Universidad de las Ciencias Informáticas

Facultad 1

Desarrollo de la versión 2 del Juez Caribeño en Línea

Trabajo de Diploma para optar por el título de

Ingeniero en Ciencias Informáticas

Autor(es): Juan Carlos Lobaina Guzmán

Jorge Luis Roque Alvarez

Tutor(es): Ing. Gladys Marsi Peñalver Romero

Ing. Dovier Antonio Ripoll Méndez

Co-tutor: Ing. Luis Enrique Sánchez Arce

UCI, La Habana, Junio de 2012

“Año 54 de la Revolución”

DECLARACIÓN DE AUTORÍA

Declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas, para que lo usen según estimen pertinente.

Para que así conste, firmamos la presente a los ___ días del mes de _____ del año _____.

Jorge Luis Roque Alvarez

_____

Firma del Autor

Juan Carlos Lobaina Guzmán

_____

Firma del Autor

Ing. Gladys Marsi Peñalver Romero

_____

Firma del Tutor

Ing. Dovier Antonio Ripoll Méndez

_____

Firma del Tutor

AGRADECIMIENTOS

Agradezco:

A mis padres Mayra y Wilson, por todo el esfuerzo y apoyo que me han brindado para que pueda convertirme hoy en un Ingeniero…

A mi familia y en especial a mis abuelas Dilia y Edipta, y a mis tías Mirta y Ada, a Pacheco y a mi tía Maribel, a mi tocayo, tío y pariente Trucopey, a mi tío Pedro…

A mis primos Israel, Irán, Yasleni, Yenely y a mi primita Angélica…

A Odelta y Orisel, por siempre apoyarme en todo lo que he necesitado…

A la gente de mi barrio por siempre confiar en mí y por todo su apoyo, en especial a "Los Serra": Geiner, Ricardo, Barbarita, Ricardito, Yeny, Amaury, Taiza, Jessika y Amaurito, y a Juana Pérez…

A Yanet, a su mamá Martha y a toda su familia por el apoyo y hospitalidad que siempre me mostraron…

A mis compañeros de aula, los viejos del 10106 y los nuevos del 1509…

A todos mis compañeros del proyecto, por estar siempre ahí tirando código codo con codo conmigo, en especial a Luis Enrique, a Yuri y a Daileny "las pillas"…

A mi compañero de tesis Jorge Luis, por todo lo que hemos logrado; además de ser el creador de la idea de construir un juez en línea desarrollado completamente por nosotros…

A mis compañeros de trabajo de la EPICA, en especial Luis, Greysi, Miriam, Iván, Pedrito…

A todas las personas que, de una forma u otra, me han apoyado siempre e hicieron posible esta investigación…

Juan Carlos

Agradezco:

A mis abuelos mama, paya y payo por convertirme en la persona que soy…

A mis padres por confiar siempre en mí y dejarme tomar mis propias decisiones…

A mi tía Luisa por aguantarme y quererme a pesar de mis pesadeces…

A tete y Raúl por hacerme sentir como un hijo más…

A Omar por estar siempre ahí de forma incondicional…

A mi novia Laura por haber compartido conmigo cuatro años maravillosos (y los que están por venir)…

A Juanky por acompañarme en la aventura de la programación y la informática…

A todos los muchachos del grupo 10106 por las risas y los buenos momentos que he pasado con ellos en la universidad…

A todos los que de una manera u otra han hecho posible este trabajo…

Jorge Luis

Agradecemos a:

Dovier cuya quisquilla logró que la versión 2 del COJ despertara admiración en personalidades de la talla de Miguel Revilla (expresando "les va bien en el Caribe con el COJ")…

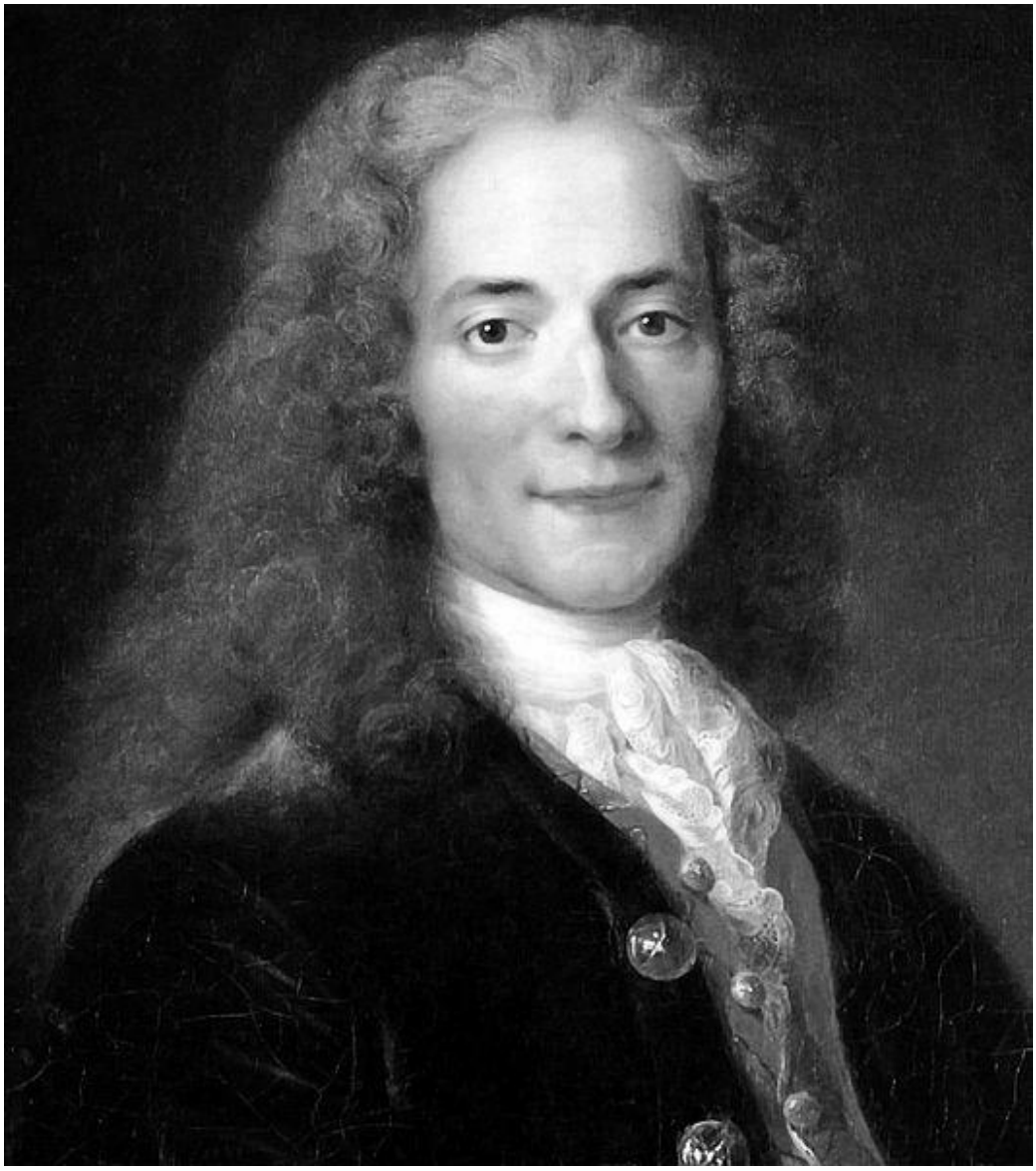
DEDICATORIA

Dedico esta tesis a mis padres Mayra y Wilson, por el apoyo y ejemplo que siempre me han brindado, y a mi familia por confiar siempre en mí.

Juan Carlos

A toda mi familia por apoyarme y guiarme en la búsqueda del conocimiento, especialmente a mis padres a los cuales debo todo lo que soy.

Jorge Luis



“Suerte es lo que sucede cuando la preparación y la oportunidad se encuentran y fusionan”

François Marie Arouet (Voltaire)

(Escritor, historiador, filósofo y abogado francés, 1694 - 1778)

RESUMEN

El presente trabajo se centra en la necesidad de desarrollar una nueva versión del Juez Caribeño en Línea (COJ, por sus siglas en inglés) con el objetivo de erradicar las principales deficiencias identificadas en la versión 1.

Para cumplimentar la presente investigación se realizó un estudio acerca de los principales jueces en línea en los diferentes ámbitos, con el objetivo de identificar tendencias en el desarrollo de este tipo de sistemas; así como las principales tecnologías utilizadas para su desarrollo y posterior despliegue. Las tecnologías seleccionadas fueron: Java como lenguaje de programación, Spring como *framework* de apoyo al desarrollo, PostgreSQL como gestor de datos, Apache Tomcat como servidor para el despliegue de la aplicación web y Ubuntu como sistema operativo para el desarrollo y despliegue del sistema. Una vez culminado el proceso de investigación, se identificaron y documentaron todas las funcionalidades que la nueva versión del COJ debía cumplir. El proceso de desarrollo estuvo guiado por la metodología ágil SXP, la cual está especialmente enfocada a equipos pequeños de trabajo con requisitos cambiantes. Una vez descritas y documentadas las funcionalidades deseadas para el sistema, se procedió a su diseño e implementación para posteriormente efectuar las pruebas que lo validaron para su uso.

Se obtuvo un sistema que erradicó las principales deficiencias identificadas en la versión 1, además de incluir un conjunto de características novedosas que posibilitarán una mejor preparación de concursantes con vistas a futuras competiciones del ACM-ICPC.

Palabras claves: ACM-ICPC, *framework*, juez en línea, metodología ágil.

ÍNDICE GENERAL

ÍNDICE DE FIGURAS

ÍNDICE DE FIGURAS DE INTERFAZ USUARIO

ÍNDICE DE TABLAS

INTRODUCCIÓN

El Concurso Internacional Universitario ACM[1] de Programación (ACM-ICPC por sus siglas en inglés), efectuado anualmente con la participación de varios equipos en representación de diferentes universidades del mundo, constituye en la actualidad uno de los más importantes y prestigiosos eventos de programación de computadoras donde se fomenta el trabajo en equipo, la resolución de tareas y el desarrollo rápido de software. Dicho concurso consta de varias etapas de competición que van desde los concursos locales, efectuados en cada una de las universidades que compiten en el evento, hasta el evento a nivel mundial realizado en distintas sedes por todo el mundo cada año, pasando por los concursos nacionales y regionales.

En sus inicios este concurso no contaba con muchos participantes (1), de manera que era posible realizar manualmente la gestión de la información del mismo (proceso de evaluación de las soluciones, confección de la tabla de posiciones y otras gestiones relacionadas con los concursos de programación). En el transcurso de los años, el crecimiento en el número de participantes complejizó el proceso de evaluación, lo cual provocó el surgimiento de los **evaluadores automáticos[2]**, que fueron utilizados durante muchos años hasta que evolucionaron con el desarrollo de la web hacia lo que se conoce en la actualidad como **jueces en línea**.

El 5 de junio de 2010 se publica en Internet la versión 1 del Juez Caribeño en Línea (COJ por sus siglas en inglés) con el objetivo fundamental de brindar más autonomía al Movimiento ACM-ICPC en la región del Caribe, proporcionando a sus miembros (concursantes, entrenadores, directores y administradores) un espacio destinado a la auto-preparación y el intercambio de conocimientos.

La versión 1 del COJ surge de la posibilidad brindada por el Juez en Línea de Pekín (POJ por sus siglas en inglés) de descargar una versión básica y restrictiva de dicho sistema, pensada para ser instalada rápidamente sobre un entorno Windows[3] y caracterizada por estar compuesta de ficheros compilados,

[1] Es una de las primeras sociedades científicas y educativas acerca de la computación.
[2] Sistema cuyo propósito general es el procesamiento y evaluación de las soluciones propuestas por los concursantes.
[3] Sistema Operativo desarrollado por la Empresa Microsoft.

siendo así el caso del motor de evaluación (ejecutable desarrollado en C++[4] del cual no se tiene acceso al código fuente). Luego de un proceso de ingeniería inversa se logró obtener el código fuente de la aplicación web del POJ, el cual se usó como base para el desarrollo de la versión 1 del COJ.

Actualmente el COJ se encuentra instalado sobre plataforma Windows en la Universidad de las Ciencias Informáticas (UCI), contrario a las políticas de seguridad informática establecidas en la institución y los principios de soberanía tecnológica del Movimiento Caribeño del ACM-ICPC; por otra parte, la institución no posee ningún tipo de licencia para prestar servicios en Internet usando sistema operativo Windows, por lo cual no puede recibir el soporte ofrecido por Microsoft[5]. Además de los problemas legales que pudiera enfrentar la institución por el uso del sistema operativo Windows en la versión 1 del COJ, también pudiera estar en riesgo de ser demandada por los desarrolladores del POJ, puesto que su licencia restringe la publicación de dicha versión o cualquiera derivada para redes internas y establece la prohibición para realizar ingeniería inversa sobre los ejecutables que componen el sistema.

La información publicada en el sistema (Problemas, Concursos, Anuncios y Estadísticas) es gestionada a través del módulo de administración, el cual posee una interfaz poco usable e intuitiva debido a su estructura ineficiente. Asimismo, no abarca más de un tercio de las necesidades de gestión de la información del Juez en Línea, causando que en ocasiones sea necesario realizar la administración de los recursos de la aplicación directamente sobre la base de datos o los ficheros de configuración, y requiera de conocimientos de programación web y base de datos por parte de los administradores para la realización de dichas tareas, provocando en varias ocasiones el retraso en el inicio de concursos y en la publicación de problemas.

La dependencia del esquema de base de datos de la versión restrictiva del POJ provoca que el almacenamiento de datos estadísticos sea casi nulo, por lo cual el sistema no posee ninguna funcionalidad que permita a los usuarios evaluar su desempeño histórico en la aplicación, lo cual es crucial en sistemas de este tipo. Asimismo, el desconocimiento del código fuente del motor de evaluación

[4] Lenguaje de programación creado por Bjarne Stroustrup. Es considerado uno de los primeros lenguajes en soportar el paradigma de programación orientado a objetos.
[5] Empresa dedicada al desarrollo de software y equipos electrónicos, considerada como una de las corporaciones más grandes del siglo XX.

obstaculiza el desarrollo de nuevos estilos de concurso, provocando que los usuarios busquen alternativas de preparación en otros sistemas similares en Internet.

La naturaleza de los jueces en línea provoca que sean sistemas propensos a recibir ataques, por lo cual el ambiente donde se ejecutan y/o prueban los códigos fuentes enviados por los usuarios debe poseer ciertas restricciones, tales como:

- Restricción de lectura/escritura.
- Restricción en la ejecución de procesos.
- Restricción en el uso de hilos y conexiones TCP o UDP.

La versión 1 del COJ no posee ninguna de las restricciones mencionadas anteriormente, provocando que los programas enviados por los usuarios se ejecuten en un entorno con los permisos suficientes para realizar cambios en el servidor, constituyendo esto un serio problema de seguridad.

Los elementos mencionados anteriormente conducen a la formulación del siguiente **problema científico**:

¿Cómo erradicar las principales deficiencias identificadas en la versión 1 del COJ?

Para la presente investigación se define como **objeto de estudio**: Proceso de evaluación en línea de habilidades.

Campo de Acción: Proceso de evaluación en línea de habilidades para la programación competitiva de computadoras.

Persiguiendo como **objetivo general**: Desarrollar la versión 2 del COJ para erradicar las principales deficiencias identificadas en la versión 1; articulando dicho objetivo general en los siguientes **objetivos específicos**:

- Sistematizar acerca de los principales jueces en línea existentes a nivel local, nacional e internacional.
- Diseñar la versión 2 del COJ.
- Implementar la versión 2 del COJ.
- Probar las funcionalidades de la versión 2 del COJ.

La **idea a defender** en el presente Trabajo de Diploma es: El desarrollo de la versión 2 del COJ erradicará las principales deficiencias identificadas en la versión 1.

Para dar cumplimiento a los objetivos se definen las siguientes **tareas de investigación**:

- Realización de un estudio sobre los principales jueces en línea existentes en la UCI, Cuba y el resto del mundo.
- Realización de un estudio sobre las principales tecnologías de desarrollo.
- Realización de un estudio sobre el comportamiento de los diferentes compiladores de lenguajes de programación y la gestión de los permisos de usuarios sobre plataformas libres.
- Investigación de patrones y buenas prácticas para la programación de hilos y *sockets*.
- Indagación sobre las tecnologías y herramientas más populares en el desarrollo de jueces en línea sobre plataformas libres.
- Realización de una investigación sobre las metodologías de desarrollo de software.
- Identificación de las principales funcionalidades requeridas para el sistema.
- Descripción de la arquitectura del sistema y patrones empleados.
- Implementación de la solución.
- Investigación de mecanismos y técnicas para la realización de pruebas de software.
- Ejecución de las pruebas funcionales.

Para cumplimentar las tareas anteriormente expuestas se empleará un enfoque científico constituido por un conjunto de pasos o etapas establecidas para dirigir el proceso de investigación de manera óptima, permitiendo alcanzar su propósito (el conocimiento científico) de la manera más eficiente posible.

Los métodos teóricos utilizados en la presente investigación son: el Histórico Lógico y el Analítico-Sintético. El primero posibilita estudiar la trayectoria histórica y la lógica del desarrollo de jueces en línea, permitiendo conocer de forma general el funcionamiento de los mismos. El segundo, permite dividir y estudiar los componentes existentes y así poder identificar qué elementos pueden ser útiles para dar solución al problema.

Los métodos empíricos utilizados en la presente investigación son: la observación y la medición; el primero permite recopilar información de los jueces en línea implementados sobre plataformas libres,

permitiendo la obtención de conocimiento del tema a partir del análisis de las funcionalidades de los mismos y el segundo permite la obtención de información generalmente numérica para establecer comparaciones entre los diferentes sistemas.

A continuación se describe brevemente la estructura del presente trabajo de diploma, sintetizando el contenido de cada capítulo.

- En el capítulo 1 (Fundamentación teórica) se procederá a realizar un breve bosquejo sobre los antecedentes de los jueces en línea y la historia de su surgimiento, posteriormente se realizará un análisis de los principales jueces en línea en la UCI, Cuba y el Mundo teniendo en cuenta las tendencias más novedosas en cuanto a arquitectura, funcionalidades, tecnologías, metodologías y otros parámetros de interés. Seguidamente se realizará un análisis de las tecnologías y metodologías de desarrollo para elegir las que serán utilizadas en la implementación de la solución. Finalmente se reflejará mediante las conclusiones la posición de los autores acerca de los temas analizados en el capítulo.
- En el capítulo 2 (Análisis y Diseño de la Solución) se realizará la propuesta de la solución mediante una descripción de las funcionalidades seleccionadas para el desarrollo y de las características que se conciben para el sistema. En dicho capítulo se elaborarán las historias de usuarios correspondientes las cuales guiaran el desarrollo de forma organizada y se modelará la solución propuesta usando la metodología de desarrollo seleccionada además de explicar los patrones y arquitecturas seleccionados para la implementación de la solución.
- En el capítulo 3 (Implementación y pruebas) se obtendrá el plan de *release* del sistema, además se realizará la transformación de los diagramas del diseño para obtener los ficheros de código fuente que conforman el sistema. Una vez concluida esta fase de implementación se le realizarán las pruebas de aceptación e integración las cuales validarán si el sistema cumple con requerimientos deseados por el cliente.

1. CAPÍTULO 1. FUNDAMENTACIÓN TEÓRICA.

1.1 Introducción.

En este capítulo se abordarán los antecedentes de los jueces en línea y se realizará un estudio acerca de los principales sistemas de este tipo que existen en la actualidad en los diferentes ámbitos, para obtener conocimiento de cuáles son la tendencias que rigen el desarrollo de los mismos. Además, se seleccionarán las herramientas a utilizar durante el proceso de desarrollo de la solución propuesta.

1.2 Antecedentes de los jueces en línea.

Los concursos de programación al estilo ACM-ICPC tienen sus antecedentes en un concurso local celebrado en 1970 en la Universidad A&M de Tejas, Estados Unidos, y no es hasta 1977 que se desarrollan varias rondas clasificatorias como parte del concurso, realizándose por primera vez una final mundial organizada con el auspicio de la ACM. A pesar de ser durante mucho tiempo el único evento de su tipo en el mundo, prácticamente sólo equipos de Estados Unidos y Canadá participaron entre los años 1977 y 1989. No es hasta 1997 cuando con el patrocinio de la empresa IBM se aprecia un aumento considerable en las cantidades de equipos, instituciones y países participantes, evidenciándose un crecimiento anual entre el 10 y el 20% (2).

En la realización de dichos concursos un conjunto de jueces proponen varias tareas a resolver por cada uno de los equipos participantes, cada tarea se compone de uno o varios casos de prueba[6] y puede traer aparejado un evaluador externo[7]. Los equipos pueden solicitar la evaluación de una tarea en cualquier momento de la competencia sin ningún tipo de restricción de cantidad de solicitudes, esperando una respuesta en el menor tiempo posible. En cada competencia se realizan un número de solicitudes de evaluación que está condicionado por la cantidad de equipos y la dificultad de las tareas propuestas, efectuándose un cúmulo significativo de envíos por cada competencia, lo cual convierte la evaluación en un proceso tan complejo que es casi imposible realizarlo eficientemente desde un enfoque manual (3).

[6] Ficheros que contienen la entrada y salida original de cada tarea.
[7] Programa informático cuyo objetivo es la evaluación de la salida generada por una solución en las tareas con múltiples salidas para un mismo caso de prueba.

Generalmente, el proceso de evaluación de las tareas era realizado por evaluadores automáticos, los cuales son sistemas informáticos cuyo propósito general es el procesamiento eficiente de las solicitudes de evaluación.

En el año 1988 surge el proyecto PC^2[8], siendo uno de los primeros evaluadores automáticos del cual se tiene conocimiento, el mismo fue desarrollado inicialmente en Turbo Pascal y se mantuvo el desarrollo con dicha tecnología hasta 1999 que fue remplazada por Java, brindando por primera vez soporte para sistemas Unix[9], FreeBSD[10] y Solaris. Hasta el año 2001 se utilizaron varias versiones del sistema para la realización de competencias alrededor de todo el mundo, ya en el año 2000 se realiza la primera final mundial haciendo uso del sistema PC^2 sobre plataforma Unix (3).

Con el surgimiento y la popularidad de la web, los evaluadores automáticos fueron evolucionando a lo que se conoce en la actualidad como **Jueces en línea**.

1.3 Jueces en línea.

Un juez en línea es un sistema, por lo general web, que permite juzgar programas de computación que intenten solucionar tareas propuestas. Estos sistemas pueden compilar y ejecutar códigos fuente, y ponerlos a prueba con los juegos de datos definidos para la tarea seleccionada. Las posibles soluciones se ejecutan con restricciones tales como: límite de tiempo de ejecución, límite de memoria, tamaño de código fuente, restricciones de seguridad, entre otras (4). La salida de cada programa enviado por el usuario es capturada por el sistema y comparada contra la salida que se tiene de la tarea en cuestión o será evaluada por un evaluador externo.

1.3.1 Principales jueces en línea a nivel internacional.

Durante la investigación realizada se han identificado alrededor de 100 jueces en línea, de los cuales la mayor cantidad está concentrada en Asia; usualmente dichos sistemas están hospedados en instituciones

[8] Sistema de control de concursos de programación creado en la Universidad Estatal de California “Sacramento”.
[9] Sistema operativo portable, multitarea y multiusuario desarrollado en principio en los laboratorios Bell de AT&T.
[10] Sistema operativo libre basado en las CPU de arquitecturas INTEL aunque en la actualidad es compatible con otro número de arquitecturas.

de Educación Superior, debido a que el mayor número de usuarios de los mismos son estudiantes universitarios. Por otra parte, los jueces en línea también han sido utilizados como herramientas de apoyo al proceso de enseñanza-aprendizaje.

Juez en Línea de la Universidad de Valladolid (UVa Online Judge)[11].

El UVa es uno de los jueces en línea más antiguos y prestigiosos del mundo. Surge en el año 1995 con el propósito de ser usado como herramienta de entrenamiento para concursos de programación, pero no es hasta el año 1997 que comienza a cobrar fuerza en la comunidad internacional (4). Hasta la fecha (28 de noviembre de 2011) el sistema consta con 121061 usuarios registrados (de los cuales la mayoría son estudiantes universitarios), 9502299 soluciones evaluadas y 294 concursos realizados.

El UVa se caracteriza por tener una interfaz amigable y fácil de usar, además de ofrecer las funcionalidades típicas de todo juez en línea y de poseer uno de los mejores archivos de problemas del mundo. Como rasgos distintivos se pueden resaltar:

- Módulo de estadísticas, el cual muestra información detallada respecto a los problemas, usuarios y concursos.
- Integración a la plataforma EduJudge[12], convirtiéndolo en uno de los pocos jueces en línea que poseen una integración con el proceso de enseñanza-aprendizaje.
- Implementado sobre plataforma UNIX.
- Tabla de posiciones, dedicada al desempeño de los usuarios en competencias.

Sin embargo, se le puede señalar la escasa disponibilidad de lenguajes de programación, pues sus usuarios solamente pueden enviar soluciones implementadas en C[13], C++, JAVA[14] y PASCAL[15]. Asimismo, el sistema brinda poca información a los usuarios en el perfil (ver Figura 1). Es necesario señalar además que el proceso de mostrar el resultado de un envío incluye varios pasos, los cuales se

[11] http://uva.onlinejudge.org
[12] Proyecto perteneciente a la Universidad de Valladolid que se dedica al desarrollo de herramientas con carácter pedagógico utilizando el Juez en Línea UVa.
[13] Es considerado uno de los lenguajes más usado en el desarrollo de sistemas operativos.
[14] http://java.com
[15] Lenguaje de programación pensado para ser usado con fines científicos.

consideran innecesarios, provocando que los usuarios tarden más tiempo en obtener el resultado de la calificación.

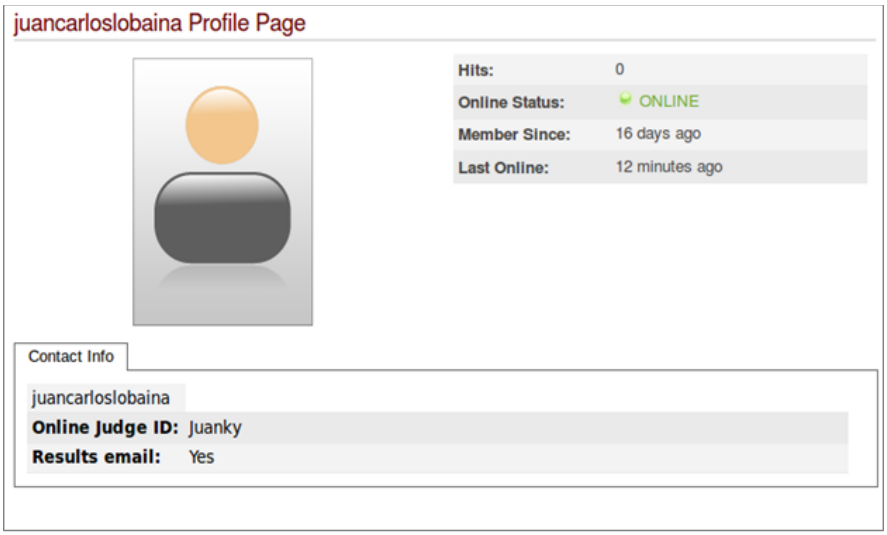


Figura 1. Perfil del Usuario en UVa.

Sphere Online Judge (SPOJ).

El SPOJ es un juez en línea desarrollado por el proyecto Sphere Research Labs, grupo dedicado al desarrollo de tecnologías en línea para proveer herramientas de evaluación automática de habilidades (5). El SPOJ se ha convertido en uno de los jueces en línea más populares en la comunidad internacional, contando hasta la fecha (30 de noviembre de 2011) con alrededor de 100000 usuarios registrados, 6109235 envíos realizados, 1866 concursos hospedados y 10141 problemas en su archivo, convirtiéndolo en uno de los archivos más grandes de problemas y de más rápido crecimiento en el mundo.

El SPOJ se caracteriza por tener una interfaz intuitiva, sencilla y ligera, propiciando a usuarios nuevos e inexpertos la fácil comprensión del propósito de la herramienta. Posee varias características novedosas que lo diferencian de la mayoría de los jueces en línea, tales como:

- Diferenciación del archivo de problemas en cuanto a dificultad y propósito.
- Brinda soporte a más de 40 lenguajes de programación.

- Posee tiempos de evaluación muy rápidos, a pesar de las pocas prestaciones del servidor sobre el cual está desplegado.
- Brinda un abundante kit de herramientas asociadas al sistema.
- Espacio de intercambio para sus usuarios mediante un foro de discusión.

Quizás las dos características más distintivas del sistema son el uso de MOSS[16], como herramienta de detección de plagios, y la posibilidad brindada a determinados usuarios de organizar concursos bajo sus propias reglas y problemas.

Se considera que el SPOJ hace un uso excesivo de anuncios publicitarios en las principales vistas del sistema; además de no brindar información referente a su historia. Por otra parte, también carece de referencias a documentación o tutoriales destinados a la preparación de los usuarios en los temas relacionados con la programación competitiva.

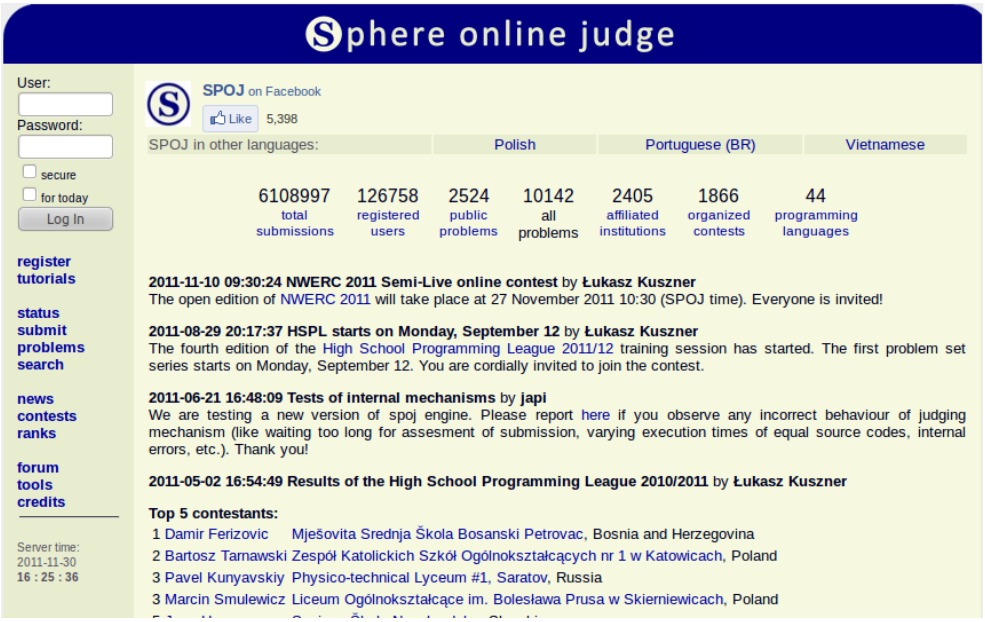


Figura 2. Página Principal de SPOJ.

Juez en Línea de Pekín (POJ).

[16] Herramienta desarrollada en la Universidad de Stanford desde el año 1994 para la detección de plagio http://theory.stanford.edu/~aiken/moss/

El POJ es uno de los jueces en línea de mayor demanda entre la comunidad universitaria china, puesto que es uno de los más antiguos de esa región. Posee características similares a varios jueces en línea de otras universidades chinas y otros lugares del mundo, debido a que por algún tiempo era posible descargar una versión disminuida y restrictiva de la aplicación. Hasta la fecha (3 de diciembre de 2011), el sistema posee alrededor de 3000 problemas, 482000 usuarios registrados y 9623000 envíos juzgados. El POJ se caracteriza por poseer una interfaz intuitiva, ligera y fácil de usar, además de ser uno de los jueces en línea con menores tiempos de respuesta en su interfaz y poseer una abundante sección de preguntas (FAQ[17] por sus siglas en ingles).

Su característica más distintiva es el módulo de estadísticas (ver Figura 3), ya que recopila y muestra los datos de uso del sistema desde su surgimiento en el año 2003 hasta la fecha. Por otra parte, ofrece un servicio de mensajería interna para los usuarios registrados, funcionalidad que se considera muy útil en este tipo de sistemas.

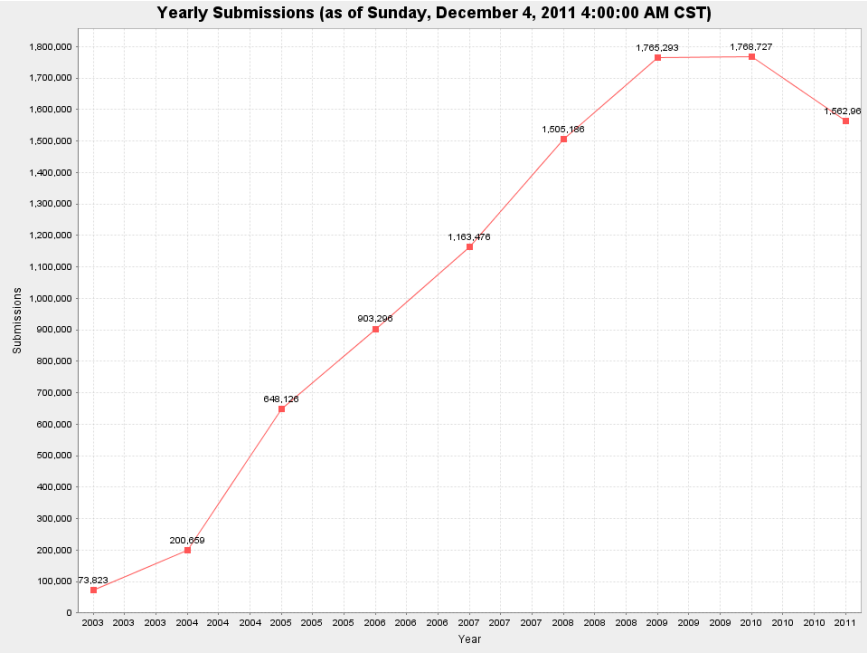


Figura 3. Gráfica de las estadísticas de los envíos en el POJ.

[17] Sección usada mayormente en sitios web para caracterizar el sistema y ayudar a los usuarios en el uso del mismo.

Se considera que el POJ brinda pocos lenguajes de programación (solamente cinco) y que la principal deficiencia del sistema radica en estar implementado sobre plataforma Windows.

Saratov State University Online Contester.

Este juez en línea fue uno de los primeros sistemas de este tipo desarrollados por y para universidades rusas; surge en la Universidad Estatal de Saratov en el año 2002. Hasta la fecha (5 diciembre de 2011), el sistema cuenta con cerca de 500 problemas en su archivo, 1254152 envíos realizados y 16000 usuarios registrados. Acorde a los datos analizados se puede asumir que el SGU es, de los jueces en línea analizados, el que menor crecimiento exhibe en su archivo de problemas. El SGU no posee un interfaz amigable, debido a que el diseño del portal muestra letras muy pequeñas lo cual tiende a confundir los enlaces.

La característica más relevante que muestra el SGU es el módulo de **Concursos Virtuales**. El SGU exhibe además un novedoso módulo de estadísticas, mostrando información referente al mejor usuario de la semana, así como a los usuarios, los países y las instituciones con mejor desempeño en un período determinado.

Comparándolo con el resto de los jueces analizados, el SGU muestra un menor crecimiento en cuanto a los principales parámetros analizados.

Juez en Línea Timus (TIMUS).

El TIMUS posee el mayor archivo ruso de problemas construido a partir de concursos de programación. El sistema está desarrollado y administrado por estudiantes de la Universidad Estatal de los Urales. Hasta la fecha (5 de diciembre de 2012), el sistema cuenta con 65000 usuarios registrados, 4020000 envíos realizados y un extenso archivo de problemas dividido en volúmenes, áreas del conocimiento y tipos de concursos. TIMUS posee una interfaz intuitiva y fácil de usar, además de un rápido tiempo de respuesta. El sistema no admite autenticación; en su lugar, a la hora de realizar algún envío, se le exige al usuario el JUDGE_ID[18] mediante el cual es posible realizar los envíos y cualquier otra funcionalidad que implique identificación. La característica más relevante del sistema, en comparación con la mayoría de los

[18] Identificador enviado por el sistema durante el proceso de registro a cada usuario.

analizados, es la internacionalización de la interfaz de la aplicación en ruso e inglés. El sistema ofrece soporte a pocos lenguajes de programación, lo que se considera una deficiencia.

Tianjin University Online Judge (TOJ).

El TOJ surge a partir de la posibilidad brindada por el POJ de descargar una versión disminuida y restrictiva de la aplicación. Posee las mismas funcionalidades y características del POJ, solo que incluye la funcionalidad de crear Concursos Virtuales Públicos. La diferencia más significativa del TOJ, respecto al POJ, radica en la no dependencia de plataforma Windows.

ACM-ICPC Live Archive.

El Archivo del ACM-ICPC es el Juez en Línea con mayor número de problemas de competencias oficiales (Concursos Regionales, Finales Mundiales, entre otros). El Archivo se caracteriza por tener una interfaz amigable, muy similar a la interfaz de UVA. Su funcionalidad más relevante, además del ordenamiento de los problemas por concursos, es la posibilidad brindada a numerosos usuarios de organizar sus propias competencias sobre la plataforma y aportar problemas que aún no estén publicados en la misma. Se considera que su principal deficiencia radica en el corto tiempo de vida de los envío de los usuarios al sistema.

USA Computing Olympiad (USACO).

USACO es una plataforma compuesta por varios portales dedicados a la preparación, información y apoyo a los competidores de IOI alrededor de todo el mundo. Los dos portales de mayor relevancia en la plataforma son el USACO Training Gateway y el USACO Contest Gateway; el primero es una plataforma destinada a la preparación de los competidores, con una serie de problemas organizados por secciones y capítulos. Los usuarios deben vencer los problemas de un capítulo para poder pasar a la próxima sección/categoría de entrenamiento (ver Figura 4). La funcionalidad más importante de este módulo es el análisis de los problemas: una vez resuelto el problema se habilita un análisis realizado por el creador. También se puede resaltar que se muestra en todo momento, a través de la interfaz, la cantidad de usuarios conectados por países.

Section 1.3	DONE 2010.11.02	TEXT Greedy Algorithm
	DONE 2010.06.02	PROB Mixing Milk [ANALYSIS]
	VIEWED 2010.06.02	PROB Barn Repair
	TODO	TEXT Winning Solutions
	TODO	PROB Calf Flac
	VIEWED 2010.06.02	PROB Prime Cryptarithm
Section 1.4	TODO	TEXT More Search Techniques
	TODO	PROB Packing Rectangles
	TODO	PROB The Clocks
	TODO	PROB Arithmetic Progressions
	TODO	PROB Mother's Milk
Section 1.5	TODO	TEXT Introduction to Binary Numbers
	TODO	PROB Number Triangles
	TODO	PROB Prime Palindromes
	TODO	PROB SuperPrime Rib
	TODO	PROB Checker Challenge
Chapter 2	TODO	Bigger Challenges
Chapter 3	TODO	Techniques more subtle
Chapter 4	TODO	Advanced algorithms and difficult drills
Chapter 5	TODO	Serious challenges
Chapter 6	TODO	Contest Practice

Figura 4. USACO Training Gateway.

Del USACO Training Gateway se considera como principal deficiencia la obligatoriedad de interactuar con ficheros (lectura y escritura de datos) desde el código fuente, en lugar de efectuar dicho proceso por la entrada y salida estándar. Además el sistema obliga al usuario a incluir datos de la solución (id del usuario, id del problema y lenguaje de programación) en el código fuente para poder juzgar el envío.

La USACO Contest Gateway es una plataforma optimizada para la realización de concursos al estilo IOI y con las mismas características de calificación que se usan en el USACO Training Gateway. La funcionalidad más importante de este módulo es la evaluación de los usuarios según su desempeño en concursos pasados, restringiendo el acceso a determinadas competencias según la categoría del usuario.

TopCoder.

Topcoder es una compañía que organiza concursos de programación de diferentes estilos:

- Algoritmos: Consiste en una prueba algorítmica, generalmente de 3 problemas, conformada por 4 fases:
    - Fase de codificación: Posee una duración de 75 minutos, durante esta primera fase los competidores codifican las soluciones a los problemas en un espectro reducido de lenguajes de programación (Java, C++, C# y Visual Basic). Los puntos recibidos por cada solución varían según la complejidad del problema y el tiempo empleado para darle solución.
    - Fase Intermedia: Es un descanso breve (5 minutos) entre la fase de codificación y la fase de retos, durante esta fase los competidores generalmente pueden pensar en los retos.
    - Fase de retos: Posee una duración de 15 minutos donde cada competidor tiene la posibilidad de desafiar las soluciones de otros competidores, probándolas con pruebas generadas por ellos mismos. El retador recibe 50 puntos por cada intento satisfactorio de reto (mientras que el retado pierde todos los puntos recibidos por el problema) y pierde 25 puntos si el intento no es satisfactorio.
    - Fase final: Cada solución que sobrevive la fase de retos es probada con varios juegos de datos establecidos por el sistema, la solución no recibe los puntos si en definitiva no pasa la prueba.
- Diseño: Posee una duración de 1 semana, donde se le proporciona a los competidores una serie de requisitos con el objetivo de obtener un producto de software usable. Las plataformas soportadas son Java y .Net.
- Marathon Matches: Se propone una tarea algorítmica que será evaluada por un sistema automático. Posee una duración entre 1 y 2 semanas.
- Competencias de Detección de Errores: Los clientes y competidores comparten un espacio para intercambiar acerca de los errores de diferentes softwares.

El principal objetivo de la compañía es proveer una plataforma donde las diferentes empresas puedan reclutar personal talentoso, basados en el desempeño de los mismos en competencias. Topcoder funciona además como una fuente de creatividad para las empresas que contratan los servicios de la plataforma.

El rasgo más distintivo de la plataforma es la integración como plataforma educativa y empresarial al mismo tiempo, además de incluir novedosos estilos de competencias, así como nuevos sistemas de puntuación.

Codeforces.

En la actualidad, Codeforces es uno de los jueces en línea más populares. Surge en el año 2010 como iniciativa de Mike Mirzanayov[19], cambiando la dinámica de los jueces en línea existentes hasta el momento. Codeforces se caracteriza por ser uno de los pocos jueces en línea que aplica la Web 2.0 y constituir un sistema predominantemente para la realización de competencias de programación en varios estilos existentes.

También aplica una funcionalidad que se considera vital para incrementar la preparación de los concursantes y es la referente a mostrar los códigos fuentes de las soluciones una vez finalizado cada concurso. Por otra parte, como resultado de aplicar la Web 2.0, ofrece la posibilidad de autenticarse en el sistema con el mismo usuario de Google. Quizás el rasgo más distintivo de Codeforces es el sistema de puntuación y ranking que aplica.

COJ versión 1.

El desarrollo del sistema base (Xtreme Online Judge) comenzó en el año 2006 a partir de la versión limitada y restrictiva del POJ, bajo la "Iniciativa Xtreme", compuesta principalmente por estudiantes y profesores de la antigua Facultad 8 de la UCI. Luego de que la UCI se uniera al Movimiento ACM-ICPC y liderara la creación de la Comunidad Caribeña del ACM-ICPC, el Xtreme Online Judge fue seleccionado para publicarse en Internet como COJ v1, siendo la fecha de publicación el 5 de junio de 2010. El COJ v1 posee las funcionalidades típicas de todos los sistemas que tuvieron como punto de partida al POJ, aunque la interfaz y la arquitectura de la información cambiaron notablemente. La característica más considerable del sistema es la rapidez en los tiempos de respuesta en la mayoría de las interfaces, no así en la vista del estado de los envíos.

El módulo de administración del sistema no cambió mucho comparado con la administración proveniente de la versión disminuida del POJ, ofreciendo funcionalidades mínimas tales como:

[19] Creador del Saratov State University Online Contester y de Codeforces.

- Adicionar, Listar y Modificar Problemas.
- Adicionar y Listar Competencias.
- Manejar los grupos de usuarios en competencias.
- Re-puntear problemas.

El resto de los procesos de administración (ver Figura 5) es obligatorio realizarlos de manera directa sobre los ficheros de configuración o en la base de datos.

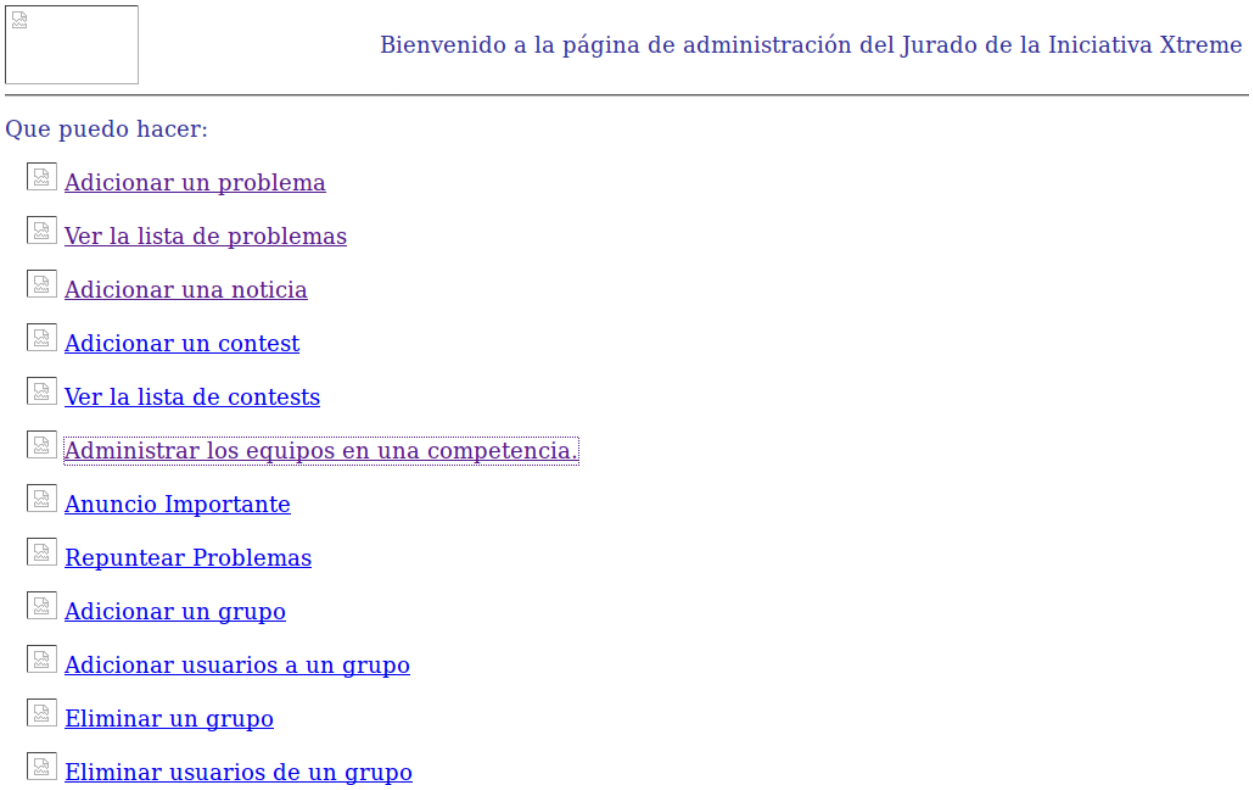


Figura 5. Módulo de Administración de COJ versión 1.

Teniendo en cuenta que la presente investigación se centra en resolver las principales deficiencias identificadas en la versión 1 del COJ; los autores consideran importante mencionarlas de forma detallada:

- Dependencia de la plataforma Windows.

- La dependencia del esquema de base de datos de la versión restrictiva del POJ, provoca que el almacenamiento de datos estadísticos sea casi nulo, por lo cual el sistema no posee ninguna funcionalidad que permita a los usuarios evaluar su desempeño histórico en la aplicación.
- El desconocimiento del código fuente del motor de evaluación, obstaculiza el desarrollo de nuevos estilos de concurso, provocando que los usuarios busquen alternativas de preparación en otros sistemas similares en Internet.
- El módulo de administración posee una interfaz poco usable e intuitiva, al mismo tiempo que no abarca más de un tercio de las necesidades de gestión del juez en línea.
- El entorno donde se ejecutan los programas enviados por los usuarios, no posee las restricciones necesarias para ser considerado seguro.

1.3.2 Principales jueces en línea a nivel nacional.

El desarrollo de jueces en línea en Cuba no ha sido una práctica muy difundida hasta la fecha, debido principalmente a la escasa participación de los estudiantes cubanos en competencias ACM-ICPC. La UCI fue la primera institución del país que incursionó en dichos desarrollos.

UCOCoach.

El sistema UCOCoach surge como iniciativa de estudiantes y profesores de la Universidad de Camagüey, con el objetivo de apoyar a los concursantes de dicha institución en la preparación para competencias de programación. Posee las características comunes de todo juez en línea, además de una interfaz amigable y sugerente. La funcionalidad más relevante del sistema es la ayuda que ofrece automáticamente a los usuarios, cuando estos fallan varias veces un mismo problema. El sistema está actualmente en desarrollo y no se encuentra disponible en Internet.

1.3.3 Principales jueces en línea a nivel local.

La UCI, a pesar de su corto período de fundación, es la institución del país con mayores resultados en el desarrollo de jueces en línea.

Cátedra de Programación Avanzada (CPAV).

El Juez en Línea de la Cátedra de Programación Avanzada (CPAV) es un sistema desarrollado con el CMS PHP[20] Fusión para su interfaz y haciendo uso del lenguaje C++ para la implementación del motor de calificación.

Actualmente cuenta con alrededor de 6000 usuarios registrados y más de 700 tareas que pueden ser resueltas en los lenguajes C, C++, Java, Perl, Python, Pascal, C# y Visual Basic.

Este sistema posee 5 módulos principales:

- Gestión de equipos.
- Gestión de concursos.
- Gestión de tablas de posiciones.
- Gestión de los volúmenes de tareas.
- Gestión de la administración.

El desarrollo del juez en línea de la CPAV sobre un CMS brinda una serie de ventajas para la incorporación de varias funcionalidades (sobre todo de gestión de contenidos), pero al mismo tiempo genera una serie de inconvenientes, principalmente en torno al rendimiento y los tiempos de respuesta.

1.4 Tendencias de los jueces en línea.

Los jueces en línea son aplicaciones que se caracterizan por el amplio consumo de recursos de hardware causado en gran medida por la demanda de procesamiento de las soluciones a evaluar. Un juez en línea medianamente popular recibe en un día promedio de 800 a 3000 solicitudes de evaluación, pudiendo incrementarse considerablemente en un día de competencia, por lo que los jueces en línea deben basar su implementación en arquitecturas eficientes, lenguajes robustos y en la optimización de sus funcionalidades. Por otra parte, los jueces en línea son sistemas propensos a recibir ataques por lo que la implementación de los mismos debe estar sustentada por una plataforma lo más segura posible.

[20] www.php.net

Los jueces en línea generalmente están compuestos por dos módulos fundamentales, el módulo de presentación (Generalmente una aplicación web) y el módulo de evaluación que puede estar embebido en el módulo de presentación o ser una aplicación independiente, generalmente conocida como evaluador automático o motor de evaluación, el cual es usado de forma centralizada o distribuida según la arquitectura seleccionada para la implementación del mismo.

1.4.1 Arquitectura.

Los jueces en línea convencionales (POJ, SPOJ, COJ v1, entre otros) implementan evaluadores de forma centralizada o embebidos como funcionalidad de la aplicación Web; los evaluadores centralizados o embebidos se caracterizan por poseer tiempos de respuesta más rápidos que los evaluadores distribuidos pues no incluyen en su funcionamiento el factor “rapidez de la red”, pero al mismo tiempo, son propensos a sobrecargar el servidor y su funcionamiento pone en riesgo la integridad de los datos del Juez en Línea y la seguridad del servidor donde se encuentra el sistema principal.

Los jueces en línea más modernos (Codeforces, UCOCoach, entre otros) implementan los evaluadores automáticos aplicando **programación distribuida**[21]. EL uso de evaluadores distribuidos puede incluir una tercera aplicación que es el distribuidor/balanceador de tareas/carga, encargado de comunicarse con cada uno de los servicios distribuidos por los nodos de la red. Los motores de evaluación distribuidos se caracterizan por minimizar los daños al servidor ante un posible ataque sobre el sistema, además de optimizar el procesamiento de las soluciones por la posibilidad de realización de tareas simultáneas en diferentes servidores. Por otra parte, el uso de sistemas distribuidos no limita la posibilidad de usarse de forma centralizada.

[21] paradigma de programación enfocado en desarrollar sistemas distribuidos, abiertos, escalables, transparentes y tolerantes a fallos. Este paradigma es el resultado natural del uso de las computadoras y las redes (6)

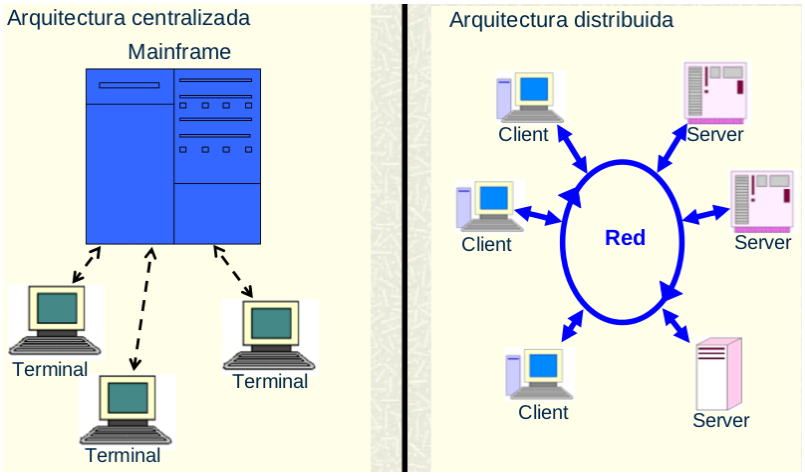


Figura 6. Arquitecturas Centralizada y Distribuida.

Existen varias tendencias/arquitecturas en la implementación de los jueces en línea para el proceso de almacenamiento de los juegos de datos de cada problema.

- Almacenamiento físico en el servidor donde se ejecuta el evaluador o motor de calificación.
- Almacenamiento en Base de Datos.
- Almacenamiento mediante un sistema de archivos de red (NFS por sus siglas en ingles).

El almacenamiento físico en el servidor es la variante más empleada entre todos los jueces en línea por ser la menos compleja de implementar aunque no es la más segura y eficiente debido a que la distribución de los juegos de datos entre varios evaluadores distribuidos es un proceso complejo. El almacenamiento en Base de Datos es la menos eficiente y es usada generalmente por sistemas personalizados para competencias, que manejan pequeños volúmenes de problemas y no se usan en entrenamiento las 24 horas. El almacenamiento usando NFS es el que mayor dificultad presenta para su configuración pero al mismo tiempo es el que exhibe mayor seguridad y eficiencia pues puede funcionar de forma centralizada y distribuida, permitiendo gestionar la seguridad e integridad de los ficheros usados por cada evaluador automático, además de mantener cada servidor actualizado de forma automática ante los cambios que puedan ocurrir en el servidor principal.

1.4.2 Tecnologías.

El estudio realizado sobre los diferentes jueces en línea develó que:

- Las 2 tecnologías más utilizadas para el desarrollo de la capa de presentación entre los jueces en línea analizados son Java y PHP. Java alcanza este auge debido a que el POJ ha sido el punto de partida para muchos jueces en línea alrededor de todo el mundo mayoritariamente en el continente asiático y PHP por su condición de lenguaje más usado y popular para el desarrollo de aplicaciones Web dinámicas en los últimos años (7).
- Para el desarrollo del módulo de evaluación las tecnologías utilizadas han sido C++, Java y Python[22] siendo el primero el que ha sido usado mayoritariamente por integrarse fácilmente con las diferentes arquitecturas y caracterizarse por ser uno de los lenguajes más rápidos de los últimos tiempos; por otra parte, Java y Python han sido las otras opciones de desarrollo destacándose Java debido a la popularidad alcanzada por la plataforma en los últimos años.
- Las tecnologías de procesamiento de datos más utilizadas entre estos tipos de aplicaciones han sido PostgreSQL y MySQL; sin existir una diferencia marcada en el uso de una u otra para el desarrollo de jueces en línea.

1.4.3 Funcionalidades.

Luego de realizado el análisis de los jueces seleccionados se identificaron un conjunto de funcionalidades que se consideran las más importantes a tener en cuenta para el desarrollo de un juez en línea exitoso y seguro.

Concursos Virtuales.

Considerando que en la fuente consultada no se encontró una definición exacta del término “Concurso Virtual”, los autores luego de analizar los diferentes sistemas que soportan dicha funcionalidad estiman necesario aportar una definición propia:

[22] http://www.python.org

Concurso Virtual: Es una herramienta, módulo o funcionalidad que permite a concursantes y entrenadores revivir en tiempo real concursos pasados, convirtiendo el proceso de análisis de un concurso anterior en una experiencia real, emocionante y dinámica; pues el concursante tiene la oportunidad de volver a enfrentarse al concurso o de competir en uno en el cual no estuvo presente en un ambiente virtual donde todo se desarrolla con la misma dinámica de la competencia pasada.

Dicha funcionalidad solo fue identificada en dos de los jueces en línea analizados (TOJ y Saratov), siendo considerada un factor determinante en la popularidad alcanzada por ambos.

Sistemas de puntuación.

Quizás uno de los rasgos más distintivos de los Jueces en Línea sean los sistemas de puntuación, dado que dicha característica es la que ubica al juez en línea en una de las 3 categorías siguientes.

- Juez en Línea de entrenamiento.
- Juez en línea de competencias.
- Juez en Línea balanceado entre competencias y entrenamiento.

Los autores consideran que un juez en línea exitoso debe lograr un equilibrio entre entrenamiento y competencias por lo cual debe soportar diferentes estilos de puntuación.

Estilos de concurso.

Algunos sistemas como el POJ solo soportan la realización de competencias de programación al estilo ACM-ICPC, debido a que son herramientas desarrolladas con el objetivo de fomentar la preparación de los estudiantes universitarios. Sin embargo, otros como Codeforces, Topcoder y SPOJ incluyen conceptos novedosos en la realización de dichos eventos.

Competencias por niveles: El acceso de los usuarios a los problemas está limitado según el nivel del mismo, el cual es determinado de acuerdo al desempeño del usuario en la competencia.

Competencias Libres: Basa su funcionamiento en la asignación estática de puntos a cada solución correcta sobre un problema.

Usabilidad.

El sistema debe caracterizarse por ser una aplicación sencilla y fácil de usar, causando en los usuarios una sensación de familiaridad y posibilitando una rápida y fácil comprensión de su propósito. Asimismo, debe incluir referencia a documentación que permita a los usuarios nuevos la preparación. La inclusión de propaganda es considerada un factor de rechazo en este tipo de sistemas, al mismo tiempo que permite a la entidad portadora la financiación (por lo que se considera que no se debe abusar de este modelo de negocio).

1.5 Tecnologías de desarrollo de software.

1.5.1 Lenguajes de programación.

Un lenguaje de programación es aquel elemento dentro de la Informática que permite crear programas mediante un conjunto de instrucciones, operadores y reglas de sintaxis, poniéndose a disposición del programador para que éste pueda comunicarse con los dispositivos de software y hardware (8).

C++.

C++ es un lenguaje de programación de propósito general basado en el lenguaje de programación C. Además de los recursos de C, C++ proporciona clases, funciones en línea (*inline*), sobrecarga de operadores, sobrecarga de nombres de funciones, tipos constantes, referencias, operadores para el manejo del almacenamiento disponible, verificación de argumentos de funciones y conversiones de tipos (9). El lenguaje C++ se comenzó a desarrollar en 1980, por Bjarne Stroustrup; en la actualidad C++ es un lenguaje versátil y potente, se puede utilizar en la construcción de todo tipo de aplicaciones aunque su mayor aplicación radica en las aplicaciones de escritorio, sistemas operativos y servicios.

PHP.

PHP es un lenguaje interpretado en el lado del servidor que se caracteriza por su potencia, versatilidad, robustez y modularidad. Los programas escritos en PHP son embebidos directamente en el código HTML y ejecutados por el servidor web a través de un intérprete antes de transferir al cliente que lo ha solicitado un resultado en forma de código HTML puro. Al ser un lenguaje que sigue la corriente *open source,* tanto el intérprete como son totalmente accesibles de forma gratuita en la red. Es un lenguaje multiplataforma,

que puedo funcionar sobre la mayoría de los servidores web y brinda soporta a más de 20 tipos de base de datos (10).

Java.

Java es un lenguaje de programación de alto nivel desarrollado por Sun Microsystem a principios de la década del 90. Su sintaxis es muy parecida a la de C y C++ ya que su desarrollo fue inspirado en los mismos, siempre buscando eliminar errores que suelen inducirse en lo que respecta a la manipulación de memoria y punteros (11).

Los programas en Java generalmente son compilados a un lenguaje intermedio llamado *bytecode*, que luego es interpretado por una máquina virtual (JVM). Esta última sirve como una plataforma de abstracción entre el hardware y el lenguaje permitiendo que se pueda ejecutar el programa sobre cualquier arquitectura. Señalar que la compilación a código máquina también es posible, brindando mayor eficiencia en la ejecución del programa pero limita su característica multiplataforma.

Entre las características principales de este lenguaje se encuentran las siguientes:

- Simple: Elimina la complejidad de los lenguajes como C y C++ dando paso al contexto de los lenguajes modernos orientados a objetos.
- Orientado a objetos: Soporta las características esenciales del paradigma de la programación orientada a objetos: encapsulamiento, herencia y polimorfismo.
- Robusto: Elimina el uso de apuntadores para referenciar áreas de memoria, además, despoja al desarrollador de la necesidad de liberar la memoria que la aplicación ya no usa. También requiere la declaración explícita tanto de los tipos de datos como de los métodos.
- Multiplataforma: El mismo código Java que funciona en un sistema operativo, funciona en cualquier otro que tenga instalada la máquina virtual de Java.
- Multitareas: Permite la ejecución concurrente de varios procesos ligeros o hilos de ejecución.

La versatilidad y eficiencia de la tecnología Java, la portabilidad de su plataforma y la seguridad que aporta, la han convertido en la tecnología ideal para el desarrollo de aplicaciones distribuidas.

Python.

Python es un lenguaje de programación creado por Guido van Rossum a principios de los años 90 cuyo nombre está inspirado en el grupo de cómicos ingleses “Monty Python”. Es un lenguaje similar a Perl, pero con una sintaxis muy limpia y que favorece un código legible. Se trata de un lenguaje interpretado o de script, con tipado dinámico, fuertemente tipado, multiplataforma y orientado a objetos (12).
En los últimos años el lenguaje se ha hecho muy popular, gracias a varias razones como:

- La cantidad de librerías que contiene, tipos de datos y funciones incorporadas en el propio lenguaje, que ayudan a realizar muchas tareas habituales sin necesidad de tener que programarlas desde cero.
- La sencillez y velocidad con la que se crean los programas. Un programa en Python puede tener de 3 a 5 líneas de código menos que su equivalente en Java o C.
- La cantidad de plataformas en las que se puede desarrollar, como Unix, Windows, OS/2, Mac.

Python es gratuito, incluso, para propósitos empresariales.

Análisis de la selección del lenguaje de programación.

Para la selección del lenguaje de programación a emplear en el desarrollo de la solución los autores tuvieron en cuenta 6 indicadores fundamentales, seleccionados de forma arbitraria de acuerdo a las necesidades del desarrollo y la experiencia de los mismos. Para la evaluación de cada uno de los indicadores se determinó una escala numérica del 1 al 4 que evalúa la aceptación de cada uno en orden ascendente, siendo la menor evaluación 1 y la mayor 4.

Indicador	**Java**	**PHP**	**Python**	**C++**
Desarrollo web	4	4	4	1
Implementación de servicios o demonios[23]	4	1	4	4
Interoperabilidad	4	1	4	3
Rendimiento	3	2	2	4
Experiencia	4	1	1	2

[23] Tipo especial de proceso informático no interactivo que se ejecuta en segundo plano.

Adaptación a las necesidades de desarrollo	4	2	4	3

Basados en la puntuación otorgada a cada uno de los indicadores, los autores seleccionan a Java como lenguaje para el desarrollo de la solución. Esta selección permite a los autores aprovechar la experiencia obtenida con el uso de la plataforma Java, así como reutilizar el conocimiento adquirido previamente en el uso de las herramientas asociadas al lenguaje, lo cual compensa las deficiencias que pueda presentar la plataforma.

1.5.2 Framework.

Un *framework,* en el idioma utilizado por los desarrolladores de software, es una estructura de soporte definido, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Típicamente, puede incluir soporte de programas, bibliotecas y un lenguaje interpretado para ayudar a desarrollar y unir los diferentes componentes de un software (13).

Spring.

Spring es un *framework* de código abierto muy popular y ampliamente desarrollado, que ayuda a los desarrolladores a construir rápidamente aplicaciones de alta calidad. Spring ofrece una programación coherente y un modelo de configuración que es bien entendido y utilizado por millones de desarrolladores de todo el mundo. A diferencia de la tradicional plataforma Java EE, Spring ofrece una gama alta de capacidades para la creación de aplicaciones en Java, que se caracterizan por ser ligeras y potentes (14).

Entre sus posibilidades más potentes radica su contenedor Inversión de Control también llamado Inyección de Dependencias (DI por sus siglas en inglés) que es una técnica alternativa a las clásicas búsquedas de recursos vía JNDI[24], permite configurar las clases en un archivo XML[25] y definir en él las

[24] Interfaz de Nombrado y Directorio Java (*Java Naming and Directory Interface*) es una Interfaz de Programación de Aplicaciones (API) de Java para servicios de directorio.

dependencias. De esta forma la aplicación se vuelve muy modular y a la vez no adquiere dependencias con Spring.

Struts.

Apache Struts es una herramienta de soporte para el desarrollo de aplicaciones Web bajo el patrón MVC con la plataforma Java EE (Java *Enterprise Edition*). Struts se desarrollaba como parte del proyecto Jakarta de la Apache *Software Foundation*, pero actualmente es un proyecto independiente conocido como Apache Struts.

Struts permite reducir el tiempo de desarrollo. Su carácter de software libre y su compatibilidad con todas las plataformas en las que Java *Enterprise* esté disponible lo convierten en una herramienta altamente disponible.

Análisis de la selección del framework.

Después de realizar un análisis sobre los *frameworks* anteriores, los autores de la solución deciden utilizar Spring como *framework* principal para el desarrollo de la solución, basándose en un estudio comparativo realizado entre ambos *frameworks,* el cual demostró que Spring posee las siguientes ventajas sobre su homólogo:

- Spring MVC es muy flexible, ya que implementa toda su estructura mediante interfaces; a diferencia de Struts, el cual obliga a heredar de clases concretas tanto en sus Controladores como en sus Formularios. Además, todas las partes del framework son configurables a través de *plugins*.
- Los controladores de Spring MVC se configuran mediante Inversión de Control como los demás objetos, lo cual los hace fácilmente integrables con otros objetos que estén en el contexto de Spring y, por consiguiente, que sean manejables por éste.

[25] XML, siglas en inglés de *eXtensible Markup Language* (lenguaje de marcas extensible), es un metalenguaje extensible de etiquetas desarrollado por el *World Wide Web Consortium* (W3C).

- Resulta más fácil probar las partes de Spring MVC que las de Struts, debido a que el primero evita forzosamente la herencia de una clase y la dependencia directa en el controlador del servlet[26] que despacha las peticiones.

1.5.3 Servidores de bases de datos.

Los servidores de Bases de datos, también conocidos como DBMS (acrónimo en inglés de *DataBase Management Systems*), son programas que permiten organizar datos en una o más tablas relacionadas. Los servidores de Bases de Datos se utilizan en todo el mundo en una amplia variedad de aplicaciones. Prácticamente cualquier aplicación que necesite almacenamiento, acceso y análisis de datos estructurados hace uso de algún tipo de DBMS. La mayoría de los servidores de bases de datos ofrecen una interfaz de texto rudimentaria que permite interactuar con el servidor usando SQL (*Structured Query Language*).

MySQL.

MySQL es un sistema de gestión de bases de datos relacional, multi-hilos y multi-usuario. MySQL AB (desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009) desarrolla MySQL como software libre en un esquema de licenciamiento dual (10).

Por un lado se ofrece bajo la GNU-GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso. Está desarrollado en su mayor parte en ANSI C.

Al contrario de proyectos como Apache, donde el software es desarrollado por una comunidad pública y los derechos de autor del código están en poder del autor individual, MySQL es patrocinado por una empresa privada, que posee el *copyright* de la mayor parte del código.

Esto es lo que posibilita el esquema de licenciamiento anteriormente mencionado. Además de la venta de licencias privativas, la compañía ofrece soporte y servicios.

[26] http://www.servlets.com

PostgreSQL.

Es un sistema de gestión de bases de datos objeto-relacional (ORDBMS). Basado en el proyecto POSTGRES, de la universidad de Berkeley. El director de este proyecto es el profesor Michael Stonebraker, y fue patrocinado por *Defense Advanced Research Projects Agency* (DARPA), el *Army Research Office* (ARO), el *National Science Foundation* (NSF), y ESL, Inc (15).

Fue el pionero en muchos de los conceptos existentes en el sistema objeto-relacional actual, incluido, más tarde en otros sistemas de gestión comerciales. PostgreSQL es un sistema objeto-relacional, ya que incluye características de la orientación a objetos, como puede ser la herencia, tipos de datos, funciones, restricciones, disparadores, reglas e integridad transaccional.

Análisis de la selección del servidor de bases de datos.

Al finalizar la comparación se decide utilizar PostgreSQL como Sistema Gestor de Bases de Datos (SGBD), debido a su condición de ser multiplataforma y estar licenciado bajo GNU/GPL. Por otra parte, posee varias características que normalmente sólo se encuentran en homólogos comerciales (por ejemplo: Oracle), al mismo tiempo que (respecto a MySQL -su principal rival-) demuestra superioridad en cuanto a las necesidades del problema planteado. Dicha superioridad se manifiesta del siguiente modo: soporta una capacidad de almacenamiento en el orden de los TB (*Tera Bytes*), posee mayor escalabilidad (soporta una mayor cantidad de peticiones concurrentes), y tiene la capacidad de comprobar la integridad referencial (15).

1.5.4 Entorno Integrado de Desarrollo.

Un Entorno Integrado de Desarrollo (IDE, por sus siglas en inglés) es un programa informático compuesto por un conjunto de herramientas de programación que le facilitan al usuario editar, compilar, depurar códigos además de construir de manera rápida interfaces gráficas de usuarios. Puede ser usado para uno o varios lenguajes de programación.

Eclipse.

Eclipse es una plataforma de desarrollo de código abierto basada en Java. Es un desarrollo de IBM cuyo código fuente fue puesto a disposición de los usuarios. Presenta una estructura en forma de *plugins* que permite que los mismos puedan ser creados utilizando una API especial brindada por los desarrolladores y que puedan ser agregados fácilmente (sólo ubicarlos dentro de un directorio); gracias a esta facilidad posee una gama amplia y creciente de *plugins* desarrollados ya sea por empresas o por usuarios comunes (16).

Netbeans.

NetBeans es un proyecto de código abierto de gran éxito con una enorme base de usuarios, una comunidad en constante crecimiento. Es una aplicación de código abierto diseñada para el desarrollo de aplicaciones fácilmente portables entre las distintas plataformas, haciendo uso de la tecnología Java, dispone de soporte para crear interfaces gráficas de forma visual, desarrollo de aplicaciones web, control de versiones, colaboración entre varias personas, creación de aplicaciones compatibles con teléfonos móviles, resaltado de sintaxis y soporta la instalación de módulos para ampliar las funcionalidades del IDE (17).

Análisis de la selección del IDE.

Los autores seleccionan Netbeans como IDE para implementar la solución, teniendo en cuenta la experiencia anterior en el uso de la herramienta, además de considerar que la misma brinda un mejor soporte a las tecnologías seleccionadas para el desarrollo que Eclipse. Algunos de los principales elementos considerados fueron:

- Netbeans brinda (de forma local) las librerías necesarias para programar con el *framework* Spring, mientras que Eclipse requiere conexión a Internet para realizar la descarga de las dependencias.
- Netbeans se integra de forma natural con el servidor web Apache Tomcat, mientras que Eclipse requiere de varias configuraciones para garantizar la integración.
- Los autores consideran que la interfaz de Netbeans es más fácil de usar e intuitiva que la de Eclipse.

1.5.5 Servidor web.

Es un programa que gestiona cualquier aplicación en el lado del servidor realizando conexiones bidireccionales y/o unidireccionales y síncronas o asíncronas con el cliente generando una respuesta en cualquier lenguaje o aplicación en el lado del cliente.

El servidor Web se ejecuta en un ordenador manteniéndose a la espera de peticiones por parte de un cliente (un navegador Web) y que responde a estas peticiones, mediante una página web que se exhibirá en el navegador o mostrando el respectivo mensaje si se detectó algún error.

Apache Tomcat.

Apache Tomcat es desarrollado, en un entorno abierto y participativo y publicado bajo la licencia Apache versión 2, por miembros de la Apache *Software Foundation* y voluntarios independientes. El proyecto tiene la intención de ser una colaboración de los mejores desarrolladores de su clase de todo el mundo.

Tomcat puede funcionar como servidor web por sí mismo. Al principio de su desarrollo existió la percepción de que la utilización de Tomcat de forma autónoma era sólo recomendable para entornos de desarrollo y entornos con mínimos requisitos de velocidad y gestión de transacciones. Actualmente ya no existe esa percepción y Tomcat es usado como servidor Web independiente en entornos con alto nivel de tráfico y alta disponibilidad (18).

Glassfish.

Glassfish es un servidor de aplicaciones de software libre desarrollado por Sun Microsystems, compañía adquirida por Oracle Corporation, que implementa las tecnologías definidas en la plataforma Java EE y permite ejecutar aplicaciones que siguen esta especificación. La versión comercial es denominada Oracle Glassfish Enterprise Server (antes *Sun Glassfish Enterprise Server*). Es gratuito y de código libre, se distribuye bajo un licenciamiento dual a través de la licencia CDDL y la GNU-GPL (19).

Análisis de la selección del servidor web.

Los autores deciden utilizar Apache Tomcat en su versión 7, pues el mismo posee una licencia completamente libre, facilitando la obtención de nuevas versiones. Sin embargo, el licenciamiento dual que posee su principal competidor (Glassfish) condiciona el uso de la versión libre, la cual puede ser cerrada en cualquier momento (provocando la perdida de soporte y actualización del servidor web).

1.5.6 Metodologías de desarrollo de software.

Las metodologías de desarrollo de software son un conjunto de procedimientos y técnicas que permiten estructurar, planificar y controlar el proceso de desarrollo en sistemas de información. Se basan en los principios de uno o más modelos para guiar el proceso de desarrollo del software. Son las encargadas de detallar la información que se debe producir como resultado de una actividad durante el proceso de desarrollo y la información necesaria para comenzarla.

Para la selección de la metodología de desarrollo a utilizar fueron consideradas cuatro de las más utilizadas; estas son el Proceso Unificado de Desarrollo (RUP), la Programación Extrema (XP), Scrum (S) y Scrum - Programación Extrema (SXP).

RUP.

La metodología de desarrollo RUP se caracteriza por estar basada en componentes y utilizar el Lenguaje Unificado de Modelado (UML) para visualizar, especificar, construir y documentar un software.

El ciclo de vida de RUP se caracteriza por ser dirigido por casos de uso, siendo los casos de uso los encargados de iniciar el proceso de desarrollo y servir como guía para el desarrollo de las siguientes fases y de los artefactos que estas generan, centrado en la arquitectura ya que presta especial atención al establecimiento temprano de una buena organización y estructura, que le permite tener una visión común entre todos los involucrados (desarrolladores y usuarios), además de brindar una perspectiva clara del sistema completo, iterativo e incremental, lo que significa que en cada una de las fases (inicio, elaboración, desarrollo y transición) se ejecutaran una o varias iteraciones que comprenden actividades de todos los flujos de trabajo.

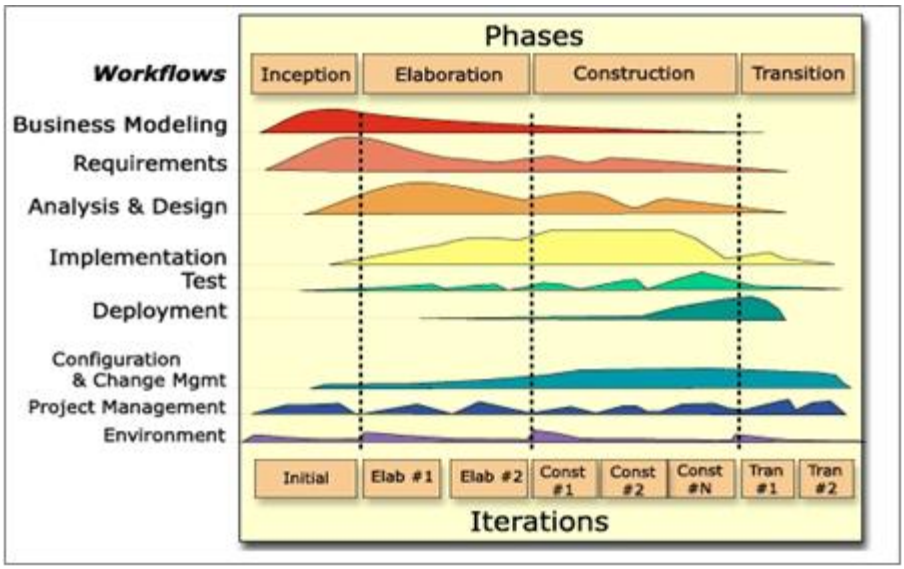


Figura 7. RUP en 2 dimensiones.

Consta de nueve flujos de trabajo fundamentales, de ellos seis denominados “de Ingeniería”, que son modelado del negocio, requerimientos, análisis y diseño, implementación, prueba y despliegue. Los restantes tres flujos de trabajo son considerados de apoyo y son la administración de proyectos, la gestión de configuración y cambios, y finalmente el ambiente (20).

La metodología RUP es más apropiada para proyectos grandes, dado que requiere un equipo de trabajo capaz de administrar un proceso complejo en varias etapas. En proyectos pequeños, es posible que no se puedan cubrir los costos de dedicación del equipo de profesionales necesarios.

XP.

La Programación Extrema es una metodología ágil concebida e implementada para dirigir las necesidades específicas del desarrollo de software conducido por equipos pequeños. Esta le da poder a los desarrolladores para responder con confianza a los requerimientos cambiantes del consumidor final. Se caracteriza además por fomentar la comunicación desarrollador-cliente desde el primer día. Es considerada ligera, flexible, predecible, de bajo riesgo, y no por ello menos científica.

Otras ventajas que no pueden pasar por alto son los pocos requerimientos de documentación y planificación siendo las historias de usuarios los principales artefactos que se generan, así como la exigencia de tener siempre el cliente disponible para el desarrollo, implicando una mejor correspondencia entre el producto y la necesidad del negocio (20).

Scrum.

Scrum es un marco de trabajo para la gestión y desarrollo de software basada en un proceso iterativo e incremental utilizado comúnmente en entornos basados en el desarrollo ágil de software. Está especialmente indicada para proyectos con un rápido cambio de requisitos. Sus principales características se pueden resumir en dos. El desarrollo de software se realiza mediante iteraciones, denominadas “sprints”, con una duración de 1 a 4 semanas. El resultado de cada “sprint” es un incremento ejecutable que se muestra al cliente. La segunda característica importante son las reuniones a lo largo del proyecto. Éstas son las verdaderas protagonistas, especialmente la reunión diaria de 15 minutos del equipo de desarrollo para coordinación e integración (20).

SXP.

SXP es una metodología compuesta por las metodologías SCRUM y XP, la cual ofrece una estrategia tecnológica a partir de la introducción de procedimientos ágiles, que permitan actualizar los procesos de software para el mejoramiento de la actividad productiva, fomentando el desarrollo de la creatividad, aumentando el nivel de preocupación y responsabilidad de los miembros del equipo y ayudando al líder del proyecto a tener un mejor control del mismo.

Consta de 4 fases principales:

- Planificación: Definición donde se establece la visión, se fijan las expectativas y se realiza el aseguramiento del financiamiento del proyecto.
- Desarrollo: Es donde se realiza la implementación del sistema hasta que esté listo para ser entregado.
- Entrega.
- Mantenimiento: Donde se realiza el soporte para el cliente.

De cada una de estas fases se realizan numerosas actividades tales como el levantamiento de requisitos, la priorización de la Lista de Reserva del Producto, definición de las Historias de Usuario, diseño, implementación, pruebas, entre otras; de donde se generan artefactos para documentar todo el proceso. Las entregas son frecuentes, y existe una refactorización continua, lo que permite mejorar el diseño cada vez que se le añada una nueva funcionalidad (21).

SXP propone los siguientes roles para el trabajo en equipo:

- Líder del proyecto (Scrum Master).
- Gerente (Management).
- Especialista.
- Cliente (Customer).
- Consultor.
- Equipo del proyecto (ScrumTeam).

El equipo del proyecto será conformado por otros roles como:

- Programadores (Programmers).
- Analista (Analyst).
- Diseñadores (Designers).
- Encargado de prueba (Tester).
- Arquitecto (Architect).

Análisis de la selección de la metodología de desarrollo.

Se selecciona SXP como metodología de desarrollo, ya que se destina especialmente para pequeños equipos de trabajo con requisitos imprecisos o cambiantes, donde existe un alto riesgo técnico. Asimismo, se orienta a una entrega rápida de resultados y posee alta flexibilidad. Además, propicia el trabajo en equipo con objetivos bien definidos, permitiendo seguir de manera clara el avance de las tareas. En definitiva se puede afirmar que, con el uso de la metodología SXP y las dimensiones del proyecto, este podrá implementarse exitosamente.

1.6 Conclusiones.

Luego de analizar las principales deficiencias existentes en la versión 1 del COJ, se evidencia la necesidad de implementar la versión 2 de dicho sistema, obviando completamente el código fuente anterior. Para ello, primeramente se tendrán en cuenta las características de la versión 1 que serán incluidas en la versión 2 y luego los resultados del estudio realizado acerca de los principales jueces en línea en el ámbito internacional, nacional y local. Por otra parte, después de un estudio realizado sobre las principales tecnologías a emplear en el desarrollo de la solución, se decide utilizar Java (como lenguaje de programación), Netbeans (como IDE de desarrollo), Spring (como *framework* para el desarrollo de la capa de presentación) y SXP (como metodología para guiar el proceso de desarrollo); ya que dichas tecnologías son las que más se ajustan para garantizar el cumplimiento de los objetivos trazados.

Una vez finalizada esta etapa, las condiciones están creadas para realizar el Análisis y Diseño de la solución propuesta.

2. CAPÍTULO 2. DESCRIPCIÓN DE SISTEMA.

2.1 Introducción.

En el presente capítulo se recogen las funcionalidades de la solución propuesta, reflejadas en las Historias de Usuarios con sus prototipos de interfaces de usuario no funcionales, así como la planificación de las iteraciones del ciclo de desarrollo. Además se realiza el diseño de la solución propuesta y se describen los patrones a emplear en el desarrollo.

2.2 Descripción de la solución.

La versión 2 del COJ estará conformada por dos capas fundamentales: la capa de presentación y la capa de evaluación. La capa de presentación es una aplicación web desarrollada en java (usando el *framework* “Spring”) y servirá como interfaz del sistema, mediante la cual los usuarios harán uso de los servicios que se presten en la plataforma. Los servicios contemplados en la presente versión son los mismos incluidos en la versión 1, algunos de los cuales fueron modificados o re-implementados. Además se incluyen nuevas funcionalidades, tales como:

- Concursos virtuales.
- Soporte para nuevos estilos de concursos (concursos progresivos y concursos libres).
- Módulo de administración completo (permite gestionar todos los recursos del sistema desde la interfaz).
- Evaluación distribuida.
- Calificación especial.
- Otras funcionalidades que facilitan el uso del sistema por parte de usuarios y administradores.

Por su parte, la capa de evaluación es la encargada de calificar/evaluar las soluciones enviadas por los usuarios, además de realizar las tareas que representan una carga en el consumo de recursos del servidor; dicha capa está conformada por dos aplicaciones, ambas implementadas usando el paradigma de programación distribuida:

Motor de Evaluación: Es la aplicación encargada de la evaluación de las soluciones propuestas por los usuarios.

Balanceador/Distribuidor de Tareas: Es el encargado de la realización de las tareas que representan un consumo excesivo de recursos en el servidor, la actualización de la base de datos, la recalificación, el envío de las soluciones al motor de evaluación, la ejecución de tareas paralelas al funcionamiento del Juez en Línea (Recalificación en lote de soluciones, recalificación de concursos y bloqueo de concursos), entre otras.

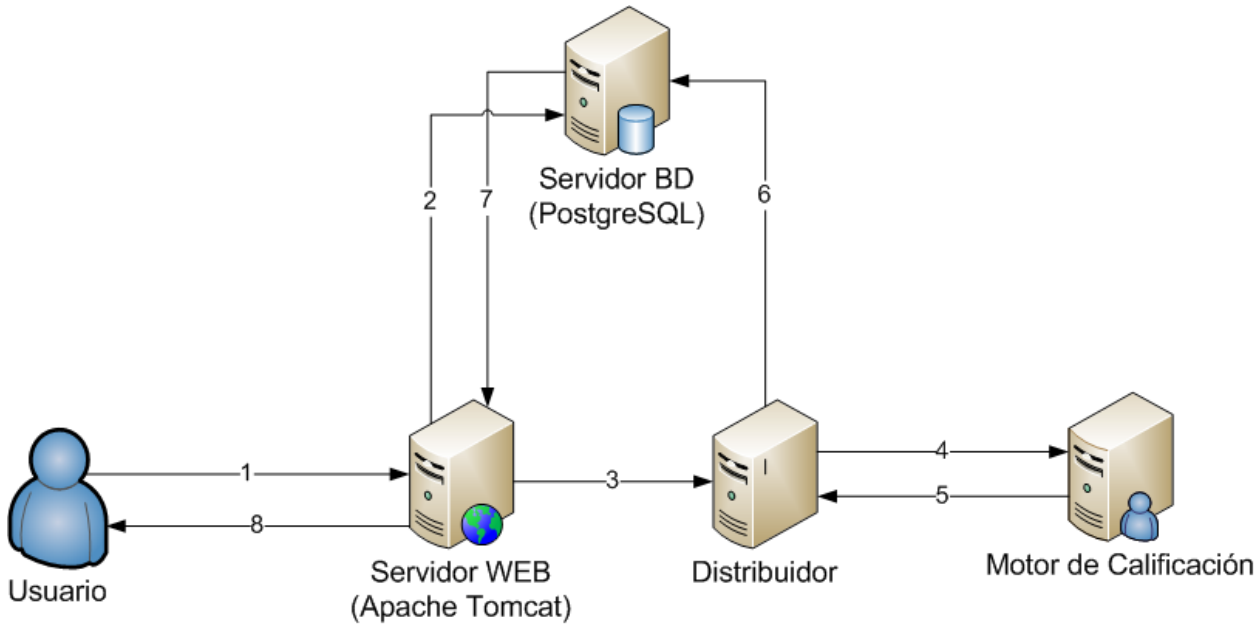


Figura 8. Funcionamiento del sistema.

1. El usuario realiza una acción sobre la interfaz web.
2. Si la acción lo requiere, se consulta la base de datos.
3. En caso de que la acción constituya una solicitud de evaluación, la misma es enviada al distribuidor para ser procesada.
4. Para ser evaluada, la información del programa es enviada a algún motor de evaluación que esté disponible.

5. Una vez culminada la evaluación, el motor envía el resultado al distribuidor.
6. El distribuidor actualiza la información del envío en la base de datos.
7. El sistema actualiza la interfaz con la información modificada.
8. El usuario consulta el resultado de la calificación.

Una vez descritas las características deseadas en el sistema se procede al levantamiento de los requisitos, proceso que pertenece al flujo de trabajo de "Planificación" (donde se genera el artefacto "Lista de Reserva del Producto").

2.3 Lista de Reserva del Producto.

La lista de reserva del producto es una colección organizada y priorizada de los requisitos sobre el producto. El objetivo de la misma radica en cubrir las cualidades requeridas en el software y determinar el orden en que se cumplimentarán durante las **iteraciones o *sprint*[27]**; además, incluye las características que el producto debe tener. A continuación se recogen, en cada una de las filas de la tabla mostrada, los elementos que describen cada requisito identificado. Se recoge el nombre del desarrollador que dará cumplimiento al requisito, el **número identificativo del requisito**[28], la descripción del requisito, el tiempo estimado para su cumplimiento y el nombre del estimador (21).

Asignado a	**Item***	**Descripción**	**Estimación**	**Estimado por**
Prioridad			**Muy Alta**	
Jorge Luis	01	Registrar Usuario	1 día	Jorge Luis
Jorge Luis	02	Modificar Usuario	1 día	Jorge Luis
Jorge Luis	03	Listar Usuario	1 día	Jorge Luis
Juan Carlos	04	Insertar Problema	1 día	Juan Carlos

[27] Ciclo iterativo o período de tiempo en el que se implementa o mejora una funcionalidad del sistema. Durante un *sprint* el producto puede ser diseñado, implementado y probado para producir nuevos entregables.
[28] Notación HU.R.I: significa la historia de usuario **HU** que dará cumplimiento al requisito **R** en la iteración o sprint **I**.

Juan Carlos	05	Modificar Problema	1 día	Juan Carlos
Juan Carlos	06	Listar Problemas	1 día	Juan Carlos
Jorge Luis	07	Insertar Concurso	2 días	Jorge Luis
Jorge Luis	08	Modificar Concurso	1 Semana	Jorge Luis
Jorge Luis	09	Listar Concursos	1 día	Jorge Luis
Juan Carlos, Jorge Luis	10	Realizar Concurso	6 Semanas	Juan Carlos, Jorge Luis
Juan Carlos, Jorge Luis	11	Modificar estado del Concurso	1 Semana	Juan Carlos, Jorge Luis
Juan Carlos	12	Insertar Solución	2 días	Juan Carlos
Juan Carlos	13	Modificar Solución	2 días	Juan Carlos
Juan Carlos	14	Listar Soluciones	1 día	Juan Carlos
Juan Carlos, Jorge Luis	15	Evaluar Solución	3 Semanas	Juan Carlos, Jorge Luis
Juan Carlos, Jorge Luis	16	Modificar resultado de la Solución	1 Semana	Juan Carlos, Jorge Luis
Prioridad			**Alta**	
Jorge Luis, Juan Carlos	17	Crear Concurso Virtual	3 días	Juan Carlos, Jorge Luis

Jorge Luis, Juan Carlos	18	Eliminar Concurso Virtual	1día	Juan Carlos, Jorge Luis
Jorge Luis, Juan Carlos	19	Listar Concurso Virtual	1día	Juan Carlos, Jorge Luis
Jorge Luis, Juan Carlos	20	Realizar Concurso Virtual	3 Semanas	Juan Carlos, Jorge Luis
Prioridad			**Media**	
Juan Carlos	21	Mostrar Estadísticas de las soluciones	1día	Juan Carlos
Juan Carlos	22	Mostrar Estadísticas de un concurso	1día	Juan Carlos
Juan Carlos	23	Mostrar estadísticas globales de los concursos	1día	Juan Carlos
Juan Carlos	24	Mostrar estadísticas globales de los concursos virtuales	1día	Juan Carlos
Jorge Luis	25	Enviar mensaje privado	1día	Jorge Luis
Jorge Luis	26	Eliminar mensajes privados	1día	Jorge Luis
Jorge Luis	27	Listar mensajes privados	1día	Jorge Luis
Juan Carlos	28	Comparar Usuarios	3 días	Juan Carlos
Prioridad			**Baja**	

Juan Carlos	29	Mostrar páginas de información	1día	Juan Carlos
Juan Carlos	30	Insertar Anuncios	1día	Juan Carlos
Juan Carlos	31	Listar Anuncios	1día	Juan Carlos
Juan Carlos	32	Modificar Anuncios	1día	Juan Carlos
Juan Carlos	33	Eliminar Anuncios	1día	Juan Carlos
Jorge Luis	34	Insertar Institución	1día	Jorge Luis
Jorge Luis	35	Modificar Institución	1día	Jorge Luis
Jorge Luis	36	Listar Instituciones	1día	Jorge Luis
Jorge Luis	37	Insertar País	1día	Jorge Luis
Jorge Luis	38	Modificar País	1día	Jorge Luis
Jorge Luis	39	Listar Países	1día	Jorge Luis
Juan Carlos	40	Insertar Lenguaje de Programación	1día	Juan Carlos
Juan Carlos	41	Modificar Lenguaje de programación	1día	Juan Carlos
Juan Carlos	42	Listar Lenguajes de Programación	1día	Juan Carlos
Requisitos No Funcionales				
Apariencia e interfaz externa				

	1	Optimizado para Firefox 7.x o superior.		
	2	Soporte para pantallas widescreen y pantallas estándares		
Usabilidad				
	3	Resaltado de los enlaces con subrayado y color azul		
	4	Arquitectura de la información distribuida en el panel izquierdo garantizando el acceso a toda la información del sitio con no más de 3 clicks.		
	5	Todos los contenidos serán mostrados en el panel derecho.		
Rendimiento				
	6	Cada página del sitio debe poseer un tiempo de respuesta rápido.		
	7	Soportar hasta 300 usuarios concurrentes con los requisitos mínimos de hardware.		
Seguridad				
	8	Gestión de los roles de usuarios para acceder a los recursos del		

		sistema		
	9	Encriptar la contraseña de los usuarios mediante un proceso irreversible		
	10	Las soluciones de los usuarios deben ser probadas en un entorno seguro		
Software				
	11	Utilizar PostgreSQL como SGBD en su versión 8.4 o superior.		
	12	Utilizar apache Tomcat 7.x o superior como servidor web		
	13	Utilizar el entorno de ejecución java “OpenJDK-JRE” en su versión 6 o superior		
Hardware				
	14	Requerimientos mínimos de hardware: Core 2 duo 2.0 GHZ y 2GB RAM		

Tabla 1. Lista de Reserva del Producto.

2.4 Historias de usuario.

Las historias de usuario son la técnica utilizada en XP para especificar los requisitos del software, lo que equivale a los casos de uso en el proceso unificado. Las mismas son escritas por los clientes como las tareas que el sistema debe hacer y su construcción depende principalmente de la habilidad que tenga el cliente para definirlas. Son escritas en lenguaje natural, no excediendo su tamaño de unas pocas líneas de texto.

Las historias de usuario guían la construcción de las pruebas de aceptación, elemento clave en XP (deben generarse una o más pruebas para verificar que la historia ha sido correctamente implementada) y son utilizadas para estimar tiempos de desarrollo. En este sentido, sólo proveen detalles suficientes para hacer una estimación razonable del tiempo que llevará implementarlas. En el momento de implementar una historia de usuario, se debe detallar a través de la comunicación con el cliente. Estas son la base para las pruebas funcionales (21).

Historia de Usuario	
Numero: 01	Nombre Historia de Usuario: Registrar Usuario
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Jorge Luis Roque Alvarez	Iteración Asignada:1
Prioridad en Negocio: Muy Alta	Puntos Estimados:1 día
Riesgo en Desarrollo: Medio	Puntos Reales:1 día
Descripción: la historia de usuario permite el registro de un usuario en el sistema. Para el registro de un usuario se necesitan los siguientes datos: usuario, apodo, nombre(s), apellido(s), género, fecha de nacimiento, país, institución, correo-e, idioma para la interfaz , lenguaje de programación, contraseña, pregunta de seguridad, respuesta de seguridad. Una vez registrado el usuario se le envía un correo de activación de la cuenta creada y se redirecciona hacia una página con un mensaje informativo “Por favor, espere unos minutos y compruebe su correo-e para la Validación de la Cuenta de Usuario”.	

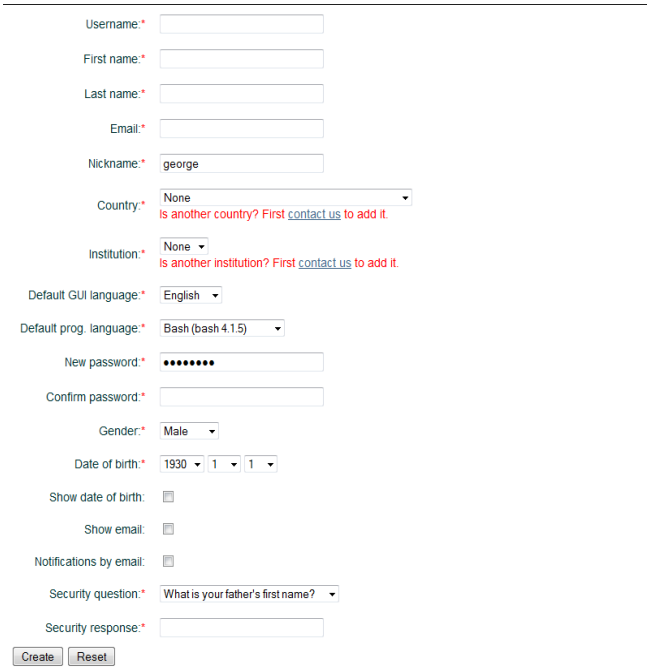
Observaciones:

Los campos nombre de usuario, apodo, nombres, apellidos y respuesta de seguridad no pueden ser vacíos.

Los campos de selección son obligatorios:

- País.
- Género.
- Institución.
- Lenguaje de la interfaz.
- Lenguaje de programación.

Prototipo de Interfaz:



IU 1. Registrar Usuario.

Tabla 2. HU - Registrar Usuario.

<table>
<tr><td colspan="2">Historia de Usuario</td></tr>
<tr><td>Numero: 02</td><td>Nombre Historia de Usuario: Insertar Concurso</td></tr>
<tr><td colspan="2">Modificación de Historia de Usuario Número: Ninguna</td></tr>
<tr><td>Usuario: Jorge Luis Roque Alvarez</td><td>Iteración Asignada:1</td></tr>
<tr><td>Prioridad en Negocio: Muy Alta</td><td>Puntos Estimados:1 día</td></tr>
<tr><td>Riesgo en Desarrollo: Alto</td><td>Puntos Reales:1 día</td></tr>
<tr><td colspan="2">Descripción: Permite la creación de un concurso en el sistema, para ello es necesario insertar los siguientes campos: id, import_data e id_concurso. Luego de insertado se redirecciona hacia la sección de administración del concurso creado.

Rol: Administrador, Contest_Setter.</td></tr>
<tr><td colspan="2">Observaciones:

• El valor del campo id debe ser entero y no puede estar en uso por otro concurso registrado en el sistema.
• Si el campo import_data es seleccionado se requiere la selección de un concurso registrado en el sistema para ser utilizado.</td></tr>
<tr><td colspan="2">Prototipo de Interfaz:</td></tr>
</table>

IU 2.Insertar Concurso.

Tabla 3. HU - Insertar Concurso.

Historia de Usuario	
Numero: 03	Nombre Historia de Usuario: Modificar Concurso
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Jorge Luis Roque Alvarez	Iteración Asignada:1
Prioridad en Negocio: Muy Alta	Puntos Estimados: 1 Semana
Riesgo en Desarrollo: Alto	Puntos Reales: 1 Semana
Descripción: Permite modificar todas las variables de un concurso separadas por secciones. Configuraciones Globales Nombre del concurso, tipo de registro, fecha límite de registro, fecha de inicio del concurso, fecha de fin del concurso, estilo de concurso, tipo de usuario, habilitado, plantilla de concurso virtual y bloqueado.	

Variables Globales

Penalidad, tiempo congelado, tiempo muerto, tiempo de descongelado, cantidad límite de registros, mostrar el estado de los envíos, mostrar el estado de los envíos al público, mostrar la tabla de posiciones, mostrar la tabla de posiciones al público, mostrar los problemas, mostrar los casos de prueba.

Problemas

Debe permitir seleccionar y añadir los problemas al concurso.

Lenguajes de programación

Debe permitir seleccionar y añadir los lenguajes de programación al concurso.

Usuarios

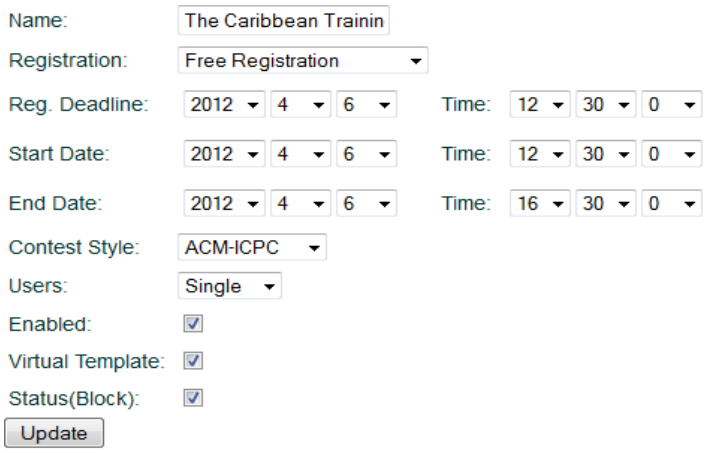
Grupo, jueces, usuarios

Rol: Administrador, Contest_Setter

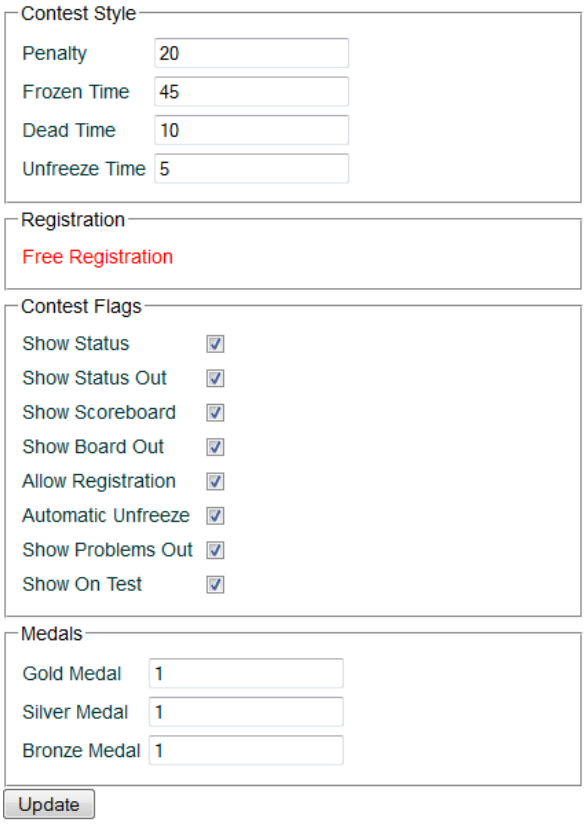
Observaciones:

- La fecha de inicio del concurso debe ser mayor que la fecha límite de registro.
- La fecha fin del concurso debe ser mayor que la fecha fin.
- Si el estilo del concurso es ACM-ICPC la duración del mismo no puede ser mayor de 8 horas.
- EL concurso debe tener al menos un lenguaje de programación asignado

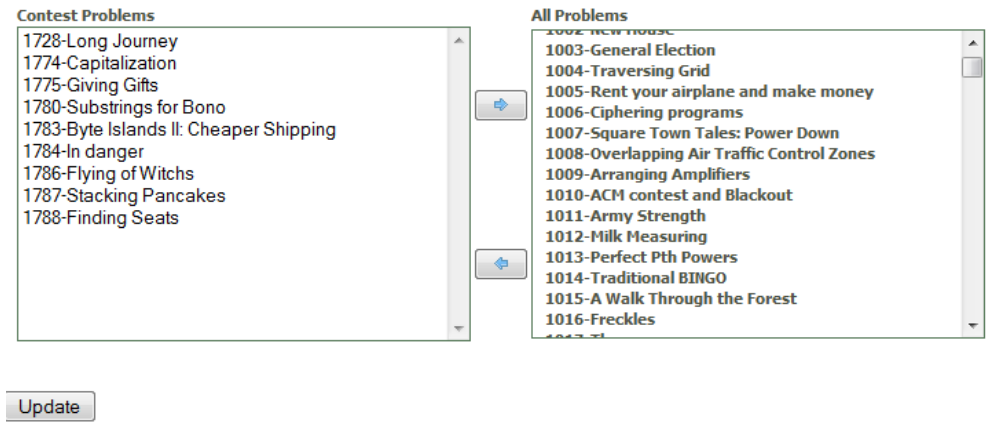
Prototipo de Interfaz:



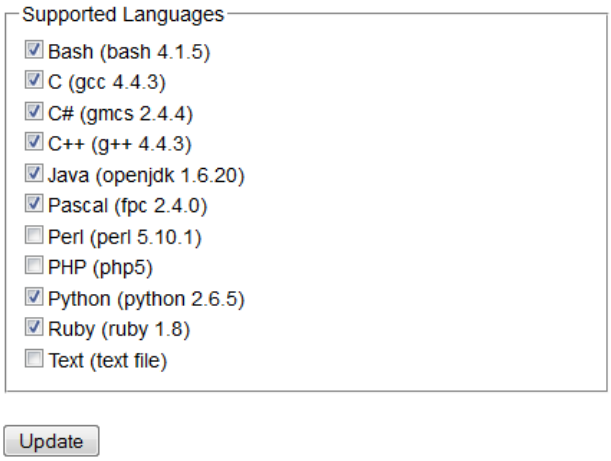
IU 3.Configuraciones Globales.



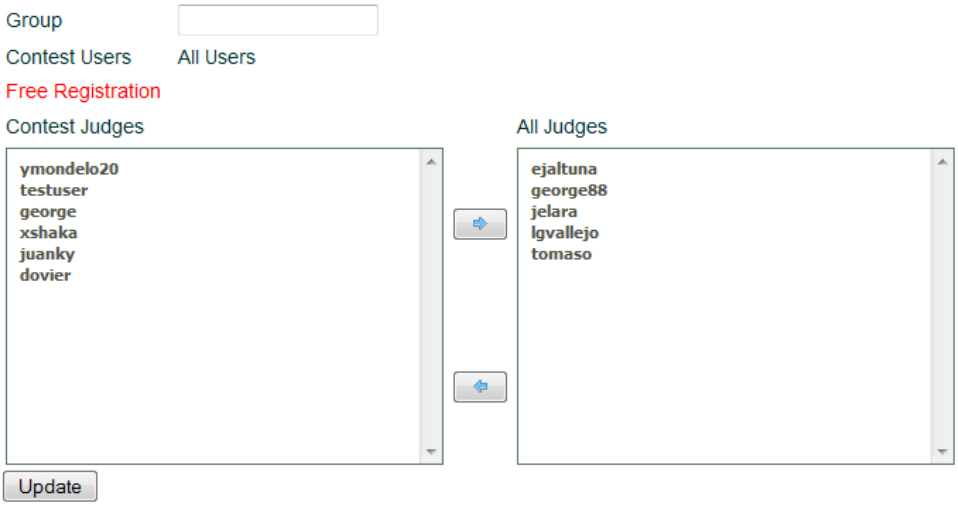
IU 4. Variables Globales.



IU 5.Problemas de un concurso.



IU 6.Lenguajes de Programación.



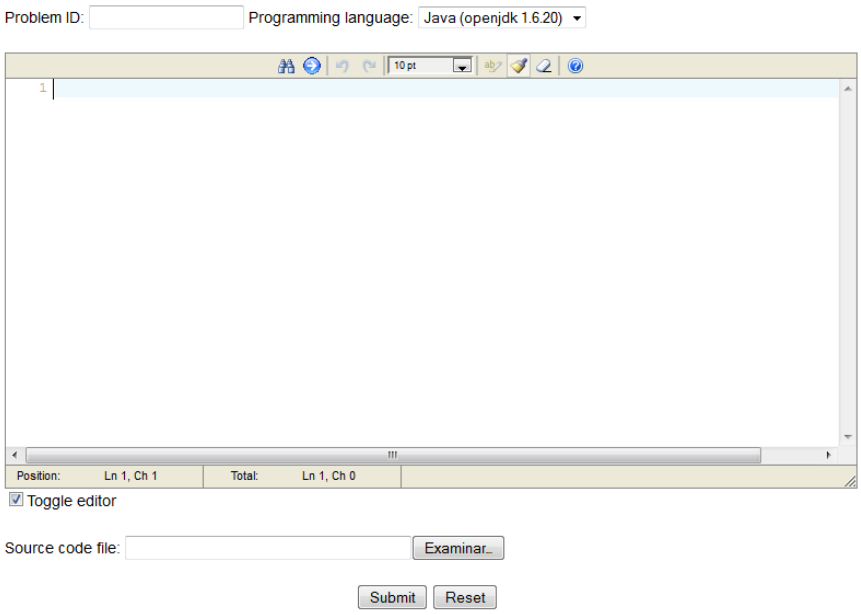
IU 7.Usuarios del Concurso.

Tabla 4. HU - Modificar Concurso.

Historia de Usuario	
Numero: 04	Nombre Historia de Usuario: Insertar Solución
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Juan Carlos Lobaina Guzmán	Iteración Asignada: 1
Prioridad en Negocio: Muy Alta	Puntos Estimados: 2 días
Riesgo en Desarrollo: Alto	Puntos Reales: 2 días
Descripción: Permite a los usuarios la inserción de una solución al sistema. Los campos requeridos son: id del problema, lenguaje de programación, código fuente y fichero. Rol: todos los roles del sistema.	
Observaciones:	

- El id del problema debe ser un valor entero y estar asignado a un problema que se encuentre habilitado en el sistema.
- EL código fuente no debe exceder los 100kb de tamaño.
- El fichero no debe exceder los 100kb de tamaño.

Prototipo de Interfaz:



IU 8. Insertar Solución.

Tabla 5. HU - Insertar Solución.

Historia de Usuario	
Numero: 05	Nombre Historia de Usuario: Evaluar Solución
Modificación de Historia de Usuario Número: Ninguna	
Usuario: Juan Carlos Lobaina Guzmán, Jorge Luis Roque Alvarez	Iteración Asignada:1

Prioridad en Negocio: Muy Alta	Puntos Estimados: 3 Semanas
Riesgo en Desarrollo: Alto	Puntos Reales: 3 Semanas
Descripción: El motor de evaluación recibe una petición de evaluación del balanceador de carga. Se crea el fichero con el código fuente, se compila y ejecuta en caso de ser necesario y se evalúa el estado de la solución y se le envía la respuesta al balanceador. Rol: Sistema.	
Observaciones: • Debe existir al menos un juego de datos completo del problema a evaluar. • Debe estar instalado un compilador para el lenguaje de programación asignado a la solución. • Si se requiere de calificación especial debe existir el evaluador externo.	
Prototipo de Interfaz:	

Tabla 6. HU - Evaluar Solución.

2.5 Diagrama de Paquetes.

Un diagrama de paquetes muestra cómo un sistema está dividido en agrupaciones lógicas y las relaciones entre las mismas. Dado que normalmente un paquete está pensado como un directorio, los diagramas de paquetes suministran una visión preliminar de la estructura de un sistema. Cada paquete puede asignarse a un individuo o a un equipo, y las dependencias entre ellos pueden indicar el orden de desarrollo requerido (22).

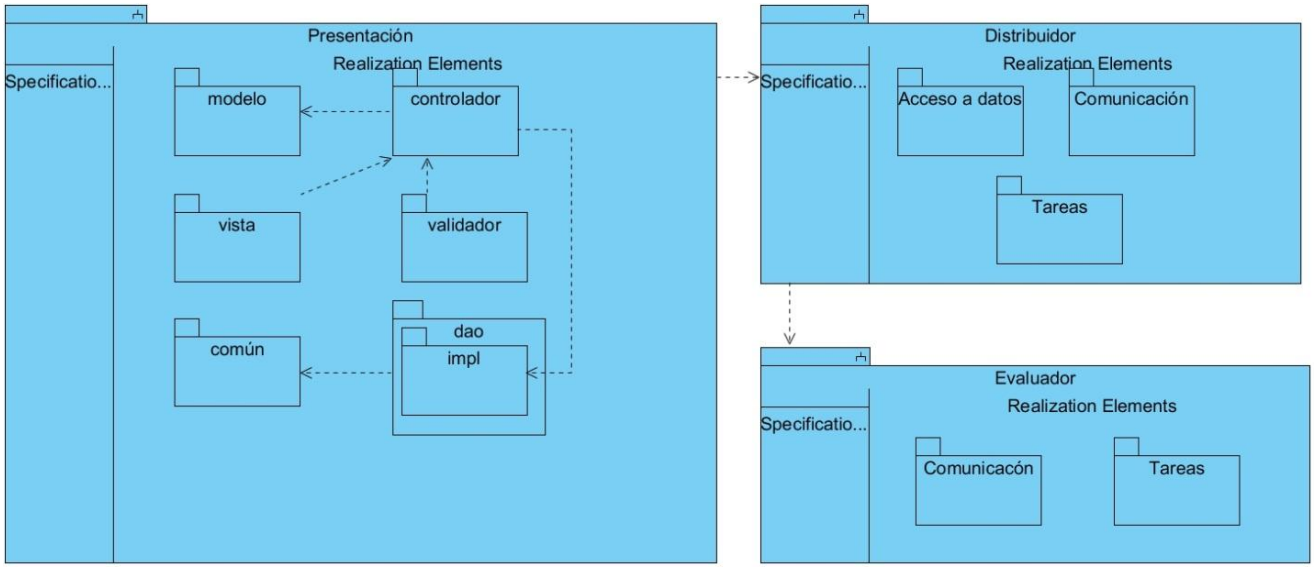


Figura 9. Diagrama de paquetes.

2.5.1 Descripción de los Paquetes.

El diagrama anterior representa la distribución física que conformará la solución donde se destacan los 3 paquetes que la conforman:

Presentación: Representa la aplicación web mediante la cual los usuarios podrán interactuar con el sistema así como su relación con el paquete “Distribuidor”. Los paquetes principales que la componen son:

- Modelo: Agrupa las clases o modelos que representan las entidades del negocio.
- Vista: Mantiene la información actualizada para ser consultada por los usuarios.
- Controlador: Contiene las clases encargadas de gestionar las acciones realizadas por los usuarios sobre las vistas.
- Validador: Almacena las clases encargadas de validar las acciones de los usuarios que requieran modificaciones o inserciones de modelos.
- Dao: Contiene las interfaces que permiten a los controladores y validadores el acceso a los datos persistentes en la Base de Datos.

- Impl: Está contenido en el Paquete Dao y contiene las implementaciones de las interfaces del mismo.
- Común: Contiene clases de uso común en el sistema.

Distribuidor: Representa el subsistema de distribución que sirve como puente entre el subsistema de evaluación y la capa de presentación por lo cual se encuentra relacionado con los otros dos paquetes principales. Los paquetes que lo componen son:

- Acceso a Datos: Contiene las clases que le permiten la comunicación con la Base de Datos.
- Tareas: Almacena un conjunto de clases que representan las tareas que se ejecutan en el subsistema.
- Comunicación: Agrupa las clases que permiten la comunicación con el subsistema de *Presentación* y el subsistema de *Evaluación*.

Evaluador: Constituye la representación del subsistema de *Evaluación* donde se califican las soluciones enviadas por los usuarios desde la interfaz. Dicho subsistema solo se comunica con el subsistema de *Distribución*; los paquetes principales que lo conforman son:

- Comunicación: Contiene las clases que permiten la comunicación con el subsistema de *Distribución*.
- Tareas: Almacena un conjunto de clases que representan las tareas que se ejecutan en el subsistema.

2.6 Diagrama de clases del diseño.

El diagrama de clases del diseño no es más que la representación gráfica de las clases que serán implementadas en el sistema. Este diagrama muestra las especificaciones y detalles más concretos de cada clase del sistema, así como su relación de asociación, composición o agregación existentes entre ellas (23).

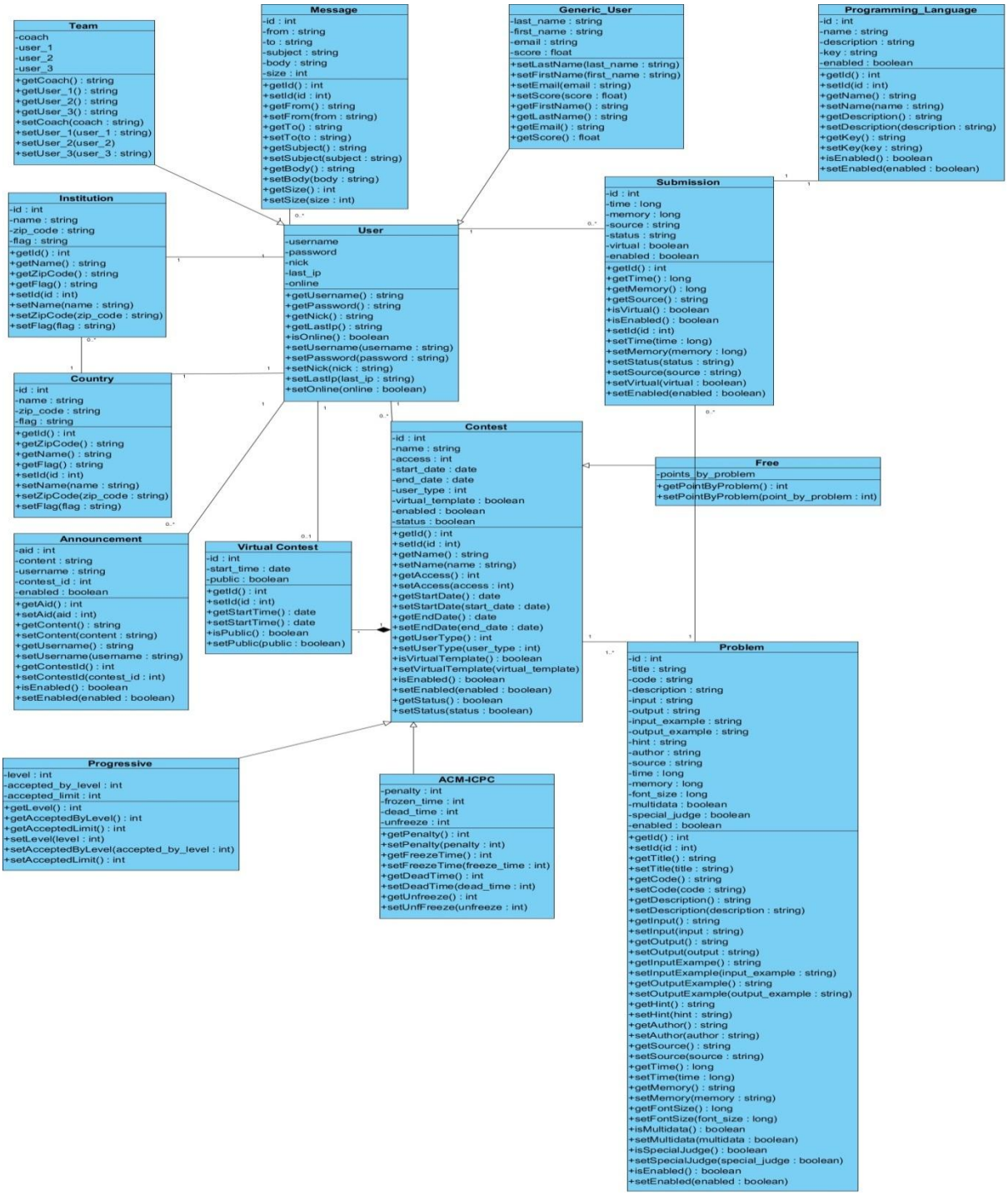


Figura 10. Diagramas de clases.

En el diagrama anterior destacan las clases “User”, “Contest”, “Problem” y “Submission” debido a que constituyen las clases que poseerán mayor impacto en el desarrollo, además de estar inter-relacionadas.

Nombre: User	
Tipo de clase Entidad	
Atributo	**Tipo**
Username	String
Password	String
Nick	String
last_ip	String
Online	Boolean
Para cada responsabilidad:	
Métodos de Acceso.	

Tabla 7. Descripción de la Clase User.

Nombre: Submission	
Tipo de clase Entidad	
Atributo	**Tipo**
submit_id	Int
Time	Long
Memory	Long
Source	long
Status	String
Virtual	boolean
Enabled	boolean
Para cada responsabilidad:	
Métodos de Acceso.	

Tabla 8. Descripción de la Clase Submission.

Nombre: Problem	
Tipo de clase Entidad	
Atributo	**Tipo**
id	Int
Title	String
Code	String
Description	String
Input	String
Output	String
Input_Example	String
Output_Example	String
Hint	String
Author	String
Source	String

Time	Long
Memory	Long
Fontsize	Long
Multidata	boolean
special_judge	boolean
Enabled	boolean
Para cada responsabilidad:	
Métodos de Acceso.	

Tabla 9. Descripción de la Clase Problem.

Nombre: Contest	
Tipo de clase Entidad	
Atributo	**Tipo**
cid	Int
name	String
Access	Int
start_date	Date
end_date	date
user_type	Int
virtual_template	Boolean
Enabled	Boolean
Status	boolean
Para cada responsabilidad:	
Métodos de Acceso.	

Tabla 10. Descripción de la clase Contest.

Nombre: Virtual Contest	
Tipo de clase Entidad	
Atributo	**Tipo**
cid	Int
Id	Int
Access	Int
start_date	Date
Para cada responsabilidad:	
Métodos de Acceso.	

Tabla 11. Descripción de la Clase Virtual Contest.

2.7 Modelo de Datos.

Un modelo de datos es un conjunto de herramientas conceptuales para describir datos, sus relaciones, su significado y sus restricciones de consistencia (24).

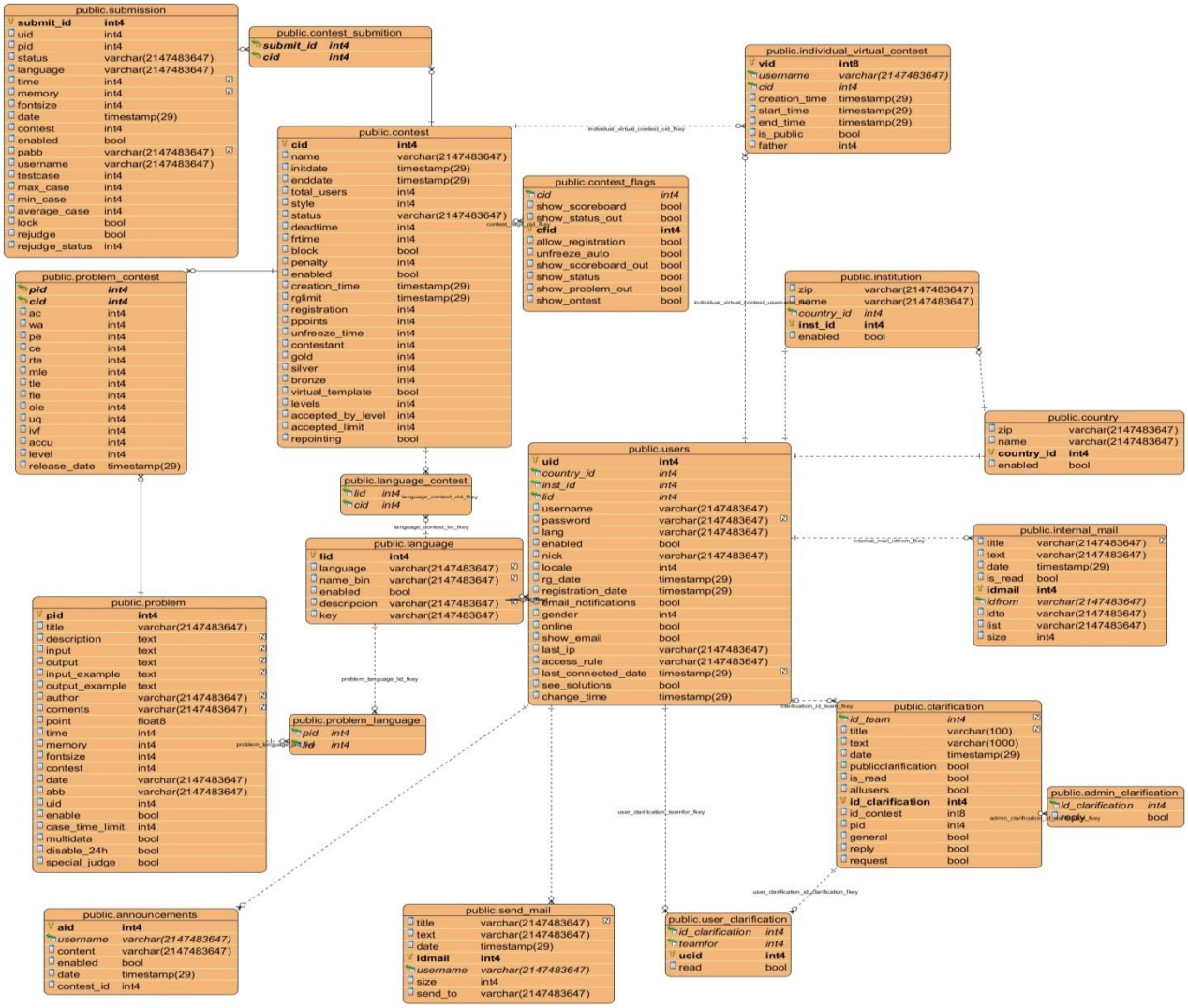


Figura 11. Modelo de Datos.

Nombre: submission		
Descripción: Almacena todos los datos relacionados con los envíos realizados por los usuarios		
Atributo	**Tipo**	**Descripción**
submit_id	Serial	Identificador de cada envío
Uid	Int	Id del usuario que realizó el envío
Pid	Int	Id del problema asociado al envío
Status	String	Estado del envío
Language	Int	Id del lenguaje de programación en el que fue realizado el

		envío
Time	Int	Tiempo de ejecución del envío.
Memory	Long	Memoria consumida en la ejecución
Fontsize	Long	Tamaño del código fuente
date	Date	Fecha de realización del envío
Contest	Int	Id del contest en el cual fue realizado o 0 si fue un envío realizado en el módulo de 24h
Enabled	Boolean	Determina si el envío está activado o no
test_case	Int	Número de pruebas realizadas
max_case	Int	Máximo tiempo de ejecución consumido en una prueba
Min_case	Int	Mínimo tiempo de ejecución consumido en una prueba
average_case	Double	Promedio de tiempo de ejecución del envío

Tabla 12. Descripción de la Entidad submision.

Nombre: contest		
Descripción: Almacena los datos relacionados con los concursos		
Atributo	**Tipo**	**Descripción**
cid	Serial	Identificador de cada concurso
Name	String	Nombre del concurso
Initdate	Date	Fecha de inicio del concurso
Enddate	Date	Fecha de fin del concurso
total_users	Int	límite de usuarios para un concurso
Deadtime	Int	Tiempo muerto del concurso
Frozentime	Int	Tiempo congelado de un concurso
Fontsize	Long	Tamaño del código fuente
Block	Boolean	Bloquea el concurso
Penalty	Int	Penalización por cada envío fallido en un concurso ACM-ICPC
Enabled	Boolean	Determina si el concurso está activado o no
creation_time	Date	Hora de creación de un concurso
rg_limit	Date	Fecha límite de registro en un concurso
Registration	Int	Tipo de registro
Ppoints	Int	Puntos asignados a un problema en concursos “Progressive” o “Free”
Unfreeze_time	Int	Representa el tiempo de desbloqueo de un concurso después de finalizar
Contestant	Int	Representa el tipo de concursante admitidos en el concurso
GOLD	INT	Cantidad de medallas de oro otorgadas en el concurso
silver	INT	Cantidad de medallas de plata otorgadas en el concurso
bronze	INT	Cantidad de medallas de bronce otorgadas en el concurso
virtual_template	Boolean	Si el concurso está activo o no para ser usado virtualmente
Levels	Int	Cantidad de niveles para un concurso “Progressive”
accepted_by_levels	Int	Límite de aceptados por niveles para un concurso

		"progressive"
accepted_limit	Int	Límite de aceptados para un concurso "progressive"

Tabla 13. Descripción de la Entidad Contest.

Nombre: problem		
Descripción: Es la tabla donde se almacenan los datos de cada problema		
Atributo	**Tipo**	**Descripción**
Pid	Int	Representa el id del problema
Title	String	Título del problema
Description	String	Descripción del problema
Input	String	Descripción de la entrada del problema
Output	String	Descripción de la salida del problema
Input_example	String	Ejemplo de entrada del problema
output_example	String	Ejemplo de salida del problema
Author	String	Referencia al autor y la fuente del problema
Hint	String	Comentario o recomendación del autor del problema.
Time	Int	Tiempo límite total de ejecución del problema
Memory	Long	Memoria límite de ejecución del problema
Fontsize	Long	Límite de fuente para el código fuente del problema
Date	Date	Fecha de adición del problema al juez en línea
Abb	String	Código del problema
Uid	Int	Representa el id del usuario que adicionó el problema
Enabled	Boolean	Representa si el problema está activo o no
case_time_limit	Int	Tiempo límite de ejecución para un caso de prueba
multi_data	Boolean	Representa si el problema tiene múltiples casos de prueba o no
special_judge	Boolean	Representa si el problema requiere calificación especial o no
disable_24h	Boolean	Representa si el problema se encuentra activo para el módulo de 24h

Tabla 14. Descripción de la Entidad problem.

Nombre: users		
Descripción: Almacena los datos comunes entre los diferentes usuarios del sistema		
Atributo	**Tipo**	**Descripción**
Uid	Int	Id del usuario
country_id	Int	Id del país del usuario
inst_id	Int	Id de la institución del usuario
Lid	Int	Id del lenguaje de programación por defecto del usuario
Username	String	Nombre de usuario
Password	String	Password encriptado del usuario
Nick	String	Alias del usuario
Enabled	Boolean	Representa si el usuario está activo o no en el sistema
Locale	String	Representa el lenguaje de la interfaz por defecto para

		dicho usuario
registration_date	Date	Fecha de registro del usuario
email_notofication	Boolean	Si/no el usuario desea recibir notificaciones automáticas del sistema via email
Gender	Int	Sexo del usuario
Online	Boolean	Representa si el usuario se encuentra online en el sistema
show_email	Boolean	Si/no el usuario desea mostrar su email a otros usuarios
last_ip	String	Dirección ip de la última conexión del usuario
access_rule	String	Regla de control de acceso del usuario
last_connected_date	Date	Fecha de la última conexión del usuario

Tabla 15. Descripción de la Entidad users.

2.8 Propuesta de arquitectura.

La arquitectura definida para el desarrollo de la versión 2 del COJ está basada en una arquitectura de tres capas lógicas: Capa de Presentación, Capa de Lógica de Negocio y Capa de Acceso a Datos. Esta arquitectura se eligió siguiendo como objetivo la separación de responsabilidades en cada una de las capas y lograr mayor facilidad de desarrollo del sistema. Dichas capas están bien delimitadas una de las otras; una capa superior interactúa con la inferior mediante interfaces que definen las funcionalidades que la misma debe brindar.

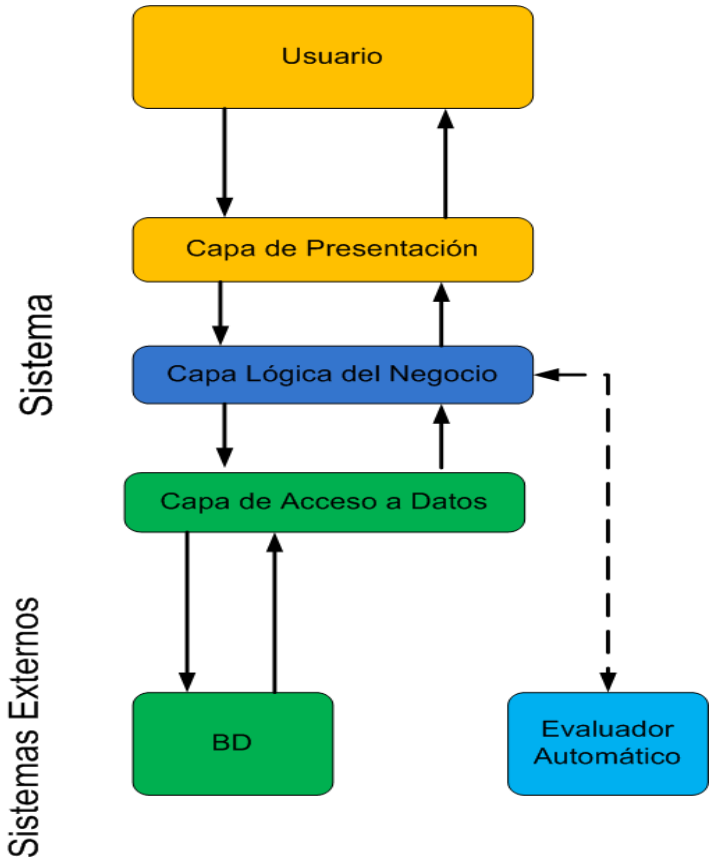


Figura 12. Propuesta de arquitectura.

2.8.1 Capa de Presentación.

La capa de presentación es la encargada de interactuar con el usuario, para la construcción de la misma se utilizará Spring MVC (módulo que provee el *framework* Spring que implementa el patrón Modelo - Vista – Controlador).

Dicho módulo ofrece una división limpia entre Controladores / Modelos / Vistas y es muy flexible, ya que implementa toda su estructura mediante interfaces. Además, provee interceptores también como controladores, lo que permite factorizar el comportamiento común en el manejo de múltiples peticiones.

2.8.2 Capa de Negocio.

La capa de negocio implementa la lógica del negocio de la aplicación, en esta capa se separa la lógica de la aplicación de los datos mediante la utilización de clases modelos, que carecen de lógica, y son utilizados como objetos que transitan por las diversas capas arquitectónicas. Además, en esta capa se establece la comunicación con el subsistema de evaluación, implementando el patrón arquitectónico Cliente-Servidor.

2.8.3 Capa de Acceso a Datos.

La capa de acceso a datos es la encargada de la manipulación de los datos que serán manejados por el sistema, como buena práctica de programación se hace uso de los Objetos de Acceso a Datos (DAO por sus siglas en inglés), los cuales constituyen una abstracción entre la persistencia de los datos y la capa de negocio. Para la implantación de esta capa se hará uso del módulo *jdbcTemplate*, el cual posee el *framework* Spring, caracterizado por ser muy sencillo y ligero de utilizar.

2.9 Patrones de diseño y arquitectónicos.

En ingeniería del software, un patrón es una solución ya probada y aplicable a un problema que se presenta una y otra vez en el desarrollo de distintas aplicaciones y en distintos contextos (25). Es importante destacar que, en general, un patrón no es una solución en forma de código directamente, sino una descripción de cómo resolver el problema y ante qué circunstancias es aplicable.

2.9.1 Modelo Vista Controlador (MVC).

El MVC (Modelo – Vista – Controlador) es un patrón de arquitectura de software que separa en tres componentes distintos los datos de una aplicación, la interfaz de usuario y la lógica de control. El modelo representa los datos y las reglas de negocio que rigen su acceso y actualización; puede verse como una representación de los procesos del mundo real. Las vistas se encargan de presentar los datos obtenidos del modelo. Es responsabilidad de las vistas mantener la información actualizada, esto se puede lograr a

través de peticiones de actualización al modelo o a través de notificaciones de cambio que el modelo emite (eventos). El controlador actúa como un traductor de las acciones que se realizan en las vistas en las operaciones que ocurren en el modelo. Las acciones realizadas por el modelo desencadenan la activación de procesos de negocio o cambian el estado del modelo. Sobre la base de las acciones del usuario y los resultados del modelo, el controlador responde mediante la selección de la vista apropiada (26). El framework *Spring* utilizado para el desarrollo de la solución basa su funcionamiento en la implementación del patrón MVC pues propone una estructura basada en modelos (Clases DAO), vistas (Ficheros ".jsp") y controladores (Ficheros ".java") que interactúan entre sí del mismo modo que define el patrón.

2.9.2 Inyección de Dependencia.

Este patrón consiste en resolver las dependencias de cada clase (atributos), generando los objetos cuando se inicia la aplicación y luego inyectándolos en los demás objetos que los necesiten a través de los métodos set o del constructor. Es una forma para mitigar la proliferación de dependencias y así fomentar el bajo acoplamiento entre los componentes, además tiene la intención de reducir la cantidad de código de infraestructura que se debe escribir(27). Spring es uno de los *frameworks* que provee un contenedor DI listo para usar, debido a que este es el mecanismo usado por el *framework* para proveer a los diferentes controladores de los objetos que necesitan para su funcionamiento, evitando de esta manera la creación de múltiples instancias de un mismo objeto y el uso innecesario de la herencia. Especial atención requiere el uso de dicho patrón en los objetos DAO que se conectan a la base de datos, ya que evita la creación de múltiples conexiones por los diferentes usuarios. Además se utiliza para inyectar en los controladores cada uno de los modelos requeridos para su correcto funcionamiento.

2.9.3 Bajo Acoplamiento.

El acoplamiento es una medida de la fuerza con que una clase está conectada a otras clases, con que las conoce y recurre a ellas. El Bajo Acoplamiento soporta un diseño de clases más independientes, que reducen el impacto de cambios, y permite que sean más reutilizables. En la solución dicho patrón es el

resultado de la utilización del patrón DI, el cual permite que los diferentes controladores sean independientes de los objetos que necesitan para su funcionamiento.

2.9.4 Alta Cohesión.

La cohesión es una medida de la fuerza con la que se relacionan las clases y el grado de focalización de las responsabilidades de un elemento. Cada elemento del diseño debe realizar una labor única dentro del sistema, no desempeñada por el resto de los elementos, y auto-identificable. Una clase con baja cohesión hace muchas cosas no relacionadas, o hace demasiado trabajo. En la solución este patrón es el resultado de asignar responsabilidades únicas a cada uno de los componentes (controladores, vistas y modelos).

2.9.5 Cliente-Servidor.

Esta arquitectura se encuentra dentro de la clasificación de estilo de llamada y retorno. El cliente y el servidor generalmente están localizados en diferentes sistemas, pero es posible que se encuentren en el mismo sistema. El cliente es el que realiza la petición por un servicio y el servidor provee el servicio correspondiente a la petición. El servicio debe procurar el resultado, el cual es retornado al cliente. Existen dos aproximaciones de dicha arquitectura; la primera ocurre cuando los clientes se comunican directamente con el servidor y la segunda cuando un tercer componente de software actúa como mediador entre los clientes y el servidor(es) (28). Dicho patrón permite la comunicación entre los tres subsistemas que componen la solución donde la capa de presentación actúa como el cliente, el distribuidor como el mediador y el motor(es) de evaluación como el servidor.

2.9.6 Agrupación de Objetos (Object Pool).

La agrupación de objetos puede ofrecer un significativo aumento del rendimiento, es más eficaz en situaciones donde el costo de la inicialización de una instancia es alto, la tasa de instancia de una clase es alta, y el número de instancias en uso en cualquier momento es bajo. En la implementación del patrón es deseable mantener todos los objetos que no están siendo usados en una misma agrupación para ello la clase de agrupación de objetos es creada usando el patrón *Singleton* (29). Dicho patrón permite en el

sistema la utilización de un *pool* de conexiones a la base de datos para minimizar el costo de creación de nuevas conexiones y poder reutilizar las existentes.

2.9.7 Patrón de Composición de Vistas (Composite View Pattern).

Consiste en la creación de páginas con estructuras similares en las cuales cada sección de la página varía en situaciones diferentes(30). Dicho patrón permite la confección de diseños diferentes para cada módulo del sistema reutilizando componentes comunes.

2.10 Conclusiones.

Al culminar el proceso de análisis y diseño de la versión 2 del COJ se logró una mayor comprensión de los requisitos, desglosados en Historias de Usuarios que permiten a los desarrolladores una mejor planificación de las tareas. Por otra parte, se obtuvieron artefactos como: el Diagrama de Clases, el Diagrama de Paquetes y el Modelo de Datos; los cuales permiten a los desarrolladores obtener una visión más detallada de los principales componentes que deben conformar el sistema.

Se concluye, además, que todos los artefactos generados por la metodología en esta etapa guiarán de forma efectiva el desarrollo de la versión 2 del COJ, o prácticamente cualquier otro subsistema que pueda ser concebido en el futuro.

3. CAPÍTULO 3. IMPLEMENTACIÓN Y PRUEBAS.

3.1 Introducción.

En este capítulo primeramente quedará confeccionado el "Plan de *release*" de la solución, permitiendo al cliente y los desarrolladores trazar una línea imaginaria de tiempo para la obtención del producto. Por otra parte se confeccionarán algunos artefactos, tales como: el Diagrama de Componentes y el Diagrama de Despliegue; el primero modela el empaquetado físico del sistema, permitiendo a los desarrolladores una mayor comprensión del mismo, y el segundo permite la concepción del despliegue físico del mismo. Finalmente se diseñarán y ejecutarán las pruebas a la solución obtenida.

3.2 Plan de release.

De acuerdo a las funcionalidades descritas en las Historias de Usuarios y la prioridad asignada a cada una, se planificaron 3 iteraciones para obtener la solución propuesta.

Release	Descripción de la iteración	Orden de la HU a implementar	Duración total
1	Implementar las HU de máxima prioridad	01-05	8 semanas
2	Implementar las HU de prioridad alta	06-18	4 semanas
3	Implementar las historias de usuario de prioridad media y baja	19-30	3 semanas

Tabla 16. Plan de *Release*.

3.3 Diagrama de Componentes.

El diagrama de componentes modela el empaquetado físico del sistema en unidades reutilizables llamadas componentes y sus relaciones. Un componente es una unidad física de implementación que

encapsula una o más clases del diseño. Este diagrama describe la descomposición del software en capas y subsistemas de implementación al igual que sus dependencias (31).

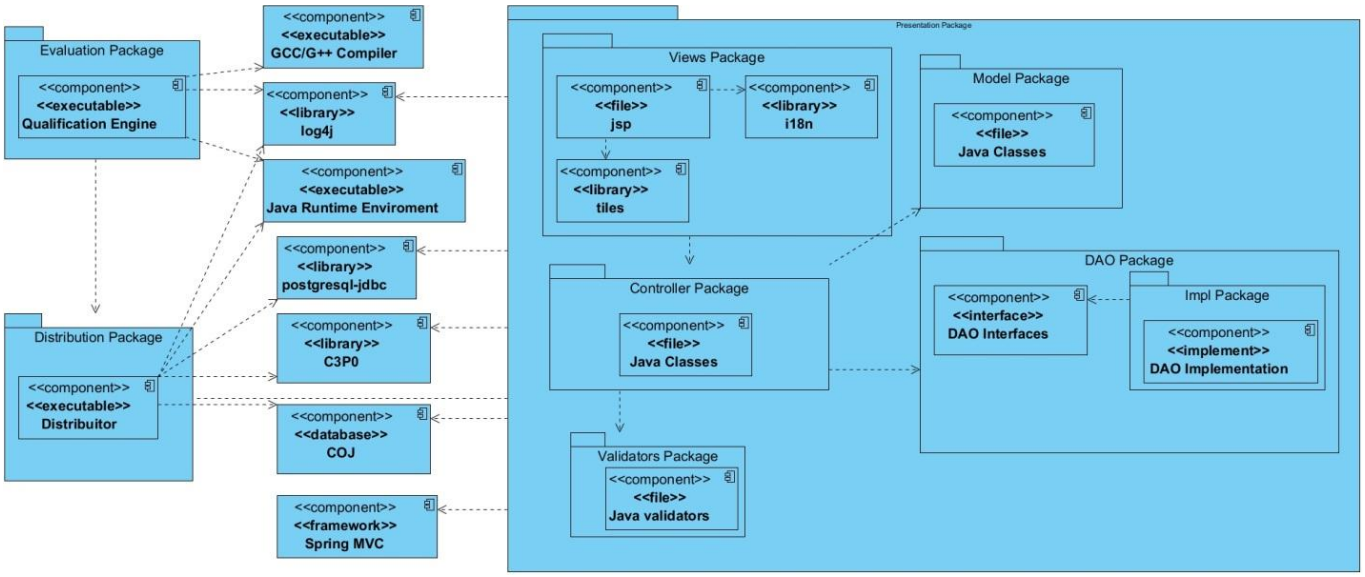


Figura 13. Diagrama de Componentes.

3.4 Diagrama de Despliegue.

El diagrama de despliegue muestra la configuración física sobre la que será desplegado el software. Este presenta los nodos computacionales que intervienen en el funcionamiento del sistema, las conexiones entre estos y los protocolos de comunicación que serán utilizados, estableciendo posibles configuraciones que se ilustran mediante los diagramas de despliegue (32).

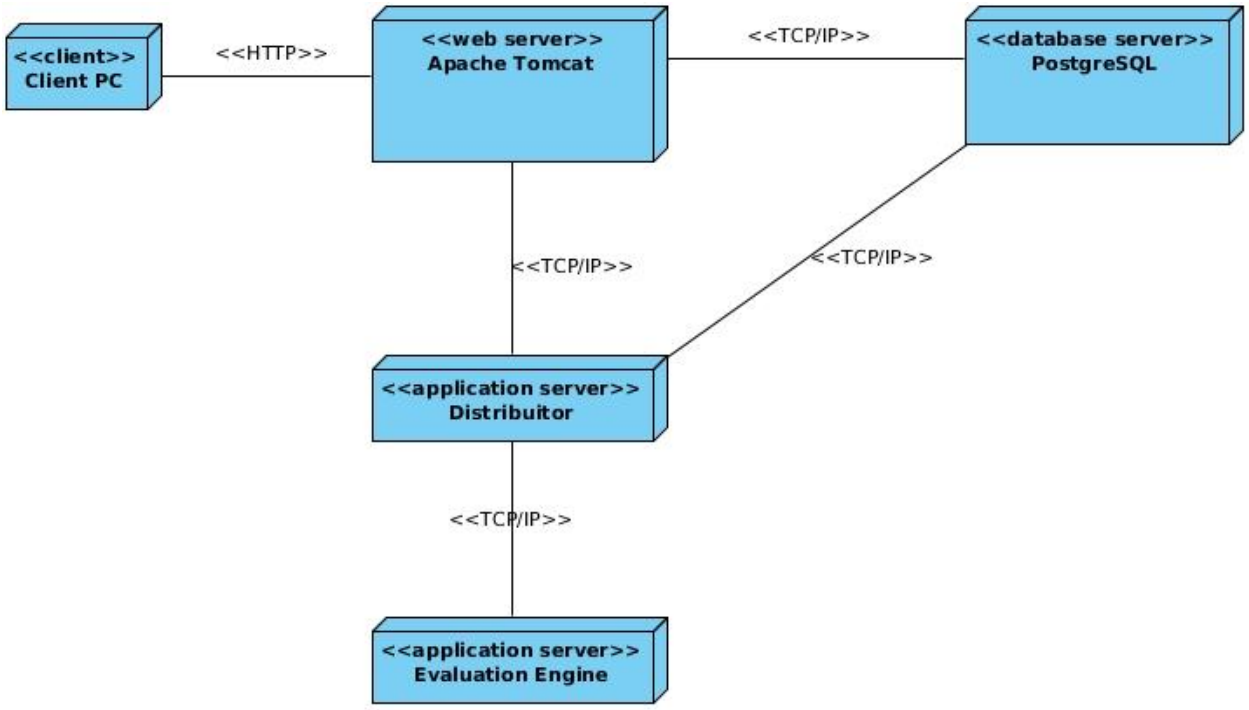


Figura 14. Diagrama de Despliegue.

3.4.1 Descripción de los nodos.

Client PC: PC desde donde se podrá visualizar e interactuar con el juez en línea a través de un navegador web (previamente instalado en la máquina).

Apache Tomcat: Servidor donde se encuentra desplegada la aplicación web, a la que se conectan los clientes por medio de sus estaciones de trabajo.

PostgreSQL: Servidor en el que se encuentran almacenados todos los datos persistentes del sistema.

Distribuitor: Ejecutable encargado de distribuir las peticiones de evaluación que realizan los clientes entre los diferentes motores de evaluación.

Evaluation Engine: Ejecutable encargado de evaluar la petición realizada por el cliente.

3.4.2 Descripción de los protocolos.

HTTP: Protocolo de comunicación estándar-básico que se utiliza en las arquitecturas web. Se ha utilizado para la comunicación establecida entre el servidor web y las estaciones de trabajo de los clientes.

TCP/IP: Protocolo mediante el cual se realizan las comunicaciones entre el servidor web y el servidor de bases de datos; además de la comunicación entre el servidor web y el distribuidor de peticiones, incluyendo también la del distribuidor con cada uno de los motores de evaluación y la base de datos.

3.5 Pruebas.

Las pruebas realizadas a todo producto de software constituyen un elemento crucial para garantizar la calidad e integridad del mismo, pues permiten a los desarrolladores identificar un conjunto de no conformidades antes de que el producto sea liberado, permitiendo entregar a los clientes un producto con mayor calidad. La realización de pruebas, a cada nueva funcionalidad, constituye una práctica recomendada en el desarrollo de software.

3.5.1 Pruebas de aceptación.

El uso de cualquier producto de software tiene que estar justificado por las ventajas que ofrece. Sin embargo, antes de empezar a usarlo es muy difícil determinar si sus ventajas realmente justifican su uso. El mejor instrumento para esta determinación es la llamada “prueba de aceptación”. En esta prueba se evalúa el grado de calidad del software con relación a todos los aspectos relevantes para que el uso del producto se justifique.

Para eliminar la influencia de conflictos de intereses, y para que sea lo más objetiva posible, la prueba de aceptación nunca debería ser responsabilidad de los ingenieros de software que han desarrollado el producto. Para la preparación, la ejecución y la evaluación de la prueba de aceptación ni siquiera hacen falta conocimientos informáticos. Sin embargo, un conocimiento amplio de métodos y técnicas de prueba y de la gestión de la calidad en general facilita esta labor.

La persona adecuada (o el equipo adecuado) para llevar a cabo la prueba de aceptación dispone de estos conocimientos y además es capaz de interpretar los requerimientos especificados por los futuros usuarios del sistema del software en cuestión. (33)

Casos de prueba asociados a la HU Registrar Usuario.

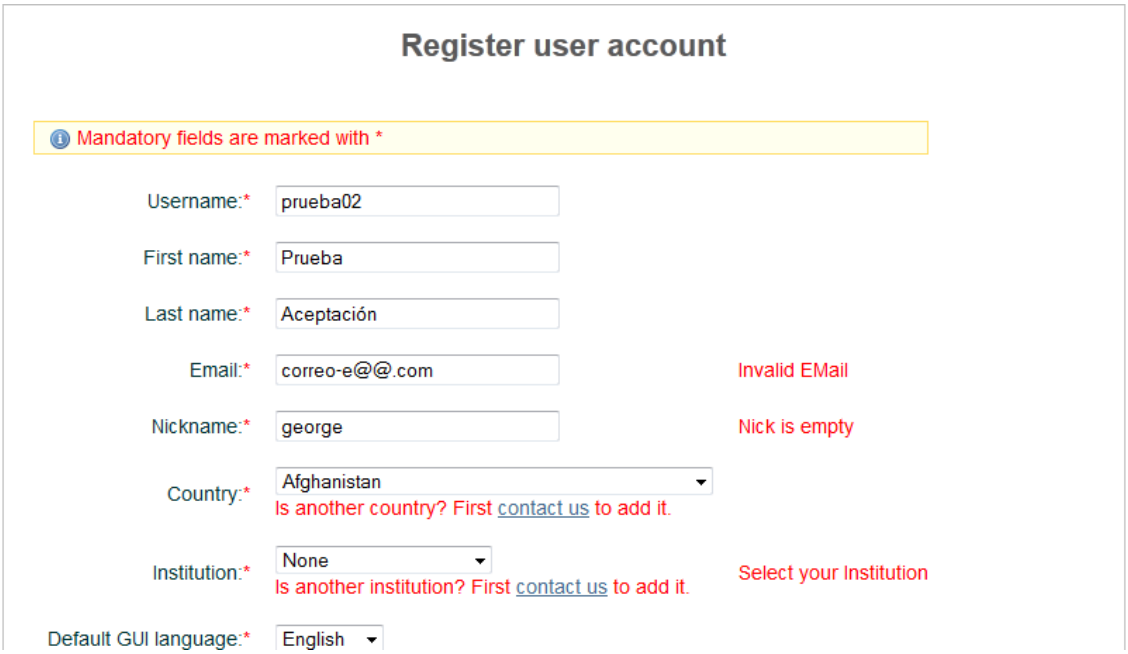
En la presente prueba se evaluará el proceso de registro de un usuario en el sistema, cuando dicho usuario inserta todos los datos solicitados por el sistema de forma correcta.

Casos de Prueba	
Número de caso de prueba : 01	Número de Historia de Usuario: 01
Nombre del Caso de prueba: Registrar Usuario con campos correctos	
Descripción de la prueba: Un usuario accede al formulario de registro (http://<host>/ user/createnewaccount.xhtml) donde llena los datos personales requeridos para dicho proceso cumpliendo con todos los requisitos propuestos para cada uno de los campos.	
Condiciones de ejecución: El usuario no debe estar autenticado. Tiene que existir almacenado y habilitado en la base de datos al menos un país, un lenguaje de programación y una pregunta de seguridad	
Entradas: - Nombre de usuario. - Alias. - Nombre. - Apellidos. - Correo-e. - País (Selección). - Institución (Selección). - Lenguaje de la Interfaz (Selección). - Lenguaje de Programación (Selección). - Contraseña - Confirmación de la Contraseña - Género (Selección). - Fecha de nacimiento (Selección). - Mostrar fecha de nacimiento (Marcar). - Mostrar correo-e (Marcar). - Recibir notificaciones vía correo-e (Marcar). - Pregunta de seguridad (Selección). - Respuesta de seguridad.	
Resultado esperado: Se inserta un nuevo usuario al sistema.	
Evaluación: Satisfactorio	

Tabla 17. Caso de Prueba Registrar Usuario.

En la siguiente prueba se evaluará el proceso de registro de un usuario en el sistema, cuando dicho usuario omite el llenado de algunos campos obligatorios, provee una dirección de correo electrónico que no cumple con el formato adecuado y no selecciona una institución.

Casos de Prueba	
Número de caso de prueba : 02	Número de Historia de Usuario: 01
Nombre del Caso de prueba: Registrar Usuario con campos incorrectos y problemas en formato.	
Descripción de la prueba: Un usuario accede al formulario de registro (http://<host>/ user/createnewaccount.xhtml) donde llena los datos personales requeridos para dicho proceso de forma parcial o con errores.	
Condiciones de ejecución: El usuario no debe estar autenticado. Tiene que existir almacenado y habilitado en la base de datos al menos un país, un lenguaje de programación y una pregunta de seguridad	
Entradas: - Nombre de usuario. - Alias (Vacío). - Nombre. - Apellidos. - Correo-e (Formato incorrecto). - País (Selección). - Institución (No seleccionada). - Lenguaje de la Interfaz (Selección). - Lenguaje de Programación (Selección). - Contraseña - Confirmación de la Contraseña - Género (Selección). - Fecha de nacimiento (Selección). - Mostrar fecha de nacimiento (Marcar). - Mostrar correo-e (Marcar). - Recibir notificaciones vía correo-e (Marcar). - Pregunta de seguridad (Selección). - Respuesta de seguridad.	
Resultado esperado: El sistema alerta al usuario de los errores cometidos a la hora de completar el formulario de registro mostrando para cada campo incorrecto un mensaje de color rojo al lado del mismo.	



Evaluación: Satisfactorio

Tabla 18. Caso de Prueba Registrar Usuario.

Casos de prueba asociados a la HU Insertar Concurso.

En la presente prueba se evaluará el proceso de creación de un concurso en el sistema teniendo en cuenta el escenario donde el usuario introduce todos los campos requeridos de forma correcta.

Casos de Prueba	
Número de caso de prueba : 03	Número de Historia de Usuario: 02
Nombre del Caso de prueba: Insertar Concurso con datos correctos.	
Descripción de la prueba: Un usuario que posea un rol con los privilegios necesarios para insertar un nuevo concurso accede al formulario (http://<host>/admin/createcontest.xhtml) una vez validados los datos el concurso es insertado a la base de datos y el usuario es redireccionado hacia una página donde podrá editar todos los recursos del mismo.	
Condiciones de ejecución: El usuario debe estar autenticado y poseer un rol con permisos asignados para realizar dicha acción (ROLE_ADMIN,ROLE_CONTEST_SETTER).	

Entradas:
- Id del concurso.
- Importar datos (Marcar) (Opcional).
  o Id del concurso a importar los datos.
  o Importar todos los datos (Marcar).
  o Importar problemas (Marcar).	

○ Importar variables globales (Marcar). ○ Importar lenguajes de programación (Marcar). ○ Importar banderas de restricción (Marcar). ○ Importar usuarios (Marcar). - Id del concurso a importar.
Resultado esperado: Se inserta un nuevo concurso al sistema y se redirecciona al usuario hacia una página para editar los recursos del mismo.
Evaluación: Satisfactorio

Tabla 19. Caso de Prueba Insertar Concurso.

En la presente prueba se evaluará el proceso de creación de un concurso en el sistema teniendo en cuenta el escenario donde el usuario selecciona la opción de importar datos de un concurso pasado sin elegir el recurso del cual desea importar los datos

Casos de Prueba	
Número de caso de prueba : 04	Número de Historia de Usuario: 02
Nombre del Caso de prueba: Importar datos de concursos anteriores	
Descripción de la prueba: Un usuario que posea un rol con los privilegios necesarios para insertar un nuevo concurso accede al formulario (http://<host>/ admin/createcontest.xhtml) luego selecciona la opción de importar datos de un concurso pasado sin seleccionar el concurso del cual desea importar los datos.	
Condiciones de ejecución: El usuario debe estar autenticado y poseer un rol con permisos asignados para realizar dicha acción (ROLE_ADMIN,ROLE_CONTEST_SETTER).	
Entradas: - Id del concurso. - Importar datos (Marcar) (Opcional). ○ Id del concurso a importar los datos. ○ Importar todos los datos (Marcar). ○ Importar problemas (Marcar). ○ Importar variables globales (Marcar). ○ Importar lenguajes de programación (Marcar). ○ Importar banderas de restricción (Marcar). ○ Importar usuarios (Marcar). - ID del concurso a importar.	
Resultado esperado: El sistema alerta al usuario mediante un mensaje en color rojo al lado del campo asociado a los concursos pasados indicando que debe seleccionar un concurso para importar.	

Evaluación: Satisfactorio

Tabla 20. Caso de Prueba Insertar Concurso.

Caso de prueba Modificar Concurso.

En la presente prueba se evaluará el proceso de configuración de un concurso en el sistema, donde se validará si la aplicación puede gestionar y validar todos los recursos asociados a un concurso de forma correcta.

Casos de Prueba	
Número de caso de prueba : 05	Número de Historia de Usuario: 03
Nombre del Caso de prueba: Modificar Concurso	
Descripción de la prueba: Un usuario accede a la sección de administración de un concurso donde realiza los cambios o inserciones que considere pertinente; una vez validados, los campos son insertados o modificados en la base de datos.	
Condiciones de ejecución: El usuario debe estar autenticado y poseer un rol con permisos asignados para realizar dicha acción (ROLE_ADMIN,ROLE_CONTEST_SETTER). Para las configuraciones generales de un concurso se requiere que exista almacenado en la base de datos: - Tipo de registro. - Estilos de concurso. - Tipos de usuarios. Debe existir al menos un lenguaje de programación almacenado en la base de datos.	
Entradas: Configuraciones Generales - Nombre del concurso. - Tipo de registro (Seleccionar) - Fecha límite de registro.	

- Fecha inicio. - Fecha fin. - Estilo de concurso (Seleccionar). - Tipo de usuarios (Seleccionar). - Habilitado (Marcar). - Plantilla Virtual (Marcar). - Bloqueado (Marcar). Banderas y Variables Globales - Penalización. - Tiempo muerto. - Tiempo congelado. - Tiempo de desbloqueo. - Mostrar estado de los envíos a los concursantes (Marcar). - Mostrar estado de los envíos a todos (Marcar). - Mostrar tabla de posiciones a los concursantes (Marcar). - Mostrar tabla de posiciones a todos (Marcar). - Mostrar problemas a los invitados (Marcar). - Permitir registro (Marcar). - Desbloqueo automático (Marcar). - Mostrar fallo en los caso de pruebas (Marcar). Problemas - Id del problema Usuarios - Id juez. - Id usuario. - Grupo. Lenguajes de programación - Id lenguaje (Marcar)
Resultado esperado: Se insertan y/o modifican los datos.
Evaluación: Satisfactorio

Tabla 21. Caso de Prueba Modificar Concurso.

Casos de prueba asociados a la HU Insertar Solución.

En la presente prueba se evaluará el proceso de envío de soluciones al sistema cuando el usuario inserta todos los datos requeridos de forma correcta.

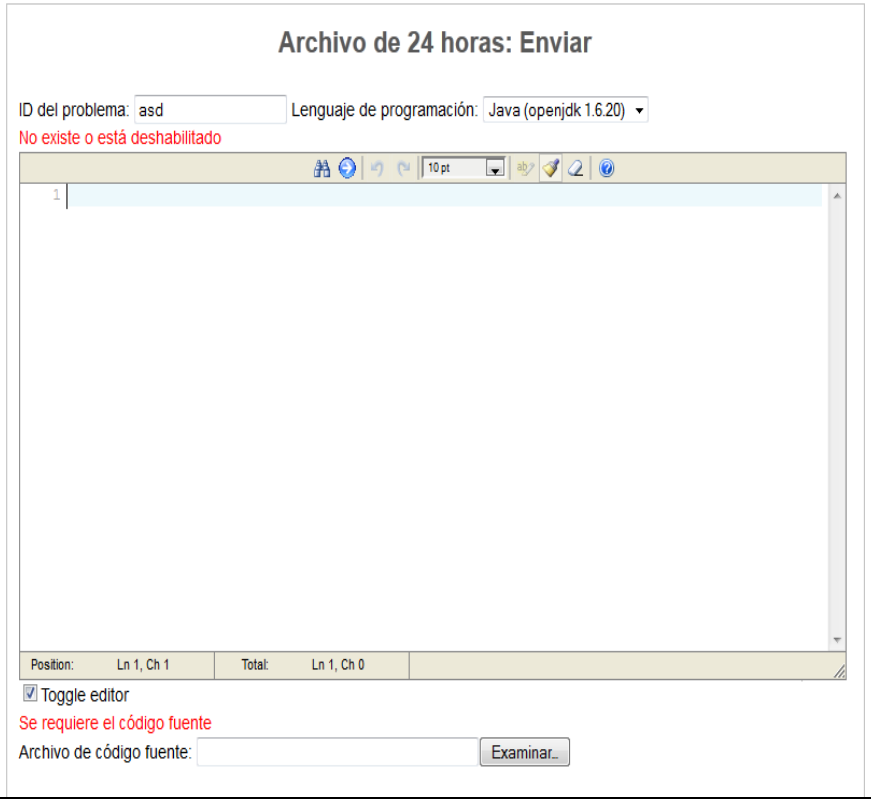
Casos de Prueba	
Número de caso de prueba : 06	Número de Historia de Usuario: 04
Nombre del Caso de prueba: Insertar solución	
Descripción de la prueba: Un usuario accede a la sección de realizar envío (http://<host>/ 24h/submit.xhtml) donde inserta todos los datos del mismo y lo envía para ser evaluado por el sistema.	

Una vez enviada la solución el usuario es redireccionado hacían una nueva página donde podrá consultar la evaluación otorgada por el sistema.
Condiciones de ejecución: El usuario debe estar autenticado y poseer un rol con permisos asignados para realizar dicha acción (ROLE_USER). Debe existir en la base de datos al menos un problema (habilitado y disponible para el módulo de 24 horas) y al menos un lenguaje de programación (habilitado).
Entradas: - Id problema. - Lenguaje de programación (Seleccionar). - Código fuente (texto plano)/Fichero de código fuente
Resultado esperado: Se inserta un envió a la base de datos y se le notifica al distribuidor. El usuario es redireccionado hacia una página donde podrá consultar el resultado de la evaluación.
Evaluación: Satisfactorio

Tabla 22. Caso de Prueba Insertar Solución.

En la presente prueba se evaluará el proceso de envío de soluciones al sistema cuando el usuario provee un id de problema incorrecto y no proporciona el código fuente.

Casos de Prueba	
Número de caso de prueba : 06	Número de Historia de Usuario: 04
Nombre del Caso de prueba: Insertar solución con datos incorrectos	
Descripción de la prueba: Un usuario accede a la sección de realizar envío (http://<host>/ 24h/submit.xhtml) proporcionando un id problema incorrecto y omitiendo el código fuente.	
Condiciones de ejecución: El usuario debe estar autenticado y poseer un rol con permisos asignados para realizar dicha acción (ROLE_USER). Debe existir en la base de datos al menos un problema (habilitado y disponible para el módulo de 24 horas) y al menos un lenguaje de programación (habilitado).	
Entradas: - Id problema (incorrecto). - Lenguaje de programación (Seleccionar). - Código fuente (texto plano)/Fichero de código fuente (Vacío).	
Resultado esperado: El sistema alerta al usuario sobre los campos incorrectos mediante un mensaje de error de color rojo asociado al(los) campos incorrectos.	



Evaluación: Satisfactorio

Tabla 23. Caso de Prueba Insertar Solución.

3.5.2 Pruebas de Rendimiento.

Pruebas dirigidas a evaluar la conformidad de un sistema o componente con requerimientos de desempeño específicos. Normalmente esto se lleva a cabo usando una herramienta de prueba automática para simular un gran número de usuarios, carga y volumen de información y para monitorear el desempeño del hardware (34). La herramienta seleccionada para llevar a cabo las pruebas de rendimiento y carga en la solución fue JMeter.

Luego de 4 meses de publicación del sistema en internet el mismo ha sido utilizado satisfactoriamente en la preparación de los concursantes caribeños; además ha permitido la realización de varios concursos oficiales y de entrenamiento cumpliendo al 100% con los requisitos necesarios para gestionar de forma efectiva y eficiente el flujo de los mismos. Así lo demuestran los avales recibidos por parte de varios

directores Caribeños mediante los cuales expresan su satisfacción con la nueva versión del COJ. Dichos avales pueden ser consultados en los anexos A, B y C.

JMeter

Apache JMeter es una herramienta de carga diseñada para realizar Pruebas de Rendimiento y Pruebas Funcionales sobre Aplicaciones Web. Desarrollado por *THE APACHE SOFTWARE FOUNDATION*, la primera versión (v1.0) data de marzo del 2001.

Originalmente el Apache JMeter fue diseñado para realizar pruebas de estrés sobre aplicaciones web (pruebas web clásicas). Sin embargo hoy en día su arquitectura ha evolucionado, ahora no sólo puede llevar a cabo pruebas en componentes típicos de Internet (HTTP), sino también puede realizar pruebas sobre Bases de Datos, scripts Perl, *servlets*, objetos java, servidores FTP y prácticamente cualquier medio de los que se pueden encontrar en la red.

Para un óptimo desarrollo de pruebas, es necesario tener ciertas nociones funcionales de la aplicación que se va a evaluar. Si esto no es así, las pruebas no serán completas al no saber por ejemplo si ha devuelto la hoja apropiada a la petición hecha o si nos ha permitido acceder con un login no apropiado.

El Apache JMeter incluye una interfaz gráfica de usuario que facilita el diseño de las pruebas. Este interfaz gráfico además de aportar un entorno cómodo de trabajo, también permite guardar y alterar tanto los test desarrollados como los componentes que lo integran. Gracias a esto se pueden reutilizar las pruebas o módulos de las mismas en el desarrollo de nuevas pruebas (35).

Plan de pruebas.

Los objetivos fundamentales para diseñar un plan de pruebas radican en la obtención de las configuraciones óptimas para el despliegue de la aplicación, la validación del estimado de carga y la concurrencia que se espera soporte la misma en condiciones extremas. Por otra parte, permiten la detección y localización de posibles errores existentes en el software.

Las pruebas fueron diseñadas para probar la estabilidad y concurrencia soportada por la aplicación, por encima de la velocidad de respuesta; debido a que se considera que la estabilidad y fiabilidad de la aplicación posee mayor importancia en situaciones extremas (evaluación de soluciones, escrutinio de

robos o crawler[29], alta concurrencia y realización de concursos) que la rapidez en los tiempos de respuesta.

Para la realización de las pruebas se utilizó un entorno de hardware y software consistente en:

- PC Hanel con procesador *core 2 duo* a 2.20 GHZ y 1 GB de RAM.
- Sistema operativo Ubuntu 10.04 en su versión para servidores.
- Entorno de ejecución java (JRE) en su versión 1.6.
- Servidor Web Apache Tomcat en su versión 7.0.
- Servidor de base de datos PostgreSQL en su versión 8.4.

Nótese que las condiciones de hardware utilizadas para la realización de las pruebas no constituyen las ideales para la utilización del sistema, debido a que en secciones anteriores se propuso una arquitectura distribuida del sistema que permite un mayor aprovechamiento de los recursos del servidor teniendo en cuenta que los diferentes subsistemas que componen la solución se desplegarían en servidores diferentes; permitiendo distribuir la carga que constituye la evaluación de las soluciones en servidores independientes al servidor web.

Antes de iniciar la prueba se identifican un conjunto de páginas caracterizadas por su alto consumo de recursos para ser utilizadas en la misma. En la solución se identificaron 3 páginas fundamentales que recogen el mayor número de peticiones realizadas al servidor debido a que manejan grandes volúmenes de datos y son solicitadas con mayor frecuencia por los usuarios. El plan de pruebas fue diseñado para simular un escenario que supere la concurrencia real estimada para el sistema. La prueba simulará una concurrencia de 350 usuarios realizando 3 peticiones en intervalos de 1 segundo de forma concurrente.

Resultados

Al culminar la prueba se puede concluir que la aplicación se encuentra lista para soportar una concurrencia muy superior a la que estará sometida; dígase 350 usuarios concurrentes realizando cada uno 3 peticiones a las 3 páginas que mayor consumo de recursos que posee la aplicación. A continuación se mostraran los resultados de las pruebas arrojados por el software JMeter a través de gráficas.

[29] Un *crawler* o araña web es un programa que inspecciona las páginas del *World Wide Web* de forma metódica y automatizada.

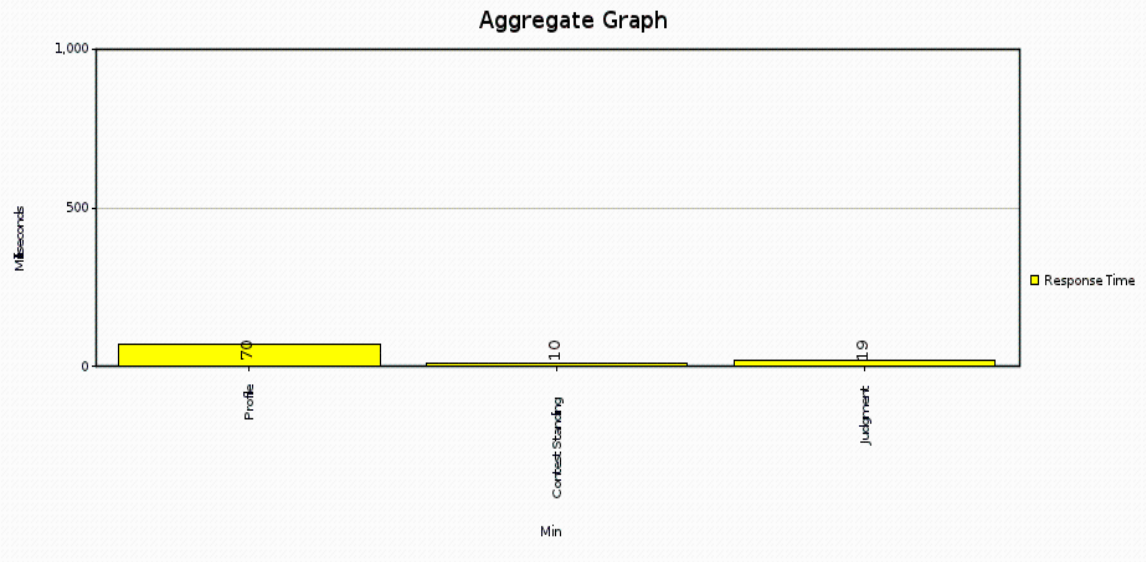


Figura 15. Tiempo mínimo de respuesta de una petición.

La gráfica anterior muestra el tiempo mínimo tomado por una petición para cada una de las páginas; nótese que la petición más rápida tardó solo 10 segundos en ser atendida sobre la “Tabla de Posiciones” de un concurso. Es importante destacar que estos casos extremos son difíciles de repetir en un ambiente donde el sistema se use por varios usuarios simultáneos.

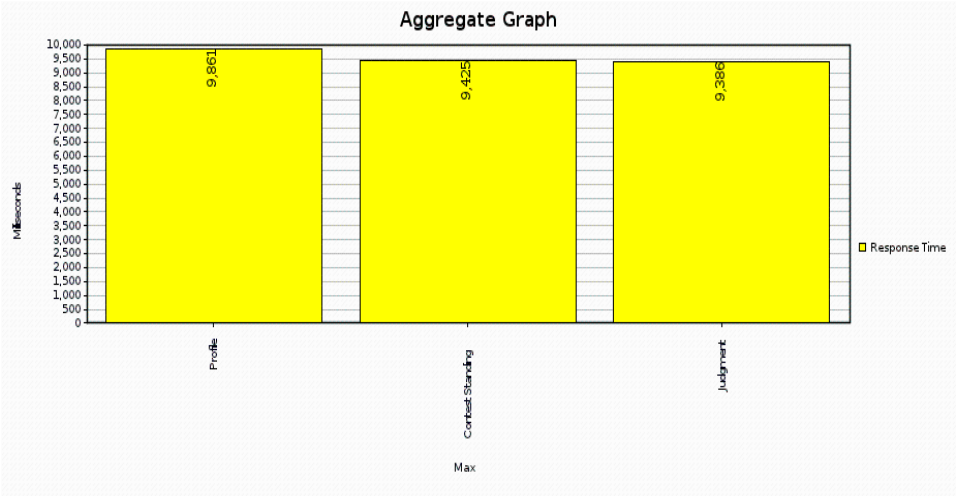


Figura 16. Tiempo máximo de respuesta de una petición.

La imagen muestra los tiempos máximos de respuesta de las peticiones realizadas a cada una de las páginas escogidas para la prueba; siendo el tiempo más alto de respuesta de 9,8 segundos. Teniendo en cuenta que las condiciones de hardware utilizadas para realizar las pruebas no son las óptimas y que la prueba simula un ambiente con mayor demanda que la mayor estimada para el sistema en un entorno real, aun en los casos más extremos (realización de concursos y escrutinio de robos) el tiempo máximo resultante es considerablemente más bajo del esperado.

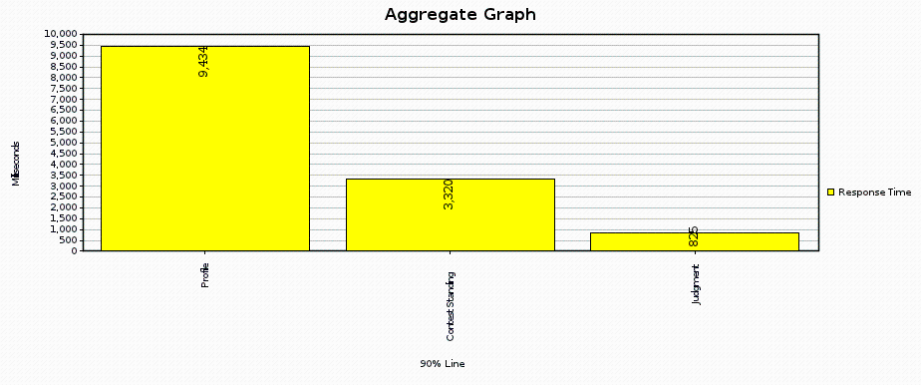


Figura 17. Línea del 90% de las peticiones.

La imagen indica el tiempo de respuesta del 90% de las peticiones para cada una de las páginas solicitadas en la prueba. La página más lenta resulta ser el “Perfil de un usuario” lo cual es esperado debido a que en dicha página se manejan grandes volúmenes de información.

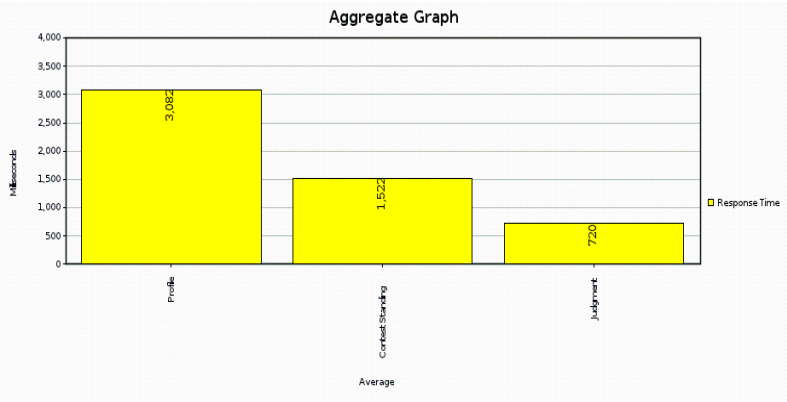


Figura 18. Tiempo promedio de respuesta.

La imagen anterior muestra el tiempo promedio de respuesta de las peticiones para cada una de las páginas analizadas, Nótese que el promedio de las páginas “Estado de las soluciones” y “Tabla de posiciones de un concurso” es considerablemente bajo, no así en la vista “Perfil de un Usuario”.

3.6 Conclusiones.

Al término de la presente fase del trabajo se han descrito los elementos de la arquitectura del sistema, así como los componentes desarrollados y la integración entre ellos. Se concibe un sistema que es resistente a cambios y cumple con los requisitos planteados.

Se considera que el uso del *framework* Spring, como base para el desarrollo de la capa de presentación, agilizó el proceso de implementación de la misma; además de permitir la obtención de un sistema modular que facilita la integración de nuevas funcionalidades mediante módulos independientes. Asimismo, la utilización del patrón arquitectónico “Cliente-Servidor” permitió la implementación distribuida del subsistema de evaluación, que lo iguala a los de más alto nivel entre sus homólogos.

Como último paso en el proceso de desarrollo, se realizaron las pruebas correspondientes que permitieron comprobar la calidad de la solución y validar al producto para ser utilizado de manera satisfactoria.

CONCLUSIONES

Luego de analizar las principales deficiencias existentes en la versión 1 del COJ, se evidenció la necesidad de implementar la versión 2 de dicho sistema. Para ello se realizó un proceso de investigación que propició la obtención del conocimiento científico como base para cumplimentar el análisis, diseño y desarrollo de la versión 2 del COJ, permitiendo a los autores arribar a las siguientes conclusiones:

- El estudio minucioso de la versión 1 del COJ demostró la necesidad de desarrollar una nueva versión del sistema sin hacer uso de ningún recurso de la versión existente.
- El análisis de sistemas similares en los diferentes ámbitos develó que los Concursos Virtuales, la Arquitectura Distribuida, los Sistemas de Puntuación, el soporte de múltiples estilos de concursos y la evaluación especial, constituyen características deseadas por los usuarios en este tipo de sistemas.
- Se identificaron y utilizaron las tecnologías más populares y efectivas en el desarrollo de jueces en línea.
- El sistema obtenido es robusto, resistente a cambios y cumple con los requisitos planteados.
- SXP es una metodología flexible que se ajustó satisfactoriamente al desarrollo del sistema.
- Se obtuvo un sistema modular que permitirá la inclusión de nuevas funcionalidades sin la necesidad de poseer un conocimiento de todo el sistema.

Los autores consideran que el sistema implementado es cualitativa y cuantitativamente superior a la versión anterior, pues la nueva versión incluye técnicas y características novedosas respecto a sistemas homólogos. En general, se considera que los objetivos de la investigación fueron cumplidos de manera satisfactoria.

RECOMENDACIONES

Aunque en esta nueva versión del COJ se recogen varias de las funcionalidades deseadas para el mismo, es esperado por todos sus miembros su constante crecimiento en cuanto a la prestación de nuevos servicios, por lo que se proponen las siguientes recomendaciones para ser incluidas en esta versión o en nuevas versiones que se desarrollen:

- Crear un proyecto comunitario en torno al COJ, el cual posibilitará la creación y extensión de un equipo de desarrollo, elemento muy necesario para el avance del movimiento ACM-ICPC en el Caribe.
- Implementar una capa de servicios que permita la interoperabilidad con otros sistemas; además de posibilitar el uso del sistema mediante el consumo de servicios web, para usuarios de nuestro país que posean conexiones muy lentas.
- Implementar funcionalidades que faciliten la integración con el proceso de enseñanza-aprendizaje.
- Establecer una integración con redes sociales como Facebook y Twitter.

REFERENCIAS BIBLIOGRÁFICAS

1. ICPCWiki: The Early Years. [En línea]. [Citado el: 7 de Marzo de 2012]. Disponible en: http://cm.baylor.edu/ICPCWiki/Wiki.jsp?page=The%20Early%20Years.

2. A History of CSUS' PC^2. [En línea]. 7 De Marzo de 2012. [Citado el: 7 de Marzo de 2012]. Disponible en: http://www.ecs.csus.edu/pc2/pc2history.html.

3. A History of CSUS' PC^2. [En línea]. [Citado el: 7 De Abril de 2012]. Disponible en: http://www.ecs.csus.edu/pc2/pc2history.html.

4. SHAHRIAR MANZOOR, RUJIA LIU and MIGUEL A. REVILLA. *Competitive Learning in Informatics: The UVa Online Judge Experience*. 7 de Abril de 2012. S.l.: s.n.

5. Sphere Research Business Solutions. [En línea]. [Citado el: 7 de Marzo de 2012]. Disponible en: http://sphere-research.com/en/.

6. PROGRAMACIÓN DISTRIBUIDA. [En línea]. [Citado el: 8 de Marzo de 2012]. Disponible en: http://www.scribd.com/doc/40114693/PROGRAMACION-DISTRIBUIDA.

7. Programming PHP - Rasmus Lerdorf, Kevin Tatroe, Peter MacIntyre - Google Libros. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.google.com.cu/books?hl=es&lr=&id=h-E1lVko-skC&oi=fnd&pg=PT7&dq=php+reference&ots=yoH2eUMrRl&sig=cmT7lr--CwS9VmeKPrrCCM-tsOE&redir_esc=y#v=onepage&q=php%20reference&f=false.

8. Definición de lenguaje de programacion. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.definicion.org/lenguaje-de-programacion.

9. C++ Manual de referencia con anotaciones - Margaret A. Ellis, Bjarne Stroustrup - Google Libros. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.google.com.cu/books?hl=es&lr=&id=8XB8-DMjKNMC&oi=fnd&pg=PR5&dq=historia+de+c%2B%2B&ots=0BHc-htb4e&sig=YIIgXhHACtF6auf2SnU1Aj9vJOQ&redir_esc=y#v=onepage&q&f=false.

10. PHP y MySQL: Tecnología para el desarrollo de aplicaciones web. - Ángel Cobo - Google Libros. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.google.com.cu/books?hl=es&lr=&id=zMK3GOMOpQ4C&oi=fnd&pg=PR17&dq=php+es+un+lenguaje+de+programaci%C3%B3n+&ots=FeizY3Hfvi&sig=6Fb7-SA1M8ZKqOyahPKJlHRplhk&redir_esc=y#v=onepage&q=php%20es%20un%20lenguaje%20de%20progr

amaci%C3%B3n&f=false.

11. Cómo programar en Java - Harvey M. Deitel, Paul J. Deitel - Google Libros. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.google.com.cu/books?hl=es&lr=&id=tR7k9ga5CjoC&oi=fnd&pg=PR17&dq=que+es+java&ots=w8_JTrRxIH&sig=UKMCH9pQRhu86vaaw2NI4zzjIKI&redir_esc=y#v=onepage&q=que%20es%20java&f=false.

12. *Python_para_todos.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Citado el: 8 de Junio de 2012]. Disponible en: http://dspace.universia.net/bitstream/2024/919/1/Python_para_todos.pdf.

13. *MarcoConceptualJISBD02.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.lcc.uma.es/~av/Publicaciones/02/MarcoConceptualJISBD02.pdf.

14. Enterprise Java Development Tools | SpringSource. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.springsource.com/developer/spring.

15. PostgreSQL: About. [En línea]. [Citado el: 8 de Junio de 2012]. Disponible en: http://www.postgresql.org/about/.

16. About the Eclipse Foundation. [En línea]. [Citado el: 12 de Junio de 2012]. Disponible en: http://www.eclipse.org/org/.

17. An Introduction to NetBeans. [En línea]. [Citado el: 12 de Junio de 2012]. Disponible en: http://netbeans.org/about/index.html.

18. Apache Tomcat - Welcome! [En línea]. [Citado el: 12 de Junio de 2012]. Disponible en: http://tomcat.apache.org/.

19. GlassFish: servidor de aplicaciones de código abierto — Java.net. [En línea]. [Citado el: 12 de Junio de 2012]. Disponible en: http://glassfish.java.net/es/.

20. *METODOLOGIAS_TRADICIONALES_VS._METODOLOGIAS_AGILES.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Citado el: 15 de Junio de 2012]. Disponible en: http://eva.uci.cu/file.php/161/Documentos/Materiales_complementarios/UD_1_Procesos/Metodologias/METODOLOGIAS_TRADICIONALES_VS._METODOLOGIAS_AGILES.pdf.

21. GLADYS MARSI PEÑALVER ROMERO. *Release_0.2_SXP*. S.l.: s.n.

22. Tutorial - UML Package Diagram. [En línea]. [Citado el: 30 de Abril de 2012]. Disponible en: http://www.visual-paradigm.com/product/vpuml/tutorials/packagediagram.jsp.

23. Introduction to UML 2 Class Diagrams. [En línea]. [Citado el: 30 de Abril de 2012]. Disponible en:

http://www.agilemodeling.com/artifacts/classDiagram.htm.

24. Introduction to Physical Data Model (PDM)s. [En línea]. [Citado el: 30 de Abril de 2012]. Disponible en: http://www.agilemodeling.com/artifacts/physicalDataModel.htm.

25. Pattern Definitions. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://st-www.cs.illinois.edu/patterns/definition.html.

26. (ootips) Model-View-Controller. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://ootips.org/mvc-pattern.html.

27. Design Patterns: Dependency Injection. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://msdn.microsoft.com/en-us/magazine/cc163739.aspx.

28. Client/Server Architectures. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://www.thedacs.com/databases/url/key/216.

29. Object Pool Design Pattern. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://sourcemaking.com/design_patterns/object_pool.

30. Apache Tiles 2 - The Composite View Pattern. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://tiles.apache.org/tutorial/pattern.html.

31. Introduction to UML 2 Component Diagrams. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://www.agilemodeling.com/artifacts/componentDiagram.htm.

32. Introduction to UML 2 Deployment Diagrams. [En línea]. [Citado el: 2 de Mayo de 2012]. Disponible en: http://www.agilemodeling.com/artifacts/deploymentDiagram.htm.

33. La prueba de aceptación es la prueba más importante para los productos software. [En línea]. [Citado el: 24 de Abril de 2012]. Disponible en: http://pruebasdesoftware.com/pruebadeaceptacion.htm.

34. Pruebas de Rendimiento de Software. [En línea]. [Citado el: 11 de Mayo de 2012]. Disponible en: http://www.greensqa.com/portal/soluciones/calidad-de-productos/55-pruebas-de-rendimiento-de-software.

35. *JMeter. Manual de usuario v1.2.pdf (objeto application/pdf)* [online]. S.l.: s.n. [Citado el: 11 de Mayo de 2012]. Disponible en: http://www.ejie.net/documentos/Herramientas/JMeter.%20Manual%20de%20usuario%20v1.2.pdf.

Anexos

Anexo A: Aval 1ra Copa UMCC de Programación

Universidad de Matanzas Camilo Cienfuegos (UMCC)
Jueves, 9 de junio de 2011
"Año 53 de la Revolución"

Aval de aplicación

A través de la presente hacemos constar nuestra gran satisfacción con el desarrollo de la nueva versión del Juez en línea del Caribe (*Caribbean Online Judge* - COJ). Vale destacar que en la UMCC se conoció de este trabajo del 17 de Enero al 20 de Febrero de 2011 pues en esta fecha cuatro estudiantes de nuestra institución y dos profesores estuvieron en la preparación de los equipos cubanos que participarían en la Final Mundial ACM-ICPC de 2011. En ese entonces era conocido como *Challenger Online Judge* (CHOJ) y a través del mismo se desarrollaron varios de los Concursos de preparación como se muestra en el Anexo # 1.

Posteriormente, del 5 al 7 de Mayo de 2011 se celebró la I Copa UMCC de Programación en saludo a los aniversarios 37 de la UMCC y de la Educación Superior en Matanzas. La nueva versión de COJ desarrollada por los estudiantes de la UCI Jorge Luis Roque Álvarez y Juan Carlos Lobaina Guzmán y el profesor Dovier Antonio Ripoll Méndez fue utilizada en cinco concursos con resultados muy satisfactorios. De los cinco concursos tres se desarrollaron como parte de la preparación de algunos de los equipos participantes en la Copa como se muestra en el Anexo # 2, así como en el Concurso de Práctica (Anexo # 3) y el Concurso Oficial (Anexo # 4) de la referida Copa.

Nos ha impresionado sobre todo que este juez brinda un grupo de nuevas opciones que serán muy útiles para la preparación de nuestros equipos con vista a los futuros eventos ACM-ICPC, así como para el desarrollo de los propios eventos.

Fraternalmente

Roger Pérez Chávez,
Director ACM-ICPC
UMCC

Anexo B: Aval 2da Copa UMCC de Programación

Universidad de Matanzas Camilo Cienfuegos (UMCC)
Miércoles, 30 de mayo de 2012
"Año 54 de la Revolución"

Aval de aplicación

A través de la presente hacemos constar nuestra gran satisfacción con el desarrollo de versión 2.0 del Juez en línea del Caribe (*Caribbean Online Judge* - COJ) desde su puesta en marcha a finales del año 2011. Por segundo año consecutivo este fue el Jurado utilizado en la Copa UMCC de Programación también en su segunda edición. Esta Copa se desarrolló del 3 al 6 de Mayo del presente año en saludo a los aniversarios 40 de la UMCC y de la Educación Superior en Matanzas.

La nueva versión de COJ desarrollada por los estudiantes de la UCI Jorge Luis Roque Álvarez y Juan Carlos Lobaina Guzmán y el profesor Dovier Antonio Ripoll Méndez fue utilizada en esta oportunidad en tres concursos con excelentes resultados y una alta satisfacción de todos los participantes. Estos tres concursos se dividieron de la siguiente forma: Concurso de Calentamiento el día 3 de Mayo (Anexo # 1), Concurso de Práctica al siguiente día (Anexo # 2) y finalmente el Concurso Oficial de la referida Copa el día 5 de Mayo (Anexo # 3).

Vale destacar que en esta ocasión se utilizó una versión del COJ muy fácil de instalar de manera local como una máquina virtual. Esta variante es muy útil para este tipo de eventos presenciales para evitarnos posibles problemas de conectividad con el Jurado oficial instalado en la UCI.

Fraternalmente

Roger Pérez Chávez,
Director Ejecutivo ACM-ICPC
UMCC

Anexo C: Aval de utilización en la UCI

Universidad de las Ciencias Informáticas (UCI)

La Habana, Mayo de 2012

"Año 54 de la Revolución"

A través del presente hacemos constar nuestra gran satisfacción con el desarrollo de la versión 2 del Juez Caribeño en Línea (COJ). Vale destacar que se tiene conocimiento de la existencia del mismo desde mediados del curso 2010-2011, donde se usó por primera vez como herramienta para gestionar los concursos de preparación que tuvieron el objetivo de apoyar el entrenamiento de los equipos cubanos con vistas a la Final Mundial 2011 del ACM-ICPC; por aquel entonces era conocido como Challenger Online Judge (CHOJ) y era usado generalmente por estudiantes de la Facultad 10 de la UCI. En Junio de 2011 se utilizó satisfactoriamente para gestionar la VI Copa UCI de Programación "Tomas López Jiménez". En ese entonces se manejaba el proyecto para sustituir la versión 1 del COJ, debido a un conjunto de deficiencias respecto a la seguridad, las funcionalidades, la administración de recursos y la plataforma.

La nueva versión del COJ desarrollada por los estudiantes Juan Carlos Lobaina Guzmán y Jorge Luis Roque Alvarez, tutorados por Dovier Antonio Ripoll Méndez (Director Ejecutivo del ACM-ICPC en el Caribe), fue publicada en Internet el 13 de octubre de 2011. Desde entonces, la versión 2 del COJ ha sido utilizada en la preparación de concursantes de la región caribeña y, además, en la realización de varios concursos oficiales y de entrenamiento.

En lo que se refiere al uso del COJ en la UCI, se debe agregar que ha sido muy útil en la organización de los concursos de preparación del recientemente creado Movimiento de Programación Competitiva "Tomás López Jiménez" (MPC-TLJ), y se empleó como la herramienta oficial para gestionar los concursos en los niveles facultad y universidad de la Copa Pascal de Programación; en ambos casos, el

sistema funcionó correctamente y cumplió con los requisitos necesarios para gestionar de forma eficiente el flujo de los concursos. Por otra parte, se evidenció que el sistema cuenta además con un conjunto de funcionalidades y módulos que facilitan la gestión de los recursos por parte de los administradores.

Es destacable, además, que la versión 2 del COJ cuenta con un conjunto de funcionalidades novedosas, las cuales serán muy útiles en la preparación de los concursantes para futuros eventos del ACM-ICPC.

Lic. Tomás Orlando Junco Vázquez
Director Ejecutivo del ACM-ICPC en la UCI

Ing. José Ernesto Lara Rodríguez
Director Asistente del ACM-ICPC en la UCI

DrC. Liesner Acevedo Martínez
Jefe del DDC de Programación