Universidad de las Ciencias Informáticas

TRABAJO DE DIPLOMA

Conjunto de aplicaciones informáticas de apoyo a la Final Caribeña del ACM-ICPC.

Autores: Adrian Enrique Ciria Jacas
Carlos González Galera
Tutores: Ing. Yolanda Mauri Pérez
Ing. Javier Bandomo Ruíz

La Habana, Junio, 2015

Declaración de autoría

Declaramos ser los únicos autores de este trabajo y concedemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los ___ días del mes de _____ del año 2015.

Agradecimientos

Adrian Enrique Ciria Jacas:

A mis padres, mi familia, a los tutores Yolanda y Javier, al compañero de tesis, a todos mis amigos, en especial a los del grupo que bastante aguantaron y a todo aquel que me apoyó a lo largo de toda la carrera.

Carlos González Galera:

A mi compañero de tesis, sería imposible haber tenido éxito sin el trabajo en equipo.
A nuestros tutores Yolanda y Javier, muchas gracias, de verdad. No imagino mejores personas ni profesionales más capaces.
A mi familia, por estar conmigo siempre.
Por último y no menos importante a mis amigos. Desafortunadamente son muchos y el espacio es corto.

Resumen

Desde 2009 la Universidad de las Ciencias Informáticas (UCI), radicada en La Habana, Cuba, ha sido una de las sedes que acoge la Final Caribeña del ACM-ICPC. Desde ese momento, ha sido de máxima prioridad para los organizadores en la sede cubana lograr que la celebración de cada edición del evento sea superior a las anteriores. Para alcanzar esta meta se han ido integrando de forma paulatina un conjunto de servicios durante la competencia, los cuales persiguen suplir las necesidades que pudiesen tener los participantes, así como acercar más el evento al público en general.

El siguiente trabajo de diploma propone a través del análisis de los servicios de impresión de código fuente, determinación de premios especiales, elaboración de la tabla de posiciones y en el servicio informativo del evento, identificar las principales deficiencias que vienen presentando los mismos y que afectan de alguna forma la celebración de la Final Caribeña para, de esta manera, presentar un conjunto de aplicaciones informáticas que establezcan una solución a las deficiencias encontradas.

Palabras clave: Final Caribeña del ACM-ICPC, servicios.

Índice

Índice de Tablas

Índice de Ilustraciones

Introducción

El ACM-ICPC constituye la competición más prestigiosa de programación y algoritmia que se desarrolla cada año entre diversas universidades del mundo. Desde su surgimiento en 1970 cada edición de la misma ha ido ganando terreno exponencialmente, en cuanto a credibilidad y participación se refiere. Solo en la edición de 2014 participaron 32,043 concursantes de 2286 universidades en 94 países, a través de más de 300 sitios.[1]

Este crecimiento acelerado se debe, principalmente, a las habilidades técnicas que desarrollan el sistema de programación online en los estudiantes de especialidades tales como la informática y las ciencias de la computación.

El ACM-ICPC tiene varios niveles de competición entre equipos (de tres estudiantes cada uno) que representan a instituciones de la Educación Superior:

- **Concursos Locales** (Nivel 1): Se realizan anualmente a nivel de institución y tienen como propósito fundamental la clasificación de equipos para niveles superiores de competición.
- **Concursos Nacionales** (Nivel 2): Se realizan anualmente a nivel de país y tienen como propósito fundamental la clasificación de equipos para el próximo nivel de competición.
- **Concursos Regionales** (Nivel 3): Se realizan en varias regiones del mundo, entre octubre y diciembre de cada año. Tienen como propósito fundamental la clasificación de equipos para el próximo nivel de competición.
- **Final Mundial** (Nivel 4): Se realiza en los primeros meses del año siguiente a los Concursos Regionales y tiene como propósito fundamental la determinación de los mejores equipos a nivel mundial.[2]

  Existen más de treinta zonas donde se desarrolla este certamen a nivel regional. Una sede puede agrupar distintos países, en otros casos se conforman por solo una parte o la totalidad de un país en específico, tal es el caso de Estados Unidos y Brasil, respectivamente.

INTRODUCCIÓN

La Universidad de las Ciencias Informáticas (UCI), es la encargada desde 2009 de la organización de la Final Caribeña del ACM-ICPC. El evento se desarrolla de la siguiente manera: en un local cerrado, completamente aislado del exterior, en el cual todos los ordenadores se encuentran en una subred privada, contando con acceso solamente a las herramientas y servicios dispuestos por los organizadores para la justa. Durante el tiempo que dura el certamen los participantes ponen a prueba sus habilidades, tratando de resolver una serie de problemas de gran complejidad algorítmica, a través del juez online disponible para el evento, el BOCA, el cual después de juzgar las soluciones notifica al equipo y actualiza la tabla de posiciones del evento. Esta tabla de posiciones permite, al terminar la competencia, elaborar el ranking oficial del ACM-ICPC para la región del Caribe, así como determinar los premios especiales que otorga la sede cubana.

Además de los servicios de ranking y premios especiales existen otros que toman parte dentro de la dinámica de la competencia. Así, existe un servicio de impresión de código fuente, a través de una herramienta autónoma de la universidad, ofreciendo una alternativa para el análisis de los problemas. Existe, además, un servicio informativo, el cual permite mantener actualizados a los espectadores acerca de la evolución del evento vía web.

Todos estos servicios, si bien se encuentran en completo funcionamiento durante el evento, presentan una serie de deficiencias que ameritan ser analizadas:

- Para actualizar la tabla de posiciones después de terminado el evento se utiliza la herramienta independiente SARIS, la cual recibe un archivo JSON serializado y genera la tabla de posiciones. Dicho archivo se recibe directamente del BOCA y contiene los envíos que hicieron todos los usuarios, incluyendo los de prueba que hacen los jueces del evento. Actualmente no existe ningún proceso automático que permita eliminar dichos envíos de prueba o de algún usuario con rol de juez, en específico. Esto implica que a la hora de elaborar la tabla de posiciones, el sistema tiene en cuenta, además de los participantes, a los jueces que sometieron casos de prueba, lo cual no es correcto. Para evitar esto, lo que se viene haciendo durante cada edición del evento es editar el archivo JSON y eliminar los envíos no deseados, de forma manual, antes de enviarlos al SARIS.
- Durante el evento se cuenta con un servidor de impresión de código fuente, conocido como Printer,

la cual fue adoptada como una solución temporal y ha continuado vigente hasta la actualidad. Esto tiene una implicación directa en el hecho de que resulta muy engorroso brindarle soporte a la aplicación, ya que la misma está desarrollada sobre Ruby, tecnología sobre la que no existe mucho dominio entre los organizadores del evento.

- La sede cubana de la Final Caribeña otorga, además de los tradicionales tres primeros lugares, cuatro premios extras o adicionales:
  - **Equipo Veloz:** Equipo que acumule la mayor cantidad de problemas donde haya sido el primero en resolver.
  - **Equipo Certero:** Equipo que resuelva la mayor cantidad de problemas sin envíos incorrectos.
  - **Equipo Exclusivo:** Equipo que resuelva la mayor cantidad de problemas que ningún otro equipo haya podido resolver.
  - **Equipo Progreso:** Equipo que escale la mayor cantidad de posiciones con respecto al ranking de los Concursos Nacionales Caribeños (CNC).

Para determinar cada uno de estos se utiliza un procedimiento manual que consiste en comparar los resultados de cada equipo con su desempeño en la edición anterior o con las estadísticas generales ofrecidas por el BOCA, en dependencia del premio en cuestión, al finalizar la competencia. Este método, aunque útil, representa una gran ineficiencia, pues se necesita de una o varias personas para realizar una tarea que pudiese ser automatizada, promoviendo así un ahorro de tiempo y esfuerzo, además de un menor margen de error.

Durante la celebración del concurso regional se identifican otros problemas relacionados con la transmisión del evento, la cual no brinda toda la cobertura que desean los organizadores, limitando el objetivo propuesto de lograr una mayor interactividad entre el evento y el público en general.

A partir de las dificultades encontradas, se identifica como **problema a resolver** del trabajo de diploma: ¿Cómo mejorar los servicios que conforman el sistema de apoyo de la Final Caribeña del ACM-ICPC?

El **objeto de estudio** se enfoca en la realización de las competencias de programación al estilo ACM-

ICPC y el **campo de acción** se sitúa en los sistemas de apoyo para la organización y realización de los concursos regionales del ACM-ICPC, específicamente en la región de América Latina.

Como **objetivo general** se propone desarrollar un conjunto de aplicaciones para mejorar servicios de información, impresión de código fuente, confección de la tabla de posiciones y determinación de premios especiales, que se realizan durante la Final Caribeña del ACM-ICPC.

Se plantean, además, los siguientes **objetivos específicos**:

- Realizar un estudio de las especificaciones y el dominio del negocio, para posteriormente conformar el marco teórico conceptual.
- Diseñar las aplicaciones que permitirán la informatización de los procesos.
- Implementar las funcionalidades definidas durante la fase de diseño.
- Someter al software a un conjunto de pruebas de calidad, según la metodología de desarrollo escogida.

De esta manera para guiar el proceso de investigación, se plantea la siguiente **hipótesis**: El desarrollo de un conjunto de aplicaciones informáticas permitiría mejorar los servicios de información, impresión de código fuente, confección de la tabla de posiciones y determinación de premios especiales, que se realizan durante la Final Caribeña del ACM-ICPC.

Para llevar a cabo la investigación se utilizan los **métodos científicos** detallados a continuación:

Métodos teóricos:

El método histórico-lógico se aplica en la investigación para consultar la bibliografía referente a las ediciones regionales pasadas del ACM-ICPC en el área de América Latina y el Caribe así como las diferentes herramientas de apoyo que se emplean en dichas ediciones.

El método analítico-sintético se utiliza para, mediante el análisis de la situación problemática planteada, determinar las características que debe tener la solución, a partir de las soluciones existentes.

Métodos empíricos:

Se utiliza la encuesta como procedimiento de recolección de datos e información sobre las distintas herramientas utilizadas como apoyo en ediciones previas del ACM-ICPC en el área de América Latina. Dicha encuesta fue respondida por un total de cuatro personas, los cuales se desempeñan como organizadores del evento en otras regiones de América Latina. Para más información ver Anexo 1.

Tareas a cumplir:

- Análisis de los servicios de información, impresión de código fuente, confección de la tabla de posiciones y determinación de premios en las diferentes sedes regionales del ACM-ICPC para América Latina.
- Investigación de las herramientas, lenguajes de programación y metodologías existentes para realizar el análisis, diseño e implementación de las aplicaciones o conjunto de aplicaciones.
- Definición de la arquitectura de las aplicaciones.
- Diseño de la(s) base de datos.
- Realización de pruebas a la solución propuesta para verificar la correspondencia de las funcionalidades con los requerimientos identificados previamente.

Estructura capitular

***Capítulo 1 "Fundamentación Teórica"*:** Se expondrán los principales conceptos relacionados con el tema de investigación. Contiene, además, una caracterización de la metodología, herramientas y tecnologías que serán utilizadas para el desarrollo de la propuesta de solución y una explicación de las razones que llevan a su selección.

***Capítulo 2 "Análisis, Exploración y Planificación"*:** Se realizará una descripción de la propuesta de solución y de las condiciones sobre las que debe operar, además de abordar el contenido referente al análisis y diseño del sistema, de acuerdo a la metodología de desarrollo seleccionada.

***Capítulo 3 "Implementación y Prueba"*:** A partir del análisis realizado en el capítulo 2, se generarán los

artefactos correspondientes a la metodología utilizada, y la propuesta de solución será sometida a una fase de pruebas.

Capítulo 1 Fundamentación Teórica

En este capítulo se describen algunos conceptos asociados al dominio de la investigación. Se abordan, además, herramientas y métodos utilizados para la administración y apoyo de otras regionales del ACM-ICPC en América Latina.

Por último, se recoge también un análisis sobre la selección de la metodología de desarrollo de software que se utilizará para elaborar la propuesta de solución. Bajo esta misma tónica se ofrece también un breve análisis sobre las diferentes herramientas y tecnologías a utilizar durante la elaboración de la propuesta de solución.

1.1 Conceptos asociados al dominio del problema

La programación competitiva es un deporte mental por equipos y, básicamente, consiste en dado un conjunto de problemas, resolver la mayor cantidad posible de estos, lo más rápido y eficiente posible.

En la realización de dichos concursos un grupo de jueces proponen varias tareas a resolver por cada uno de los equipos participantes, cada tarea se compone de uno o varios casos de pruebas y puede traer aparejado un evaluador externo. 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.

Existen dos tipos de formatos para estos eventos: a largo plazo, donde cada ronda del evento puede durar días y los de corto plazo, cuya duración oscila entre 1 y 3 horas. Dentro de estos últimos destaca, por su prestigio y nivel el ACM-ICPC, en el cual participan cada año estudiantes de varias universidades de todo el mundo.

Juez en línea: Un juez en línea es un sistema para evaluar soluciones en competencias de programación,

sirviendo, además, para practicar para dichos certámenes. Muchos de estos sistemas organizan sus propios eventos.

El sistema puede compilar y ejecutar código, así como evaluarlo con datos pre-construidos. El código presentado puede correr con restricciones, incluyendo límite de tiempo y de memoria, restricciones de seguridad, entre otras. 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.[3]

1.2 Análisis de soluciones similares

Con el objetivo de obtener una panorámica general sobre el comportamiento de los servicios brindados por la sede cubana durante la Final Caribeña así como de las herramientas que los brindan, en otras sedes de América Latina, se hizo circular una encuesta en la región (ver Anexo 1).

Tomando como medidores las regiones que respondieron a la encuesta (Sudamérica/Norte, Colombia, Argentina, México y Centroamérica) se obtuvieron los siguientes resultados:

SERVICIO DE IMPRESIÓN	SERVICIO DE RANKING	TRANSMISIÓN DEL EVENTO	PREMIOS ESPECIALES DE LA REGIONAL
75% BOCA	75% BOCA	25%TWITTER	50%

Tabla 1 Resultado de la encuesta

- De manera general se utiliza el BOCA para brindar los servicios especificados. Esto se debe, en gran medida, a la posibilidad de integración de estos servicios en el sistema, dado que es el oficial para estos certámenes en el área de América Latina y el Caribe, lo que garantiza un control centralizado.
- En el caso de la transmisión del evento vía web, la encuesta arrojó que no existe en estas sedes un sistema para la transmisión del evento. En pocas ocasiones algunas sedes han mantenido actualizada a la comunidad vía Twitter o en el caso del ITESO (Instituto Tecnológico y de Estudios

Superiores de Occidente es la Universidad Jesuita de Guadalajara, mediante un sistema de transmisión en vivo que brinda la institución.

- En el caso de los premios especiales, en las regiones entrevistadas solo se premian los tradicionales oro, plata y bronce, a diferencia de la sede cubana que ofrece 4 premios extras.
- Además, en el caso del servicio de impresión de código fuente, si bien es mayoritario el uso de la funcionalidad que incorpora el BOCA para ello, en la región de Sudamérica/Norte se trabaja con un sistema propio, por razones de confiabilidad y ajuste a la infraestructura interna de los organizadores.

El análisis de estos resultados permitió arribar a la conclusión de que no existe o hay muy poca personalización de estos servicios en las sedes encuestadas, por lo que estas suelen ajustarse a las necesidades de las aplicaciones existentes, cuando deberían ser las aplicaciones las que se ajusten a las necesidades del evento. Ninguna de las herramientas utilizadas por dichas sedes es adaptable a los intereses de los organizadores cubanos, por esta razón se decide elaborar una aplicación o un conjunto de ellas que den cumplimiento a estos intereses.

1.3 Metodologías de desarrollo de software

Los modelos prescriptivos(o tradicionales) de proceso se propusieron originalmente para ordenar el caos del desarrollo del software. La historia ha indicado que estos modelos tradicionales han traído consigo cierta cantidad de estructuras útiles para el trabajo en la Ingeniería de Software. Y han proporcionado un camino a seguir razonablemente efectivo para los equipos de software.[4]

“...Si los modelos prescriptivos del proceso buscan estructura y orden, ¿éstos resultan inapropiados para un mundo del software que se basa en el cambio?...”[5]

Para mayor entendimiento sobre las principales diferencias entre los enfoques prescriptivos y ágil de una forma simple, se propone la siguiente tabla comparativa.[6]

Criterio		Ágil	Tradicional
Aplicación	**Tamaño**	Pequeños equipos y proyectos	Pequeños equipos y proyectos
	Objetivo primario	Evaluación rápida, respuesta al cambio	Alta garantía, alta estabilidad, previsibilidad
	Ambiente	Turbulento	Estable

Ilustración 1 Metodologías de desarrollo de software

Tomando como referencia lo anterior expuesto y, enmarcándolo a las condiciones actuales del trabajo de diploma, es decisión de los autores centrarse en un desarrollo ágil para la elaboración de la solución, siguiendo para ello las siguientes pautas:

"…un proceso ágil de software debe adaptarse de forma incremental. Para llevar a cabo una adaptación incremental, un equipo ágil requiere de retroalimentación con el cliente. Un catalizador efectivo para la retroalimentación del cliente es un prototipo operacional o una porción del sistema operacional. Por lo tanto, debe instituirse una estrategia incremental de desarrollo. Los incrementos deben entregarse en cortos períodos para que la adaptación mantenga un buen ritmo con el cambio (imprevisibilidad). Este enfoque iterativo le permite al cliente evaluar el incremento del software de manera regular, proporcionar retroalimentación necesaria al equipo de software, e influir sobre las adaptaciones del proceso que se realizan para adecuar la retroalimentación."[5]

1.3.1 Selección de la metodología

Al ser un equipo pequeño, el cual se debe desenvolver en un ambiente con requisitos cambiantes, donde es necesario el trabajo en equipo sólido y la superación continua de los integrantes y es necesario que el

cliente se viese integrado en el proceso de confección del proyecto, los autores se decidieron por la metodología Programación Extrema o XP por sus siglas en inglés. Esta metodología sugiere algunas técnicas innovadoras y poderosas que permiten a un equipo ágil crear frecuentes lanzamientos de software al entregar características y funcionalidad que describe y después prioriza el cliente.

Algunas de las características que justifican el uso de esta metodología en la propuesta de solución son:

1. Producir rápidamente versiones del sistema que sean operativas, aunque no cuenten con toda la funcionalidad del sistema.
2. Se debe diseñar la solución más simple que pueda funcionar y ser implementada en un momento determinado del proyecto.
3. Toda la producción de código debe realizarse con trabajo en parejas de programadores.[5]

A pesar de todo esto, XP es una metodología centrada la implementación más que en cualquier otra fase del desarrollo de software, por lo que se hace necesario introducir otros artefactos en aras de lograr una comprensión del negocio lo más fiable posible. En este sentido se decide agregar el diagrama conceptual del negocio durante la fase de exploración, así como el modelo de datos.

Uno de los principios de XP es la programación en parejas. En este sentido, es decisión del equipo realizar una distribución del trabajo que permita a un integrante ir desarrollando una funcionalidad o un conjunto de estas, mientras el otro miembro del equipo realiza pruebas a las funcionalidades terminadas.

1.4 Lenguaje de modelado

UML son las siglas de “Unified Modeling Language” o “Lenguaje Unificado de Modelado”. Se trata de un estándar que se ha adoptado a nivel internacional por numerosos organismos y empresas para crear esquemas, diagramas y documentación relativa a los desarrollos de software (programas informáticos).[7] **UML** es un lenguaje de modelado de propósito general que pueden usar todos los modeladores. No tiene propietario y está basado en el común acuerdo de gran parte de la comunidad informática.

1.4.1 Selección de la herramienta CASE

Visual Paradigm

Se decidió utilizar como herramienta CASE para el modelado de la aplicación el Visual Paradigm en su versión 8.0 por ser multiplataforma y tener licencia de uso libre, lo que permitirá transferir los modelos obtenidos al cliente, al finalizar el desarrollo del sistema. Además el equipo de desarrollo posee experiencia previa con el uso del mismo.

Posee, además, las siguientes características, las cuales se ajustan a las necesidades del equipo de desarrollo:

- Integración con distintos Ambientes de Desarrollo Integrados (IDE): entre ellos el Netbeans.
- Disponibilidad en múltiples plataformas: Microsoft Windows (98, 2000, XP, Vista, Seven y 8), Linux, Mac OS X, Solaris o Java.

1.5 Tecnologías para el desarrollo de software

1.5.1 Sistemas Gestores de Base de Datos

Un sistema de gestión de bases de datos (**SGBD**) es un conjunto de programas que permiten el almacenamiento, modificación y extracción de la información en una base de datos, además de proporcionar herramientas para añadir, borrar, modificar y analizar los datos. Los usuarios pueden acceder a la información usando herramientas específicas de interrogación y de generación de informes, o bien mediante aplicaciones al efecto.[8]

Un SGBD debe prestar los siguientes servicios:

- Definición y creación de la base de datos.
- Manipulación de los datos realizando consultas, inserciones y actualizaciones.
- Acceso controlado a los datos mediante mecanismos de seguridad de acceso a los usuarios.
- Mantener la integridad de los datos.

- Controlar la concurrencia a la base de datos.
- Mecanismos de copias de respaldo y recuperación para restablecer la información en caso de fallos en el sistema.[8]

Los sistemas gestores de bases de datos se dividen en varias categorías, entre ellas:

- SQL: Bases de datos tradicionales que basan su funcionamiento en tablas, joins y transacciones.
- NoSQL: Las bases de datos no imponen una estructura de datos en forma de tablas y relaciones entre ellas, sino que proveen un esquema mucho más flexible.

1.5.1.1 Selección del sistema gestor de base de datos

Según los requisitos de la solución, definidos por el cliente, el SGBD a utilizar debe ser portable, o sea, no debe ser necesario instalar el mismo. Además, el sistema debe garantizar la integridad de los datos y el acceso a los mismos. Actualmente, existen dos ramas de clasificación para los SGBD: SQL y NoSQL. Primeramente se procedió a determinar cuál de estas utilizar para la propuesta de solución. Mediante un estudio se determinó utilizar los gestores SQL, ya que NoSQL destaca a la hora de dar solución a problemas de persistencia y almacenamiento masivo de datos para las organizaciones. En aplicaciones de poco tráfico de datos, a menos que existan condiciones específicas, es irrelevante utilizar una u otra variante. Debido a esto y, debido a que los autores se encuentran familiarizados con el lenguaje SQL, se decidió utilizar un SGBD de este tipo.

Después de realizar una investigación de las posibles herramientas SQL a utilizar, el equipo se decidió por SQLite, por ser portable, compatible con el modelo ACID (Atomicity, Consistency, Isolation and Durability) debido a su simplicidad y a su curva de aprendizaje.

1.5.2 Selección de los lenguajes de programación

1.5.2.1 HTML

HTML (Hyper Text Markup Language) es el lenguaje que se emplea para el desarrollo de páginas de internet. Está compuesto por una serie de etiquetas que el navegador interpreta y da forma en la pantalla.

HTML dispone de etiquetas para imágenes, hipervínculos que nos permiten dirigirnos a otras páginas, saltos de línea, listas, tablas, etc.[9]

Durante la presente investigación se utilizará HTML versión 5.0, ya que esta es la última hasta la fecha, incorpora varios elementos que sirven para estructurar mejor una página web, estableciendo qué es cada s*ección, así como un mayor soporte para realizar animaciones web basadas en JavaScript y CSS.*

1.5.2.2 CSS

CSS (Cascade Style Sheets) es un lenguaje utilizado en la presentación de documentos HTML. Un documento HTML viene siendo coloquialmente "una página web". Entonces podemos decir que el lenguaje CSS sirve para organizar la presentación y aspecto de una página web. Este lenguaje es principalmente utilizado por parte de los navegadores web de internet y por los programadores web informáticos para elegir multitud de opciones de presentación como colores, tipos y tamaños de letra, etc.[10]

La filosofía de CSS se basa en intentar separar lo que es la estructura del documento HTML de su presentación. Por decirlo de alguna manera: la página web sería lo que hay debajo (el contenido) y CSS sería un cristal de color que hace que el contenido se vea de una forma u otra. Usando esta filosofía, resulta muy fácil cambiarle el aspecto a una página web: basta con cambiar "el cristal" que tiene delante.[10]

Durante la investigación se utilizará CSS versión 3.0, por ser esta la de mayor referencia y usabilidad en la actualidad, además de poseer nuevas pseudoclases y selectores definidos, muy útiles para realizar animaciones web y estilizar sitios.

1.5.2.3 JAVASCRIPT

JavaScript es un lenguaje de programación que se utiliza principalmente para crear páginas web dinámicas. Una página web dinámica es aquella que incorpora efectos como texto que aparece y desaparece, animaciones, acciones que se activan al pulsar botones y ventanas con mensajes de aviso al

usuario.[11]

Técnicamente, JavaScript es un lenguaje de programación interpretado, por lo que no es necesario compilar los programas para ejecutarlos. En otras palabras, los programas escritos con JavaScript se pueden probar directamente en cualquier navegador sin necesidad de procesos intermedios.[11]

1.5.2.4 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.

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 en la que pueda correr la JVM. 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.[12]

1.5.2.5 JSON

JSON (JavaScript Object Notation) es un formato de texto que es completamente independiente de cualquier lenguaje de programación, pero usa convenciones que son familiares a los programadores de la familia del lenguaje C, incluyendo C, C++, C#, Java, JavaScript, Perl, Python, entre otros. Estas propiedades hacen de JSON un lenguaje ideal para el intercambio de datos.

JSON es construido en dos estructuras:

- Una colección de pares nombre/valor. En diversos lenguajes esto se realiza mediante un objeto, record, estructura, diccionario, tabla hash o array asociativo.
- Una lista ordenada de valores. En la mayoría de los lenguajes de programación, esto es llevado a cabo mediante un array, vector, lista o secuencia.[13]

1.5.3 Framework

En general, se refiere a una estructura software compuesta de componentes personalizables e intercambiables para el desarrollo de una aplicación. En otras palabras, un framework se puede considerar como una aplicación genérica incompleta y configurable a la que podemos añadirle las últimas piezas para construir una aplicación concreta.[14]

Un framework Web, por tanto, podemos definirlo como un conjunto de componentes (por ejemplo clases en java y descriptores y archivos de configuración en XML) que componen un diseño reutilizable que facilita y agiliza el desarrollo de sistemas Web.[14]

1.5.3.1 JQUERY

Este framework JavaScript, nos ofrece una infraestructura con la que tendremos mucha mayor facilidad para la creación de aplicaciones complejas del lado del cliente. Por ejemplo, con JQuery obtendremos ayuda en la creación de interfaces de usuario, efectos dinámicos, aplicaciones que hacen uso de Ajax,

etc. Cuando programemos JavaScript con JQuery tendremos a nuestra disposición una interfaz para programación que nos permitirá hacer cosas con el navegador que estemos seguros que funcionarán para todos nuestros visitantes. Simplemente debemos conocer las bibliotecas del framework y programar utilizando las clases, sus propiedades y métodos para la consecución de nuestros objetivos.[15]

Además, todas estas ventajas, con JQuery las obtenemos de manera gratuita, ya que el framework tiene licencia para uso en cualquier tipo de plataforma, personal o comercial. Para ello simplemente tendremos que incluir en nuestras páginas un script JavaScript que contiene el código de JQuery, que podemos descargar de la propia página web del producto y comenzar a utilizar el framework.[15]

1.5.3.2 ANGULARJS

AngularJS es una tecnología del lado del cliente, un framework JavaScript opensource desarrollado por Google utilizado principalmente para crear aplicaciones web de una sola página; funciona con las tecnologías web más asentadas a lo largo del tiempo (HTML, CSS y JavaScript) para hacer el desarrollo de aplicaciones web más fácil y rápido que nunca. El código fuente de Angular está disponible gratuitamente en Github bajo la licencia MIT. Esto significa que cualquier persona puede contribuir y ayudar en su desarrollo.[16]

Angular permite construir aplicaciones web modernas e interactivas mediante el aumento del nivel de abstracción entre el desarrollador y las tareas de desarrollo de aplicaciones web más comunes. Extendiendo el tradicional HTML con etiquetas propias (directivas), se podría decir que utiliza el patrón MVC (Model-ViewController), aunque ellos mismos lo definen como un MVW (Model-ViewWhatever (whatever works for you)).[16]

1.5.3.3 BOOTSTRAP

Bootstrap, es un framework originalmente creado por Twitter, que permite crear interfaces web con CSS y JavaScript, cuya particularidad es la de adaptar la interfaz del sitio web al tamaño del dispositivo en que se visualice. Es decir, el sitio web se adapta automáticamente al tamaño de una PC, una Tablet u otro dispositivo. Esta técnica de diseño y desarrollo se conoce como “responsive design” o diseño

adaptativo.[17]

1.5.3.4 SPRING

Spring es la solución Java Web creada como contraposición a los EJB (Enterprise Java Beans) de Java, basada en los tradicionales POJO. Como tal es un framework que se caracteriza por su arquitectura modular caracterizada por el bajo acoplamiento y la posibilidad de integrar y delegar funcionalidades a distintos framework. Spring está pensado bajo los principios de la programación orientada a objetos y como tal funciona bajo tales, entre los que se encuentran la inyección de dependencias, uso de singletons, etc.[18]
En la actualidad la combinación Spring+Struts+Hibernate es considerada el estándar defacto de las soluciones Java Web.

1.5.4 Selección del entorno de desarrollo (IDE)

Para la investigación que se lleva a cabo se decide utilizar el Netbeans en su versión 8.0 ya que existe una experiencia previa de trabajo con el mismo. Además, incorpora los servicios de servidor web (Apache Tomcat y GlassFish), control de versiones y servicios de base de datos. Algunos elementos considerados fueron:

- Suele dar soporte a casi todas las novedades en el lenguaje Java.
- Asistentes para la creación y configuración de distintos proyectos, incluida la elección del framework Spring, el cual viene agregado de forma nativa en el IDE.
- Buen editor de código, multilenguaje, con el habitual coloreado y sugerencias de código, acceso a clases pinchando en el código, control de versiones, localización de ubicación de la clase actual, comprobaciones sintácticas y semánticas, plantillas de código, coding tips, herramientas de refactorización, entre otras.[19]

Conclusiones parciales

Luego de haber realizado un análisis sobre la prestación de los servicios definidos en la encuesta (ver anexo 1) durante la celebración de la Final Caribeña del ACM-ICPC, dentro del área de Latinoamérica, se

evidenció la generalidad del juez en línea BOCA como herramienta para ello. Sin embargo, la sede cubana ha desarrollado sus propias aplicaciones para prestar dichos servicios, debido a que se adaptan mejor a sus intereses y estructura, además de posibilitar una mejor integración con otras funcionalidades propias de la universidad, las cuales no están integradas en el BOCA, ni se llevan a cabo en otras sedes regionales, tal es el caso de la transmisión en vivo, vía web, del evento. Por ende, se decide trabajar sobre la base de esta cartera de servicios ya existentes en la sede cubana, en aras de mejorar su eficiencia y usabilidad, en lugar de utilizar las funcionalidades del BOCA que sirven a este propósito.

Por otra parte, mediante el estudio sobre metodologías, herramientas y lenguajes de programación, se decide utilizar Visual Paradigm 8.0 como herramienta CASE, Java, JavaScript ,CSS y HTML como lenguajes de programación, Spring, Twitter Bootstrap, AngularJS y JQuery como framework de desarrollo y como entorno integrado de desarrollo el Netbeans 8.0. Además, se definió el uso de SQLite para el trabajo con las bases de datos.

Capítulo 2 Análisis, Exploración y Planificación

En este capítulo se brinda una panorámica del escenario en el que se desarrollarán las aplicaciones elaboradas como propuesta solución, así como una breve descripción de las mismas. Luego se sigue el cumplimiento de las fases a deben seguir al aplicar una metodología de desarrollo, la cual en nuestro caso es XP, generan los diferentes artefactos que dan paso al desarrollo exitoso de la propuesta de solución.

2.1 Modelo conceptual del negocio

La solución propuesta está dedicada a informatizar los servicios de información del evento y confección de la tabla de posiciones, así como actualizar los de impresión de código fuente y determinación de premios especiales. El comportamiento de dichos servicios durante la Final Caribeña se detalla a continuación:

Durante la celebración del evento cada equipo puede, de manera opcional, acceder a una aplicación de código fuente para generar una solución a un problema del certamen. Este sistema pone a disposición de los usuarios todos los problemas presentes en la Final Caribeña del ACM-ICPC. Cada usuario puede generar una o varias soluciones para cualquiera de estos problemas, para posteriormente imprimirlas. Adicionalmente, un equipo puede al terminar la competencia, obtener todos los envíos que generó como respuesta a cada uno de los problemas. Estos servicios son exclusivamente para los equipos de la justa y se encuentran controlados en su totalidad por los organizadores de la final.

Existen, además de los anteriormente expuestos, otros servicios que son de acceso general, tanto para los participantes como para la audiencia en general, los cuales se encargan de brindar información sobre el evento, tanto del desarrollo del evento como de la historia y ediciones anteriores del mismo. Así, se tiene el caso de la tabla de posiciones, la cual es actualizada periódicamente por el jurado en línea después de cada envío que realiza un usuario. Se cuenta, además, con una serie de premios especiales, los cuales están disponibles después de finalizado el evento y se determinan siguiendo las estadísticas generales que recoge el BOCA durante el certamen.

Con el objetivo de lograr una mayor comprensión de lo anteriormente expuesto, se procede a elaborar un

diagrama conceptual, en aras de capturar y expresar el entendimiento ganado sobre el negocio.

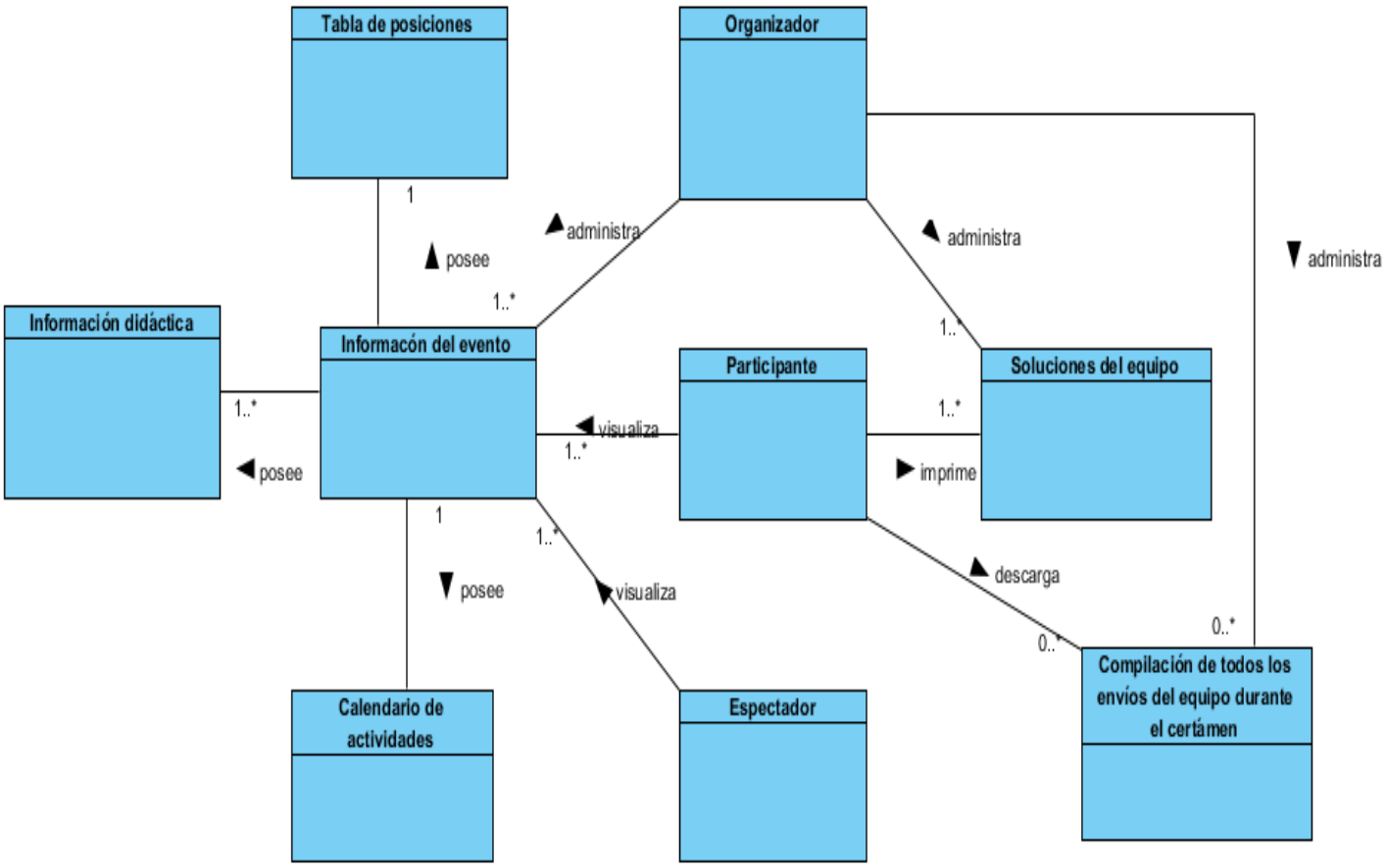


Ilustración 2 Diagrama conceptual

- **Participante**: Identifica a los equipos que participan directamente en la Final Caribeña del ACM-ICPC, los cuales tendrán acceso a la totalidad de los servicios.
- **Organizador:** Identifica a los organizadores del evento, los cuales, además de tener acceso a la totalidad de los servicios poseen privilegios administrativos en cada uno de ellos.
- **Espectador**: Identifica a todos los usuarios que forman parte de la audiencia del evento. Los espectadores cuentan con acceso únicamente a los servicios informativos del evento.
- **Soluciones del equipo**: Identifica cada una de las soluciones a un problema, generadas a través de la herramienta de impresión de código fuente.

- **Información del evento**: Constituye el servicio informativo del evento y se encuentra disponible para la totalidad de los roles involucrados en el negocio (Participante, Organizador y Espectador).
- **Tabla de posiciones**: Identifica el servicio de ranking oficial de la final caribeña del ACM-ICPC.
- **Información didáctica**: Identifica una aplicación que brinda información sobre la historia y desarrollo del ACM-ICPC, así como sobre de los países caribeños que participan en ella.

2.2 Descripción de la solución

Producto a la diversidad de servicios sobre los que va a estar centrada la propuesta de solución del trabajo, se hace necesario fraccionar la misma en múltiples aplicaciones, cada una con funcionalidades específicas y totalmente independientes del resto. En general, las aplicaciones a desarrollar son las siguientes:

- **PrintServer**: como servidor de impresión de código fuente.
- **DataTransformer**: brinda el servicio de calcular los premios especiales de la sede cubana, además de brindar soporte al servicio de ranking.
- **Slider**: se encarga de proveer información a los que sigan el evento.

A continuación se brinda una descripción de cada una de estas herramientas.

2.2.1 PrintServer

La aplicación tiene como objetivo administrar y gestionar los servicios de impresión de código fuente y descarga de las soluciones individuales que se brindan durante el certamen en la sede cubana. Para esto, el sistema cuenta con cuatro módulos iniciales: módulo de usuarios, de problemas, de código y de descargas, los cuales permiten llevar un registro de la actividad de cada usuario, a la vez que permiten a los mismos la impresión de código fuente y la descarga de las soluciones que cada equipo realizó en la competencia. Cada uno de estos módulos posee dos niveles de acceso: nivel usuario y nivel administrador. El nivel de administrador es el responsable de la gestión del sistema, pudiendo acceder a la totalidad de las funcionalidades del mismo, entre las que se incluyen la creación de nuevos usuarios, los

cuales, inicialmente, tendrán nivel de acceso de usuario.

El ser una aplicación web, el sistema posee una arquitectura de tipo cliente servidor. Del lado del cliente destacan las tecnologías HTML, javascript mediante el uso de JQuery 1.9.2 y CSS 3 a través del framework Bootstrap 2. Por su parte, del lado del servidor se utilizó Apache Tomcat 8 como servidor web y Java 1.8 como lenguaje de programación, además de SQlite como gestor de bases de datos.

Dentro de la herramienta existen dos funcionalidades que constituyen el aporte real a los servicios ofertados durante la Final Caribeña. Por una parte se encuentra la funcionalidad de impresión de código fuente, la cual se detalla a continuación: El proceso comienza realizando una petición al servidor. Dentro de esta petición se envía, adicionalmente, el código deseado. Paso seguido la petición es capturada por un controlador, el cual realiza una llamada a un servicio de impresión junto con los parámetros: nombre del problema al que está relacionado el código, contenido del código y nombre del usuario que realizó la petición. Estos parámetros son enviados al script imprimir.sh, disponible bajo el directorio printscript/scripts de la aplicación, mediante una llamada al mismo. Este script es el encargado, como tal, de gestionar la cola y la actividad de impresión. Estas peticiones de impresión, al llevar implícitas un código, pueden ser realizadas de dos formas diferentes, bien a través de la opción imprimir disponible en cualesquiera de la lista de códigos generados del usuario, o bien mediante la creación de un nuevo código, debido a que al generar un nuevo código automáticamente el sistema procede a su impresión. Generar un nuevo código por su parte, puede llevarse a cabo de dos formas: pulsando en la opción imprimir disponible para cada uno de los problemas, en cuyo caso el sistema asocia el nuevo código con dicho problema, o mediante la opción nuevo código, en cuyo caso no ocurre la asociación anterior de forma automática, para lo que debemos especificar el nombre del problema en cuestión. El siguiente diagrama ofrece una representación de esta funcionalidad.

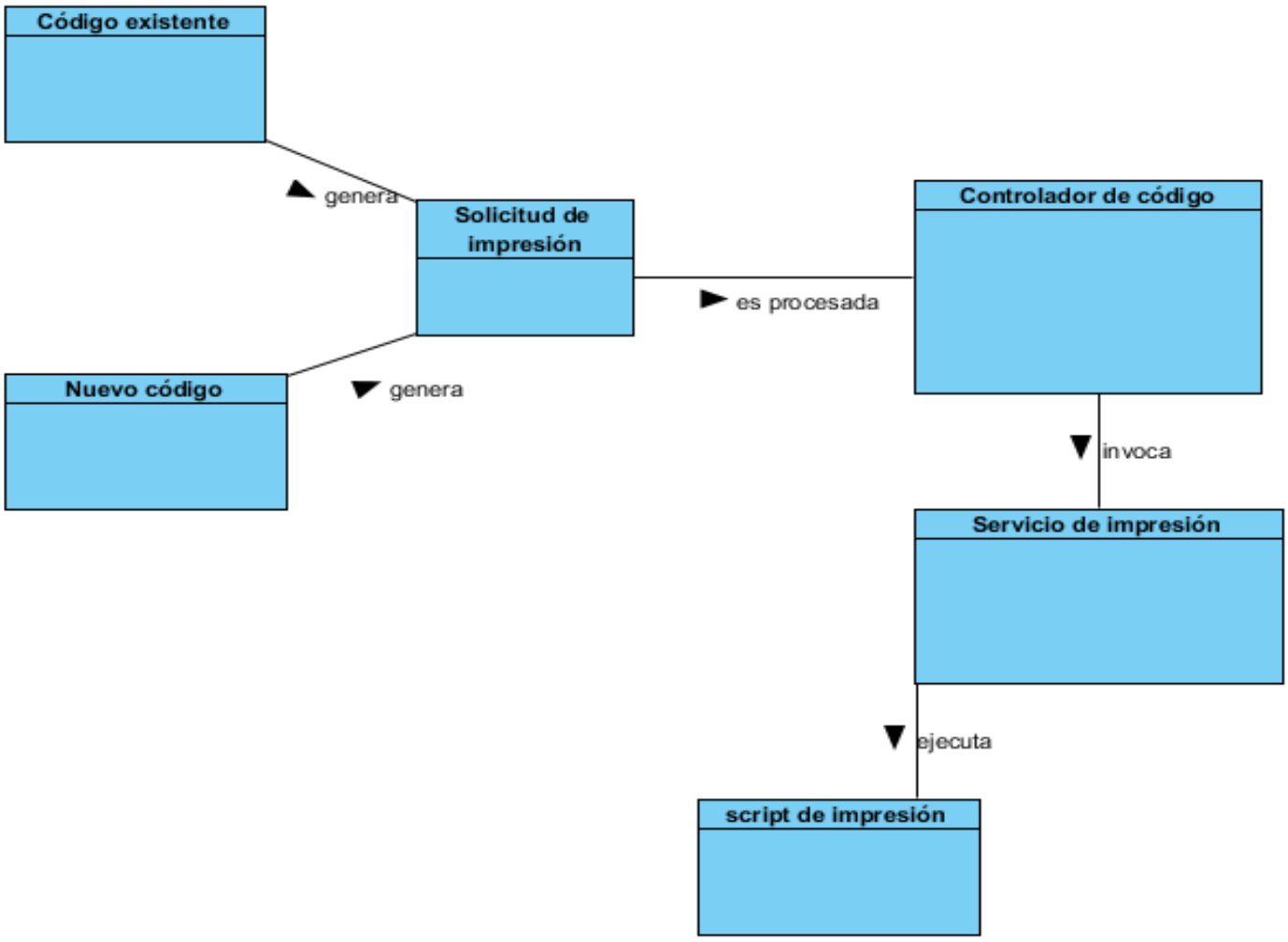


Ilustración 3 Funcionalidad imprimir.

La otra funcionalidad de alto valor consiste en la opción que se le brinda a cada equipo de descargar todas las soluciones que el mismo generó durante la Final Caribeña del ACM-ICPC. El procedimiento comienza cuando el administrador realiza una llamada al script gpt.sh, disponible bajo el directorio printscript/scripts de la aplicación, el cual se encarga de crear, a partir de un respaldo de la base de datos del BOCA obtenido después de finalizado el certamen, un conjunto de archivos bajo la extensión .tar.bz2. El nombre de cada archivo coincide con el de cada equipo que participa en el evento y contiene todos los envíos que el mismo realizó. Después de generados, el sistema asocia cada uno de estos archivos a un

usuario, de forma tal que tanto el nombre del usuario como del archivo coincidan. De no existir coincidencia con ningún usuario, se brinda la posibilidad de asignarle uno de forma manual. Finalmente, se permite el acceso a cada archivo de descarga a todos los usuarios que tienen asociado alguno de estos archivos. A continuación se ofrece una descripción visual de esta funcionalidad.

Ilustración 4 Funcionalidad descargas.

Existen, además de estas funcionalidades, existen otras que permiten el control y seguimiento de las actividades de los usuarios dentro de la aplicación. En este sentido se encuentran las operaciones CRUD[1], comunes a los cuatro módulos.

Cabe destacar que la herramienta fue pensada considerando cada usuario del sistema como un equipo del certamen. Por ende, un usuario puede tener una y solo una sesión activa. Esto, junto con otras cuestiones como las políticas de control de acceso, se logra a través del framework Spring Security 3.2.6.

[1] Create Read Update Delete, por sus siglas en inglés.

2.2.2 DataTransformer

La aplicación brinda soporte al servicio de elaboración del ranking del ACM-ICPC para la región del Caribe, además de informatizar el servicio concerniente a la determinación de los premios especiales. Debido a que ambos servicios se llevan a cabo una sola vez, después de finalizada la competencia, se decide el DataTransformer como una aplicación Java sencilla, orientada al ambiente de escritorio en vez de la web. La misma se encuentra dividida en dos módulos, el módulo SARIS y el módulo Premios.

El módulo Saris se encarga de revisar el archivo a partir del cual la herramienta SARIS elabora el ranking oficial del evento, permitiendo la exclusión de dicho fichero de los usuarios se sabe no son participantes oficiales, como los jueces. Esto se lleva a cabo a través de la biblioteca Gson, la cual permite el acceso, modificación y creación de ficheros de tipo JSON, como es el caso. Adicionalmente, exporta los cambios realizados en un nuevo archivo, permitiendo así la preservación del original. De modo general, el método utilizado para llevar a cabo estas operaciones es el siguiente: una vez leído el archivo se procede a almacenar en una lista todos los usuarios recogidos en el mismo, así como cada uno de sus envíos. Posteriormente, cada vez que se decide eliminar un usuario, el sistema lo excluye de esta lista y lo guarda en otra destinada a los usuarios que van siendo descartados. Esto permite que no se pierda la información concerniente a estos usuarios, en caso de que se quiera restaurar el mismo a la lista original, mediante la opción restaurar usuario. A continuación se brinda una representación visual del proceso.

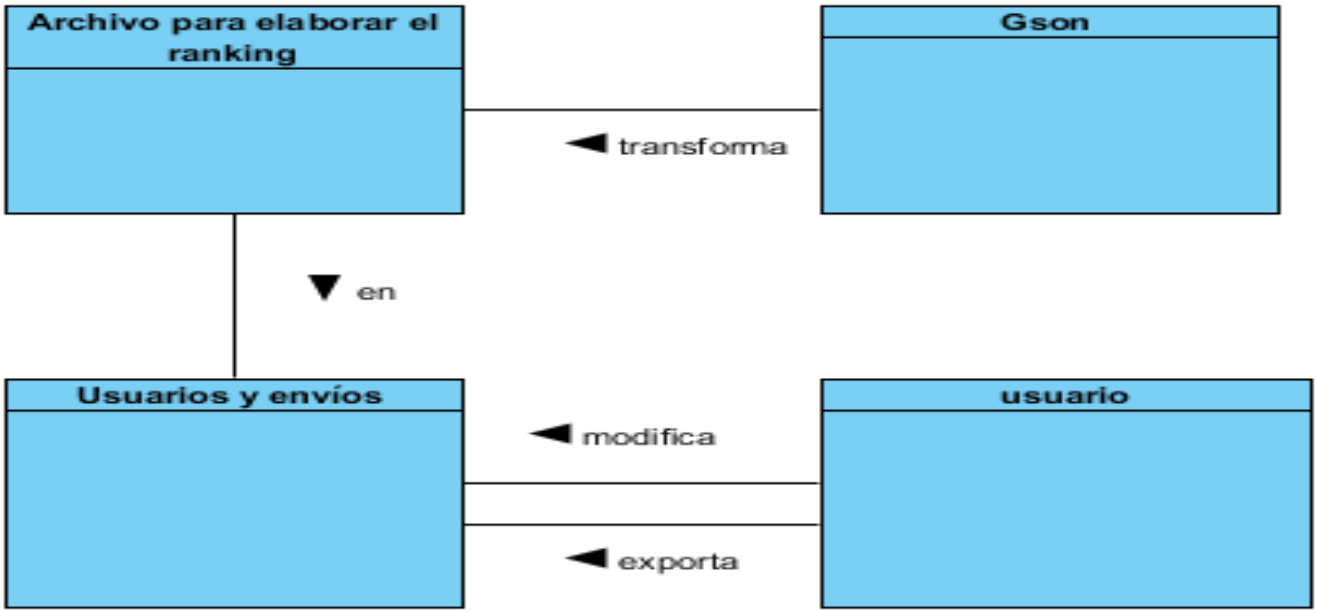


Ilustración 5 Módulo Saris.

El módulo Premios, por su parte, permite determinar los ganadores de los premios que se otorgan en la sede cubana, visualizarlos y exportarlos a una página web para su publicación. Para calcular los premios se utiliza el ranking oficial emitido por el juez en línea BOCA, al finalizar el evento competitivo. La información de este archivo es transformada mediante la biblioteca JSOUP, a través de una entidad especializada, en información entendible para la clase encargada de determinar los premios. Cabe destacar que mientras la información pueda ser transformada, otro archivo de entrada es igualmente válido para determinar los premios. El siguiente diagrama ofrece una representación de cómo se lleva a cabo este proceso.

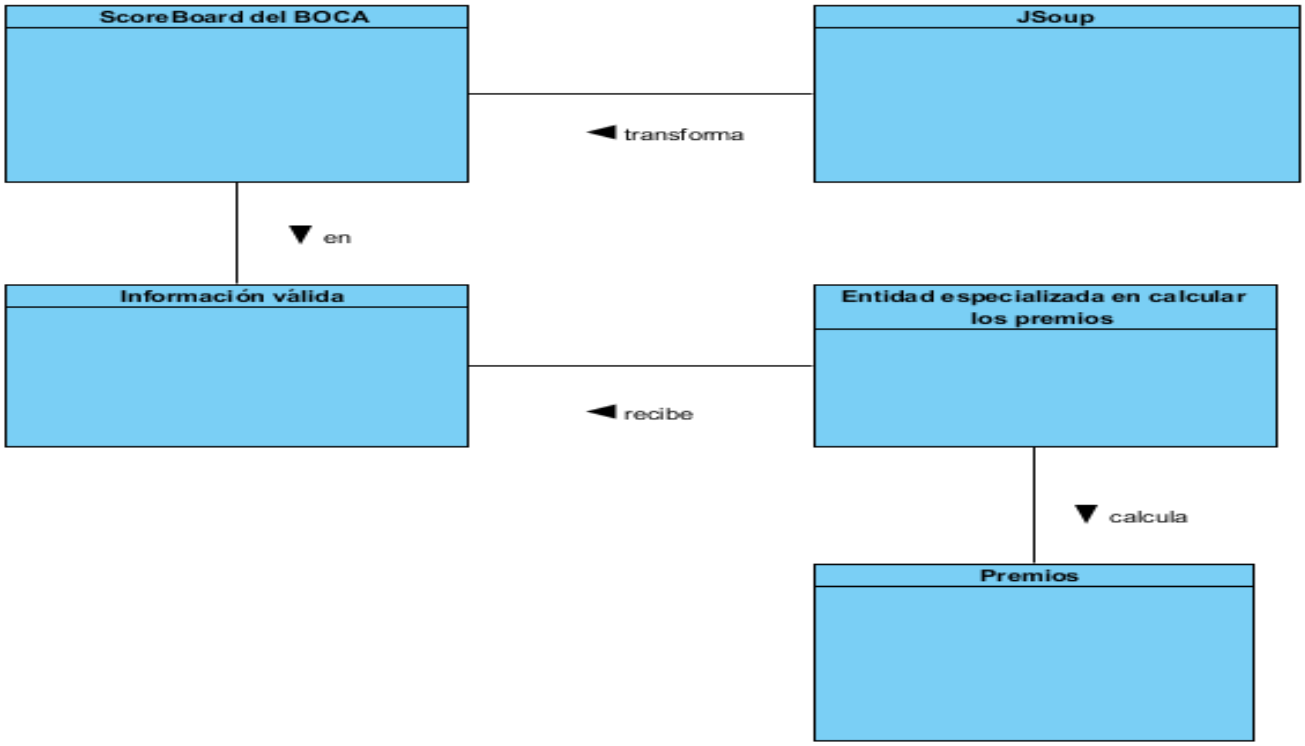


Ilustración 6 Módulo Premios.

2.2.3 Slider

La aplicación tiene como objetivo principal proveer de información a los que siguen el evento o deciden acceder a la misma la misma se compone de tres secciones: Counter, Activities y Gallery.

- En la sección Counter se mostrará un reloj que indica el tiempo restante para un evento previamente anunciado. El mismo puede ser utilizado además para indicar el tiempo de duración luego de iniciado el evento en su respectivo local de desarrollo. La fecha del evento será introducida a través de una función javascript.
- En la sección Activities se muestran las actividades planificadas para la realización de un evento. Además la misma cuenta con una opción de búsqueda, por si se desea buscar una actividad en específico.
- En la sección Gallery contiene otras tres secciones donde se realiza la presentación de los países

y sus respectivas universidades que integran el ACM-ICPC en el Caribe, una galería de fotos sobre eventos pasados de clasificación y una breve reseña histórica acerca del ACM-ICPC, además de mostrar datos sobre las ediciones pasadas de las finales mundiales del evento competitivo en cuestión.

A continuación se muestra un mapa de la aplicación:

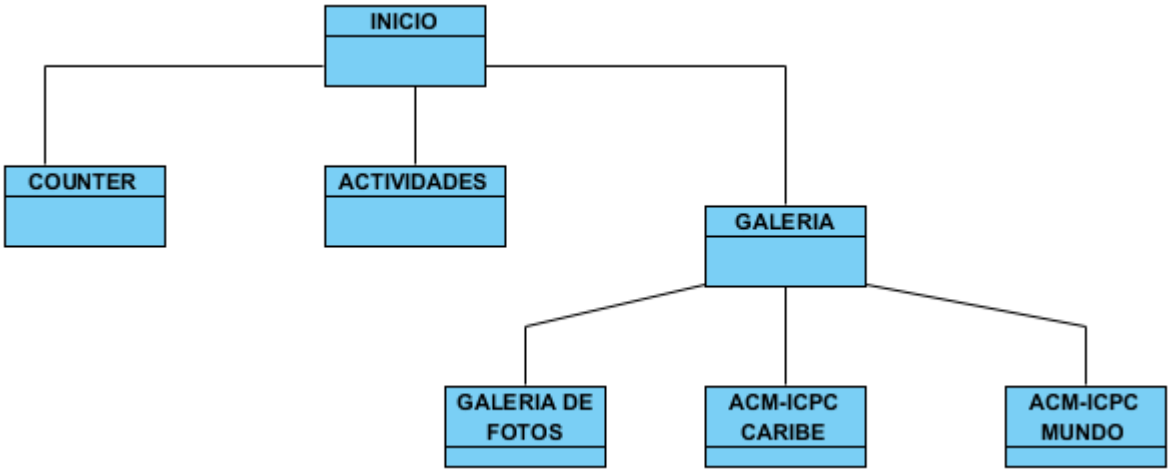


Ilustración 7 Mapa del Slider.

El Slider fue realizado bajo el concepto de diseño adaptativo, más conocido como “responsive design” para que el mismo fuera visto y accedido desde varios dispositivos y adaptarse a su tamaño. También fueron utilizados los framework AngularJS y jQuery para facilitar el manejo de los eventos realizados en el mismo, a través del uso de directivas y funciones del framework AngularJS. En el mismo se hizo uso de una biblioteca javascript llamada Ammap para la realización de mapas sin tener la necesidad de estar conectado a internet, ni hacer uso de las API que ofrece Google, ya que las mismas requieren de conexión permanente a internet, además que los bancos de imágenes existentes poseen capacidades de almacenamientos muy grandes, el tamaño mínimo hallado fue de 1Gb, y ello no permitía la realización de una aplicación que cumpliera con los requerimientos del cliente.

2.3 Lista de reserva del producto

Las listas de reserva del producto son el equivalente a los requerimientos funcionales que se obtienen al utilizar la metodología de desarrollo RUP. A continuación se presentan los diferentes requerimientos funcionales que se definieron con el cliente para llevar a cabo la realización de las distintas aplicaciones.

2.3.1 Para el PrintServer

2.3.1.1 Gestionar problema

- Crear problema: Adiciona un nuevo problema al sistema.
- Listar problemas: Muestra en una lista todos los problemas existentes en el sistema.
- Editar problema: Permite modificar los datos del problema seleccionado.
- Visualizar problema: Visualiza el problema seleccionado.
- Eliminar problema: Elimina el problema seleccionado.

2.3.1.2 Gestionar código

- Crear código a partir de un problema conocido: Permite crear un nuevo código al elegir la opción imprimir presente en cada uno de los problemas listados.
- Crear código sin conocer el problema. Permite crear un nuevo código especificando el nombre del problema al cual se vinculará.
- Listar todos los códigos: Muestra en una lista todos los códigos existentes en el sistema.
- Listar códigos del usuario: Muestra en una lista todos los códigos del usuario autenticado.
- Editar código: Permite modificar los datos del código seleccionado.
- Visualizar código: Visualiza el código seleccionado.
- Eliminar código: Elimina el código seleccionado.
- Imprimir código: Permite imprimir un código previamente creado.

2.3.1.3 Gestionar usuario

- Crear usuario: Adiciona un nuevo usuario al sistema.
- Listar usuarios: Muestra en una lista todos los usuarios existentes en el sistema.
- Editar usuario: Permite modificar los datos del usuario seleccionado.
- Editar perfil: Permite editar la clave de acceso del usuario autenticado.
- Visualizar usuario: Visualiza el usuario seleccionado.
- Eliminar usuario: Elimina el usuario seleccionado.
- Expulsar usuario: Permite expulsar a un usuario cuya sesión esté activa.

2.3.1.4 Gestionar descargas

- Gestionar descargas: Permite generar todas las soluciones de cada usuario durante la competencia, para posteriormente brindar la opción a cada usuario de descargar sus y solo sus soluciones.
- Obtener descargas: Permite a un usuario obtener sus soluciones a los problemas del evento.

2.3.2 Para el DataTransformer

- Editar Sarislog: El sistema permitirá importar el archivo Sarislog.txt, utilizado para elaborar el ranking final de la competencia.
- Generar premios especiales: A partir del ranking oficial emitido por el BOCA, genera los premios especiales, propios de la región caribeña.

2.3.3 Para el Slider

- Mostrar actividades definidas para un evento previamente anunciado.
- Mostrar reloj con el tiempo restante para un evento previamente anunciado.
- Mostrar galería de fotos acerca de eventos realizados.
- Mostrar presentación de los países del Caribe que participan en la edición regional del ACM-ICPC.
- Mostrar breve reseña histórica acerca del ACM-ICPC, además de la ubicación geográfica de las

finales del mismo desde sus inicios.

2.4 Requisitos no funcionales del sistema

Los requisitos no funcionales son características que de una u otra forma puedan limitar el sistema, por ejemplo el rendimiento, (en tiempo y espacio), interfaces de usuario, fiabilidad (robustez del sistema, disponibilidad del equipo), mantenimiento, seguridad, portabilidad, estándares, entre otros.[20] Los requerimientos no funcionales definidos para las distintas aplicaciones son:

Software:

- Se requiere JDK en su versión 7.0 o superior.
- Sistema operativo: Alguna de las distribuciones de GNU/Linux.
- Servidor web: Apache Tomcat 8 o GlassFish 4.
- Navegador Web.

Hardware:

- Memoria RAM: 512 MB a 1 Gb como mínimo.
- Espacio en disco duro: 60 MB mínimo.

2.5 Exploración

Es la fase en la que se define el alcance general del proyecto. En esta fase, el cliente define lo que necesita mediante la redacción de sencillas “historias de usuarios”. Los programadores estiman los tiempos de desarrollo en base a esta información. Debe quedar claro que las estimaciones realizadas en esta fase son primarias (ya que estarán basadas en datos de muy alto nivel), y podrían variar cuando se analicen más en detalle en cada iteración. Esta fase dura típicamente un par de semanas, y el resultado es una visión general del sistema, y un plazo total estimado.[21]

2.5.1 Historias de usuario de la propuesta de solución

La historia de usuario es una técnica aplicada en la metodología de desarrollo ágil XP, para sustituir los documentos de especificación funcional y a los casos de uso y, de esa manera, especificar los requisitos de software. Estas historias son escritas por el cliente en su propio lenguaje, como descripciones cortas de lo que el sistema debe realizar. Las historias de usuario además se utilizan para realizar la estimación del tiempo de implementación por lo deben ser programadas en un tiempo de duración entre una y tres semanas. Si la estimación es superior a las tres semanas debe ser dividida en dos o más historias.[21]

Hasta ahora la metodología XP no ha definido una estructura específica para representar las historias de usuario, por lo que se realiza una plantilla de acuerdo a las necesidades del proyecto en cuestión.

A continuación se exponen las diferentes historias de usuario, agrupadas por aplicación.

2.5.1.1 PrintServer

Historia de usuario	
Número: 1	**Nombre:** Gestionar problema
Usuario: Administrador y Usuario	
Prioridad: Media	**Complejidad:** Medio
Estimación: 1 semana	**Iteración:** 1
Descripción: El sistema permitirá crear un problema, así como listar todos los problemas existentes y, para cada uno, brindar las opciones de editar, visualizar y eliminar. De todas estas acciones, listar problemas es de dominio público (no se requiere estar autenticado en el sistema para tener acceso a él), visualizar es del dominio del usuario estándar, al resto solo se tiene acceso con el rol Administrador.	
Observación: Cuando se listan los problemas, existe para cada uno otra opción, la cual genera un nuevo código a partir del problema seleccionado.	

Tabla 2 HU 1-Gestionar problema

Historia de usuario	
Número: 2	**Nombre:** Gestionar código

Usuario: Administrador y Usuario	
Prioridad: Media	**Complejidad:** Alto
Estimación: 3 semanas	**Iteración:** 2
Descripción: El sistema permitirá • 1-Crear un nuevo código de dos formas diferentes. i. -Generar un nuevo código a partir de un problema seleccionado. A esta funcionalidad se tiene acceso desde la lista de problemas del sistema. ii. -Generar un nuevo código especificando el nombre del problema. El usuario debe especificar a qué problema está relacionado el nuevo código a generar. Se accede mediante el menú MyCodes->New Code. • Independientemente de la opción seleccionada, cada vez que se genera un nuevo código, el mismo se imprime automáticamente. • 2-Listar todos los códigos. Lista los códigos de todos los usuarios, ordenados por usuario. Esta opción es accesible solo al rol Administrador. • 3-Listar códigos del usuario. Lista los códigos del usuario autenticado. • 4-Editar, visualizar, Re-imprimir y eliminar código.	
Observación: Todas las funcionalidades requieren estar autenticado.	

Tabla 3 HU 2-Gestionar código

Historia de usuario	
Número: 3	**Nombre:** Gestionar usuarios
Usuario: Administrador	
Prioridad: Media	**Complejidad:** Alto
Estimación: 3 semanas	**Iteración:** 1

Descripción: El sistema permitirá • 1-Crear un nuevo usuario. • Por defecto, todos los usuarios que se creen tendrán rol de usuario estándar. Para promover un usuario estándar a administrador o lo contrario, se debe hacer por la acción de editar usuario. • 2-Editar usuario. Permite cambiar la contraseña de un usuario, así como el rol que posee. Después de modificar el rol, el usuario editado, de encontrarse autenticado, será expulsado del sistema para que pueda volver a acceder con los nuevos permisos. • 3-Mostrar usuario. Muestra la información de usuario (contraseña y nombre de usuario), así como una gráfica con todos los códigos creados por el usuario hasta el momento, agrupados por problema. • 4-Eliminar usuario. • 5-Expulsar usuario. Permite expulsar un usuario cuya sesión esté activa. • 6-Listar usuarios. Muestra información de todos los usuarios del sistema en cuanto a última dirección ip que utilizó, si se encuentra activo (autenticado) y brinda las opciones 2-6 por cada usuario.
Observación: Todas las funcionalidades requieren estar autenticado como administrador.

Tabla 4 HU 3-Gestionar usuarios

Historia de usuario	
Número: 4	**Nombre:** Gestionar descargas
Usuario: Administrador y Usuario	
Prioridad: Media	**Complejidad:** Bajo
Estimación: 2 semanas	**Iteración:** 2

Descripción: El sistema permitirá, a partir de una salva de la base de datos del BOCA, generar todas las soluciones de cada usuario durante la competencia, para posteriormente brindar la opción a cada usuario de descargar sus y solo sus soluciones. Debido a que el nombre del usuario en el PrintServer no necesariamente debe coincidir con el nombre del usuario durante la competencia, se brindará la posibilidad de asignar de forma manual un usuario del sistema a un archivo de soluciones, en caso de que el nombre del archivo no coincida con el de ningún usuario. Esta funcionalidad es válida solo para administradores. Se brindará, además, la opción de descarga de soluciones para cada usuario, la cual se hará visible solo cuando el administrador del sistema estime conveniente. Esta funcionalidad no estará accesible para el rol administrador, ya que se entiende que el mismo no participa en la competencia.
Observación: Las soluciones serán generadas en un compactado con extensión .tar.gz.

Tabla 5 HU 4-Gestionar descargas

Historia de usuario	
Número: 5	**Nombre:** Funcionalidades básicas.
Usuario: Usuario, Administrador	
Prioridad: Baja	**Complejidad:** Bajo
Estimación: 1 semana	**Iteración:** 2
Descripción: El sistema permitirá la opción de autenticarse en el sistema. De no existir previamente el usuario, el mismo será registrado y autenticado automáticamente. En este mismo sentido se brindará la opción de salir del sistema. El sistema brindará, además, la opción de editar perfil, la cual permitirá al usuario autenticado cambiar su contraseña de acceso.	
Observación:	

Tabla 6 HU 5-Funcionalidades básicas

2.5.1.2 DataTransformer

Historia de usuario	
Número: 6	**Nombre:** Editar Sarislog
Usuario: Administrador	

Prioridad: Baja	**Complejidad:** Bajo
Estimación: 1 semana	**Iteración:** 3
Descripción: El sistema permitirá importar el archivo Sarislog.txt, utilizado para elaborar el ranking final del certamen, con el objetivo de eliminar usuarios que pudiesen no ser participantes sino jueces. Adicionalmente, se guardarán los cambios realizados en un nuevo documento.	
Observación:	

Tabla 7 HU 6- Editar sarislog

Historia de usuario	
Número: 7	**Nombre:** Generar premios especiales
Usuario: Administrador	
Prioridad: Media	**Complejidad:** Bajo
Estimación: 1 semana	**Iteración:**3
Descripción: El sistema permitirá, a partir del ranking oficial emitido por el BOCA, para la región de América Latina y el Caribe, generar los premios especiales, propios de la regional caribeña para, posteriormente, exportarlos a una página web que puede ser utilizada a conveniencia.	
Observación: Existe un premio especial, llamado equipo progreso, para el cual necesita importar el ranking correspondiente a la edición anterior del evento.	

Tabla 8 HU 7-Generar premios especiales

2.5.1.3 Slider

Historia de usuario	
Número: 8	**Nombre:** Generar Slider
Usuario: Administrador	
Prioridad: Media	**Complejidad:** Bajo
Estimación: 1 semana	**Iteración:**3

Descripción: Este sistema se encarga de proveer al usuario de información referente acerca del ACM-ICPC siendo el mismo dividido en varias secciones para lograr su objetivo, dichas secciones se muestran a continuación: • Mostrar actividades definidas para un evento previamente anunciado. • Mostrar reloj con el tiempo restante para un evento previamente anunciado. • Mostrar galería de fotos acerca de eventos realizados. • Mostrar presentación de los países del Caribe que participan en la edición regional del ACM-ICPC. • Mostrar breve reseña histórica acerca del ACM-ICPC, además de la ubicación geográfica de las finales del mismo desde sus inicios.
Observación: Estas opciones son accedidas con el rol de usuario estándar de la página.

Tabla 9 HU 8-Generar Slider

2.6 Planificación

Esta fase de la metodología XP se plantea como objetivos estimar la prioridad de cada una de las historias de usuario que se generan en esta misma etapa y establecer el contenido de la primera entrega.[22] Estos objetivos se logran a través del diálogo constante entre las partes involucradas en el proyecto, incluyendo al cliente, los programadores y los coordinadores o gerentes.[21]

Como resultado de la realización de esta fase se encuentran la realización del Plan de entregas y el Plan de iteraciones.

2.6.1 Plan de entregas

El plan de entregas establece qué historias de usuario serán agrupadas para conformar una entrega, y el orden de las mismas. Este cronograma será el resultado de una reunión entre todos los actores del proyecto (cliente, desarrolladores, gerentes, etc.). Típicamente el cliente ordenará y agrupará según sus prioridades las historias de usuario. El plan de entregas se realiza en base a las estimaciones de tiempos de desarrollo realizadas por los desarrolladores. Luego de algunas iteraciones es recomendable realizar nuevamente una reunión con los actores del proyecto, para evaluar nuevamente el plan de entregas y

ajustarlo si es necesario.[21]

A continuación se presenta el plan de entregas definidos por el cliente y sus prioridades definidas, incluyendo al equipo de desarrollo con las estimaciones previamente realizadas.

Entregable	1ra entrega (3ra semana marzo)	2da entrega (2da semana abril)	3ra entrega (1ra semana mayo)
Propuesta solución.	V 0.1	V 0.2	V 0.3

Tabla 10 Plan de entregas

Historia de Usuario	1ra entrega	2da entrega	3ra entrega
HU1. Gestionar problema.	X	X	X
HU2. Gestionar código.		X	X
HU3. Gestionar usuario	X	X	X
HU4. Gestionar descargas		X	X
HU5. Funcionalidades básicas		X	X
HU6. Editar SarisLog			X
HU7. Generar premios especiales			X
HU8. Generar Slider			X

Tabla 11 Funcionalidades por entrega

2.6.2 Plan de iteraciones

Esta es la fase principal en el ciclo de desarrollo de XP. Las funcionalidades son desarrolladas en esta fase, generando al final de cada una un entregable funcional que implementa las historias de usuario asignadas a la iteración. Como las historias de usuario no tienen suficiente detalle como para permitir su análisis y desarrollo, al principio de cada iteración se realizan las tareas necesarias de análisis, recabando

con el cliente todos los datos que sean necesarios. El cliente, por lo tanto, también debe participar activamente durante esta fase del ciclo. Las iteraciones son también utilizadas para medir el progreso del proyecto. Una iteración terminada sin errores es una medida clara de avance.[21]

Iteraciones	Orden de las HU a implementar	Cantidad de tiempo de trabajo
Iteración 1	HU1. Gestionar problema HU3. Gestionar usuario.	3 semanas
Iteración 2	HU2. Gestionar código HU4. Gestionar descargas HU5. Funcionalidades básicas	3 semanas
Iteración 3	HU6. Editar SarisLog HU7. Generar premios especiales HU8. Generar Slider	3 semanas

Tabla 12 Plan de iteraciones

Conclusiones parciales

Luego de realizar una descripción acerca del diseño de la propuesta de solución, se analizaron las diferentes fases que componen la metodología de desarrollo XP y se generaron los artefactos para cada una de ellas, obteniendo de esa forma 8 historias de usuario. Después se analizaron los requerimientos funcionales mediante la lista de reservas, artefacto que genera la metodología, definiendo 29 requerimientos funcionales que recogen la totalidad de la propuesta de solución. Se definen, además, los requisitos no funcionales que plantean las condiciones mínimas de hardware y software para el funcionamiento de las aplicaciones desarrolladas, las cuales serán liberadas al cliente a lo largo de 3 iteraciones de desarrollo definidas en el plan de entregas definido.

Capítulo 3 Implementación y Prueba

En este capítulo se definen elementos estructurales y de diseño del sistema, así como las convenciones tomadas para ello. Se recoge, además, todo lo concerniente a la fase de pruebas del sistema, incluyendo un análisis de los resultados de la misma.

3.1 Tarjetas Clase-Responsabilidades-Colaboradores (CRC)

Las tarjetas CRC (Clase Responsabilidad Colaboración), constituyen uno de los artefactos de la metodología XP que guía el proceso de desarrollo de la solución propuesta. Se dividen en tres secciones que contienen la información del nombre de la clase, sus responsabilidades y sus colaboradores.[23]

Una clase es cualquier persona, evento, concepto, pantalla o reporte. Las responsabilidades de una clase son las funcionalidades que realiza y sus atributos. Los colaboradores son las demás clases con las que interactúa con el fin de cumplir sus responsabilidades.[23]

A continuación se definen las tarjetas CRC de la solución propuesta, agrupadas por aplicación

3.1.1 Aplicación PrintServer

CLASE: ProblemController	
Responsabilidades:	**Colaboraciones:**
- Listar todos los problemas.	ProblemService
- Mostrar un problema.	ProblemService
- Eliminar un problema.	ProblemService
- Editar un problema.	ProblemService ProblemValidator
- Adicionar un problema.	ProblemService ProblemValidator

Tabla 13 Tarjeta CRC # 1

CLASE: DownloadController	
Responsabilidades:	**Colaboraciones:**
- Generar archivos de descarga.	DownloadService
- Obtener archivos de descarga.	ProblemService
- Reparar archivos con estado PENDING.	ProblemService

- Permitir a los usuarios obtener sus descargas.	ProblemService UserService
- Adicionar un problema.	ProblemService ProblemValidator

Tabla 14 Tarjeta CRC # 2

CLASE: UserController	
Responsabilidades:	**Colaboraciones:**
- Listar todos los usuarios.	UserService
- Mostrar un usuario.	UserService
- Eliminar un usuario.	UserService
- Permitir a los usuarios obtener sus descargas.	UserService
- Editar un usuario.	UserValidator UserService
- Adicionar un usuario.	UserValidator UserService
- Expulsar usuario del sistema.	UserService

Tabla 15 Tarjeta CRC # 3

CLASE: CodeController	
Responsabilidades:	**Colaboraciones:**
- Listar todos los códigos.	CodeService
- Mostrar un código.	CodeService
- Eliminar un código.	CodeService
- Editar un código.	CodeService CodeValidator
- Adicionar un código conociendo el problema.	CodeService CodeValidator
- Adicionar un código sin conocer el problema.	CodeService CodeValidator
- Imprimir un código.	DefaultCode DefaultCodePrinter

Tabla 16 Tarjeta CRC # 4

CLASE: CodeService	
Responsabilidades:	**Colaboraciones:**
- Buscar todos los códigos.	CodeDao
- Buscar un código dado su identificador.	CodeDao
- Eliminar un código.	CodeDao
- Actualizar un código.	CodeDao
- Adicionar un código.	CodeDao
- Imprimir un código.	CodeDao

- Buscar todos los códigos asociados a un usuario.	CodeDao

Tabla 17 Tarjeta CRC # 5

CLASE: UserService	
Responsabilidades:	**Colaboraciones:**
- Buscar todos los usuarios.	UserDao
- Buscar un usuario dado su nombre.	UserDao
- Eliminar un usuario.	UserDao
- Actualizar un usuario.	UserDao
- Adicionar un usuario.	UserDao
- Obtener roles de un usuario.	UserDao
- Buscar todos los códigos asociados a un usuario.	UserDao
- Dar o quitar permisos de usuario.	UserDao
- Actualizar si un usuario se encuentra activo o no.	UserDao
- Lanzar evento de re autenticar usuario.	ApplicationEventPublisher

Tabla 18 Tarjeta CRC # 6

CLASE: DownloadService	
Responsabilidades:	**Colaboraciones:**
- Adicionar una descarga.	DownloadDao
- Actualizar todas las descargas que posean estado de PENDING.	DownloadDao
- Buscar todas las descargas.	DownloadDao
- Buscar una descarga dado su nombre	DownloadDao
- Buscar la descarga asociada a un nombre de usuario.	DownloadDao
- Listar los usuarios que no tengan asociada alguna descarga.	DownloadDao
- Actualizar una descarga con el estado de DOWNLOADED.	DownloadDao
- Encontrar todas las descargas que posean un estado en específico.	DownloadDao

Tabla 19 Tarjeta CRC # 7

CLASE: ProblemService	
Responsabilidades:	**Colaboraciones:**
- Buscar todos los problemas.	ProblemDao
- Buscar un problema dado su nombre.	ProblemDao
- Eliminar un problema.	ProblemDao

- Actualizar un problema.	ProblemDao
- Adicionar un problema.	ProblemDao
- Buscar un problema dado su identificador.	ProblemDao
- Obtener los nombres de los problemas	ProblemDao

Tabla 20 Tarjeta CRC # 8

CLASE: DefaultCodePrinter	
Responsabilidades:	**Colaboraciones:**
- Imprimir código.	

Tabla 21 Tarjeta CRC # 9

3.1.2 Aplicación DataTransformer

CLASE: HtmlToTeamResultTransformer	
Responsabilidades:	**Colaboraciones:**
- Devuelve los resultados de todos los equipos de la región del Caribe.	

Tabla 22 Tarjeta CRC # 10

CLASE: ResultsToHtmlTransformer	
Responsabilidades:	**Colaboraciones:**
- Exporta los ganadores de los premios especiales de la región del Caribe a una página web.	

Tabla 23 Tarjeta CRC # 11

CLASE: DefaultAnalizer	
Responsabilidades:	**Colaboraciones:**
- Permite calcular los premios que otorga la sede cubana al finalizar la Final Caribeña del ACM-ICPC.	TeamResult

Tabla 24 Tarjeta CRC # 12

3.2 Arquitectura

Debido a la variedad de aplicaciones realizadas no se definió una arquitectura en común para cada una de ellas sino que se realizaron arquitecturas que se adaptaran a las funcionalidades de cada una de las aplicaciones.

3.2.1 PrintServer

En el caso del PrintServer, la arquitectura del sistema está dada por el uso del framework Spring 3, el cual propone la conocida arquitectura Modelo-Vista-Controlador (MVC) para la creación de aplicaciones web.

El Modelo-Vista-Controlador se centra en separar los distintos componentes de la aplicación en tres capas diferentes: modelo, encargada de las operaciones de acceso a dato; vista, procesa la información obtenida a través del modelo en contenido digerible del usuario y finalmente la capa controlador, encargada de recibir, procesar y responder las diferentes peticiones de los usuarios.

3.2.2 DataTransformer

La siguiente imagen muestra la estructura empleada en la aplicación:

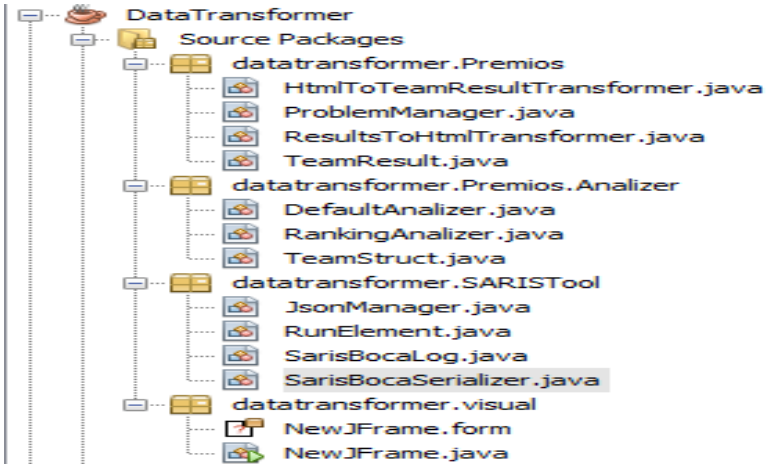


Ilustración 8 Arquitectura DataTransformer.

Dentro del paquete Premios se agrupan la entidad TeamResult que constituye el tipo de dato primario en el que se transforma la información de la tabla de posiciones. Se encuentra, además, la clase encargada de realizar la transformación del contenido de la tabla de posiciones en entidades de tipo TeamResult: HtmlToTeamResultTransformer, así como la encargada de realizar el proceso contrario: ResultsToHtmlTransformer. Forma parte de este paquete, además, la clase encargada de controlar todo

este proceso de transformación: ProblemManager. El paquete Premios contiene, a su vez, otro empaquetado de clases llamado Analizer, el cual posee las entidades especializadas en calcular los premios: DefaultAnalizer y RankingAnalizer, así como una estructura de datos utilizada durante el cálculo de los premios: TeamStruct.

De manera similar el paquete SARISTOOL contiene la clase JsonManager, la cual transforma el contenido del archivo de entrada en elementos de tipo SarisBocaLog, el cual, a su vez, almacena entre otros datos los envíos de cada usuario en un arreglo de tipo RunElement.

El paquete visual agrupa, por su parte, los componentes visuales de la aplicación.

3.2.3 Slider

La arquitectura empleada para esta aplicación es organizada por carpetas de acuerdo a la extensión de los archivos que se utilizaron en la misma, tales como los archivos javascript, css, html, imágenes cada uno de esos tipos contenidos en una carpeta que tiene a todos los de su tipo. Esta se utiliza con el fin de facilitar la búsqueda y acceso de los archivos.

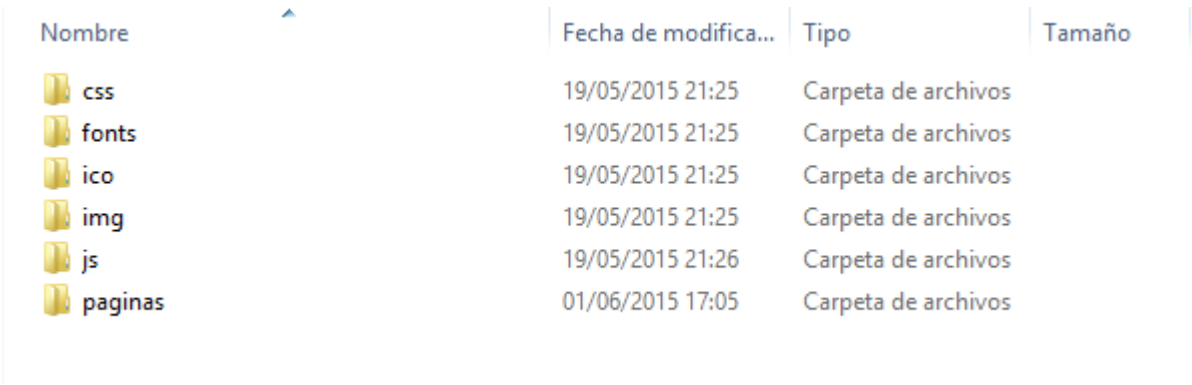


Ilustración 9 Arquitectura Slider.

3.3 Implementación

En esta fase se pasará a implementar las historias de usuarios definidas en el capítulo anterior, para de

esa manera dar cumplimiento los requisitos funcionales del sistema. A lo largo de este proceso se debe cumplir con una serie de estándares y reglas definidas por el equipo de desarrollo, basándose en las especificidades del cliente.

3.3.1 Estándares de codificación

Entre las prácticas a aplicar durante el proceso de desarrollo de software que propone la metodología XP se encuentran la refactorización del código y la propiedad compartida de éste, de forma que todo el personal pueda corregir y extender cualquier parte del producto. Para complementar estas prácticas, la metodología enfatiza en que la comunicación de los programadores es a través del código, por lo cual es indispensable que se sigan ciertos estándares de programación que le provean legibilidad.[24] En la propuesta de solución para declarar el nombre de las variables, métodos y clases se tendrán en cuenta las siguientes convenciones:

- Se utiliza el estilo de escritura CamelCase, en la variante lowerCamelCase para los nombres de variables y métodos.

```
private Validator codeValidator;

@InitBinder("code")
private void initCodeBinder(WebDataBinder binder) {
    binder.setValidator(codeValidator);
}
```

Ilustración 10 Estilo lowerCamelCase.

- Se utiliza el estilo de escritura CamelCase, en la variante HigherCamelCase para los nombres de clases.

```
@Controller
@RequestMapping(value="/code")
public class CodeController extends BaseController{
```

Ilustración 11 Estilo HigherCamelCase.

- No se utilizará una misma línea para definir más de una variable y siempre que sea posible estas se inicializarán en su misma línea de declaración.

3.3.2 Patrones de diseño

Una de las premisas fundamentales de la programación orientada a objetos es la elaboración de un producto que pueda ser lo suficientemente adaptable al cambio como para no crear un efecto dominó dentro de la arquitectura del sistema cada vez que varía un elemento del diseño. “La causa inmediata de la degradación del diseño se entiende bien. Los requisitos van cambiando en formas que el diseño inicial no anticipó. A menudo, estos cambios deben hacerse rápidamente, y pueden ser realizados por ingenieros que no están familiarizados con la filosofía de diseño original. Así, a pesar de que el cambio en el diseño funciona, viola de alguna manera el diseño original... Tenemos que encontrar alguna manera de hacer nuestros diseños resistentes a tales cambios y protegerlos de la degradación."[25]

La solución a este problema se encuentra en los patrones de diseño, los cuales proveen soluciones abordables a todos los problemas se sabe pueden aparecer en el transcurso del diseño. Los patrones de diseño comunican los estilos y soluciones consideradas como "buenas prácticas", que los expertos en el diseño orientado a objetos utilizan para la creación de sistemas.[26]

A continuación se exponen los patrones de diseños utilizados en las aplicaciones, ejemplificando su uso en las aplicaciones desarrolladas.

3.3.2.1 Patrones GRASP

- **Alta cohesión y bajo acoplamiento**: El nivel de cohesión de una clase nos indica la coherencia existente entre la clase y la información que maneja. Una alta cohesión garantiza que cada una de las clases del sistema realiza una labor única con respecto al resto. Por ende, toda información que procese una clase debe estar lo más relacionada posible con la clase que la maneja.[27] Una clase tendrá una cohesión alta si sus métodos están relacionados entre sí, trabajan con tipos similares. Las clases altamente cohesivas permiten disminuir la dependencia existente con respecto a otros componentes del sistema, las hace más desacopladas, permitiendo así disminuir

la propagación del cambio que pudiese generar una modificación en cualquiera de las clases. Spring Framework fomenta un bajo acoplamiento mediante la Inyección de dependencias. La siguiente imagen ejemplifica este proceso en la aplicación, usando para ello la anotación @Autowired.

```
public class DownloadService_Impl implements DownloadService {

    @Autowired
    @Qualifier("downloadDao")
    DownloadDao downloadDao;

    @Override
    @Transactional(propagation = Propagation.REQUIRED, isolation = Isola
    public int registerDownload(List<DownloadEntity> e) {
        downloadDao.cleanPreviousRecords();
        for (Iterator<DownloadEntity> it = e.iterator(); it.hasNext();)
            DownloadEntity downloadEntity = it.next();
            downloadDao.registerDownload(downloadEntity);

        }
        return 1;
    }
```

Ilustración 12 Patrón Bajo acoplamiento

- **Experto en Información**: La responsabilidad de la creación de un objeto o la implementación de un método, debe recaer en la clase que conoce toda la información necesaria para ello, de esta forma se garantiza una alta cohesión y un bajo acoplamiento.[27] En general, el trabajo con el framework Spring está determinado por este patrón.
- **Creador**: El patrón creador nos ayuda a identificar quien debe ser el responsable de la creación o instanciación de nuevos objetos o clases. En la aplicación DataTransformer la clase TeamResult constituye la entidad que permite el trabajo en el módulo SARIS. La creación e instanciación de este tipo de objetos TeamResult es llevada a cabo única y exclusivamente por la clase HtmlToTeamResultTransformer del sistema.
- **Controlador**: Es la C del patrón MVC. Este patrón sugiere que la lógica de negocio debe estar separada de la capa de presentación, siendo el controlador quien recibe los datos del usuario y los

envía a las distintas clases según el método llamado.[27]

3.3.2.2 Patrones GoF

- **Singleton**: Patrón creacional que permite la unicidad de un objeto. Su intención consiste en garantizar la existencia de una única instancia de la entidad u objeto, así como proporcionar un punto de acceso global a este.[28] Spring cuenta con un mecanismo para regular el proceso de creación de objetos u entidades denominado ámbito(scope en inglés).[18] Así, existen diversos tipos de ámbitos, siendo el singleton el utilizado por defecto. Al establecer este ámbito a una entidad, se garantiza una instanciación de la misma según el patrón singleton.
- **Bridge**: Promueve la separación de la interfaz de una abstracción de su implementación. Utilizando el patrón de puente se puede lograr un diseño más eficiente y manejable de una abstracción. Aplicando este patrón, tanto interfaces como las implementaciones de una abstracción se pueden poner en jerarquías de clase separadas.[29] El uso de este patrón se evidencia en el servicio de impresión, ubicado en el paquete printer.PrintingWithBridgePattern.
- **Observer**: Define una dependencia de uno a muchos entre los objetos de manera que cuando alguno de estos cambia de estado, todos sus dependientes son notificados y actualizados automáticamente.[30] Dentro de Spring este patrón se evidencia en el uso de eventos. Dentro de la aplicación se puede observar el uso de este patrón mediante los eventos agrupados en el paquete printer.Security.Events.
- **Decorator**: Permite agregar responsabilidades adicionales a un objeto, de forma dinámica, sin afectar a otros objetos.[30] El uso del patrón decorator viene dado por el empleo del motor de plantillas sitemesh, el cual inserta, dentro de una plantilla base, el contenido de las diferentes páginas del sitio, permitiendo así la reutilización de componentes comunes a todas las vistas.

3.3.3 Tareas de ingeniería

Para realizar la implementación del sistema a partir de las especificaciones recogidas en las historias de usuarios definidas, es necesario descomponer cada HU en tareas de ingeniería según lo definido en el plan de iteraciones, el cual puede estar sujeto a cambios durante este proceso. Estas tareas son del dominio del equipo de desarrollo, por lo que estarán elaboradas en un lenguaje puramente técnico, no

necesariamente entendible por el cliente.

A continuación se expone un listado de las tareas definidas para llevar a cabo la fase de implementación:

Historia de Usuario	Tareas de Ingeniería
HU1_Gestionar problema	Adicionar problema
	Eliminar problema
	Editar problema
	Ver problema
	Listar problemas
	Refactorizar módulo de problemas
HU2_Gestionar código	Crear código a partir de un problema
	Crear código sin conocer el problema
	Ver código
	Editar código
	Eliminar código
	Listar códigos del sistema
	Listar códigos del usuario
	Imprimir código del usuario
	Refactorizar módulo de códigos
HU3_Gestionar usuario	Profundizar en el manejo de usuarios y sesiones del framework Spring Security 3
	Elaborar la seguridad del sitio
	Adicionar usuario
	Eliminar usuario
	Editar usuario
	Ver usuario
	Listar usuarios
	Expulsar usuario
	Refactorizar módulo de usuarios
HU4_Gestionar descargas	Investigar sobre la descarga de archivo en Spring 3

	Generar las soluciones de cada usuario
	Asignar usuarios a cada envío
	Publicar descargas
	Refactorizar módulo de descargas.
HU5_Funcionalidades básicas	Acceder al sistema
	Salir del sistema
	Editar perfil
HU6_Editar SarisLog	Investigar sobre el trabajo con la biblioteca Gson
	Editar SarisLog
	Diseñar y elaborar la interfaz de la aplicación DataTransformer
	Refactorizar módulo SarisLog
HU7_Generar premios especiales	Investigar sobre el trabajo con la biblioteca Jsoup
	Calcular premios
	Refactorizar módulo Premios
HU8_Generar Slider	Investigar acerca del framework AngularJS
	Mostrar las actividades
	Mostrar Counter
	Mostrar la galería de fotos
	Mostrar los países del Caribe que participan en el ACM-ICPC
	Mostrar ubicación de ediciones anteriores del ACM-ICPC

Para mayores detalles de las mismas, el anexo 2 recoge las tareas de ingeniería definidas.

3.3.4 Modelo de datos

La aplicación DataTransformer basa su funcionamiento en el acceso y modificación de un fichero de texto, generando uno nuevo con los cambios realizados sobre el primero. En este proceso, el tiempo de vida de los datos está determinado por el tiempo que permanezca activa la aplicación, por lo que no se requiere de bases de datos.

De forma similar, la aplicación Slider permite mostrar información interactiva sobre el evento, con datos que se encuentran almacenados dentro de los mismos archivos de la aplicación, lo que determina que no

se necesite de un gestor de bases de datos en el sistema.

Para la aplicación PrintServer, al ser necesario la persistencia de objetos se debe definir un modelo físico para la elaboración de la base de datos del sistema. Dicho modelo se describe a continuación.

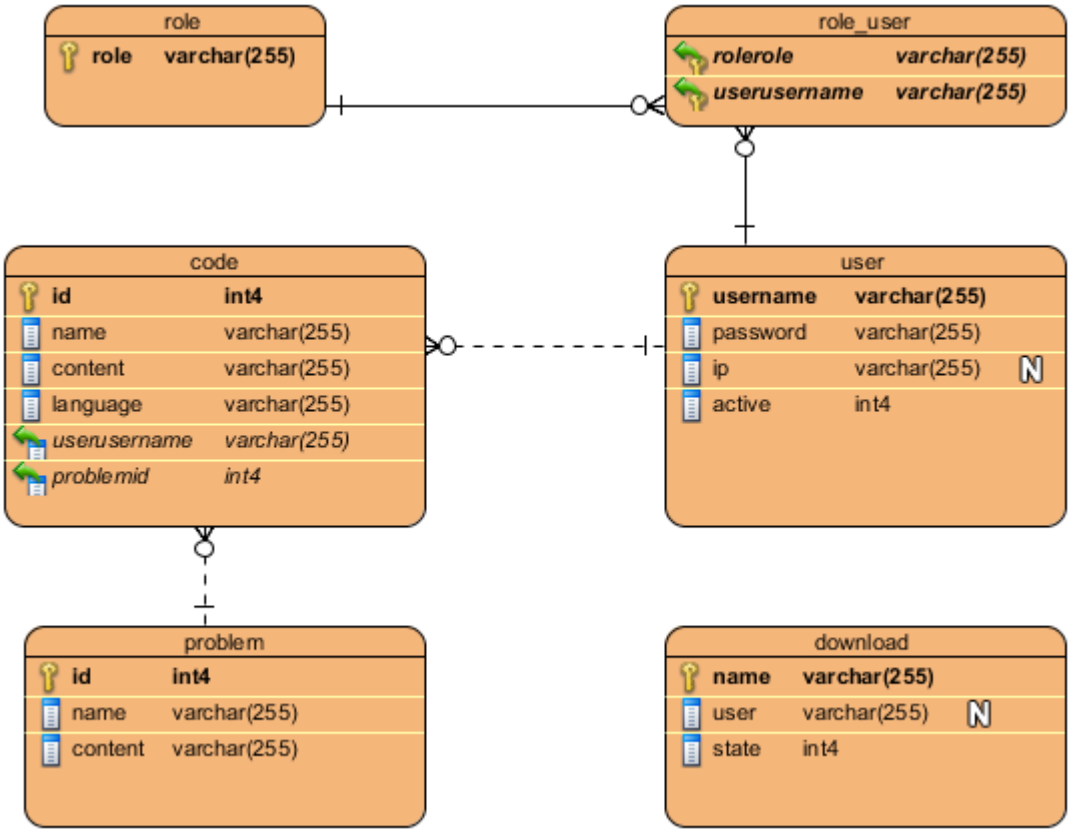


Ilustración 13 Modelo de datos

La responsabilidad de cada una de las tablas es la siguiente:

- **problem:** Contiene los datos de un problema.
- **code:** Almacena lo concerniente a los códigos existentes en el sistema.
- **download:** Guarda una dirección a cada archivo que constituye una descarga, así como una referencia al usuario al que está asociado dicho archivo, así como el estado actual del mismo.

- **user:** Permite mantener un registro de los usuarios registrados en el sistema.
- **role:** Almacena los roles disponibles en la aplicación.
- **role_user:** Permite controlar los roles asociados a cada usuario del sistema.

3.4 Pruebas

Las pruebas de software son fundamentales para verificar la calidad del producto software teniendo como base el control de la satisfacción de los requisitos. Su principal objetivo es detectar errores. La metodología XP establece dos tipos de pruebas: unitarias y de aceptación.

3.4.1 Prueba Unitarias

Las pruebas unitarias son elaboradas por los mismos desarrolladores desde el inicio mismo de la fase de desarrollo, siguiendo el principio de la metodología XP de desarrollo guiado por pruebas, para comprobar el correcto funcionamiento del código, a medida que se van implementando nuevas funcionalidades.

De las aplicaciones elaboradas, el PrintServer y el DataTransformer ameritan este tipo de pruebas, las cuales se realizan, en este caso, a través de la herramienta JUnit, disponible para aplicaciones Java. Sin embargo, la inexperiencia de los desarrolladores en el trabajo con esta aplicación, unido a la falta de tiempo disponible, implicaron no realizar pruebas unitarias automatizadas. Se utilizó, como alternativa, una distribución del trabajo en la cual, mientras un miembro del equipo se encarga de implementar una funcionalidad, el otro integrante se encarga de ir realizando pruebas unitarias de forma manual a la misma, así como a otras desarrolladas anteriormente. Adicionalmente, se realizó un proceso de refactorización al terminar cada iteración.

3.4.2 Pruebas de aceptación

La prueba de aceptación del usuario es la prueba final antes del despliegue del producto obtenido. Su objetivo es verificar que el software esté listo y que puede ser usado por los usuarios finales para ejecutar aquellas funciones y tareas para las cuales fue construido.

En el caso de XP las pruebas de aceptación no están basadas en las historias de usuario, sino en la

funcionalidad que conoce el cliente la cual está asociada a dicha historia de usuario. Las pruebas de aceptación son consideradas como “pruebas de caja negra”. Los clientes son responsables de verificar que sus resultados sean correctos, y en caso de que fallen varias pruebas, deben indicar el orden de prioridad de resolución. Una historia de usuario no se puede considerar terminada hasta tanto pase correctamente todas las pruebas de aceptación.[21]

Los casos de prueba de aceptación definidos por el cliente para las historias de usuario se especifican en el Anexo 3.

A continuación se expone, un caso de prueba por aplicación, así como los resultados que arrojó la misma. Todas las no conformidades detectadas durante el proceso de pruebas de aceptación fueron resueltas, con excepción de CP1_HU4. Habilitar descargas, la cual requería de un script proporcionado al equipo de desarrollo, el cual nunca funcionó.

PrintServer

Caso de prueba de aceptación	
Código: CP1_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se cree un código correctamente.	
Condiciones de ejecución: Se requiere estar autenticado en el sistema. Debe existir al menos un problema en el sistema.	

FLUJO 1: Crear un nuevo código a partir de un problema listado. **Entrada / Pasos de ejecución:** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Validar si tanto el contenido como el nombre del problema no están vacíos y que se haya seleccionado un lenguaje de programación. **FLUJO 2: Crear un nuevo código sin conocer el problema al que está asociado.** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Seleccionar el nombre del problema al que se asociará el nuevo código. 4. Seleccionar un lenguaje de programación. Si todo está correcto se debe proceder a guardar el nuevo código e imprimirlo a continuación.
Resultado esperado: Se crea el nuevo código y se redirecciona a la lista de códigos del usuario, notificando con un mensaje de éxito**.**
Evaluación de la prueba: Satisfactoria.

Tabla 25 Ejemplo 1 Caso de prueba de aceptación.

Durante la ejecución de este caso de prueba se detectaron 1 no conformidad significativa y 2 no conformidades no significativas. En el primer caso se detectó que el sistema no verificaba la existencia del nuevo código en la base de datos, antes de insertarlo. Por su parte, las no conformidades no significativas se relacionaban con la disposición de los datos en la página, así como problemas de validación de la entrada de datos.

DataTransformer

Caso de prueba de aceptación	
Código: CP1_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se cree un código correctamente.	
Condiciones de ejecución: Se requiere estar autenticado en el sistema. Debe existir al menos un problema en el sistema.	

FLUJO 1: Crear un nuevo código a partir de un problema listado. **Entrada / Pasos de ejecución:** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Validar si tanto el contenido como el nombre del problema no están vacíos y que se haya seleccionado un lenguaje de programación. **FLUJO 2: Crear un nuevo código sin conocer el problema al que está asociado.** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Seleccionar el nombre del problema al que se asociará el nuevo código. 4. Seleccionar un lenguaje de programación. Si todo está correcto se debe proceder a guardar el nuevo código e imprimirlo a continuación.
Resultado esperado: Se crea el nuevo código y se redirecciona a la lista de códigos del usuario, notificando con un mensaje de éxito**.**
Evaluación de la prueba: Satisfactoria.

Tabla 26 Ejemplo 2 Caso de prueba de aceptación.

Al ejecutar el caso de prueba se detectó 1 no conformidad significativa. Al ejecutar el paso 4 el usuario se agrega al final de la lista, en lugar de como se describe en el Resultado esperado. Esto implica que al exportar el nuevo archivo modificado que recibirá la herramienta SARIS, el orden de los datos, con respecto al original, no es el mismo.

Slider

Caso de prueba de aceptación	
Código: CP3_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para verificar la galería de fotos.	
Condiciones de ejecución: Se deben haber introducido las fotos de eventos realizados o que ocurren en ese momento.	

Entrada / Pasos de ejecución: 1. El sistema mostrar una imagen inicial de cada evento que se quiera exponer. 2. El usuario debe elegir una de ellas. 3. El sistema muestra una galería de fotos acerca del evento escogido. 4. El usuario cierra la galería elegida y se deben mostrar las imágenes iniciales.
Resultado esperado: Mostrar las fotos elegidas.
Evaluación de la prueba: Satisfactoria

Tabla 27 Ejemplo 3 Caso de prueba de aceptación

Durante la realización de este caso de prueba se detectó 1 no conformidad significativa y 1 no significativa. Para el caso de la significativa en la ejecución del paso 2, el mismo no mostraba la galería que era escogida, y en el caso de la no significativa, no se mostraban algunas fotos dentro de una galería elegida.

3.4.2.1 Resultados de las pruebas

Los resultados obtenidos al realizar las pruebas en cada iteración se muestran en la siguiente tabla:

	Iteración 1	Iteración 2	Iteración 3
CP de Aceptación	8	9	5
No conformidades	7	12	6
Significativas	2	6	4
No significativas	5	6	2
Resueltas	7	11	6
Pendientes	0	1	0

Tabla 28 Resultado de las pruebas

Conclusiones Parciales

Durante el transcurso de este capítulo se realizó uno de los artefactos que genera la metodología escogida para guiar el proceso de implementación, las tarjetas CRC (Clase Responsabilidad

Colaboración), además se definieron las distintas arquitecturas por cada una de las aplicaciones en las que se divide la propuesta de solución. Se llevó a cabo las etapas de implementación de la propuesta solución. En las mismas se evidenciaron uso de los patrones de diseño y de los estándares de codificación resultaron en un código más reutilizable y un mejor diseño funcional del sistema, partiendo siempre de las descripciones recogidas en las historias de usuario y posteriormente transformadas en tareas de ingeniería.

Por otra parte, la estrategia de pruebas, apoyada en los procesos de refactorización de código, permitió,- eventualmente, corregir la casi totalidad de las no conformidades detectadas por el cliente, así como las deficiencias remanentes del proceso de implementación.

Conclusiones Generales y Recomendaciones

La culminación de la presente investigación permitió arribar a las siguientes conclusiones:

- La experiencia que ha tenido el equipo organizador de las finales regionales del ACM-ICPC en el área de América Latina evidenció la necesidad de desarrollar un conjunto de aplicaciones informáticas para la determinación de premios especiales, divulgación de las actividades que involucra el evento y confección de la tabla de posiciones.
- La selección de las tecnologías y herramientas estuvo marcada por los conocimientos que tenían los desarrolladores, el enfoque ágil que se le dio al proceso de desarrollo y a los requerimientos del cliente.
- Para el desarrollo de la aplicación Slider se seleccionaron tecnologías web única y exclusivamente del lado del cliente, lo cual permite que pueda ser embebida dentro de cualquier otro sistema de este tipo, sin importar la compatibilidad entre tecnologías del lado del servidor.
- A pesar de no haber realizado pruebas unitarias automatizadas, la realización de las pruebas de aceptación, así como los procesos de refactorización y pruebas de caja negra, permitieron comprobar el buen funcionamiento de las herramientas, así como el cumplimiento de las especificidades planteadas por el cliente.

Recomendaciones

- En el caso del PrintServer, si bien sqlite3 garantiza la funcionalidad de la aplicación no garantiza la escalabilidad al agregar nuevas funcionalidades al sistema que impliquen incremento de peticiones al gestor de base de datos. En dicho caso, el uso de las variantes no-sql, principalmente Radis, pudiese ser considerado.
- Los premios de la competencia pudiesen tener como entrada, en lugar del ranking público emitido por el BOCA, otro documento en el que se contemplen los milisegundos de cada envío, en aras de garantizar una mayor precisión. De cualquier forma, los algoritmos utilizados para determinar los premios son independientes del tipo de entrada de datos.
- De contar con tiempo suficiente, existen bibliotecas javascript como ScrollMagic que permiten

hacer animaciones muy elaboradas y detalladas. El problema radica en que dichas bibliotecas poseen una documentación muy escasa.

- Google maps es una de las API más competentes para el trabajo con mapas, pero necesita de conexión directa a internet, lo que imposibilita su uso en ambientes sin conectividad. Según el sitio oficial de CNET http://www.cnet.com/news/google-maps-goes-offline-complete-with-turn-by-turn-directions/ para finales de 2015, Google Maps sin conexión estará disponible. De consolidarse esta noticia, pudiese ser posible utilizar esta tecnología en lugar del actual Ammap, para la representación de países en la aplicación Slider.

Bibliografía

1. ***acm***, *ICPC FACT SHEET – 4rd Release **2014**, **Disponible** en http://icpc.baylor.edu/g/Factsheet, **Accedido** el 05/02/2015*
2. ***COJ***, *Reglas oficiales. **2014**, **Disponible** en http://coj.uci.cu/downloads/final-carib/2014/Reglas.pdf, **Accedido** el 5 de febrero de 2015*
3. ***Revilla, M.A., S. Manzoor, and R. Liu***, *Competitive learning in informatics: The UVa online judge experience. Olympiads in Informatics, **2008**. **2**: p. 131-148, **Disponible** en http://www.mii.lt/olympiads_in_informatics/htm/INFOL035.htm, **Accedido** el enero, 2015*
4. ***Juan C. Nogueira, C.J., Luqi***, *Surfing the Edge of Chaos: Applications to Software Engineering. **2000**, **Disponible** en http://dodccrp.org/events/2000_CCRTS/html/pdf_papers/Track_4/075.pdf, **Accedido** el 5 de febrero de 2015*
5. ***Pressman, R.S.***, *Ingeniería de Software: Un enfoque práctico. **2005**, **Disponible** en http://eva.sepyc.gob.mx:8383/greenstone3/sites/localsite/collect/ciencia1/index/assoc/HASH015f/ceb375c1.dir/33040073.pdf, **Accedido** el 5 de febrero de 2015*
6. ***Larman, C.***, *Agile and iterative development: a manager's guide**2004**: Addison-Wesley Professional, **Disponible** en **Accedido** el*
7. ***Krall, C.***, *¿Qué es y para qué sirve UML? Versiones de UML (Lenguaje Unificado de Modelado). Tipos de diagramas UML. **2009**, **Disponible** en http://www.aprenderaprogramar.com/index.php?option=com_content&id=688:ique-es-y-para-que-sirve-uml-versiones-de-uml-lenguaje-unificado-de-modelado-tipos-de-diagramas-uml&Itemid=163, **Accedido** el 6 de febrero de 2015*
8. ***Ramírez, R.Z.***, *Sistemas Gestores de Base de Datos. **2008**, **Disponible** en http://www.csi-csif.es/andalucia/modules/mod_ense/revista/pdf/Numero_14/RAQUEL_ZAMBRANO_2.pdf , **Accedido** el 5 de febrero de 2015*
9. ***González, E.***, *Entrega nº4 del Tutorial básico del programador web: HTML desde cero. **2013**, **Disponible** en http://www.aprenderaprogramar.com/index.php?option=com_content&view=article&id=435:ique-es-y-para-que-sirve-html-el-lenguaje-mas-importante-para-crear-paginas-webs-html-tags-cu00704b&catid=69:tutorial-basico-programador-web-html-desde-cero&Itemid=192, **Accedido** el 5 de febrero de 2015*
10. ***Sierra, M.***, *Qué es y para qué sirve el lenguaje CSS (Cascading Style Sheets - Hojas de Estilo). **2013**, **Disponible** en http://www.aprenderaprogramar.com/index.php?option=com_content&id=546:que-es-y-para-que-sirve-el-lenguaje-css-cascading-style-sheets-hojas-de-estilo&Itemid=163,*

Accedido *el 5 de febrero de 2015*

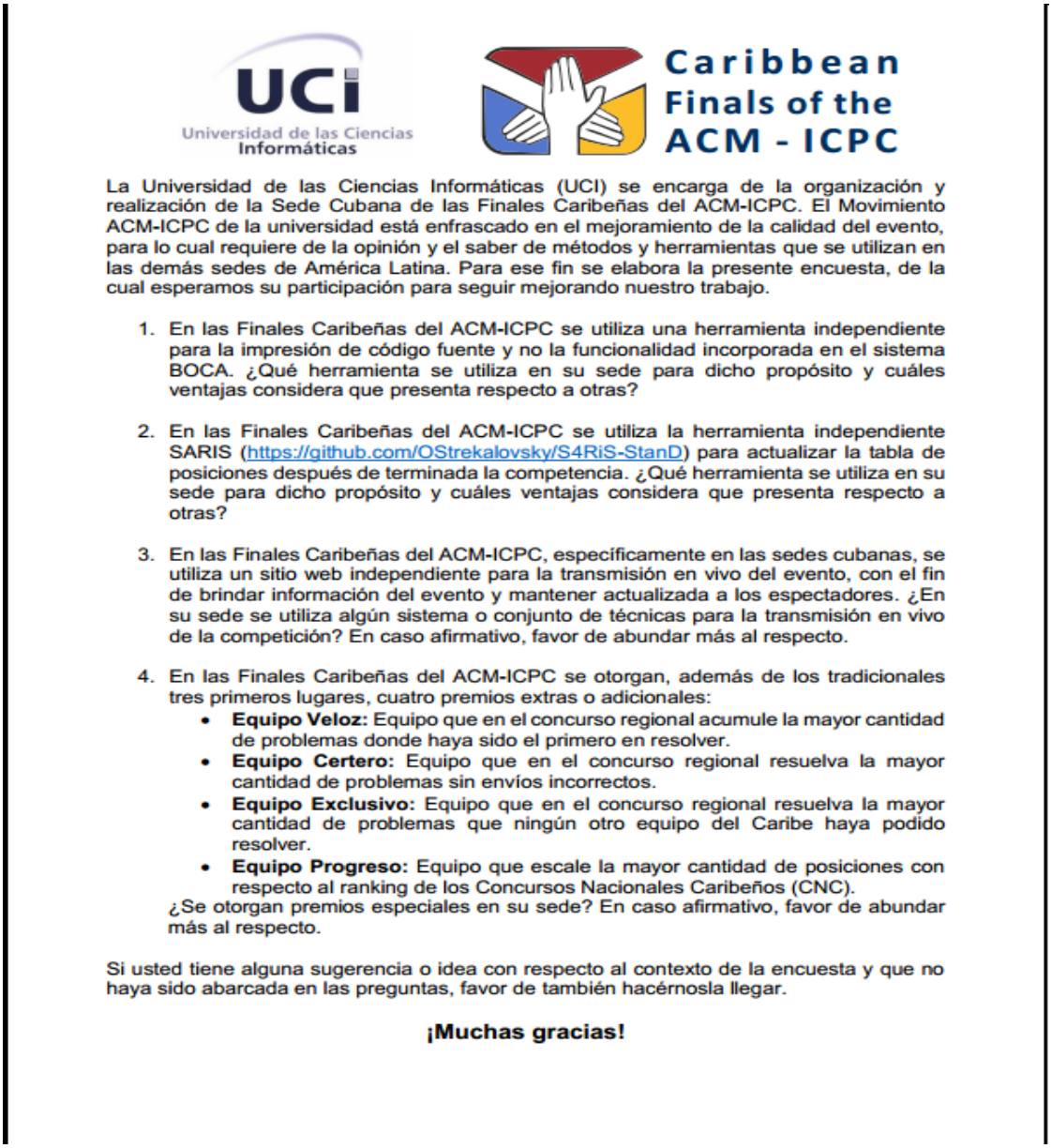
11. ***Pérez, J.E.***, *Introducción a JavaScript* ***2009***, ***Disponible*** *en* http://librosweb.es/javascript/capitulo_1.html, ***Accedido*** *el 5 de febrero de 2015*
12. ***Méndez, J.***, *Lenguajes de programación.* ***2011***, ***Disponible*** *en* http://www.monografias.com/trabajos/lengprog/lengprog.shtml, ***Accedido*** *el 7 de febrero de 2015*
13. ***JSON***, *Introducing JSON.* ***2014***, ***Disponible*** *en* http://www.json.org/, ***Accedido*** *el 5 de febrero de 2015*
14. ***Gutiérrez, J.J.***, *¿Qué es un framework web?* ***2009***, ***Disponible*** *en* http://www.lsi.us.es/~javierj/investigacion_ficheros/Framework.pdf, ***Accedido*** *el 5 de febrero de 2015*
15. ***Álvarez, M.A.***, *Manual de jQuery.* ***2009***, ***Disponible*** *en* http://www.desarrolloweb.com/articulos/introduccion-jquery.html, ***Accedido*** *el 5 de febrero de 2015*
16. ***López, F.J.A.***, *Desarrollo de una aplicación web para compartir medio de transporte con AngularJS.* ***2014***, ***Disponible*** *en* http://repositorio.bib.upct.es:8080/jspui/bitstream/10317/4411/1/tfg396.pdf, ***Accedido*** *el 14 de marzo de 2015*
17. ***Solis, J.***, *¿Qué es Bootstrap y cómo funciona el diseño en la web?* ***2014***, ***Disponible*** *en* http://www.arweb.com/chucherias/editorial/%C2%BFque-es-bootstrap-y-como-funciona-en-el-diseno-web.htm, ***Accedido*** *el 6 de febrero de 2015*
18. ***Johnson, R., et al.***, *The spring framework, reference documentation.". Interface21.(accessed 30.04. 07),* ***2004***, ***Disponible*** *en* ***Accedido*** *el 5 de abril de 2015*
19. ***calendamaia***, *Netbeans.* ***2014***, ***Disponible*** *en* http://www.genbetadev.com/herramientas/netbeans-1, ***Accedido*** *el 5 de febrero de 2015*
20. ***Chaves, M.A.***, *La ingeniería de requerimientos y su importancia en el desarrollo de proyectos de software. InterSedes,* ***2011***. ***6****(10),* ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*
21. ***Joskowicz, J.***, *Reglas y Prácticas en eXtreme Programming.* ***2008***, ***Disponible*** *en* http://iie.fing.edu.uy/~josej/docs/XP%20-%20Jose%20Joskowicz.pdf, ***Accedido*** *el 28 de marzo de 2015*
22. ***Luis Calabria, P.P.***, *Metodología XP.* ***2003***, ***Disponible*** *en* http://fi.ort.edu.uy/innovaportal/file/2021/1/metodologia_xp.pdf, ***Accedido*** *el 20 de marzo de 2015*
23. ***Anaya Villegas, A.***, *A propósito de programación extrema XP (extreme programming),*

2007, *URL: http://www. monografias. com,* ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*

24. ***Letelier, P.a.M.C.P.***, *Métodologías ágiles para el desarrollo de software: eXtreme Programming (XP).* ***2006***, ***Disponible*** *en http://www.cyta.com.ar/ta0502/b_v5n2a1.htm,* ***Accedido*** *el 5 de febrero de 2015*
25. ***Martin, R.C.***, *Design Principles and Design Patterns.* ***2000***, ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*
26. ***Larman, C. and U. y Patrones***, *Introducción al análisis y diseño orientado a objetos. Ed. Prentice Hall, México,* ***1999***, ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*
27. ***Carmona, J.G.***, *Solid y Grasp. Buenas prácticas hacia el éxito en el desarrollo de sorftware.* ***2012***, ***Disponible*** *en https://jbravomontero.files.wordpress.com/2012/12/solid-y-grasp-buenas-practicas-hacia-el-exito-en-el-desarrollo-de-software.pdf,* ***Accedido*** *el 2 de junio de 2015*
28. ***javadepend***, *Spring: The art of using GRASP Patterns.* ***2011***, ***Disponible*** *en https://javadepend.wordpress.com/2011/10/26/spring-the-art-of-using-grasp-patterns/,* ***Accedido*** *el 18 de mayo de 2015*
29. ***Kuchana, P.***, *Software architecture design patterns in Java****2004****: CRC Press,* ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*
30. ***Gamma, E., et al.***, *Design patterns: elements of reusable object-oriented software****1994****: Pearson Education,* ***Disponible*** *en* ***Accedido*** *el 2 de junio de 2015*

Anexos

Anexo 1



UCI
Universidad de las Ciencias Informáticas

Caribbean Finals of the ACM - ICPC

La Universidad de las Ciencias Informáticas (UCI) se encarga de la organización y realización de la Sede Cubana de las Finales Caribeñas del ACM-ICPC. El Movimiento ACM-ICPC de la universidad está enfrascado en el mejoramiento de la calidad del evento, para lo cual requiere de la opinión y el saber de métodos y herramientas que se utilizan en las demás sedes de América Latina. Para ese fin se elabora la presente encuesta, de la cual esperamos su participación para seguir mejorando nuestro trabajo.

1. En las Finales Caribeñas del ACM-ICPC se utiliza una herramienta independiente para la impresión de código fuente y no la funcionalidad incorporada en el sistema BOCA. ¿Qué herramienta se utiliza en su sede para dicho propósito y cuáles ventajas considera que presenta respecto a otras?

2. En las Finales Caribeñas del ACM-ICPC se utiliza la herramienta independiente SARIS (https://github.com/OStrekalovsky/S4RiS-StanD) para actualizar la tabla de posiciones después de terminada la competencia. ¿Qué herramienta se utiliza en su sede para dicho propósito y cuáles ventajas considera que presenta respecto a otras?

3. En las Finales Caribeñas del ACM-ICPC, específicamente en las sedes cubanas, se utiliza un sitio web independiente para la transmisión en vivo del evento, con el fin de brindar información del evento y mantener actualizada a los espectadores. ¿En su sede se utiliza algún sistema o conjunto de técnicas para la transmisión en vivo de la competición? En caso afirmativo, favor de abundar más al respecto.

4. En las Finales Caribeñas del ACM-ICPC se otorgan, además de los tradicionales tres primeros lugares, cuatro premios extras o adicionales:
   - **Equipo Veloz:** Equipo que en el concurso regional acumule la mayor cantidad de problemas donde haya sido el primero en resolver.
   - **Equipo Certero:** Equipo que en el concurso regional resuelva la mayor cantidad de problemas sin envíos incorrectos.
   - **Equipo Exclusivo:** Equipo que en el concurso regional resuelva la mayor cantidad de problemas que ningún otro equipo del Caribe haya podido resolver.
   - **Equipo Progreso:** Equipo que escale la mayor cantidad de posiciones con respecto al ranking de los Concursos Nacionales Caribeños (CNC).

   ¿Se otorgan premios especiales en su sede? En caso afirmativo, favor de abundar más al respecto.

Si usted tiene alguna sugerencia o idea con respecto al contexto de la encuesta y que no haya sido abarcada en las preguntas, favor de también hacérnosla llegar.

¡Muchas gracias!

Ilustración 14 Encuesta

A continuación las respuestas recibidas de la encuesta

Director General del ACM-ICPC en la región de México y Centroamérica. Ver.

Director Asistente del ACM-ICPC en Argentina. Ver.

Director General del ACM-ICPC en Colombia. Ver.

Gerente de Sistemas del ACM-ICPC en la región de Sudamérica/Norte. Ver.

Anexo 2

Tareas de ingeniería por iteración

Iteración 1

Tarea	
Número: 1	Número de HU: 1
Nombre: Adicionar problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 16 de febrero de 2015	**Fecha fin:** 17 de febrero de 2015
Responsable: Carlos	
Descripción: Adicionar un problema al sistema.	

Tarea	
Número: 2	Número de HU: 1
Nombre: Ver problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 16 de febrero de 2015	**Fecha fin:** 17 de febrero de 2015
Responsable: Carlos	
Descripción: Ver datos de un problema.	

Tarea	
Número: 3	**Número de HU**: 1
Nombre: Eliminar problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 17 de febrero de 2015	**Fecha fin:** 18 de febrero de 2015
Responsable: Carlos	
Descripción: Eliminar un problema del sistema.	

Tarea	
Número: 4	**Número de HU**: 1
Nombre: Actualizar problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 17 de febrero de 2015	**Fecha fin:** 18 de febrero de 2015
Responsable: Carlos	
Descripción: Actualizar un problema existente.	

Tarea	
Número: 5	**Número de HU**: 1
Nombre: Listar problemas	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 18 de febrero de 2015	**Fecha fin:** 19 de febrero de 2015
Responsable: Carlos	
Descripción: Listar los problemas registrados en el sistema.	

Tarea	
Número: 6	**Número de HU**: 1
Nombre: Refactorizar módulo de problemas.	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 19 de febrero de 2015	**Fecha fin:** 21 de febrero de 2015
Responsable: Carlos	

Descripción:	
Tarea	
Número: 7	**Número de HU**: 3
Nombre: Profundizar en el manejo de usuarios y sesiones del framework Spring Security 3	
Tipo de tarea: Investigación	**Estimación:** 4 días
Fecha inicio: 21 de febrero de 2015	**Fecha fin:** 25 de febrero de 2015
Responsable: Carlos	
Descripción:	
Tarea	
Número:8	**Número de HU**: 3
Nombre: Elaborar la seguridad del sitio.	
Tipo de tarea: Desarrollo	**Estimación:** 4 días
Fecha inicio: 25 de febrero de 2015	**Fecha fin:** 1 de marzo de 2015
Responsable: Carlos	
Descripción: Plantear las restricciones de seguridad, así como los niveles de acceso de los usuarios.	
Tarea	
Número: 9	**Número de HU**: 3
Nombre: Adicionar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 1 de marzo de 2015	**Fecha fin:** 2 de marzo de 2015
Responsable: Carlos	
Descripción: Adicionar un usuario al sistema.	
Tarea	
Número: 10	**Número de HU**: 3
Nombre: Mostrar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 1 de marzo de 2015	**Fecha fin:** 2 de marzo de 2015

Responsable: Carlos	
Descripción: Mostrar el nombre y clave de acceso de un usuario, así como los códigos por problema que tiene asociados.	
Tarea	
Número: 11	**Número de HU**: 3
Nombre: Editar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 2 de marzo de 2015	**Fecha fin:** 3 de marzo de 2015
Responsable: Carlos	
Descripción: Editar los datos de un usuario. Permite promoverlo a administrador o reiterarlo a un usuario estándar.	
Tarea	
Número: 12	**Número de HU**: 3
Nombre: Eliminar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 2 de marzo de 2015	**Fecha fin:** 3 de marzo de 2015
Responsable: Carlos	
Descripción: Elimina un usuario. Debe eliminarse en cascade buscando llaves foráneas que sean el nombre del usuario.	
Tarea	
Número: 13	**Número de HU**: 3
Nombre: Listar usuarios	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 2 de marzo de 2015	**Fecha fin:** 3 de marzo de 2015
Responsable: Carlos	
Descripción: Listar todos los usuarios del sistema. Por cada uno dar acceso a las funcionalidades básicas del CRUD.	
Tarea	

Número: 14	**Número de HU**: 3
Nombre: Expulsar usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 3 de marzo de 2015	**Fecha fin:** 4 de marzo de 2015
Responsable: Carlos	
Descripción: Expulsar usuarios del sistema.	
Tarea	
Número: 15	**Número de HU**: 3
Nombre: Refactorización	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Fecha inicio: 4 de marzo de 2015	**Fecha fin:** 7 de marzo de 2015
Responsable: Carlos	
Descripción: Refactorizar módulo de usuarios.	

Tabla 29 Tareas de ingeniería- iteración 1

Iteración 2

Tarea	
Número: 16	**Número de HU**: 5
Nombre: Acceder al sistema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 13 de marzo de 2015	**Fecha fin:** 14 de marzo de 2015
Responsable: Carlos	
Descripción: Permite loguear un usuario usando sus credenciales. De no existir dicho usuario se registrará automáticamente.	
Tarea	
Número: 17	**Número de HU**: 5
Nombre: Salir del sistema	

Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 13 de marzo de 2015	**Fecha fin:** 14 de marzo de 2015
Responsable: Carlos	
Descripción: Invalida la sesión actual del usuario.	

Tarea	
Número: 17	**N**úmero de HU: 5
Nombre: Editar perfil	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 14 de marzo de 2015	**Fecha fin:** 15 de marzo de 2015
Responsable: Carlos	
Descripción: Edita la información personal del usuario. No incluye funcionalidades de administración como promover usuario.	

Tarea	
Número: 18	**N**úmero de HU: 2
Nombre: Crear código a partir de un problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 15 de marzo de 2015	**Fecha fin:** 16 de marzo de 2015
Responsable: Carlos	
Descripción: Permite crear un nuevo código a partir de la opción Print que brinda cada problema. El nuevo código ya estará asociado a dicho problema y luego de ser creado será impreso de forma automática.	

Tarea	
Número: 19	**N**úmero de HU: 2
Nombre: Crear código sin conocer el problema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 15 de marzo de 2015	**Fecha fin:** 16 de marzo de 2015
Responsable: Carlos	

Descripción: Permite crear un nuevo código especificando a que problema se encontrará asociado. El nuevo código luego de ser creado será impreso de forma automática.	
Tarea	
Número: 20	**N**úmero de HU: 2
Nombre: Ver código	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 15 de marzo de 2015	**Fecha fin:** 16 de marzo de 2015
Responsable: Carlos	
Descripción: Permite visualizar la información de un código.	
Tarea	
Número: 21	**N**úmero de HU: 2
Nombre: Editar código	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 16 de marzo de 2015	**Fecha fin:** 17 de marzo de 2015
Responsable: Carlos	
Descripción: Editar la información de un código.	
Tarea	
Número: 22	**N**úmero de HU: 2
Nombre: Eliminar código	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 16 de marzo de 2015	**Fecha fin:** 17 de marzo de 2015
Responsable: Carlos	
Descripción: Elimina un código.	
Tarea	
Número: 23	**N**úmero de HU: 2
Nombre: Listar códigos del sistema	
Tipo de tarea: Desarrollo	**Estimación:** 1 día

Fecha inicio: 18 de marzo de 2015	**Fecha fin:** 19 de marzo de 2015
Responsable: Carlos	
Descripción: Permite listar todos los códigos existentes en el sistema, ordenados por usuario. Disponible solo para el rol administrador. Por cada código se deben mostrar las opciones de mostrar y eliminar código.	
Tarea	
Número: 24	**N**úmero de HU: 2
Nombre: Listar códigos del usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 19 de marzo de 2015	**Fecha fin:** 20 de marzo de 2015
Responsable: Carlos	
Descripción: Permite listar todos los códigos pertenecientes al usuario. Por cada código se deben mostrar las opciones de mostrar, editar, imprimir y eliminar código.	
Tarea	
Número: 25	**N**úmero de HU: 2
Nombre: Imprimir código del usuario	
Tipo de tarea: Desarrollo	**Estimación:** 1 día
Fecha inicio: 23 de marzo de 2015	**Fecha fin:** 24 de marzo de 2015
Responsable: Carlos	
Descripción: Imprime un código seleccionado, previamente creado.	
Tarea	
Número: 26	**N**úmero de HU: 4
Nombre: Investigar sobre la descarga de archivo en Spring 3	
Tipo de tarea: Investigación	**Estimación:** 1 día
Fecha inicio: 24 de marzo de 2015	**Fecha fin:** 25 de marzo de 2015
Responsable: Carlos	
Descripción: Indagar como el framework maneja el trabajo de descarga de archivos.	
Tarea	

Número: 27	**N**úmero de HU: 4
Nombre: Generar las soluciones de cada usuario.	
Tipo de tarea: Desarrollo	**Estimación:** 2 día
Fecha inicio: 24 de marzo de 2015	**Fecha fin:** 26 de marzo de 2015
Responsable: Carlos	
Descripción: A partir de un backup del BOCA, generar los envíos de cada usuario durante la competencia en un compactado, cuyo nombre coincidirá con el nombre del usuario, de acuerdo a la base de datos. Para esto se cuenta con un script bin/Shell. Después de generar dichos archivos el sistema debe ser capaz de registrar como "PENDING" y notificar los compactados cuyo nombre coincida con el de algún usuario en el sistema, así como registrar como "MISSING" y notificar los compactados para los que no hubo coincidencia.	
Tarea	
Número: 28	**N**úmero de HU: 4
Nombre: Asignar usuarios a cada envío	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 26 de marzo de 2015	**Fecha fin:** 28 de marzo de 2015
Responsable: Carlos	
Descripción: Después de finalizada la tarea 27, permitir asociar de forma manual, mediante privilegios de administrador, un usuario del sistema con un compactado que no estuvo asociado a ningún usuario. Automáticamente después de realizar esta acción el archivo quedará registrado como "PENDING".	
Tarea	
Número: 29	**N**úmero de HU: 4
Nombre: Publicar descargas	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 28 de marzo de 2015	**Fecha fin:** 30 de marzo de 2015
Responsable: Carlos	
Descripción: Permite a los usuarios que tengan asociado algún archivo de soluciones del certamen descargarlo. Después de realizar esta acción el archivo quedara registrado como "DOWNLOADED". Esta opción no estará disponible para los administradores.	

Tarea	
Número: 30	**N**úmero de HU: 4
Nombre: Refactorización del módulo descargas.	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Fecha inicio: 1 de marzo de 2015	**Fecha fin:** 3 de marzo de 2015
Responsable: Carlos	
Descripción:	
Tarea	
Número: 31	**N**úmero de HU: 4
Nombre: Refactorización del módulo código.	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Fecha inicio: 4 de marzo de 2015	**Fecha fin:** 6 de marzo de 2015
Responsable: Carlos	
Descripción:	

Tabla 30 Tareas de ingeniería- iteración 2

Iteración 3

Tarea	
Número: 32	**N**úmero de HU: 6
Nombre: Investigar sobre el trabajo con la biblioteca Gson.	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 13 de marzo de 2015	**Fecha fin:** 15 de marzo de 2015
Responsable: Carlos	
Descripción: Gson es una biblioteca que permite manipular archivos JSON en Java. Aprender a trabajar y si es viable.	
Tarea	
Número: 33	**N**úmero de HU: 7

Nombre: Jsoup	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 13 de marzo de 2015	**Fecha fin:** 15 de marzo de 2015
Responsable: Carlos	
Descripción: Jsoup es una biblioteca de Java que permite manipular archivos HTML.	
Tarea	
Número: 34	**N**úmero de HU:6, 7
Nombre: Diseñar y elaborar la interfaz de la aplicación DataTransformer	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 16 de marzo de 2015	**Fecha fin:** 18 de marzo de 2015
Responsable: Carlos	
Descripción: Jsoup es una biblioteca de Java que permite manipular archivos HTML.	
Tarea	
Número: 35	**N**úmero de HU:6
Nombre: Editar Sarislog	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 18 de marzo de 2015	**Fecha fin:** 20 de marzo de 2015
Responsable: Carlos	
Descripción: Brindar la opción de editar el archivo SarisLog para eliminar usuarios no deseados. Permitir guardar los cambios en un nuevo archivo. Notificar al usuario de las acciones.	
Tarea	
Número: 36	**N**úmero de HU:6
Nombre: Elaborar premios del certamen	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Fecha inicio: 21 de marzo de 2015	**Fecha fin:** 24 de marzo de 2015
Responsable: Carlos	

Descripción: Determinar los premios especiales de la sede cubana a partir de la tabla de posiciones general generada por el BOCA y siguiendo las pautas para cada premio. Verificar los milisegundos para comparar mejores soluciones. Exportar los resultados a una página web, notificando al usuario en todo momento.	
Tarea	
Número: 37	**N**úmero de HU:6,7
Nombre: Refactorizar	
Tipo de tarea: Desarrollo	**Estimación:** 1 días
Fecha inicio: 24 de marzo de 2015	**Fecha fin:** 25 de marzo de 2015
Responsable: Carlos	
Descripción: Refactorizar las tareas 36 y 37.	
Tarea	
Número: 38	**N**úmero de HU:8
Nombre: Investigar acerca del framework AngularJS.	
Tipo de tarea: Investigación	**Estimación:** 2 días
Fecha inicio: 13 de marzo de 2015	**Fecha fin:** 15 de marzo de 2015
Responsable: Adrian Enrique	
Descripción: Lectura de documentos y búsqueda de tutoriales acerca del uso y aprendizaje del framework AngularJS.	
Tarea	
Número: 39	**N**úmero de HU:8
Nombre: Mostrar las actividades	
Tipo de tarea: Desarrollo	**Estimación:** 2 días
Fecha inicio: 15 de marzo de 2015	**Fecha fin:** 17 de marzo de 2015
Responsable: Adrian Enrique	
D0escripción: Mostrar las actividades definidas para la realización de un evento anunciado en una página web.	
Tarea	
Número: 40	**N**úmero de HU:8
Nombre: Mostrar Counter.	

Tipo de tarea: Desarrollo	**Estimación:** 4 días
Fecha inicio: 18 de marzo de 2015	**Fecha fin:** 22 de marzo de 2015
Responsable: Adrian Enrique	
Descripción: Mostrar tiempo restante para un evento previamente anunciado. El counter que se muestra puede ser utilizado como cronómetro, reloj, countdown debido a los métodos que trae incorporado el mismo.	
Tarea	
Número: 41	**N**úmero de HU:8
Nombre: Mostrar la galería de fotos.	
Tipo de tarea: Desarrollo	**Estimación:** 5 días
Fecha inicio: 23 de marzo de 2015	**Fecha fin:** 28 de marzo de 2015
Responsable: Adrian Enrique	
Descripción: Mostrar la galería de fotos sobre varios eventos ya realizados y luego escoger uno y exponer todas las fotos acerca del mismo.	
Tarea	
Número: 42	**N**úmero de HU:8
Nombre: Mostrar los países del Caribe que participan en el ACM-ICPC.	
Tipo de tarea: Desarrollo	**Estimación:** 6 días
Fecha inicio: 29 de marzo de 2015	**Fecha fin:** 4 de abril de 2015
Responsable: Adrian Enrique	
Descripción: Mostrar geográficamente los países caribeños que participan en el ACM-ICPC además de información acerca del mismo y las universidades de cada uno de ellos que participan en el evento.	
Tarea	
Número: 43	**N**úmero de HU:8
Nombre: Mostrar ubicación de ediciones anteriores del ACM-ICPC.	
Tipo de tarea: Desarrollo	**Estimación:** 3 días
Fecha inicio: 5 de abril de 2015	**Fecha fin:** 8 de abril de 2015
Responsable: Adrian Enrique	

Descripción: Mostrar geográficamente de las finales mundiales del evento ACM-ICPC hasta la fecha, además de realizar una breve reseña acerca del surgimiento del mismo, así como mostrar las universidades ganadoras de cada una de las finales ocurridas.

Tabla 31 Tareas de ingeniería- iteración 3

Anexo 3

Casos de prueba de aceptación.

Aplicación PrintServer

Caso de prueba de aceptación	
Código: CP1_HU1	**HU_1:** Gestionar problema
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que un problema se cree correctamente.	
Condiciones de ejecución: Se debe estar autenticado con privilegios de administrador.	
Entrada / Pasos de ejecución: 1. Entrada del nombre y contenido del problema. 2. Validar si ya existe un problema con ese nombre. 3. Validar si tanto el contenido como el nombre del problema no están vacíos.	
Resultado esperado: Problema creado satisfactoriamente. Redireccionamiento a la lista de problemas y notificación de éxito.	
Evaluación de la prueba: Satisfactoria	

Tabla 32 CP1_HU1. Insertar problema

Caso de prueba de aceptación	
Código: CP2_HU1	**HU_1:** Gestionar problema
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que un problema se muestre correctamente.	

Condiciones de ejecución: No se requiere estar autenticado
Entrada / Pasos de ejecución: 1. Seleccionar la opción mostrar de la lista de problemas. 2. Se debe mostrar, además del contenido del problema, su nombre independientemente de su longitud. El formato en el que se presenta el mismo debe ser uniforme.
Resultado esperado:
Evaluación de la prueba: Satisfactoria

Tabla 33 CP2_HU1. Mostrar problema

Caso de prueba de aceptación	
Código: CP3_HU1	**HU_1:** Gestionar problema
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que un problema se edite correctamente.	
Condiciones de ejecución: Se requiere estar autenticado como administrador.	
Entrada / Pasos de ejecución: 1. Escribir un nuevo nombre, contenido o ambos. 2. Validar que los datos introducidos sean correctos de la misma forma que al insertar un problema.	
Resultado esperado: Los cambios se registran y se redirecciona a la lista de problemas, mostrando un mensaje de éxito**.**	
Evaluación de la prueba: Satisfactoria	

Tabla 34 CP3_HU1. Editar problema

Caso de prueba de aceptación	
Código: CP4_HU1	**HU_1:** Gestionar problema
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se listen los problemas correctamente.	
Condiciones de ejecución: Se requiere estar autenticado como administrador para tener acceso a todas las funcionalidades, como usuario del sistema para las opciones de mostrar e imprimir problema y no se requiere estar autenticado para tener acceso a la opción de mostrar problema.	

Entrada / Pasos de ejecución: 1. De acuerdo al rol, se debe listar por cada problema las siguientes opciones. -Administrador: Opciones de mostrar, editar, eliminar e imprimir. -Usuario del sistema: Opciones de mostrar e imprimir. -Otro caso: Opción de mostrar.
Resultado esperado:
Evaluación de la prueba: Satisfactoria

Tabla 35 CP4_HU1. Listar problemas

Caso de prueba de aceptación	
Código: CP1_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se cree un código correctamente.	
Condiciones de ejecución: Se requiere estar autenticado en el sistema. Debe existir al menos un problema en el sistema.	
FLUJO 1: Crear un nuevo código a partir de un problema listado. **Entrada / Pasos de ejecución:** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Validar si tanto el contenido como el nombre del problema no están vacíos y que se haya seleccionado un lenguaje de programación. **FLUJO 2: Crear un nuevo código sin conocer el problema al que está asociado.** 1. Entrada del nombre del nuevo código. 2. Validar si ya existe un código con ese nombre. 3. Seleccionar el nombre del problema al que se asociará el nuevo código. 4. Seleccionar un lenguaje de programación. 5. Si todo está correcto se debe proceder a guardar el nuevo código e imprimirlo a continuación.	
Resultado esperado: Se crea el nuevo código y se redirecciona a la lista de códigos del usuario, notificando con un mensaje de éxito**.**	
Evaluación de la prueba: Satisfactoria	

Tabla 36 CP1_HU2. Crear código

Caso de prueba de aceptación	
Código: CP2_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se edite correctamente un código perteneciente al usuario.	
Condiciones de ejecución: Se requiere estar autenticado como usuario.	
Entrada / Pasos de ejecución: 1. Escribir un nuevo nombre para el código, contenido, lenguaje, o todos los anteriores. 2. Validar que los datos introducidos sean correctos de la misma forma que al insertar un nuevo código conociendo el problema. 3. El nombre del problema al que se encuentra asociado el código debe mostrarse.	
Resultado esperado: Se guardan los cambios hechos al código y se redirecciona a la lista de códigos del usuario, notificando con un mensaje de éxito.	
Evaluación de la prueba: Satisfactoria	

Tabla 37 CP2_HU2. Editar código perteneciente a un usuario

Caso de prueba de aceptación	
Código: CP3_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se muestren correctamente la lista de código correspondiente.	
Condiciones de ejecución: Se requiere estar autenticado como usuario o como administrador.	
Entrada / Pasos de ejecución: 1. Seleccionar el tipo de listado, ya sea listar los códigos pertenecientes al usuario o listar todos los códigos del sistema. 2. Independientemente del tipo del listado, debe aparecer el nombre del código y el nombre del problema al que está asociado. En caso de que cualesquiera de estos fuese muy grande el(los) nombre(s) deben ajustarse al tamaño de la pantalla. **FLUJO 1: Listar códigos pertenecientes al usuario.** 1. Por cada código se deben mostrar las opciones de editar, mostrar, eliminar e imprimir código. **FLUJO 2: Listar códigos del sistema.** 1. Deben mostrarse todos los códigos existentes en el sistema, ordenados por usuario. Por cada código deben mostrarse las opciones de mostrar y eliminar.	

Resultado esperado:
Evaluación de la prueba: Satisfactoria

Tabla 38 CP3_HU2. Listar códigos

Caso de prueba de aceptación	
Código: CP4_HU2	**HU_2:** Gestionar código
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se reimprima correctamente un código previamente creado.	
Condiciones de ejecución: Se requiere estar autenticado. El código a imprimir debe existir.	
Entrada / Pasos de ejecución: 1. Presionar la opción imprimir para el código deseado.	
Resultado esperado: Se guardan los cambios hechos al código y se redirecciona a la lista de códigos del usuario, notificando con un mensaje de éxito.	
Evaluación de la prueba: Satisfactoria	

Tabla 39 CP4_HU2. Imprimir código

Caso de prueba de aceptación	
Código: CP1_HU3	**HU_3:** Gestionar usuario
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se crea correctamente un usuario.	
Condiciones de ejecución: Se requiere estar autenticado como administrador.	
Entrada / Pasos de ejecución: 1. Insertar el nombre y contraseña del nuevo usuario. 2. Validar que no exista el nombre de usuario y que el campo contraseña no se encuentre vacío.	
Resultado esperado: Se crea un nuevo usuario en el sistema con el rol de usuario estándar.	
Evaluación de la prueba: Satisfactoria	

Tabla 40 CP1_HU3. Crear usuario

Caso de prueba de aceptación	
Código: CP2_HU3	**HU_3:** Gestionar usuario
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se edite correctamente un usuario.	
Condiciones de ejecución: Se requiere estar autenticado como administrador.	

Entrada / Pasos de ejecución: 1. Insertar la contraseña del usuario. Por defecto mostrar la actual. 2. Validar que el campo contraseña no se encuentre vacío. 3. Mostrar el rol del usuario junto con la opción de promoverlo si es usuario estándar o degradarlo si es administrador.
Resultado esperado: Se guardan los cambios del usuario.
Evaluación de la prueba: Satisfactoria

Tabla 41 CP2_HU3. Editar usuario

Caso de prueba de aceptación	
Código: CP3_HU3	**HU_3:** Gestionar usuario
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se muestren correctamente los datos de un usuario.	
Condiciones de ejecución: Se requiere estar autenticado como administrador.	
Entrada / Pasos de ejecución: 1. Se deben mostrar, además del nombre y contraseña del usuario, las estadísticas del mismo en cuanto a códigos generados, reflejadas en una gráfica de cantidad de código contra problema.	
Resultado esperado:	
Evaluación de la prueba: Satisfactoria	

Tabla 42 CP3_HU3. Mostrar usuario

Caso de prueba de aceptación	
Código: CP1_HU4	**HU_4:** Gestionar descargas
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que se generen correctamente las descargas asociadas a cada usuario.	
Condiciones de ejecución: Se requiere estar autenticado como administrador. Deben existir archivos de descarga.	

Entrada / Pasos de ejecución: 1. Se deben mostrar los pasos necesarios para generar las descargas, según el orden de ejecución. 2. Al pulsar sobre el comando generar (Paso 1) se deben extraer los envíos de cada usuario del backup del BOCA y mostrarlos en una tabla. 3. Todas las descargas generadas en el paso 1 deben asociarse con un usuario del sistema, tal que el nombre del usuario del sistema coincida con el nombre del archivo de descarga. En caso de no coincidir, se debe dar al administrador la opción de asociarle un usuario al archivo de descarga. 4. Los archivos de descargas deben transitar por los siguientes estados: *PENDING: si hubo coincidencia entre el nombre del archivo y el nombre de algún usuario del sistema, pero este último no ha descargado el archivo aún. *DOWNLOADED: Ídem al anterior, solo que el usuario ya descargó el recurso. *MISSING: Si no existe coincidencia entre el nombre de un usuario con respecto al nombre del archivo, en cuyo caso se debe permitir al administrador asignarle un usuario asociado al recurso, e inmediatamente hacerlo transitar de este estado al de PENDING. 5. Al publicar las descargas se le dará acceso a los archivos de código a todos los usuarios que tengan alguno de estos archivos asociado. En este sentido, debe ser posible volver a publicar los archivos si hubo alguna transición de MISSING a PENDING y los usuarios afectados deben tener disponible la descarga. 6. Al hacer pública la opción de descarga solo los usuarios estándar tendrán derecho a la misma, los usuarios con rol administrador no podrán efectuar la operación.
Resultado esperado: El usuario puede descargar sus envíos de la competencia.
Evaluación de la prueba: Satisfactoria

Tabla 43 CP1_HU4. Habilitar descargas

Caso de prueba de aceptación	
Código: CP1_HU5	**HU_5:** Funcionalidades básicas
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que los usuarios se autentiquen en el sistema correctamente.	
Condiciones de ejecución: No se debe estar autenticado en el sistema.	

Entrada / Pasos de ejecución: 1. El usuario escribe sus credenciales y pulsa en el botón de login. 2. El sistema valida las credenciales, pudiendo devolver una de las siguientes respuestas: **Excepción**: Si coincide el nombre de usuario, pero la clave de acceso no. Si existe una sesión activa del usuario en cuestión. * En cualquier caso se le debe notificar al usuario sobre la excepción. **Éxito**: Si el nombre de usuario y la clave de acceso coinciden con las credenciales registradas en el sistema. Si no existe el nombre de usuario en el sistema, en cuyo caso se procederá a registrar un nuevo usuario con las credenciales brindadas y posteriormente se le dará acceso al mismo al sistema.
Resultado esperado: El usuario se autentica en el sistema.
Evaluación de la prueba: Satisfactoria

Tabla 44 CP1_HU5. Autenticación de usuario

Aplicación DataTransformer

Caso de prueba de aceptación	
Código: CP1_HU6	**HU_6:** Editar Sarislog
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que el archivo de entrada sea el correcto.	
Condiciones de ejecución: Haber seleccionado un archivo correctamente.	
Entrada / Pasos de ejecución: 1. El usuario selecciona el archivo a tratar. 2. El sistema debe validar que el archivo sea correcto, verificando el nombre y el formato del mismo. En caso de ser incorrecto el archivo se debe notificar al usuario y detener la ejecución del sistema, caso contrario proceder con el flujo normal.	
Resultado esperado: El usuario puede editar el archivo SarisLog	
Evaluación de la prueba: Satisfactoria	

Tabla 45 CP1_HU6. Comprobar archivo

Caso de prueba de aceptación	
Código: CP1_HU7	**HU_7:** Calcular premios

Responsable de la prueba: Adrian Enrique Ciria Jacas
Descripción: Prueba para verificar que el archivo de entrada sea el correcto.
Condiciones de ejecución:
Entrada / Pasos de ejecución: 1. El usuario importa el ranking oficial del BOCA. 2. El sistema debe validar que el archivo sea correcto, verificando el nombre y el formato del mismo. 3. En caso de ser incorrecto el archivo se debe notificar al usuario y detener la ejecución del sistema, caso contrario proceder con el flujo normal. 4. De la misma forma, se procede para el ranking de la edición anterior.
Resultado esperado: El usuario puede proceder a calcular los premios del certamen.
Evaluación de la prueba: Satisfactoria

Tabla 46 CP1_HU7. Comprobar entrada de datos

Caso de prueba de aceptación	
Código: CP2_HU7	**HU_7:** Calcular premios
Responsable de la prueba: Adrian Enrique Ciria Jacas	
Descripción: Prueba para verificar que los premios se hayan generado correctamente.	
Condiciones de ejecución: Se deben haber importado el ranking oficial del BOCA y el ranking oficial del BOCA de la edición anterior, de forma correcta.	
Entrada / Pasos de ejecución: 1. El sistema debe determinar los premios del certamen y mostrarlos en una lista. 2. El usuario selecciona la opción exportar para generar los premios en formato HTML. 3. El sistema debe guardar los premios en la página results, en el directorio page de la aplicación.	
Resultado esperado: El usuario posee los premios del evento.	
Evaluación de la prueba: Satisfactoria	

Tabla 47 CP2_HU7. Generar premios

Aplicación Slider

Caso de prueba de aceptación	
Código: CP1_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para verificar las actividades.	

Condiciones de ejecución: Se deben haber introducido las actividades para un evento a realizar.
Entrada / Pasos de ejecución: 1. El sistema mostrar las actividades en una lista. 2. El usuario escribe la información que busca en el campo Buscar.
Resultado esperado: Mostrar las actividades en una lista.
Evaluación de la prueba: Satisfactoria

Tabla 48 CP1_HU8. Mostrar Actividades

Caso de prueba de aceptación	
Código: CP2_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para verificar el counter.	
Condiciones de ejecución: Se deben haber introducido la fecha para el evento a realizar.	
Entrada / Pasos de ejecución: 1. Se introduce la fecha del evento a realizar. 2. El sistema muestra un reloj bastante visible con el tiempo que falta para llevar a cabo un evento determinado.	
Resultado esperado: El usuario sabe el tiempo que falta para un evento específico.	
Evaluación de la prueba: Satisfactoria	

Tabla 49 CP2_HU8. Mostrar Counter

Caso de prueba de aceptación	
Código: CP3_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para verificar la galería de fotos.	
Condiciones de ejecución: Se deben haber introducido las fotos de eventos realizados o que ocurren en ese momento.	
Entrada / Pasos de ejecución: 1. El sistema mostrar una imagen inicial de cada evento que se quiera exponer. 2. El usuario debe elegir una de ellas. 3. El sistema muestra una galería de fotos acerca del evento escogido. 4. El usuario cierra la galería elegida y se deben mostrar las imágenes iniciales.	

Resultado esperado: Mostrar las fotos elegidas.
Evaluación de la prueba: Satisfactoria

Tabla 50 CP3_HU8. Mostrar Galería

Caso de prueba de aceptación	
Código: CP4_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para los países del Caribe en el ACM-ICPC.	
Condiciones de ejecución: Se deben haber introducido los datos de los países del Caribe que participan en el ACM-ICPC.	
Entrada / Pasos de ejecución: 1. El sistema muestra en un mapa el área del Caribe con los países que participan en el ACM-ICPC, con la cantidad de instituciones involucradas en el evento, además de las banderas de cada uno de ellos en una pequeña galería. 2. El usuario escoge una de las banderas y se muestran en otra galería las instituciones del país escogido, ocultando el mapa que se muestra al inicio, al dejar el espacio concerniente a la galería de instituciones se muestra el mapa una vez más. 3. El usuario escoge uno de los países y el sistema debe realizar una ampliación del mismo.	
Resultado esperado: Mostrar los países que participan en el evento y las instituciones de cada uno de ellos.	
Evaluación de la prueba: Satisfactoria	

Tabla 51 CP4_HU8. Mostrar países

Caso de prueba de aceptación	
Código: CP5_HU8	**HU_8:** Generar Slider
Responsable de la prueba: Carlos González Galera	
Descripción: Prueba para verificar las ediciones.	
Condiciones de ejecución: Se deben haber introducido los datos de las ediciones finales del ACM-ICPC.	

Entrada / Pasos de ejecución: 1. El sistema muestra en un mapa las ediciones finales del ACM-ICPC. 2. El usuario escoge la opción History, el sistema oculta el mapa y muestra una breve reseña histórica acerca del ACM-ICPC, al dejarla se muestra el mapa inicial. 3. El usuario escoge la opción About ICPC, el sistema oculta el mapa y muestra una breve información acerca del ICPC. 4. El usuario escoge cualquiera de las opciones restantes y el sistema oculta el mapa y muestra una galería con los ganadores durante el período de tiempo escogido, al dejarla se muestra el mapa inicial.
Resultado esperado: Mostrar información requerida por el usuario acerca del ACM-ICPC.
Evaluación de la prueba: Satisfactoria

Tabla 52 CP5_HU8. Mostrar ediciones