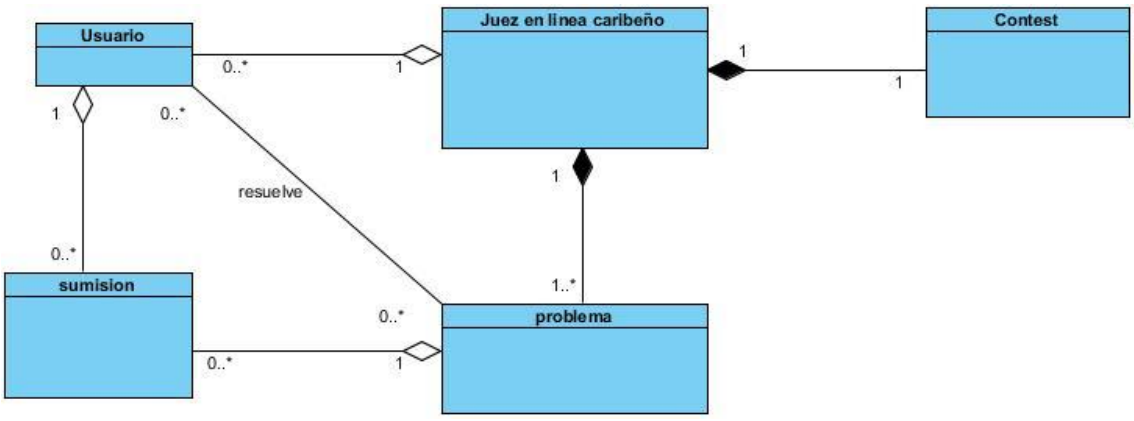


Figura 9: Modelo de Dominio

2.3 Requerimientos

Luego de analizarse el dominio del problema, es necesario definir lo que hará el sistema, para ello se determinan los requisitos que, según Jacobson [29], no son más que las condiciones o capacidades que tienen que ser alcanzadas por un sistema para

satisfacer las necesidades del cliente. Los mismos se clasifican en requisitos funcionales y no funcionales. Los requisitos funcionales son capacidades o condiciones que el sistema debe cumplir, mientras que los no funcionales son propiedades o cualidades que el producto debe tener [29].

2.3.1 Requisitos funcionales

La suite de pruebas están enmarcadas en dos tipos de pruebas, los cuales, una parte es mediante la herramienta Selenium y la otra en el framework JUnit. Los requisitos de las pruebas del Selenium están divididas en dos departamentos el primero con usuario anónimo y el segundo con usuario autenticado los cuales tienes la siguiente estructura:

2.3.1.1 Con usuario anónimo:

RF1: Permitir autenticarse con usuario.

RF2: Fallar autenticación con usuario.

RF3: Permitir autenticarse con correo.

RF4: Permitir cambiar el lenguaje del sitio a inglés.

RF5: Permitir cambiar el lenguaje del sitio a español.

RF6: Mostrar el problema.

RF7: Ver envíos del problema.

RF8: Ver mejores soluciones del problema.

RF9: Permitir cambiar el lenguaje del problema a inglés.

RF10: Permitir cambiar el lenguaje del problema a español.

RF11: Permitir cambiar el lenguaje del problema a portugués.

RF12: Probar filtrado del volumen de problemas por nombre.

RF13: Probar filtrado del volumen de problemas por ID de problema.

RF14: Probar ordenamiento ascendente del volumen de problemas por envíos.

RF15: Probar ordenamiento descendente del volumen de problemas por envíos.

RF16: Probar ordenamiento ascendente del volumen de problemas por aceptados.

RF17: Probar ordenamiento descendente del volumen de problemas por aceptados.

RF18: Probar la marca de juzgado especial (problema 2988).

RF19: Probar totales de envíos en la página Mejores Soluciones de problema.

RF20: Probar filtrado de la página de estado por nombre.

RF21: Probar filtrado de la página de estado por problema.

RF22: Probar filtrado de la página de estado por resultado.

RF23: Probar filtrado de la página de estado por lenguaje.

RF24: Probar filtrado de la página de estado por criterios combinados.

RF25: Probar Ver Descripción de Concursos Pasados.

RF26: Probar Ver Problemas de Concursos Pasados.

RF27: Probar ranking agrupado de Concursos Pasados.

RF28: Probar ranking desagrupado de Concursos Pasados.

RF29: Probar Ver Descripción de Concursos Próximos.

2.3.1.2 Con usuario autenticado:

RF30: Permitir cambiar el lenguaje del sitio a inglés.

RF31: Permitir cambiar el lenguaje del sitio a español.

RF32: Mostrar el problema.

RF33: Ver envíos del problema.

RF34: Ver mejores soluciones del problema.

RF35: Enviar el problema 1000 correctamente.

RF36: Enviar el problema 1000 solución AC.

RF37: Enviar el problema 1000 solución WA.

RF38: Permitir cambiar el lenguaje del problema a inglés.

RF39: Permitir cambiar el lenguaje del problema a español.

RF40: Permitir cambiar el lenguaje del problema a portugués.

RF41: Probar filtrado del volumen de problemas por nombre.

RF42: Probar filtrado del volumen de problemas por ID de problema.

RF43: Probar filtrado del volumen de problemas por clasificación.

RF44: Probar filtrado del volumen de problemas por complejidad.

RF45: Probar ordenamiento ascendente del volumen de problemas por envíos.

RF46: Probar ordenamiento descendente del volumen de problemas por envíos.

RF47: Probar ordenamiento ascendente del volumen de problemas por aceptados.

RF48: Probar ordenamiento descendente del volumen de problemas por aceptados.

RF49: Probar la marca de juzgado especial (problema 2988).

RF50: Probar totales de envíos en la página Mejores Soluciones de problema.

RF51: Probar filtrado de la página de estado por nombre.

RF52: Probar filtrado de la página de estado por problema.

RF53: Probar filtrado de la página de estado por resultado.

RF54: Probar filtrado de la página de estado por lenguaje.

RF55: Probar filtrado de la página de estado por criterios combinados.

RF56: Probar Ver Descripción de Concursos Pasados.

RF57: Probar Ver Problemas de Concursos Pasados.

RF58: Probar ranking agrupado de Concursos Pasados.

RF59: Probar ranking desagrupado de Concursos Pasados.

RF60: Probar Ver Descripción de Concursos Próximo.

2.3.1.3 Requisitos mediante JUnit

RF61: Adicionar problema

RF62: Editar problema

RF63: Deshabilitar problema para 24h

RF64: Traducir el problema

RF65: Adicionar Concurso

RF66: Adicionar usuarios a concursos

RF67: Autenticarse

2.4 Requisitos no funcionales

Los requisitos no funcionales, como su nombre sugiere, son aquellos requisitos que no se refieren directamente a las funciones específicas que proporciona el sistema, sino a las propiedades emergentes de éste como la fiabilidad, el tiempo de respuesta y la capacidad de almacenamiento. [30] Los mismos son las propiedades o cualidades que un producto debe tener, que se refieren a las características que hacen al producto usable, rápido o confiable. Para el desarrollo de la presente investigación se definen los siguientes requisitos no funcionales:

Usabilidad

La ejecución del sistema debe ser intuitiva para el desarrollador promedio. Los resultados de la misma deben ser claramente legibles para el desarrollador promedio.

Disponibilidad

La aplicación deberá estar disponible en todo momento sin necesidad de requisitos previos.

Soporte

La aplicación se mantiene actualizada con los cambios del sistema COJ constantemente. Se mantiene el código fuente en un control de versiones, sincronizado con el control de versiones principal del COJ.

Rendimiento

El sistema tratará de cumplir con el rendimiento previsto, dándole respuesta de manera rápida a cada una de las solicitudes ejecutadas por el desarrollador principal.

Portabilidad

El sistema de pruebas se ejecutará en versiones de Firefox 24+ y Eclipse 3.4+, sin importar el sistema operativo subyacente. Software Mozilla Firefox 24 o superior, Eclipse 3.4 o superior.

Legales

Se usarán herramientas de software libre bajo las licencias GNU/GPL.

2.5 Selenium IDE

Selenium IDE es un plugin de Firefox que pertenece al juego de herramientas SeleniumHQ, permite realizar juegos de pruebas sobre aplicaciones web. Para ello

realiza la grabación de la acción seleccionada (navegación por una página) en un "script", el cual se puede editar y paramétrica para adaptarse a los diferentes casos, y lo que es más importante su ejecución se puede repetir tantas veces como se quiera.

El principal objetivo de este plugin es crear pruebas funcionales, aunque no se puede pasar por alto que este tipo de herramientas permite automatizar tareas que requieren un cierto "procesamiento" mental básico:

- Rellenar formularios (autenticación o cualquier otro tipo)
- Navegación web
- Acciones de gestión (CRUD de comentarios blog / correos / noticias / entre otros)

Esta herramienta permite al desarrollador web ahorrarse mucho esfuerzo (mucho esfuerzo = muchas horas) cada vez que se resuelve alguna incidencia o se genera una versión nueva. Para ello permite automatizar la realización de las pruebas ya sean o bien pruebas específicas (una acción en particular) o bien juegos de pruebas (un conjunto de acciones).

2.5.1 Características:

- Facilidad de registro y ejecución de los test.
- Referencia a objetos DOM en base al ID, nombre o a través de XPath.
- Autocompletado para todos los comandos.
- Las acciones pueden ser ejecutadas paso a paso.
- Herramientas de depuración y puntos de ruptura (*breakpoints*).
- Los test pueden ser almacenados como HTML y scripts Ruby, entre otros formatos.
- Soporte para Selenium user-extensions.js.
- Ejecución en varios navegadores.
- Uso de diferentes API's en diferentes lenguajes (PHP, Ruby, JAVA, Javascript, entre otros).

2.5.2 Requerimientos de Selenium.

En este punto se indicará la compatibilidad de este plugin con diferentes configuraciones (navegador / sistema operativo / lenguaje de programación), describiendo las acciones permitidas o bien los problemas encontrados al ejecutarlo con esa configuración.

Navegador

Navegador	Funcionamiento
Firefox 3	Grabar y reproducir test
Firefox 2	Grabar y reproducir test
IE 8b1	No está soportado
IE 7	No está soportado
Safari 3	No está soportado
Safari 2	No está soportado
Opera 9	No está soportado
Opera 8	No está soportado
Otros	No está soportado

Sistema Operativo

Sistema operativo	Funcionamiento
Windows	Trabaja con Firefox 2 o más
OS X	Trabaja con Firefox 2 o más
Linux	Trabaja con Firefox 2 o más
Solaris	Trabaja con Firefox 2 o más
Otros	Debería de trabajar con Firefox 2 o mas

Lenguaje de programación

Lenguaje	Funcionamiento
C#	Genera código
Java	Genera código
Perl	Genera código
PHP	Genera código
Python	Genera código
Ruby	Genera código
Otros	Genera código personalizado

2.5.3 Panel Selenium IDE.

En este punto se va a dar una explicación detallada del panel que nos va a permitir controlar el plugin. Para ello se explicarán las diferentes áreas en las que se compone, así como las diferentes opciones permitidas.

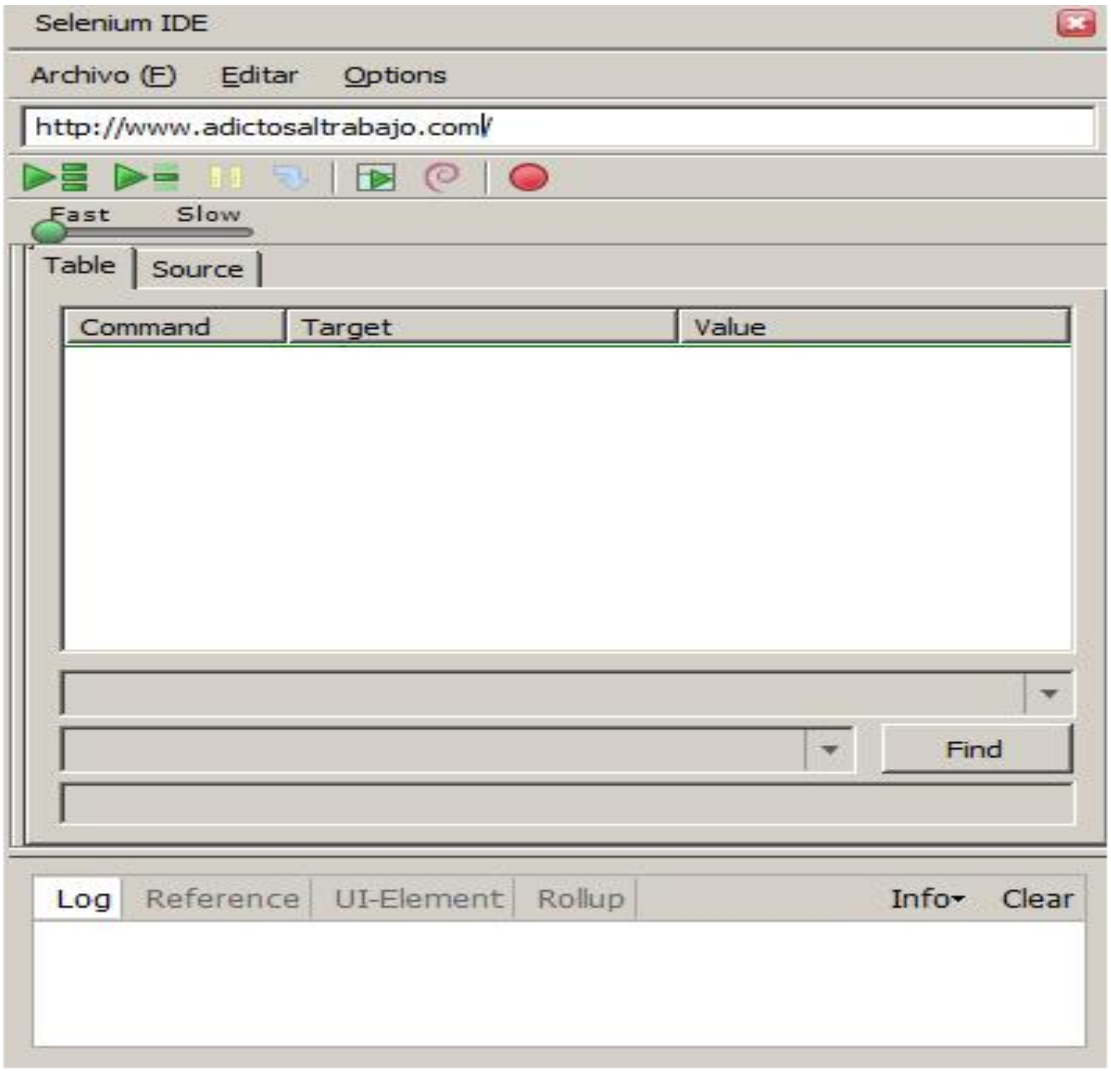
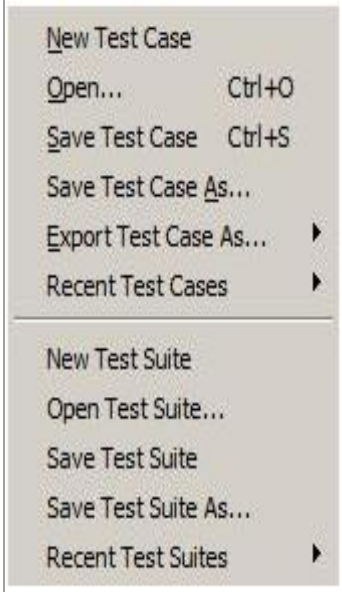
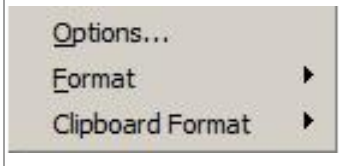


Figura 10: Vista principal de Selenium

Tabla2: Opciones del plugin

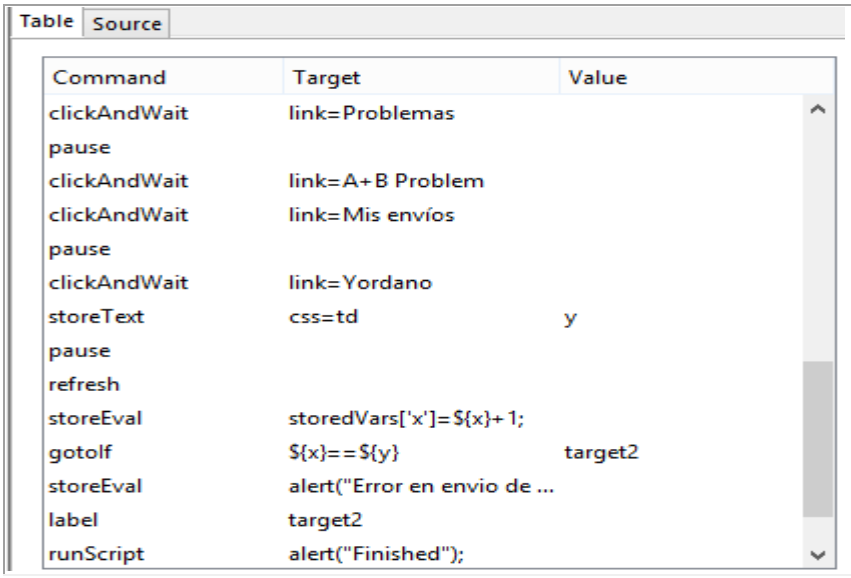
Acción	Detalle	Descripción
Opciones Generales del Plugin	Archivo (F) Editar Options	Esta es la barra de menús principales.

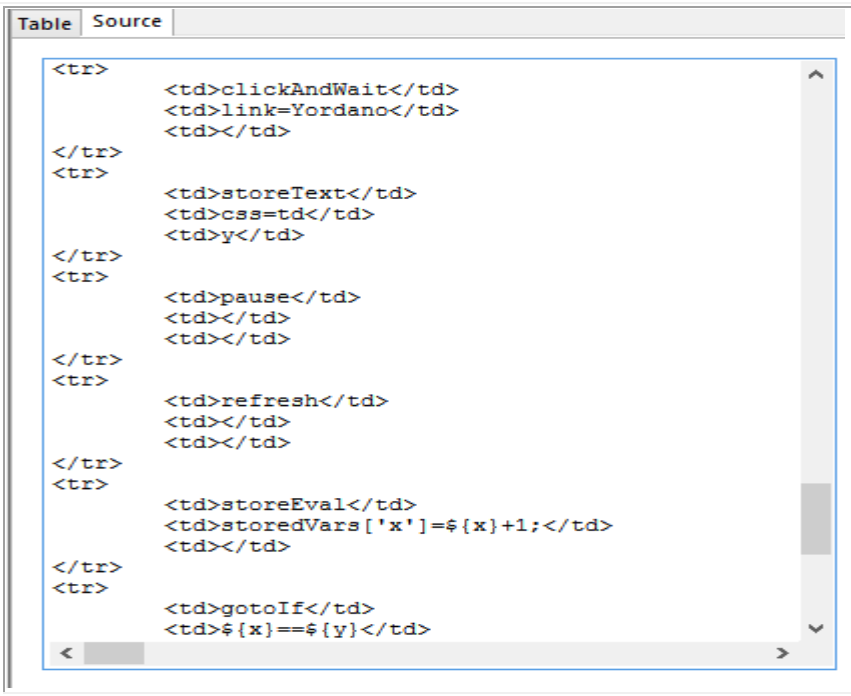
Acciones Archivo		• New Test Case : Generación de un caso de prueba • Open... : Abrir caso de prueba • Save Test Case : Guardar caso de prueba • Save Test Case As... : Guardar caso de prueba como se indique • Export Test Case As... : Exportar caso de prueba en formato de lenguaje de programación Selenium RC • Recent Test Cases : Casos de prueba usados recientemente • New Test Suite : Generación de un juego de pruebas • Open Test Suite... : Abrir juego de pruebas • Save Test Suite : Guardar juego de pruebas • Save Test Suite As... : Guardar juego de pruebas como se indique • Recent Test Suite : Juegos de prueba usados recientemente
Acciones Edición		Acciones típicas de edición

Acciones Options		• Options :
		  ○ Codificación del script (Por ejemplo : UTF-8)
		  ○ Valor por defecto del timeout en la grabación
		  ○ Extensiones Selenium Core
		  ○ Extensiones Selenium IDE
		  ○ Recordar URL base
		  ○ Grabar assertTitle automáticamente
		  ○ Grabar URL absoluta
		  ○ Opciones de formato
		• Format : Lenguaje de formato de grabación del script

		• Clipboard Format : Lenguaje del formato de grabación en el Portapapeles

Tabla 3: Detalle del panel

Elemento	Funcionamiento
	URL sobre la que se realizará la grabación
	Reproducir Prueba
	Reproducir Prueba
	Parar prueba
	Ejecución paso a paso
	Reproducir con Selenium TestRunner
	
	Grabar Prueba
	Velocidad de ejecución de la prueba
	Panel de comandos

	Panel de código
	Comando
	Destino
	Valor
	Área de información

2.6 Solución con Selenium IDE

La solución se concibe como una suite de casos de prueba ejecutables desde Firefox, mediante la herramienta anteriormente mencionada Selenium IDE. Esta suite de casos de prueba automática estará divida en dos secciones las cuales serán pruebas con usuario anónimo y pruebas con usuario autenticado.

A continuación se representa una tabla en donde se encuentra una lista de los casos de prueba y una lista de los requisitos funcionales respondidos por las pruebas.

Pruebas con usuario anónimo:

Tabla 4: Pruebas con usuario anónimo

Casos de prueba	Requisitos funcionales respondidos por las pruebas
Logueo Usuario	RF1: Permitir autenticarse con usuario.

Logueo fallido	RF2: Fallar autenticación con usuario.
Logueo correo	RF3: Permitir autenticarse con correo.
Cambio de idiomas	RF4: Permitir cambiar el lenguaje del sitio a inglés.
	RF5: Permitir cambiar el lenguaje del sitio a español.
Ver problema	RF6: Mostrar el problema.
Ver envíos del problema	RF7: Ver envíos del problema.
Ver las mejores soluciones del problema	RF8: Ver las mejores soluciones del problema.
Cambio de idiomas en problemas	RF9: Permitir cambiar el lenguaje del problema a inglés.
	RF10: Permitir cambiar el lenguaje del problema a español.
	RF11: Permitir cambiar el lenguaje del problema a portugués.
Filtrado del volumen de problemas por nombre	RF12: Probar filtrado del volumen de problemas por nombre.
Filtrado del volumen de problemas por identificador	RF13: Probar filtrado del volumen de problemas por ID de problema.
Ordenamiento ascendente del volumen de problemas por envíos	RF14: Probar ordenamiento ascendente del volumen de problemas por envíos.
Ordenamiento descendente del volumen de problemas por envíos	RF15: Probar ordenamiento descendente del volumen de problemas por envíos.
Ordenamiento ascendente del volumen de problemas por aceptados	RF16: Probar ordenamiento ascendente del volumen de problemas por aceptados.
Ordenamiento descendente del volumen de problemas por aceptados	RF17: Probar ordenamiento descendente del volumen de problemas por aceptados.
Marca de juzgado especial	RF18: Probar la marca de juzgado especial (problema 2988).

Totales de envíos en la página Mejores Soluciones de problema.	RF19: Probar totales de envíos en la página Mejores Soluciones de problema.
Filtrado de la página de estado por nombre de usuario	RF20: Probar filtrado de la página de estado por nombre.
Filtrado de la página de estado por problema	RF21: Probar filtrado de la página de estado por problema.
Filtrado de la página de estado por resultado	RF22: Probar filtrado de la página de estado por resultado.
Filtrado de la página de estado por lenguaje	RF23: Probar filtrado de la página de estado por lenguaje.
Filtrado de la página de estado por criterios combinados	RF24: Probar filtrado de la página de estado por criterios combinados.
Descripción de Concursos Pasados	RF25: Probar Ver Descripción de Concursos Pasados.
Ver Problemas de Concursos Pasados	RF26: Probar Ver Problemas de Concursos Pasados.
Ranking agrupado de Concursos Pasados	RF27: Probar ranking agrupado de Concursos Pasados.
Ranking desagrupado de Concursos Pasados	RF28: Probar ranking desagrupado de Concursos Pasados.
Ver Descripción de Concursos Próximos	RF29: Probar Ver Descripción de Concursos Próximos.

Tabla 5: Pruebas con usuario autenticado

Casos de prueba	Requisitos funcionales respondidos por las pruebas
Cambiar lenguaje del sitio a inglés (Autenticado)	RF30: Permitir cambiar el lenguaje del sitio a inglés.
Cambiar lenguaje del sitio a español (Autenticado)	RF31: Permitir cambiar el lenguaje del sitio a español.
Ver problema (autenticado)	RF32: Mostrar el problema.
Ver envíos del problema (autenticado)	RF33: Ver envíos del problema.
Ver mejores soluciones del problema (autenticado)	RF34: Ver mejores soluciones del problema.

Enviar problema correctamente (Autenticado)	RF35: Enviar el problema 1000 correctamente.
Enviar problema correctamente AC (Autenticado)	RF36: Enviar el problema 1000 solución AC.
Enviar problema correctamente WA (Autenticado)	RF37: Enviar el problema 1000 solución WA.
Cambiar lenguaje del problema a inglés (Autenticado)	RF38: Permitir cambiar el lenguaje del problema a inglés.
Cambiar lenguaje del problema a español (Autenticado)	RF39: Permitir cambiar el lenguaje del problema a español.
Cambiar lenguaje del problema a portugués (Autenticado)	RF40: Permitir cambiar el lenguaje del problema a portugués.
Filtrado de problemas por nombre (Autenticado)	RF41: Probar filtrado del volumen de problemas por nombre.
Filtrado de problemas por id (Autenticado)	RF42: Probar filtrado del volumen de problemas por ID de problema.
Filtrado de problemas por clasificación (Autenticado)	RF43: Probar filtrado del volumen de problemas por clasificación.
Filtrado de problemas por complejidad (Autenticado)	RF44: Probar filtrado del volumen de problemas por complejidad.
Ordenamiento ascendente del volumen de problemas por envíos (Autenticado)	RF45: Probar ordenamiento ascendente del volumen de problemas por envíos.
Ordenamiento descendente del volumen de problemas por envíos (Autenticado)	RF46: Probar ordenamiento descendente del volumen de problemas por envíos.
Ordenamiento ascendente del volumen de problemas por aceptados (Autenticado)	RF47: Probar ordenamiento ascendente del volumen de problemas por aceptados.
Ordenamiento descendente del volumen de problemas por aceptados (Autenticado)	RF48: Probar ordenamiento descendente del volumen de problemas por aceptados.
Marca de juzgado especial (Autenticado)	RF49: Probar la marca de juzgado especial (problema 2988).

Mejores soluciones del problema correctamente (Autenticado)	RF50: Probar totales de envíos en la página Mejores Soluciones de problema.
Filtrado de sentencias por nombre (Autenticado)	RF51: Probar filtrado de la página de estado por nombre.
Filtrado de sentencias por id problema (Autenticado)	RF52: Probar filtrado de la página de estado por problema.
Filtrado de sentencias por resultado (Autenticado)	RF53: Probar filtrado de la página de estado por resultado.
Filtrado de sentencias por lenguaje (Autenticado)	RF54: Probar filtrado de la página de estado por lenguaje.
Filtrado de sentencias por criterios combinados (Autenticado)	RF55: Probar filtrado de la página de estado por criterios combinados.
Ver descripción de concursos anteriores (Autenticado)	RF56: Probar Ver Descripción de Concursos Pasados.
Ver problemas de concursos anteriores (Autenticado)	RF57: Probar Ver Problemas de Concursos Pasados.
Posiciones agrupadas de los concursos pasados (Autenticado)	RF58: Probar ranking agrupado de Concursos Pasados.
Posiciones desagrupadas de los concursos pasados (Autenticado)	RF59: Probar ranking desagrupado de Concursos Pasados.
Ver descripción de concursos próximos (Autenticado)	RF60: Probar Ver Descripción de Concursos Próximos.

2.7 JUnit

JUnit es un paquete Java utilizado para automatizar los procesos de prueba. Mediante la creación de Tests, JUnit realizará una prueba en el código que indique el usuario.

Siempre que se vaya a desarrollar algún tipo de software, habrá que tener en cuenta las pruebas a realizar, con esto, se cerciora que nuestro programa o librería funcionen correctamente.

Normalmente las pruebas se realizan por parte del programador, esto incluye que el orden elegido podría no ser correcto y con ello alargar demasiado el trabajo.

Aunque inicialmente el proceso sea más rápido, con el avance de la aplicación podría complicarse el trabajo, ya que cada vez que se fuera a probar algo, habría que volver a escribir el código para realizar la prueba ya que no se tiene la certeza de cuáles serán

los módulos afectados con varios cambios, ni se podrá adivinar exactamente la línea donde se ha generado el error.

Existen varias razones para utilizar JUnit a la hora de hacer pruebas de código:

•La herramienta no tiene coste alguno, por lo que se podrá descargar directamente desde la Web oficial.

•Es una herramienta muy utilizada, por lo que no será complicado buscar documentación.

•Existen varios plugins para poder utilizar con diferentes Entornos de Desarrollo Integrado (IDE).

•Existen también muchas herramientas de pruebas de cobertura que utilizarán como base JUnit.

•Con JUnit, ejecutar tests es tan fácil como compilar tu código. El compilador "testea" la sintaxis del código y los tests "validan" la integridad del código.

•Los resultados son chequeados por la propia aplicación y dará los resultados inmediatamente.

•Los tests JUnit se pueden organizar en suites, las que contendrán ejemplares de tests, incluso podrán contener otras suites.

•Utilizando los tests programados en JUnit, la estabilidad de nuestras aplicaciones mejorará sustancialmente.

•Los tests realizados se podrán presentar junto con el código, para validar el trabajo realizado. [31]

2.8 JUnit en Eclipse

Utilizando el IDE Eclipse se pueden crear clases JUnit Test Case y Test Suites. Cuando se cree el primer Test Case se pedirá la versión de JUnit a utilizar (versión 3 o 4) y dependiendo de la utilizada sugerirá la introducción de la librería en el *classpath* (paquete de clases) del proyecto. A continuación se mostrarán ejemplos de cómo crear pruebas utilizando las dos versiones.

El primer paso sería crear un TestCase. Para ello sobre el paquete deseado se presionara el botón derecho del ratón y se va a New > Other y selecciona dentro de la carpeta Java > JUnit > JUnit TestCase.

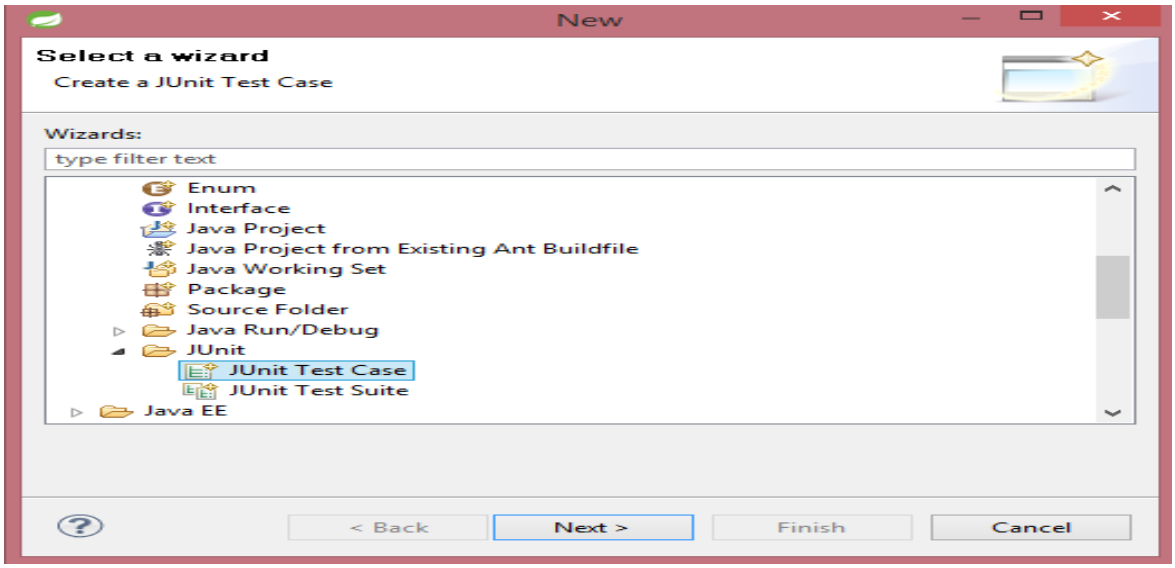


Figura 11: Creación de una clase JUnit en el Eclipse

Una vez ahí se decidirá la versión de JUnit a utilizar:

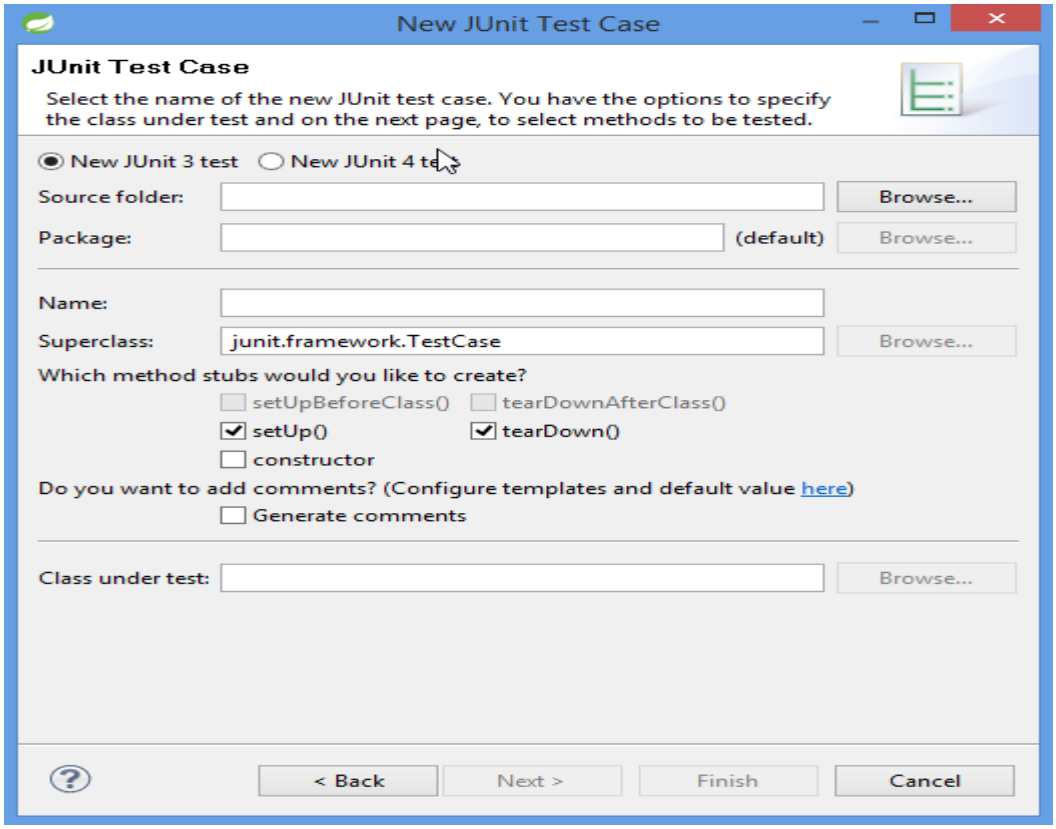


Figura 12: Crear clase JUnit

2.9 Características de JUnit 4

En primer lugar es importante que el nombre de la clase tenga la siguiente estructura: "test"+ "nombre", por ejemplo "testAutenticarse".

Métodos

A grandes rasgos, una clase de Test realizada para ser tratada por JUnit 4 tiene una estructura con 4 tipos de métodos:

- Método setUp: se asignan valores iniciales a variables antes de la ejecución de cada test. Si se quiere que se inicialicen al principio una vez, el método se debe llamar "setUpClass"
- Método tearDown: Es llamado después de cada test y puede servir para liberar recursos o similar. Igual que antes, si queremos que solo se llame al final de la ejecución de todos los test, se debe llamar "tearDownClass"
- Métodos Test: Contienen las pruebas concretas que vamos a realizar.

- Métodos auxiliares.

Anotaciones

En versiones anteriores de JUnit no existían caracteres especiales, que se llaman anotaciones, y que se han incluido en su versión 4 para intentar simplificar más la labor del programador. Se trata de palabras clave que se colocan delante de los definidos antes y que indican a las librerías JUnit instrucciones concretas.

A continuación se revelan las más importantes:

- @RunWith: Se le asigna una clase a la que JUnit invocará en lugar del ejecutor por defecto de JUnit.
- @Before: Indica que el siguiente método se debe ejecutar antes de cada test (precede al método setUp). Si tiene que preceder al método setUpClass, la notación será "@BeforeClass".
- @After: indica que el siguiente método se debe ejecutar después de cada test (precede al método tearDown). Si tiene que preceder al método tearDownClass, la notación será "@AfterClass".
- @Test: Indica a Junit que se trata de un método de Test. En versiones anteriores de JUnit los métodos tenían que tener un nombre con la siguiente estructura: "test". Con esta notación colocada delante de los métodos se puede elegir el nombre libremente.

Funciones de aceptación/rechazo

Una vez que se han creado las condiciones para probar que una funcionalidad concreta funciona es necesario que un validador diga si se está obteniendo el resultado esperado o no. Para esta labor se definen una lista de funciones (incluidas en la clase Assert) que se pueden ver detalladas en el javadoc de JUnit: [32]

Ahora se detallarán las más comunes:

- assertArrayEquals: Recibe como parámetro 2 *arrays* (arreglos) y comprueba si son iguales. Devuelve assertionError si no se produce el resultado esperado.
- assertEquals: Realiza la comprobación entre 2 valores de tipo numérico. Devuelve assertionError si no se produce el resultado esperado.
- assertTrue: Comprueba si una condición se cumple. Devuelve assertionError si no se produce el resultado esperado.
- fail: devuelve una alerta informando del fallo en el test.

2.10 Caso de prueba

Un caso de prueba o test case es, en ingeniería del software, un conjunto de condiciones o variables bajo las cuales un analista determinará si una aplicación, un sistema software (software system), o una característica de estos es parcial o completamente satisfactoria.

Se pueden realizar muchos casos de prueba para determinar que un requisito es completamente satisfactorio. Con el propósito de comprobar que todos los requisitos de una aplicación son revisados, debe haber al menos un caso de prueba para cada requisito a menos que un requisito tenga requisitos secundarios. En ese caso, cada requisito secundario deberá tener por lo menos un caso de prueba. Algunas metodologías como RUP recomiendan el crear por lo menos dos casos de prueba para cada requisito. Uno de ellos debe realizar la prueba positiva de los requisitos y el otro debe realizar la prueba negativa.

Si la aplicación es creada sin requisitos formales, entonces los casos de prueba se escriben basados en la operación normal de programas de una clase similar.

Lo que caracteriza un escrito formal de caso de prueba es que hay una entrada conocida y una salida esperada, los cuales son formulados antes de que se ejecute la prueba. La entrada conocida debe probar una precondición y la salida esperada debe probar una post-condición.

Bajo circunstancias especiales, podría haber la necesidad de ejecutar la prueba, producir resultados, y luego un equipo de expertos evaluaría si los resultados se pueden considerar como "correctos". Esto sucede a menudo en la determinación del número del rendimiento de productos nuevos. La primera prueba se toma como línea base para los subsecuentes ciclos de pruebas/lanzamiento del producto.

Los casos de prueba escritos, incluyen una descripción de la funcionalidad que se probará, la cual es tomada ya sea de los requisitos o de los casos de uso, y la preparación requerida para asegurarse de que la prueba pueda ser dirigida.

Los casos de prueba escritos se recogen generalmente en una suite de pruebas.

Las variaciones de los casos de prueba son comúnmente utilizadas en pruebas de aceptación. La prueba de aceptación es realizada por un grupo de usuarios finales o los clientes del sistema, para asegurarse que el sistema desarrollado cumple sus requisitos. La prueba de aceptación de usuario se distingue generalmente por la incorporación de un trayecto feliz o casos de prueba positivos.

2.10.1 Estructura de los casos de prueba

Formalmente, los casos de prueba escritos consisten principalmente en tres partes con subdivisiones:

- Introducción/visión general: Contiene información general acerca de los Casos de Prueba.
    - Identificador: Es un identificador único para futuras referencias, por ejemplo, mientras se describe un defecto encontrado.
    - Caso de prueba dueño/creador: Es el nombre del analista o diseñador de pruebas, quien ha desarrollado pruebas o es responsable de su desarrollo.
    - Versión: La actual definición del caso de prueba.
    - Nombre: El caso de prueba debe ser un título entendible por personas, para la fácil comprensión del propósito del caso de prueba y su campo de aplicación.
    - Identificador de requerimientos el cual está incluido por el caso de prueba. También aquí puede ser un identificador de casos de uso o de especificación funcional.
    - Propósito: Contiene una breve descripción del propósito de la prueba, y la funcionalidad que chequea.
    - Dependencias: Indica qué otros subsistemas están involucrados y en qué grado.
- Actividades de los casos de prueba
    - Ambiente de prueba/configuración: Contiene información acerca de la configuración del hardware o software en el cuál se ejecutará el caso de prueba.

  - Inicialización: Describe acciones, que deben ser ejecutadas antes de que los casos de prueba se hayan inicializado. Por ejemplo, se debe abrir algún archivo.
  - Finalización: Describe acciones, que deben ser ejecutadas después de realizado el caso de prueba. Por ejemplo si el caso de prueba estropea la base de datos, el analista debe restaurarla antes de que otro caso de prueba sea ejecutado.
  - Acciones: Pasos a realizar para completar la prueba.
  - Descripción de los datos de entrada.
- Resultados
  - Salida esperada: Contiene una descripción de lo que el analista debería ver tras haber completado todos los pasos de la prueba.
  - Salida obtenida: Contiene una breve descripción de lo que el analista encuentra después de que los pasos de prueba se hayan completado.
  - Resultado: Indica el resultado cualitativo de la ejecución del caso de prueba, a menudo con un Correcto/Fallido.
  - Severidad: Indica el impacto del defecto en el sistema: Grave, Mayor, Normal, Menor.
  - Evidencia: En los casos que aplica, contiene un link al imprimir de pantalla (*screenshot*) donde se evidencia la salida obtenida.
  - Seguimiento: Si un caso de prueba falla, frecuentemente la referencia al defecto implicado se debe enumerar en esta columna. Contiene el código correlativo del defecto, a menudo corresponde al código del sistema de tracking de bugs que se esté usando.
  - Estado: Indica si el caso de prueba está: No iniciado, En curso, o terminado.

2.10.2 Suite de pruebas

En el desarrollo de software, un conjunto de pruebas (test suite), menos conocido comúnmente como una suite de validación, es un conjunto de casos de prueba que están destinados a ser utilizados para poner a prueba un programa de software para mostrar que tiene un conjunto específico de comportamientos. Un conjunto de pruebas a menudo contiene instrucciones o metas detalladas para cada colección de casos de prueba e información acerca de la configuración del sistema que se utilizarán durante la prueba. Un grupo de casos de prueba también puede contener estados previos o pasos, y las descripciones de los siguientes exámenes. Colecciones de casos de prueba son a

veces erróneamente denominan un plan de pruebas, un script de prueba, o incluso un escenario de prueba.

2.11 Solución con JUnit

La solución se concibe en el mismo IDE mediante la interfaz que se muestra en la figura 13. Después de ejecutar todas las pruebas se muestra de color verde las pruebas exitosas y de color rojo las pruebas fallidas, comportándose también cada método de esta misma forma. Además, la barra que se muestra arriba es de color verde solo si todas las pruebas son satisfactorias, en caso contrario se muestra roja, más arriba t muestra el número de pruebas que se ejecutaron, la cantidad de errores y la cantidad de fallos en ellos.

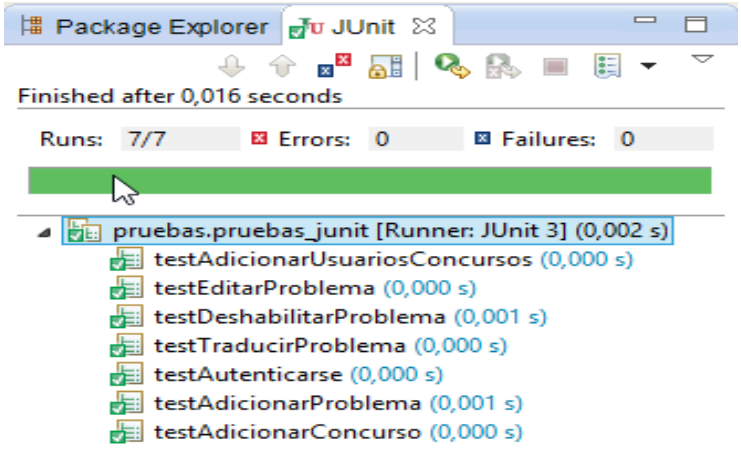


Figura 13: Interfaz de pruebas de JUnit.

A continuación se representa una tabla en donde se encuentra una lista de los casos de prueba y una lista de los requisitos funcionales respondidos por las pruebas.

Tabla 6: Pruebas de JUnit

Casos de prueba	Requisitos funcionales respondidos por las pruebas
Adicionar problema	RF61: Adicionar problema
Editar problema	RF62: Editar problema
Deshabilitar problema para 24h	RF63: Deshabilitar problema para 24h

Traducir el problema	RF64: Traducir el problema
Adicionar concurso	RF65: Adicionar Concurso
Adicionar usuarios a concursos	RF66: Adicionar usuarios a concursos
Autenticarse	RF67: Autenticarse

2.12 Personas relacionadas con el sistema

Como aspecto fundamental, debido a las características de dicho sistema y la audición en la que está enmarcada el mismo, se hace necesario delimitar el personal relacionado directamente con la aplicación. Dentro de esta audiencia se identifica como personal vinculado en este proceso al personal de desarrollo del sitio web que hasta ahora es un solo desarrollador principal.

Por consiguiente y partiendo de la propuesta de diseño se hace necesaria la presencia de un solo grupo de usuario el cual es el desarrollador principal.

Tabla 7: Personal relacionado con el sistema

Personal relacionado con el sistema	Descripción
Desarrollador principal del sistema	Este usuario es el encargado de llevar el control de la ejecución de las pruebas y el encargado de corregir los errores posteriormente.

2.13 Fase de exploración

Esta fase es la encargada de analizar a grandes rasgos, el alcance del subsistema que tiene como objetivo fundamental, el desarrollo y entrega del mismo. En este proceso los clientes juegan un papel muy importante con vistas a suplir sus necesidades, las cuales van a verse reflejadas en las historias de usuario. Desde ese entonces los desarrolladores estiman el tiempo en el que se van a programar cada una de las funcionalidades. El tiempo estimado puede variar o sufrir cambios, cuando a partir de cada iteración se haga un análisis más detallado de estos datos con los que se van a trabajar.

2.14 Historias de usuario (HU)

Las HU son el primer paso de cualquier proyecto que siga la metodología XP. Tienen la misma finalidad que los casos de uso (CU) utilizados por la metodología RUP, pero con la diferencia que constan de 3 ó 4 líneas escritas por el cliente en un lenguaje no técnico sin muchos detalles, existiendo diferencias entre estas y la especificación de requisitos, como por ejemplo el nivel de detalles, ya que las HU solo contienen información sobre la estimación del riesgo y el tiempo que requerirá su implementación.

Clasificación de las Historias de Usuario

Las HU están compuestas por tablas divididas en las siguientes secciones:

Número: número de la HU.

Nombre: nombre que identifica la HU.

Usuario: nombre de usuarios involucrados en la HU.

Prioridad en el negocio: se clasifican en alta (funcionalidades fundamentales en el desarrollo del sistema), media (funcionalidades a tener en cuenta pero que no tienen una afectación sobre el sistema) y baja (sirven de ayuda a los desarrolladores pero no tienen nada que ver con el sistema).

Nivel de complejidad: también se clasifican en alta (un error en la implementación, lleva a la inoperatividad del código), media (un error en la implementación retrasa la entrega de la versión) y baja (el error se puede tratar sin que traiga problemas mayores)

Estimación: tiempo estimado que se demorará el desarrollo de la HU.

Iteración asignada: número de la iteración en la que será realizada la HU.

Descripción: breve descripción de la HU.

Observaciones: a qué requisito hace referencia la HU.

Durante este proceso se identifican 60 HU. A continuación se muestran 6 de las 60 HU.

Tabla 8: HU Autenticar usuario

No: 1	**Nombre: Autenticar usuario**
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de autenticar automáticamente a un usuario previamente introducido
Observaciones:	Da cumplimiento al requisito 1

Tabla 9: HU Mostrar el problema

No: 1	**Nombre: Mostrar el problema**
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar el problema A+B
Observaciones:	Da cumplimiento al requisito 6

Tabla 10: HU Cambiar el lenguaje del sitio a inglés

No: 1	**Nombre: Cambiar el lenguaje del sitio a inglés**
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del sitio a inglés.
Observaciones:	Da cumplimiento al requisito 30

Tabla 11: HU Probar filtrado de la página de estado por criterios combinados

No: 1	**Nombre: Probar filtrado de la página de estado por criterios combinados**

Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 55

Tabla 12: HU Probar ver descripción de concursos próximos

No: 1	**Nombre: Probar ver descripción de concursos próximos**
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver la descripción de los concursos próximos correctamente
Observaciones:	Da cumplimiento al requisito 60

Tabla 13: HU Adicionar un concurso

No: 5	**Nombre: Adicionar un concurso**
Usuario	Administrador
Prioridad: Alta	Complejidad: Alta
Estimación:15d	Iteración:5

Descripción:	El administrador cuando ejecuta esta prueba, verifica que se adicione correctamente el concurso.
Observaciones:	Da cumplimiento al requisito 56

2.15 Fase de planificación

Durante la planificación el cliente establece la prioridad de cada HU las cuales son organizadas en iteraciones. Además se acuerda el alcance de la entrega y se realiza una estimación del esfuerzo que costará implementar cada HU.

2.15.1 Estimación de esfuerzo por Historias de Usuario

Para que el subsistema propuesto tenga un buen desempeño, se hizo una estimación de cada una de las Suites de Pruebas a realizar que se determinaron arrojando los siguientes resultados:

Tabla 14: Estimación de esfuerzo por HU

Suites de pruebas	Puntos de estimación
Pruebas en Selenium	40
Pruebas JUNIT	25

2.15.2 Plan de iteraciones

Después de determinar, atendiendo las necesidades del cliente las historias de usuario, y tras haber realizado un análisis previo por parte de los desarrolladores, del tiempo estimado para realizar cada una de las mismas, se realiza un proceso de planificación que incluye la etapa de implementación del sistema. De esta manera, quedan agrupadas por iteraciones cada una de estas historias de usuario, identificadas específicamente las que serán desarrolladas en cada iteración en el proceso de implementación. Atendiendo a lo que se ha mencionado se ha decidido que la etapa de implementación va a quedar dividida en cinco iteraciones, las que serán explicadas a continuación:

- **Iteración 1**

Esta iteración comprende las HU imprescindibles para el funcionamiento de la aplicación:

- ✓ HU Permitir autenticarse con usuario.
- ✓ HU Fallar autenticación con usuario.
- ✓ HU Permitir autenticarse con correo.

- ✓ HU Permitir cambiar el lenguaje del sitio a inglés.
- ✓ HU Permitir cambiar el lenguaje del sitio a español.
- ✓ HU Mostrar el problema.
- ✓ HU Ver envíos del problema.
- ✓ HU Permitir cambiar el lenguaje del sitio a inglés.
- ✓ HU Permitir cambiar el lenguaje del sitio a español.
- ✓ HU Mostrar el problema.
- ✓ HU Ver envíos del problema.
- ✓ HU Ver mejores soluciones del problema.
- ✓ HU Enviar el problema 1000 correctamente.
- ✓ HU Ver mejores soluciones del problema.
- ✓ HU Permitir cambiar el lenguaje del problema a inglés.
- ✓ HU Permitir cambiar el lenguaje del problema a español.
- ✓ HU Permitir cambiar el lenguaje del problema a portugués.
- ✓ HU Enviar el problema 1000 solución AC.
- ✓ HU Enviar el problema 1000 solución WA.
- ✓ HU Permitir cambiar el lenguaje del problema a inglés.
- ✓ HU Permitir cambiar el lenguaje del problema a español.
- ✓ HU Permitir cambiar el lenguaje del problema a portugués.

- **Iteración 2**

Aunque las HU que se insertan dentro de esta iteración son de prioridad alta, se agrupan en una segunda evolución de las iteraciones debido a que en ellas se determinan el trabajo con los ordenamientos de volúmenes de información:

- ✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos.
- ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos.
- ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados.
- ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados.
- ✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos.
- ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos.
- ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados.
- ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados.

- **Iteración 3**

Las HU que se implementan en esta iteración también pertenecen al grupo que presentan alta prioridad. Estas determinan el trabajo de filtrado de información.

- ✓ HU Probar filtrado del volumen de problemas por nombre.
- ✓ HU Probar filtrado del volumen de problemas por ID de problema.
- ✓ HU Probar filtrado de la página de estado por nombre.
- ✓ HU Probar filtrado de la página de estado por problema.
- ✓ HU Probar filtrado de la página de estado por resultado.
- ✓ HU Probar filtrado de la página de estado por lenguaje.
- ✓ HU Probar filtrado de la página de estado por criterios combinados.
- ✓ HU Probar filtrado del volumen de problemas por nombre.
- ✓ HU Probar filtrado del volumen de problemas por ID de problema.
- ✓ HU Probar filtrado del volumen de problemas por clasificación.
- ✓ HU Probar filtrado del volumen de problemas por complejidad.
- ✓ HU Probar filtrado de la página de estado por nombre.
- ✓ HU Probar filtrado de la página de estado por problema.
- ✓ HU Probar filtrado de la página de estado por resultado.
- ✓ HU Probar filtrado de la página de estado por lenguaje.
- ✓ HU Probar filtrado de la página de estado por criterios combinados.

- **Iteración 4**

Al inicio de esta iteración se realiza la primera entrega de la aplicación con las principales funcionalidades definidas por el usuario para ello. Las HU que pertenecen al grupo que presentan prioridad alta y son funcionalidades del módulo Concursos Reales:

- ✓ HU Probar Ver Descripción de Concursos Pasados.
- ✓ HU Probar Ver Problemas de Concursos Pasados.
- ✓ HU Probar ranking agrupado de Concursos Pasados.
- ✓ HU Probar ranking desagrupado de Concursos Pasados.
- ✓ HU Probar Ver Descripción de Concursos Próximos.
- ✓ HU Probar Ver Descripción de Concursos Pasados.
- ✓ HU Probar Ver Problemas de Concursos Pasados.
- ✓ HU Probar ranking agrupado de Concursos Pasados.
- ✓ HU Probar ranking desagrupado de Concursos Pasados.
- ✓ HU Probar Ver Descripción de Concursos Próximo.

- **Iteración 5**

En esta iteración se implementan las funcionalidades referidas en las HU de prioridad alta, vinculadas específicamente con la herramienta JUnit:

- ✓ HU Adicionar problema

- ✓ HU Editar problema
- ✓ HU Deshabilitar problema para 24h
- ✓ HU Traducir el problema
- ✓ HU Adicionar Concurso
- ✓ HU Adicionar usuarios a concursos
- ✓ HU Autenticarse

Tabla 15: Plan de duración de las iteraciones

Iteración	Orden de Historias de usuario a implementar	Duración total de iteraciones
Iteración 1	✓ HU Permitir autenticarse con usuario. ✓ HU Fallar autenticación con usuario. ✓ HU Permitir autenticarse con correo. ✓ HU Permitir cambiar el lenguaje del sitio a inglés. ✓ HU Permitir cambiar el lenguaje del sitio a español. ✓ HU Mostrar el problema. ✓ HU Ver envíos del problema. ✓ HU Permitir cambiar el lenguaje del sitio a inglés. ✓ HU Permitir cambiar el lenguaje del sitio a español. ✓ HU Mostrar el problema. ✓ HU Ver envíos del problema. ✓ HU Ver mejores soluciones del problema. ✓ HU Enviar el problema 1000 correctamente. ✓ HU Ver mejores soluciones del problema. ✓ HU Permitir cambiar el lenguaje del problema a inglés. ✓ HU Permitir cambiar el lenguaje del problema a español.	2 Semanas

	✓ HU Permitir cambiar el lenguaje del problema a portugués. ✓ HU Enviar el problema 1000 solución AC. ✓ HU Enviar el problema 1000 solución WA. ✓ HU Permitir cambiar el lenguaje del problema a inglés. ✓ HU Permitir cambiar el lenguaje del problema a español. ✓ HU Permitir cambiar el lenguaje del problema a portugués.	
Iteración 2	✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados. ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados. ✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados. ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados.	2 semanas
Iteración 3	✓ HU Probar filtrado del volumen de problemas por nombre. ✓ HU Probar filtrado del volumen de problemas por ID de problema.	3 semanas

	✓ HU Probar filtrado de la página de estado por nombre. ✓ HU Probar filtrado de la página de estado por problema. ✓ HU Probar filtrado de la página de estado por resultado. ✓ HU Probar filtrado de la página de estado por lenguaje. ✓ HU Probar filtrado de la página de estado por criterios combinados. ✓ HU Probar filtrado del volumen de problemas por nombre. ✓ HU Probar filtrado del volumen de problemas por ID de problema. ✓ HU Probar filtrado del volumen de problemas por clasificación. ✓ HU Probar filtrado del volumen de problemas por complejidad. ✓ HU Probar filtrado de la página de estado por nombre. ✓ HU Probar filtrado de la página de estado por problema. ✓ HU Probar filtrado de la página de estado por resultado. ✓ HU Probar filtrado de la página de estado por lenguaje. ✓ HU Probar filtrado de la página de estado por criterios combinados.	
Iteración 4	✓ HU Probar Ver Descripción de Concursos Pasados. ✓ HU Probar Ver Problemas de Concursos Pasados. ✓ HU Probar ranking agrupado de Concursos Pasados.	3 semanas

	✓ HU Probar ranking desagrupado de Concursos Pasados. ✓ HU Probar Ver Descripción de Concursos Próximos. ✓ HU Probar Ver Descripción de Concursos Pasados. ✓ HU Probar Ver Problemas de Concursos Pasados. ✓ HU Probar ranking agrupado de Concursos Pasados. ✓ HU Probar ranking desagrupado de Concursos Pasados. ✓ HU Probar Ver Descripción de Concursos Próximo.	
Iteración 5	✓ HU Adicionar problema ✓ HU Editar problema ✓ HU Deshabilitar problema para 24h ✓ HU Traducir el problema ✓ HU Adicionar Concurso ✓ HU Adicionar usuarios a concursos ✓ HU Autenticarse	4 semanas

2.15.3 Plan de entrega

Posteriormente a través del plan de entrega que se realiza para la fase de implementación, se establece de una forma mejor elaborada, como se han ido agrupando las HU relacionadas con los diferentes suites de pruebas, tal y como se muestra a continuación:

Tabla 16: Entrega de HU

Suites de Pruebas	Historias de usuario
Suite de pruebas en Selenium	✓ HU Permitir autenticarse con usuario. ✓ HU Fallar autenticación con usuario. ✓ HU Permitir autenticarse con correo. ✓ HU Permitir cambiar el lenguaje del sitio a inglés.

	✓ HU Permitir cambiar el lenguaje del sitio a español. ✓ HU Mostrar el problema. ✓ HU Ver envíos del problema. ✓ HU Permitir cambiar el lenguaje del sitio a inglés. ✓ HU Permitir cambiar el lenguaje del sitio a español. ✓ HU Mostrar el problema. ✓ HU Ver envíos del problema. ✓ HU Ver mejores soluciones del problema. ✓ HU Enviar el problema 1000 correctamente. ✓ HU Ver mejores soluciones del problema. ✓ HU Permitir cambiar el lenguaje del problema a inglés. ✓ HU Permitir cambiar el lenguaje del problema a español. ✓ HU Permitir cambiar el lenguaje del problema a portugués. ✓ HU Enviar el problema 1000 solución AC. ✓ HU Enviar el problema 1000 solución WA. ✓ HU Permitir cambiar el lenguaje del problema a inglés. ✓ HU Permitir cambiar el lenguaje del problema a español. ✓ HU Permitir cambiar el lenguaje del problema a portugués. ✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados. ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados.

	✓ HU Probar ordenamiento ascendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento descendente del volumen de problemas por envíos. ✓ HU Probar ordenamiento ascendente del volumen de problemas por aceptados. ✓ HU Probar ordenamiento descendente del volumen de problemas por aceptados. ✓ HU Probar filtrado del volumen de problemas por nombre. ✓ HU Probar filtrado del volumen de problemas por ID de problema. ✓ HU Probar filtrado de la página de estado por nombre. ✓ HU Probar filtrado de la página de estado por problema. ✓ HU Probar filtrado de la página de estado por resultado. ✓ HU Probar filtrado de la página de estado por lenguaje. ✓ HU Probar filtrado de la página de estado por criterios combinados. ✓ HU Probar filtrado del volumen de problemas por nombre. ✓ HU Probar filtrado del volumen de problemas por ID de problema. ✓ HU Probar filtrado del volumen de problemas por clasificación. ✓ HU Probar filtrado del volumen de problemas por complejidad. ✓ HU Probar filtrado de la página de estado por nombre. ✓ HU Probar filtrado de la página de estado por problema.

	✓ HU Probar filtrado de la página de estado por resultado. ✓ HU Probar filtrado de la página de estado por lenguaje. ✓ HU Probar filtrado de la página de estado por criterios combinados. ✓ HU Probar Ver Descripción de Concursos Pasados. ✓ HU Probar Ver Problemas de Concursos Pasados. ✓ HU Probar ranking agrupado de Concursos Pasados. ✓ HU Probar ranking desagrupado de Concursos Pasados. ✓ HU Probar Ver Descripción de Concursos Próximos. ✓ HU Probar Ver Descripción de Concursos Pasados. ✓ HU Probar Ver Problemas de Concursos Pasados. ✓ HU Probar ranking agrupado de Concursos Pasados. ✓ HU Probar ranking desagrupado de Concursos Pasados. ✓ HU Probar Ver Descripción de Concursos Próximo.
Suite de pruebas en JUnit	✓ HU Adicionar problema ✓ HU Editar problema ✓ HU Deshabilitar problema para 24h ✓ HU Traducir el problema ✓ HU Adicionar Concurso ✓ HU Adicionar usuarios a concursos ✓ HU Autenticarse

Después de confeccionado el plan de entrega, se debe especificar que este va a contener una serie de releases[2] del sistema, los que a medida que vaya avanzando la aplicación se le irán incorporando las diferentes suites de pruebas, indicándose las fechas de estos releases a continuación:

Tabla 17: Plan de duración de entregas

Módulos	Final de la iteración 1 9/03/2015	Final de la iteración 2 23/03/2015	Final de la iteración 3 13/04/2015	Final de la iteración 4 4/05/2015	Final de la iteración 5 1/06/2015
Suite de pruebas en Selenium	0.1	0.3	0.4	0.2	1.0
Suite de pruebas en JUnit			0.1	0.9	1.0

Se comenzaron a implementar las pruebas el día 23 de febrero del año 2015.

2.16 Conclusiones parciales

Al realizar los diferentes tipos de pruebas surgió la necesidad de explicar cómo se utilizaron las herramientas seleccionadas, por lo que se describen las principales características de los sistemas. Se realiza el modelo de dominio, la especificación de los requisitos funcionales y no funcionales, así como la descripción de los casos de prueba del sistema que conllevan a la implementación de las pruebas.

[2] **Releases**: en español entregas.

Conclusiones

Como resultado de desarrollar un sistema de pruebas automáticas para el sitio “Juez en Línea Caribeño” para contribuir con la calidad del mismo, se arribó a las siguientes conclusiones.

El estudio y desarrollo de la fundamentación teórica posibilitó dar inicio a la elaboración de la suite de pruebas automáticas una vez seleccionadas las herramientas, tecnologías y metodología de desarrollo de software a utilizar.

Se describieron las características de las herramientas y se identificaron 67 requisitos funcionales y 7 requisitos no funcionales.

La investigación realizada, proporcionó una entrada para la implementación de las pruebas tanto de Selenium como de JUnit y la realización de las diferentes historias de usuarios generadas por la metodología XP (Extreme Programming).

Recomendaciones

Para dar continuidad a la presente investigación, se recomienda por parte del equipo de desarrollo:

- Dar mantenimiento a las pruebas automáticas ya que las mismas fueron concebidas sobre el COJ master.
- Extender la implementación de pruebas en Selenium y JUnit hacia todos los módulos del COJ.

Referencias Bibliográficas

1. «Alternativa de comunicación para el Juez Caribeño en Línea | Acosta Labrada | Serie Científica». [En línea]. Disponible en: http://publicaciones.uci.cu/index.php/SC/article/view/1202. [Accedido: 15-jun-2015].
2. «Las pruebas de software forman el único instrumento adecuado para determinar la calidad de software». [En línea]. Disponible en: http://pruebasdesoftware.com/laspruebasdesoftware.htm. [Accedido: 22-marz-2015].
3. Metrics and Models in Software Quality Engineering. S. Kan. Boston: Addison – Wesley 2003.
4. «Introducción de pruebas de software». [En línea]. Disponible en: http://es.slideshare.net/mstabare/introduccin-de-pruebas-de-software. [Accedido: 25-marz-2015].
5. «Pruebas Funcionales - Software Testing and QA». [En línea]. Disponible en: http://www.calidadysoftware.com/testing/pruebas_funcionales.php. [Accedido: 20-marz-2015].
6. «HtmlUnit – Welcome to HtmlUnit». [En línea]. Disponible en: http://htmlunit.sourceforge.net/. [Accedido: 30-marz-2015].
7. «Apache JMeter - Apache JMeter™». [En línea]. Disponible en: http://jmeter.apache.org/. [Accedido: 11-may-2015].
8. «An introduction to Sahi: Part 1 | ThoughtWorks». [En línea]. Disponible en: http://www.thoughtworks.com/es/insights/blog/introduction-sahi-part-1. [Accedido: 30-marz -2015].
9. «¿Qué es Selenium?». [En línea]. Disponible en: http://softwarequality.bligoo.com/content/view/434556/Que-es-Selenium.html. [Accedido: 30-marz -2015].
10. Seco, José Antonio González. El lenguaje de programación C#. 2002.
11. Stewart, Celeste. The advantage of PHP. Designer's Playground. [En línea] 3 de enero de 2006. [Citado el: 22 de febrero de 2010.] http://www.designersplayground.com/articles/118/1/The-Advantages-of-PHP/Page1.html.
12. García de Jalón, Javier, y otros, y otros. Aprenda Java como si estuviera en primero. s.l.: San Sebastián, 2000.

13. «TestNG Tutorial». [En línea]. Disponible en: http://www.mkyong.com/tutorials/testng-tutorials/. [Accedido: 07-marzo-2015].
14. The Eclipse Foundation. The Eclipse Foundation. [En línea]. [Accedido el: 25 de diciembre de 2013.]. Disponible en: http://www.eclipse.org
15. «Ingeniería de sw Junit». [En línea]. Disponible en: http://es.slideshare.net/pattyand89/junit-38895990. [Accedido: 07-marzo-2015].
16. Karenny Brito Acuña Selección de metodologías de desarrollo para aplicaciones web en la facultad de informática de la Universidad de Cien Fuegos. [En línea]. [Citado el: 25 de enero 2015] http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm
17. Jacobson, Ivar, Booch, Grady y Rumbaugh, James. El proceso unificado de desarrollo de software. Primera edición. Madrid: Pearson Educación S.A, 2000.
18. Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo. Orjuela Duarte, Ailin y Rojas C., Mauricio. Colombia: s.n., 2 de 6 de 2008, Avances en Sistemas e Informática, Vol. 5. Disponible en: http://redalyc.uaemex.mx/src/inicio/ArtPdfRed.jsp?iCve=133115027022. ISSN (Versión impresa): 1657-7663.
19. Metodologías de desarrollo de software. [En línea]. [Citado el: 28 de enero de 2015] http://www.psl.com.co/desarrollo-de-software/metodologias-desarrollo-de-software.html
20. Roland F. Jeffries Xprogramin. [En línea]. [Citado el: 3 de febrero de 2015] http://xprogramming.com/xpmag/whatisxp
21. Extreme-Programing. [En línea]. [Citado el: 7 de febrero de 2015] http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html
22. Beck, Kent. Extreme Programming Explained. First Edition. s.l: s.l: Notes, 1999.
23. LaWebDelProgramador. [En línea] [Citado el: 21 de enero de 2015.] http://www.lawebdelprogramador.com.
24. Informes. [En línea]. [Citado el: 6 de febrero de 2015] http://ceds.nauta.es/informes/case01.htm
25. «Sistemas de gestión empresarial | Plusformación». [En línea]. Disponible en: http://www.plusformacion.com/Recursos/r/Sistemas-gestion-empresarial. [Accedido: 30-marz -2015].

26. Sistema Gestor de Base de Datos - EcuRed». [En línea]. Disponible en: http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos. [Accedido: 10-feb-2015].
27. Otero, Abraham. MySQL vs PostgreSQL ¿cuándo emplear cada una de ellas? javaHispano. [En línea] 10 de 9 de 2007. [Citado el: 22 de febrero de 2010.] http://www.anchor.com.au/hosting/dedicated/mysql_vs_postgres.
28. Modelo Dominio. [En línea]. [Accedido el: 11 de marzo de 2015]. Disponible en: http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf
29. James Rumbaugh, Ivar Jacobson, Grady Booch. El lenguaje Unificado de Modelado. Manual de Referencia. s.l. Addison Wesley.
30. Sommerville, Ian. Ingeniería del software.Séptima edición. Madrid: Pearson Educación.p.111
31. «EJIE S.A., Eusko Jaurlaritzaren Informatika Elkartea». [En línea]. Disponible en: http://www.ejie.eus/y79-02/eu/?r01kQry=i%3Ar01mpd0142239b23c11b702e36d3caba6ccb39e11%3Bp%3AInter%2CInter_portal%3Bm%3AfullText.LIKE.junit%3B&fullText=junit. [Accedido: 5-may-2015].
32. «Pruebas Software con Junit 4 y Eclipse». [En línea]. Disponible en: http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=Junit4Eclipse. [Accedido: 5-may-2015].
33. Licencia Pública GNU. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://gugs.sindominio.net/gnu-gpl/gples.html
34. Riehle, Dirk (2000), Framework Design: A Role Modeling Approach, Swiss Federal Institute of Technology [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.riehle.org/computer-science/research/dissertation/diss-a4.pdf
35. Entorno de Desarrollo Integrado (IDE). | fergarciac. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/
36. PDF Definición y explicación. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.masadelante.com/faqs/pdf

37. PHP: ¿Qué es PHP? - Manual. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.php.net/manual/es/intro-whatis.php
38. Definición de html - Qué es, Significado y Concepto. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://definicion.de/html/

Bibliografía

- «Automatización y Gestión de las Pruebas Funcionales usando Herramientas Open Source». [En línea]. Disponible en: http://sedici.unlp.edu.ar/handle/10915/21787. [Accedido: 28-ene-2015].

- «Como importar y ejecutar en GXtest un caso de prueba generado con Selenium - GXtest Wiki». [En línea]. Disponible en: http://gxtest.abstracta.com.uy/wiki/index.php?title=Como_importar_y_ejecutar_en_GXtest_un_caso_de_prueba_generado_con_Selenium. [Accedido: 21-ene-2015].

- «data2type GmbH: Tecnologías XML | XPath | Introducción a XPath». [En línea]. Disponible en: http://www.data2type.de/xml-xslt-xslfo/xpath/?L=3. [Accedido: 10-ene-2015].

- «Eclipse Luna». [En línea]. Disponible en: http://www.eclipse.org/. [Accedido: 10-ene-2015].

- «EJIE S.A., Eusko Jaurlaritzaren Informatika Elkartea». [En línea]. Disponible en: http://www.ejie.eus/y79-02/eu/?r01kQry=i%3Ar01mpd0142239b23c11b702e36d3caba6ccb39e11%3Bp%3AInter%2CInter_portal%3Bm%3AfullText.LIKE.junit%3B&fullText=junit. [Accedido: 19-may-2015].

- «EJIE S.A., Eusko Jaurlaritzaren Informatika Elkartea». [En línea]. Disponible en: http://www.ejie.eus/y79-02/eu/?r01kQry=i%3Ar01mpd0142239b23c11b702e36d3caba6ccb39e11%3Bp%3AInter%2CInter_portal%3Bm%3AfullText.LIKE.junit%3B&fullText=junit. [Accedido: 19-may-2015].

- «Enfoque para pruebas de unidad basado en la generación aleatoria de objetos». [En línea]. Disponible en: http://sedici.unlp.edu.ar/handle/10915/34969. [Accedido: 19-may-2015].

- «Firebug». [En línea]. Disponible en: http://getfirebug.com/. [Accedido: 10-ene-2015].

- «fitnesse/licenseHeader.txt at master · unclebob/fitnesse · GitHub». [En línea]. Disponible en: https://github.com/unclebob/fitnesse/blob/master/licenseHeader.txt. [Accedido: 10-ene-2015].

- Center for History and New Media, «Guía rápida». [En línea]. Disponible en: http://zotero.org/support/quick_start_guide.

- «IMS Learning Design Information Model». [En línea]. Disponible en: http://www.imsglobal.org/learningdesign/ldv1p0/imsld_infov1p0.html. [Accedido: 21-ene-2015].

- «JMeter. Manual de usuario v1.2 - JMeter. Manual de usuario v1.2.pdf». .

- «Pruebas Software con Junit 4 y Eclipse». [En línea]. Disponible en: http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=Junit4Eclipse. [Accedido: 19-may-2015].
- «Selenium - Web Browser Automation». [En línea]. Disponible en: http://docs.seleniumhq.org/. [Accedido: 10-ene-2015].
- «subclipse.tigris.org». [En línea]. Disponible en: http://subclipse.tigris.org/. [Accedido: 10-ene-2015].
- Codificación | Recursos web. [En línea]. [Accedido el: 1 de mayo de 2015]. Disponible en: http://recursosweb.unam.mx/recursos-web/creacion-de-paginas-web/estandares-de-codificacion/
- ConvecionesCodigoJava - ConvencionesCodigoJava.pdf [En línea]. [Accedido el: 1 de mayo de 2015]. Disponible en: http://www.um.es/docencia/vjimenez/ficheros/practicas/ConvencionesCodigoJava.pdf
- Cristóbal González Almirón «Introducción a JSF Java». 2009. [En línea]. [Accedido el: 19 de diciembre de 2014]. Disponible en: http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=IntroduccionJSFJava
- Cuba. Ministerio de la Informática y las Comunicaciones. Informatización de la Sociedad. [En línea] [Accedido el: 4 de diciembre de 2014]. Disponible en: http://www.mic.gov.cu/sitiomic/hinfosoc.asp
- Definición. De, Definición de reporte - Qué es, Significado y Concepto. [En línea]. [Accedido el: 13 de diciembre de 2014]. Disponible en: http://definicion.de/reporte/
- Definición de html - Qué es, Significado y Concepto. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://definicion.de/html/
- Definicion_Estandares_2.doc [En línea]. [Accedido el: 1 de mayo de 2015]. Disponible en: http://www.inf.utfsm.cl/~visconti/xp/Definicion_Estandares_2.doc
- Definición de modelo de datos - Qué es, Significado y Concepto. [En línea]. [Accedido el: 23 De abril de 2013]. Disponible en: http://definicion.de/modelo-de-datos/
- Definición de MySQL - Significado y definición de MySQL. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.mastermagazine.info/termino/6051.php
- Desarrollo JAVA EE. [En línea] [Accedido el: 26 de enero de 2015.] Disponible en: http://www.neuronet.cl/downloads/J2EE.pdf

- Diagrama de Componentes. [En línea]. [Accedido el: 28 De abril de 2015]. Disponible en: http://diagramacomponente.blogspot.com/
- Diagrama de Interacción. [En línea]. [Accedido el: 15 de marzo de 2015]. Disponible en: http://uxmcc1.iimas.unam.mx/~cursos/Objetos/Cap18/cap18.html
- Dugarte, Ana, y otros. 2009. Lenguaje Unificado de Modelado. [En línea] 2009. [Accedido el: 19 de diciembre de 2014]. Disponible en: http://www.oocities.org/es/annadugarte/ads1/PRINCIPAL.htm
- Estándares de codificación y buenas prácticas | inside software quality - Un blog de Optimyth en español. [En línea]. [Accedido el: 1 de mayo de 2015]. Disponible en: http://blog.optimyth.com/es/2013/11/estandares-de-codificacion-y-buenas-practicas
- EDUCATION. Modelo de dominio. p 10.
- «Alternativa de comunicación para el Juez Caribeño en Línea | Acosta Labrada | Serie Científica». [En línea]. Disponible en: http://publicaciones.uci.cu/index.php/SC/article/view/1202. [Accedido: 15-jun-2015].
- «Las pruebas de software forman el único instrumento adecuado para determinar la calidad de software». [En línea]. Disponible en: http://pruebasdesoftware.com/laspruebasdesoftware.htm. [Accedido: 22-marz-2015].
- Metrics and Models in Software Quality Engineering. S. Kan. Boston: Addison – Wesley 2003.
- «Introducción de pruebas de software». [En línea]. Disponible en: http://es.slideshare.net/mstabare/introduccin-de-pruebas-de-software. [Accedido: 25-marz-2015].
- «Pruebas Funcionales - Software Testing and QA». [En línea]. Disponible en: http://www.calidadysoftware.com/testing/pruebas_funcionales.php. [Accedido: 20-marz-2015].
- «HtmlUnit – Welcome to HtmlUnit». [En línea]. Disponible en: http://htmlunit.sourceforge.net/. [Accedido: 30-marz-2015].
- «Apache JMeter - Apache JMeter™». [En línea]. Disponible en: http://jmeter.apache.org/. [Accedido: 11-may-2015].
- «An introduction to Sahi: Part 1 | ThoughtWorks». [En línea]. Disponible en: http://www.thoughtworks.com/es/insights/blog/introduction-sahi-part-1. [Accedido: 30-marz -2015].

- «¿Qué es Selenium?». [En línea]. Disponible en: http://softwarequality.bligoo.com/content/view/434556/Que-es-Selenium.html. [Accedido: 30-marz -2015].
- Seco, José Antonio González. El lenguaje de programación C#. 2002.
- Stewart, Celeste. The advantage of PHP. Designer's Playground. [En línea] 3 de enero de 2006. [Citado el: 22 de febrero de 2010.] http://www.designersplayground.com/articles/118/1/The-Advantages-of-PHP/Page1.html.
- García de Jalón, Javier, y otros, y otros. Aprenda Java como si estuviera en primero. s.l.: San Sebastián, 2000.
- «TestNG Tutorial». [En línea]. Disponible en: http://www.mkyong.com/tutorials/testng-tutorials/. [Accedido: 07-marzo-2015].
- The Eclipse Foundation. The Eclipse Foundation. [En línea]. [Accedido el: 25 de diciembre de 2013.]. Disponible en: http://www.eclipse.org
- «Ingeniería de sw Junit». [En línea]. Disponible en: http://es.slideshare.net/pattyand89/junit-38895990. [Accedido: 07-marzo-2015].
- Karenny Brito Acuña Selección de metodologías de desarrollo para aplicaciones web en la facultad de informática de la Universidad de Cien Fuegos. [En línea]. [Citado el: 25 de enero 2015] http://www.eumed.net/libros-gratis/2009c/584/Metodologias%20tradicionales%20y%20metodologias%20agiles.htm
- Jacobson, Ivar, Booch, Grady y Rumbaugh, James. El proceso unificado de desarrollo de software. Primera edición. Madrid: Pearson Educación S.A, 2000.
- Las Metodologías de Desarrollo Ágil como una Oportunidad para la Ingeniería del Software Educativo. Orjuela Duarte, Ailin y Rojas C., Mauricio. Colombia: s.n., 2 de 6 de 2008, Avances en Sistemas e Informática, Vol. 5. Disponible en: http://redalyc.uaemex.mx/src/inicio/ArtPdfRed.jsp?iCve=133115027022. ISSN (Versión impresa): 1657-7663.
- Metodologías de desarrollo de software. [En línea]. [Citado el: 28 de enero de 2015] http://www.psl.com.co/desarrollo-de-software/metodologias-desarrollo-de-software.html
- Roland F. Jeffries Xprogramin. [En línea]. [Citado el: 3 de febrero de 2015] http://xprogramming.com/xpmag/whatisxp

- Extreme-Programing. [En línea]. [Citado el: 7 de febrero de 2015] http://ingenieriadesoftware.mex.tl/52753_XP---Extreme-Programing.html
- Beck, Kent. Extreme Programming Explained. First Edition. s.l: s.l: Notes, 1999.
- LaWebDelProgramador. [En línea] [Citado el: 21 de enero de 2015.] http://www.lawebdelprogramador.com.
- Informes. [En línea]. [Citado el: 6 de febrero de 2015] http://ceds.nauta.es/informes/case01.htm
- «Sistemas de gestión empresarial | Plusformación». [En línea]. Disponible en: http://www.plusformacion.com/Recursos/r/Sistemas-gestion-empresarial. [Accedido: 30-marz -2015].
- Sistema Gestor de Base de Datos - EcuRed». [En línea]. Disponible en: http://www.ecured.cu/index.php/Sistema_Gestor_de_Base_de_Datos. [Accedido: 10-feb-2015].
- Otero, Abraham. MySQL vs PostgreSQL ¿cuándo emplear cada una de ellas? javaHispano. [En línea] 10 de 9 de 2007. [Citado el: 22 de febrero de 2010.] http://www.anchor.com.au/hosting/dedicated/mysql_vs_postgres.
- Modelo Dominio. [En línea]. [Accedido el: 11 de marzo de 2015]. Disponible en: http://is.ls.fi.upm.es/docencia/is2/documentacion/ModeloDominio.pdf
- James Rumbaugh, Ivar Jacobson, Grady Booch. El lenguaje Unificado de Modelado. Manual de Referencia. s.l. Addison Wesley.
- Sommerville, Ian. Ingeniería del software. Séptima edición. Madrid: Pearson Educación.p.111
- «EJIE S.A., Eusko Jaurlaritzaren Informatika Elkartea». [En línea]. Disponible en: http://www.ejie.eus/y79-02/eu/?r01kQry=i%3Ar01mpd0142239b23c11b702e36d3caba6ccb39e11%3Bp%3AInter%2CInter_portal%3Bm%3AfullText.LIKE.junit%3B&fullText=junit. [Accedido: 5-may-2015].
- «Pruebas Software con Junit 4 y Eclipse». [En línea]. Disponible en: http://www.adictosaltrabajo.com/tutoriales/tutoriales.php?pagina=Junit4Eclipse. [Accedido: 5-may-2015].
- Licencia Pública GNU. [En línea]. [Accedido el: 10 de mayo de 201]. Disponible en: http://gugs.sindominio.net/gnu-gpl/gples.html

- Riehle, Dirk (2000), Framework Design: A Role Modeling Approach, Swiss Federal Institute of Technology [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.riehle.org/computer-science/research/dissertation/diss-a4.pdf
- Entorno de Desarrollo Integrado (IDE). | fergarciac. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://fergarciac.wordpress.com/2013/01/25/entorno-de-desarrollo-integrado-ide/
- PDF Definición y explicación. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.masadelante.com/faqs/pdf
- PHP: ¿Qué es PHP? - Manual. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://www.php.net/manual/es/intro-whatis.php
- Definición de html - Qué es, Significado y Concepto. [En línea]. [Accedido el: 10 de mayo de 2015]. Disponible en: http://definicion.de/html/

Glosario de términos

GNU GPL: Es la conocida GNU Public License (GPL), versión 2 (de junio de 1.991), que cubre la mayor parte del software de la Free Software Foundation. Su propósito es declarar que el software cubierto por esta licencia es software libre y protegerlo de intentos de apropiación que restrinjan esas libertades a los usuarios. Esta licencia fue creada originalmente por Richard Stallman fundador de la Free Software Foundation (FSF) para el proyecto GNU. [33]

Framework: La palabra inglesa "framework" (marco de trabajo) define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar. [34]

IDE: Un entorno de desarrollo integrado, es un entorno de programación que ha sido empaquetado como un programa de aplicación, es decir, consiste en un editor de código, un compilador, un depurador y un constructor de interfaz gráfica (GUI). Los IDE proveen un marco de trabajo amigable para la mayoría de los lenguajes de programación tales como C++, PHP, Python, Java, C#, Delphi, Visual Basic. [35]

PDF: PDF, de Portable Document Format (formato de documento portable) es el formato de archivos desarrollado por Adobe Systems y creado con los programas Adobe Acrobat Reader, Acrobat Capture, Adobe Distiller, Adobe Exchange, y el plugin Amber de Adobe Acrobat. [36]

PHP: PHP (acrónimo recursivo de PHP: Hypertext Preprocessor) es un lenguaje de código abierto muy popular especialmente adecuado para el desarrollo web y que puede ser incrustado en HTML. [37]

HTML: HTML es un lenguaje de programación que se utiliza para el desarrollo de páginas de Internet. Se trata de la sigla que corresponde a HyperText Markup Language, es decir, Lenguaje de Marcas de Hipertexto, que podría ser traducido como Lenguaje de Formato de Documentos para Hipertexto. [38]

Anexos

Anexo 1

Guía de observación: Aspectos a tener en cuenta

Objetivo: Seleccionar aquellos sistemas que permitan desarrollar pruebas automáticas a sitios web.

Tipos de pruebas:

- Tipos de pruebas que se pueden realizar a través de ellas()

Seguridad

- Mecanismos de seguridad establecidos
- Roles definidos para el acceso al sistema

Plataforma

- Multiplataforma

Elementos generales

- Principales funcionalidades
- Tecnologías de desarrollo (libres o propietarias)
- Tipos de aplicación (web o de escritorio)
- País donde se desarrollaron

Lenguajes de programación

- Tipos de lenguajes de programación

Anexo 2

Guía con aspectos a considerar en la entrevista

Objetivo: obtener criterios para comprender los elementos necesarios para el desarrollo de la solución informática.

- Principales procesos que se realizan actualmente en el COJ.
- Estructura del centro COJ.
- Tipo de aplicación a desarrollar (sistema independiente o funcionalidad integrada al sitio COJ).
- Roles que van a interactuar con la solución informática a desarrollar.
- Funcionalidades que el sistema debe permitir.
- Requisitos no funcionales a tener en cuenta para la creación de las funcionalidades definidas por el cliente.
- Tipo de pruebas a realizar al sistema del centro de COJ

Anexo 3: Historias de Usuarios

Tabla 1. HU Autenticar usuario

No: 1	**Nombre:** Autenticar usuario
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de autenticar automáticamente a un usuario previamente introducido
Observaciones:	Da cumplimiento al requisito 1

Tabla 2. HU Fallar autenticación de usuario

No: 2	**Nombre:** Fallar autenticación de usuario
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe reconocer que no se autenticó el usuario correctamente.
Observaciones:	Da cumplimiento al requisito 2

Tabla 3. HU Autenticar usuario por correo

No: 3	**Nombre:** Autenticar usuario por correo
Usuario	Administrador

Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de autenticar automáticamente a un usuario por su correo previamente introducido
Observaciones:	Da cumplimiento al requisito 3

1.1 Usuario sin autenticarse

Tabla 4. HU Cambiar el lenguaje del sitio a inglés

No: 4	**Nombre:** Cambiar el lenguaje del sitio a inglés
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del sitio a inglés.
Observaciones:	Da cumplimiento al requisito 4

Tabla 5. HU Cambiar el lenguaje del sitio a español

No: 5	**Nombre:** Cambiar el lenguaje del sitio a español
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1

Descripción:	La prueba debe ser capaz de cambiar el leguaje del sitio a español.
Observaciones:	Da cumplimiento al requisito 5

Tabla 6. HU Mostrar el problema

No: 6	**Nombre:** Mostrar el problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar el problema A+B
Observaciones:	Da cumplimiento al requisito 6

Tabla 7. HU Ver envíos del problema

No: 7	**Nombre:** Ver envíos del problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar los envíos realizados del problema
Observaciones:	Da cumplimiento al requisito 7

Tabla 8. HU Ver mejores soluciones del problema

No: 8	**Nombre:** Ver mejores soluciones

Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar las mejores soluciones realizadas del problema
Observaciones:	Da cumplimiento al requisito 8

Tabla 9. HU Cambiar el lenguaje del problema a inglés

No: 9	**Nombre:** Cambiar el lenguaje del problema a inglés
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a inglés
Observaciones:	Da cumplimiento al requisito 9

Tabla 10. HU Cambiar el lenguaje del problema a español

No: 10	**Nombre:** Cambiar el lenguaje del problema a español
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1

Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a español
Observaciones:	Da cumplimiento al requisito 10

Tabla 11. HU Cambiar el lenguaje del problema a portugués

No: 11	**Nombre:** Cambiar el lenguaje del problema a portugués
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a portugués
Observaciones:	Da cumplimiento al requisito 11

Tabla 12. HU Probar filtrado del volumen de problemas por nombre

No: 12	**Nombre:** Probar filtrado del volumen de problemas por nombre
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de filtrar los problemas por nombre correctamente
Observaciones:	Da cumplimiento al requisito 12

Tabla 13. HU Probar filtrado del volumen de problemas por id

No: 13	**Nombre:** Probar filtrado del volumen de problemas por id
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de filtrar los problemas por id correctamente
Observaciones:	Da cumplimiento al requisito 13

Tabla 14. HU Probar ordenamiento ascendente del volumen de problemas por envíos

No: 14	**Nombre:** Probar ordenamiento ascendente del volumen de problemas por envíos
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:2
Descripción:	La prueba debe ser capaz de ordenar los problemas ascendentemente por envíos correctamente
Observaciones:	Da cumplimiento al requisito 14

Tabla 15. HU Probar ordenamiento descendente del volumen de problemas por envíos

No: 15	**Nombre:** Probar ordenamiento descendente del volumen de problemas por envíos
Usuario	Administrador

Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:2
Descripción:	La prueba debe ser capaz de ordenar los problemas descendentemente por envíos correctamente
Observaciones:	Da cumplimiento al requisito 15

Tabla 16. HU Probar ordenamiento ascendente del volumen de problemas por aceptados

No: 16	**Nombre:** Probar ordenamiento ascendente del volumen de problemas por aceptados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:2
Descripción:	La prueba debe ser capaz de ordenar los problemas ascendentemente por aceptados correctamente
Observaciones:	Da cumplimiento al requisito 16

Tabla 17. HU Probar ordenamiento descendente del volumen de problemas por aceptados

No: 17	**Nombre:** Probar ordenamiento descendente del volumen de problemas por aceptados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:2
Descripción:	La prueba debe ser capaz de ordenar los problemas descendentemente por aceptados correctamente
Observaciones:	Da cumplimiento al requisito 17

Tabla 18. HU Probar marca de juzgado especial

No: 18	**Nombre:** Probar marca de juzgado especial
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que aparezca la prueba de juzgado especial en el problema con id 2988
Observaciones:	Da cumplimiento al requisito 18

Tabla 19. HU Probar totales de envíos en la página Mejores Soluciones

No: 19	**Nombre:** Probar totales de envíos en la página Mejores Soluciones
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que el total de envíos de la página mejores soluciones sea correcto

Observaciones:	Da cumplimiento al requisito 19

Tabla 20. HU Probar filtrado de la página de estado por nombre

No: 20	**Nombre:** Probar filtrado de la página de estado por nombre
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por nombre
Observaciones:	Da cumplimiento al requisito 20

Tabla 21. HU Probar filtrado de la página de estado por problema

No: 21	**Nombre:** Probar filtrado de la página de estado por problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por problema
Observaciones:	Da cumplimiento al requisito 21

Tabla 22. HU Probar filtrado de la página de estado por resultado

No: 22	**Nombre:** Probar filtrado de la página de estado por resultado
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 22

Tabla 23. HU Probar filtrado de la página de estado por lenguaje

No: 23	**Nombre:** Probar filtrado de la página de estado por lenguaje
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 23

Tabla 24. HU Probar filtrado de la página de estado por criterios combinados

No: 24	**Nombre:** Probar filtrado de la página de estado por criterios combinados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:3
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 24

Tabla 25. HU Probar ver descripción de concursos pasados

No: 25	**Nombre:** Probar ver descripción de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ver la descripción de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 25

Tabla 26. HU Probar ver problemas de concursos pasados

No: 26	**Nombre:** Probar ver problemas de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ver los problemas de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 26

Tabla 27. HU Probar ranking agrupado de concursos pasados

No: 27	**Nombre:** Probar ranking agrupado de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ver el ranking agrupado de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 27

Tabla 28. HU Probar ranking desagrupado de concursos pasados

No: 28	**Nombre:** Probar ranking desagrupado de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ver el ranking desagrupado de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 28

Tabla 29. HU Probar ver descripción de concursos próximos

No: 29	**Nombre:** Probar ver descripción de concursos próximos

Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ver la descripción de los concursos próximos correctamente
Observaciones:	Da cumplimiento al requisito 29

1.2 Usuario autenticado

Tabla 30. HU Cambiar el lenguaje del sitio a inglés

No: 30	**Nombre:** Cambiar el lenguaje del sitio a inglés
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del sitio a inglés.
Observaciones:	Da cumplimiento al requisito 30

Tabla 31. HU Cambiar el lenguaje del sitio a español

No: 31	**Nombre:** Cambiar el lenguaje del sitio a español
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del sitio a español.
Observaciones:	Da cumplimiento al requisito 31

Tabla 32. HU Mostrar el problema

No: 32	**Nombre:** Mostrar el problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar el problema A+B
Observaciones:	Da cumplimiento al requisito 32

Tabla 33. HU Ver envíos del problema

No: 33	**Nombre:** Ver envíos del problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar los envíos realizados del problema
Observaciones:	Da cumplimiento al requisito 33

Tabla 34. HU Ver mejores soluciones del problema

No: 34	**Nombre:** Ver mejores soluciones del problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de mostrar las mejores soluciones realizadas del problema
Observaciones:	Da cumplimiento al requisito 34

Tabla 35. HU Enviar el problema 1000 correctamente

No: 35	**Nombre:** Enviar el problema 1000 correctamente
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de verificar que se envía correctamente el problema 1000
Observaciones:	Da cumplimiento al requisito 35

Tabla 36. HU Enviar el problema 1000 solución AC

No: 36	**Nombre:** Enviar el problema 1000 solución AC
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de verificar que se al enviar el problema 1000 resulta una solución AC
Observaciones:	Da cumplimiento al requisito 36

Tabla 37. HU Enviar el problema 1000 solución WA

No: 37	**Nombre:** Enviar el problema 1000 solución WA
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	Esta prueba debe ser capaz de verificar que se al enviar el problema 1000 resulta una solución WA
Observaciones:	Da cumplimiento al requisito 37

Tabla 38. HU Cambiar el lenguaje del problema a inglés

No: 38	**Nombre:** Cambiar el lenguaje del problema a inglés
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a inglés
Observaciones:	Da cumplimiento al requisito38

Tabla 39. HU Cambiar el lenguaje del problema a español

No: 39	**Nombre:** Cambiar el lenguaje del problema a español
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a español
Observaciones:	Da cumplimiento al requisito 39

Tabla 40. HU Cambiar el lenguaje del problema a portugués

No: 40	**Nombre:** Cambiar el lenguaje del problema a portugués
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de cambiar el leguaje del problema a portugués
Observaciones:	Da cumplimiento al requisito 40

Tabla 41. HU Probar filtrado del volumen de problemas por nombre

No: 41	**Nombre:** Probar filtrado del volumen de problemas por nombre
Usuario	Administrador

Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de filtrar los problemas por nombre correctamente
Observaciones:	Da cumplimiento al requisito 41

Tabla 42. HU Probar filtrado del volumen de problemas por id

No: 42	**Nombre:** Probar filtrado del volumen de problemas por id
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de filtrar los problemas por id correctamente
Observaciones:	Da cumplimiento al requisito 42

Tabla 43. HU Probar filtrado del volumen de problemas por clasificación

No: 43	**Nombre:** Probar filtrado del volumen de problemas por clasificación
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de filtrar los problemas por clasificación correctamente

Observaciones:	Da cumplimiento al requisito 43

Tabla 44. HU Probar filtrado del volumen de problemas por complejidad

No: 44	**Nombre:** Probar filtrado del volumen de problemas por complejidad
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de filtrar los problemas por complejidad correctamente
Observaciones:	Da cumplimiento al requisito 44

Tabla 45. HU Probar ordenamiento ascendente del volumen de problemas por envíos

No: 45	**Nombre:** Probar ordenamiento ascendente del volumen de problemas por envíos
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ordenar los problemas ascendentemente por envíos correctamente
Observaciones:	Da cumplimiento al requisito 45

Tabla 46. HU Probar ordenamiento descendente del volumen de problemas por envíos

No: 46	**Nombre:** Probar ordenamiento descendente del volumen de problemas por envíos
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ordenar los problemas descendentemente por envíos correctamente
Observaciones:	Da cumplimiento al requisito 46

Tabla 47. HU Probar ordenamiento ascendente del volumen de problemas por aceptados

No: 47	**Nombre:** Probar ordenamiento ascendente del volumen de problemas por aceptados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ordenar los problemas ascendentemente por aceptados correctamente
Observaciones:	Da cumplimiento al requisito 47

Tabla 48. HU Probar ordenamiento descendente del volumen de problemas por aceptados

No: 48	**Nombre:** Probar ordenamiento descendente del volumen de problemas por aceptados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de ordenar los problemas descendentemente por aceptados correctamente
Observaciones:	Da cumplimiento al requisito48

Tabla 49. HU Probar marca de juzgado especial

No: 49	**Nombre:** Probar marca de juzgado especial
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que aparezca la prueba de juzgado especial en el problema con id 2988
Observaciones:	Da cumplimiento al requisito 49

Tabla 50. HU Probar totales de envíos en la página Mejores Soluciones

No: 50	**Nombre:** Probar totales de envíos en la página Mejores Soluciones
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que el total de envíos de la página mejores soluciones sea correcto
Observaciones:	Da cumplimiento al requisito 50

Tabla 51. HU Probar filtrado de la página de estado por nombre

No: 51	**Nombre:** Probar filtrado de la página de estado por nombre
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por nombre
Observaciones:	Da cumplimiento al requisito 51

Tabla 52. HU Probar filtrado de la página de estado por problema

No: 52	**Nombre:** Probar filtrado de la página de estado por problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por problema

Observaciones:	Da cumplimiento al requisito 52

Tabla 53. HU Probar filtrado de la página de estado por resultado

No: 53	**Nombre:** Probar filtrado de la página de estado por resultado
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 53

Tabla 54. HU Probar filtrado de la página de estado por lenguaje

No: 54	**Nombre:** Probar filtrado de la página de estado por lenguaje
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:1
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 54

Tabla 55. HU Probar filtrado de la página de estado por criterios combinados

No: 55	**Nombre:** Probar filtrado de la página de estado por criterios combinados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de verificar que se filtre correctamente la página de estado por resultado
Observaciones:	Da cumplimiento al requisito 55

Tabla 56. HU Probar ver descripción de concursos pasados

No: 56	**Nombre:** Probar ver descripción de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver la descripción de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 56

Tabla 57. HU Probar ver problemas de concursos pasados

No: 57	**Nombre:** Probar ver problemas de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media

Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver los problemas de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 57

Tabla 58. HU Probar ranking agrupado de concursos pasados

No: 58	**Nombre:** Probar ranking agrupado de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver el ranking agrupado de los concursos pasados correctamente
Observaciones:	Da cumplimiento al requisito 58

Tabla 59. HU Probar ranking desagrupado de concursos pasados

No: 59	**Nombre:** Probar ranking desagrupado de concursos pasados
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver el ranking desagrupado de los concursos pasados correctamente

Observaciones:	Da cumplimiento al requisito 59

Tabla 60. HU Probar ver descripción de concursos próximos

No: 60	**Nombre:** Probar ver descripción de concursos próximos
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:1d	Iteración:4
Descripción:	La prueba debe ser capaz de ver la descripción de los concursos próximos correctamente
Observaciones:	Da cumplimiento al requisito 60

1.3 Historias de usuario de JUnit

Tabla 61. HU Adicionar problema

No: 61	**Nombre:** Adicionar Problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que se adicione correctamente un problema.
Observaciones:	Da cumplimiento al requisito 61

Tabla 62. HU Editar problema

No: 62	**Nombre:** Editar problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que se editen correctamente los problemas.
Observaciones:	Da cumplimiento al requisito 62

Tabla 63. HU Deshabilitar problema para 24h

No: 63	**Nombre:** Deshabilitar problema para 24h
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que se deshabilite correctamente un problema.
Observaciones:	Da cumplimiento al requisito 63

Tabla 64. HU Traducir problema

No: 64	**Nombre:** Traducir problema
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5

Descripción:	El administrador cuando ejecuta esta prueba, verifica que se traduzca correctamente un problema.
Observaciones:	Da cumplimiento al requisito 64

Tabla 65. HU Adicionar un concurso

No: 65	**Nombre:** Adicionar un concurso
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que se adicione correctamente el concurso.
Observaciones:	Da cumplimiento al requisito 65

Tabla 66. HU Adicionar usuario a concurso

No: 66	**Nombre:** Adicionar usuario a concurso
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que se adicione correctamente un usuario a un concurso.
Observaciones:	Da cumplimiento al requisito 66

Tabla 67. HU Autenticarse

No: 67	**Nombre:** Autenticarse
Usuario	Administrador
Prioridad: Alta	Complejidad: Media
Estimación:5d	Iteración:5
Descripción:	El administrador cuando ejecuta esta prueba, verifica que los usuarios puedan autenticarse correctamente.
Observaciones:	Da cumplimiento al requisito 67