Facultad 4

Análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe.

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

Autor

Anabel Reyes Rosa

Tutores

Ing. Jorge Amado Soria Ramírez

Ing. Enrique José Alguna Castillo

Ing. Nersa Doraines Acosta Labrada

La Habana, 19 de junio de 2012

“Año 54 de la Revolución”

Declaración de autoría

Declaración de autoría

Declaro que soy la única autora del trabajo titulado: "**Análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe**" y autorizo a la Universidad de las Ciencias Informáticas a hacer uso del mismo en su beneficio.

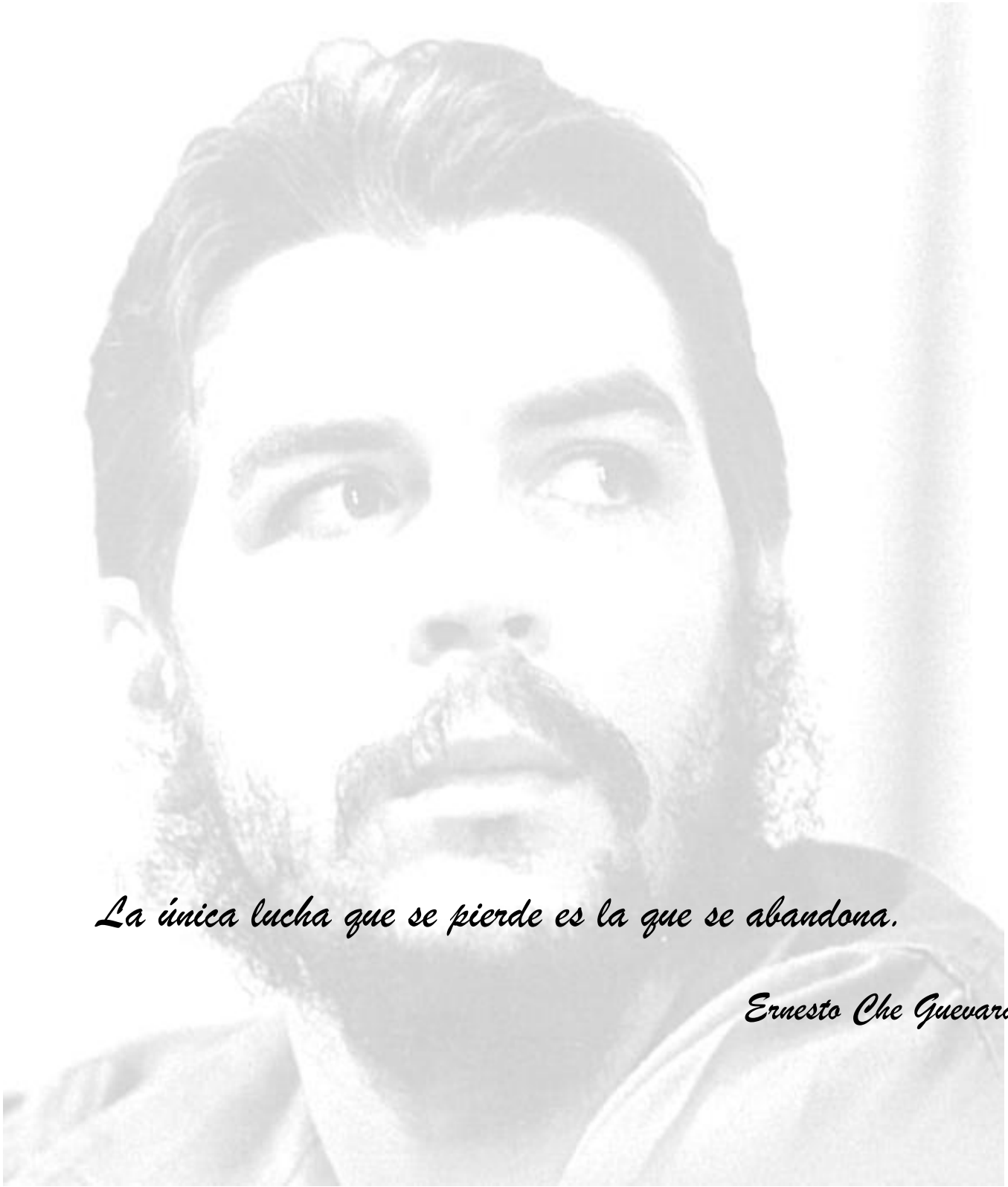
Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

_____ _____

Autor: Anabel Reyes Rosa Tutor: Jorge Amado Soria Ramírez

_____ _____

Tutor: Enrique José Altuna Castillo Tutor: Nersa Doraines Acosta Labrada



La única lucha que se pierde es la que se abandona.

Ernesto Che Guevara

Dedicatoria

A mi mamita, por no fallarme nunca, por estar alentándome en mis momentos más difíciles y darme hasta lo imposible, por hacer más suyo este sueño que mío propio, por quererme como se que lo hace, porque nadie como ella para merecerse este título.

A mí papito que con su ejemplo ha influido en lo que soy, por haber dedicado cada segundo a mi lado para enseñarme algo valeroso en la vida, porque ha sido mi orgullo e inspiración desde que tengo uso de razón y ha disfrutado de lo grande con cada triunfo mío.

A mi hermanita Lulú por confiar siempre en mí y alentarme a ir tras cada uno de mis sueños, y porque sé que aunque me he pasado muchos años lejos de ella la distancia lo que ha conseguido es que me quiera más.

Agradecimientos

A Fidel y mi revolución que me han dado la posibilidad de descubrir lo que me gusta y desarrollarlo, sueños casi seguro imposibles para mí en otro lugar. Gracias por permitir que me pueda estar graduando de ingeniera.

A mis padres por su apoyo incondicional, sus sacrificios y su fe en mí todo el tiempo. A Félix y Ana María por haber sido especiales y por preocuparse por mí como si fuera una hija más.

A lulú, mi abuela Isa y mi tío Cuqui por estar siguiendo cada paso mío y contribuir a que alcanzara éxitos. A toda la familia.

Especialmente a Machito porque ha estado más cerca de mí en estos 4 años que nadie, por ser tan lindo conmigo y entregarse a fondo en todo para que me sienta como en casa. Por su amor y entrega. A sus padres.

A mis tutores por su apoyo, especialmente a Nersa por motivarme primero con la tesis, luego por convertirse en mi compañera y terminar siendo mi tutora.

A mis amigos del tenis, que nos hemos unido como una familia y en estos 6 años con ellos he compartido lo que más me gusta.

Resumen

Los jurados en línea son aplicaciones web disponibles todo el tiempo; que proveen problemas y evalúan las soluciones que los usuarios envían. Contribuyen al desarrollo de habilidades en las técnicas de análisis y diseño de algoritmos. En la mayoría de los entornos cubanos donde se utilizan estos sistemas, no es posible aprovecharlos bien debido a que la velocidad de la red es muy lenta. A esto se le agregan aquellos lugares donde a pesar de que se cuenta con servicio de correo electrónico no existe conexión y por tanto el uso de estas aplicaciones es totalmente nulo. En la Universidad de las Ciencias Informáticas se encuentra desplegado el Jurado Online del Caribe que está disponible en la red nacional e internet pero por las razones explicadas el acceso a este se dificulta. El objetivo del presente trabajo es elaborar el análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe, que provea el conjunto mínimo de funcionalidades de un jurado online estándar y genere información a un formato estándar de visualización de documentos. Para dar cumplimiento al objetivo del trabajo se realizó un estudio del estado del arte sobre los sistemas similares y las tecnologías, se definieron los requerimientos, se elaboró el diagrama de casos de uso del sistema y sus descripciones, se realizaron los diagramas de clases del análisis y del diseño, los cuales constituyen un punto de partida para las actividades de implementación.

Índice de tablas

Índice de figuras

Índice de contenido

I

Introducción

En la actualidad, la carrera de Ingeniería Informática incluye como parte central del plan de estudio la enseñanza de la programación; ya que es a través de esta materia que los estudiantes desarrollan las habilidades que serán necesarias en su vida laboral. Sin embargo, a partir de la experiencia de La Universidad de las Ciencias Informáticas (UCI) es palpable en los alumnos que ingresan a la misma la presencia de dificultades en el dominio de este contenido; carencias generalmente dadas por el desconocimiento de los paradigmas que rigen la programación, la falta de habilidades algorítmicas y de abstracción.

Para enfrentar este problema han surgido como soluciones los concursos de programación. Estos eventos fueron gestados en la década del 70 por William B. Poucher quien se desempeña como director general del ACM International Collegiate Programming Contest (ACM-ICPC), evento organizado anualmente por la Association for Computing Machinery (ACM). Estas competencias han favorecido el desarrollo de habilidades en las técnicas de análisis y diseño de algoritmos, la cooperación en un ambiente de equipo, la productividad y el pensamiento algorítmico, contribuyendo a la formación de mejores profesionales.

Para apoyar la preparación de estos concursos y la enseñanza de la programación en sentido general, surgen los jurados online. Estos básicamente son aplicaciones web disponibles todo el tiempo que automatizan la evaluación de las respuestas a los problemas de programación de competencia.

En los últimos años se ha incrementado el acceso a los jurados online desde diversas instituciones cubanas (universidades, joven clubs, centros de trabajo e institutos preuniversitarios) fruto del despliegue y uso del Juez en Línea del Caribe (http://coj.uci.cu), que está disponible para usuarios del país y el mundo a través de internet. Sin embargo se puede afirmar que en estos centros continúa siendo muy limitado el aprovechamiento de los jurados online, debido a que la velocidad de la red es muy baja. Se afecta así uno de los pilares fundamentales de este tipo de software: su disposición constante.

Aun cuando la red desde la cual se desee acceder al sistema de evaluación automática no sufra de la afectación anteriormente mencionada, otros factores pueden interferir en la disponibilidad de la interacción del usuario con el sistema. Los mismos pueden ser organizacionales (horarios de clases, pases, reuniones, o demás afectaciones del cronograma de actividades); tecnológicos (disponibilidad de un puesto de trabajo o de herramientas de programación adecuadas a la tecnología que se desea utilizar); o

simplemente humanos (para prácticas en equipo, dificultad de coordinar la presencia de todos los integrantes del mismo en un solo lugar, o para prácticas individuales, la disposición particular de una persona en resolver un problema determinado en un momento dado). A esto se le agregan aquellos lugares donde a pesar de que los usuarios cuentan con servicio de correo electrónico no existe conexión a la red nacional y por tanto el uso de estas aplicaciones es totalmente nulo.

A partir de los resultados obtenidos en la UCI en cuanto a la aplicación de jurados online en competencias y a la auto preparación de los estudiantes se puede afirmar que contar con el acceso a estos sistemas debe ser un requisito de cualquier facultad de ciencias informáticas o ingenierías relacionadas.

Atendiendo a la problemática anteriormente explicada se define como **problema a resolver:** ¿cómo lograr que usuarios con redes de poco ancho de banda o limitados en su acceso a un sistema de evaluación automática, puedan interactuar con el mismo obteniendo resultados semejantes que con un jurado online estándar?

El problema a resolver identificado anteriormente se enmarca en el **objeto de estudio:** proceso de desarrollo de aplicaciones de evaluación automática, protocolos de transferencia de correo y generación de documentos.

El **objetivo general** de la investigación es: elaborar el análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe que provea el conjunto mínimo de funcionalidades de un jurado online estándar y genere documentos a un formato estándar de visualización.

Se tiene como **campo de acción:** análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe y generación de documentos a un formato estándar de visualización.

Se sustenta la **idea a defender** siguiente: con el análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe que provea información en un formato estándar de visualización de documentos, se podrá mejorar la disponibilidad de este tipo de software en centros con redes de poco ancho de banda o limitadas en su acceso a internet.

En función de dar cumplimiento al objetivo trazado, se decide desarrollar las siguientes **tareas de investigación:**

- Análisis del estado del arte de los jurados online actuales, los protocolos de transferencia de correo y los formatos de visualización de documentos
- Selección de las herramientas y tecnologías adecuadas para utilizar en la aplicación.
- Definición de las funcionalidades a incluir en el sistema.
- Confección del diseño del sistema, con la correspondiente documentación.
- Diseño de la integración de la aplicación con el Jurado del Caribe.
- Demostración de la factibilidad de la propuesta de solución.

Para el desarrollo de dichas tareas se combinan diferentes **métodos científicos**, los fundamentales son:

A nivel teórico:

Analítico-sintético: Empleado al analizar toda la teoría y documentos que permiten la extracción de los elementos más importantes relacionados con los jurados online de programación.
Inductivo-deductivo: Se manifiesta al analizar los principales jurados online de programación y determinar de entre todas sus funcionalidades cuáles son las que debe tener el sistema; así como las herramientas y frameworks que debe utilizar.

Histórico-lógico: Utilizado durante la investigación de los antecedentes y las tendencias actuales relacionadas con el desarrollo de los jurados online y en la profundización de los conceptos y términos relacionados con el objeto de estudio y el campo de acción.

El presente trabajo está conformado por 3 capítulos, además cuenta con conclusiones, recomendaciones, bibliografía, glosario y anexos. El primer capítulo, Fundamentación teórica, incluye la revisión del estado del arte comenzando por las funcionalidades de los principales jurados online actuales y la investigación de las tecnologías, metodologías y herramientas a utilizar en el desarrollo de la solución, así como la argumentación de la selección hecha. El segundo capítulo, Características del sistema, contiene la identificación y descripción de los conceptos asociados al ámbito del sistema. Son definidos los requisitos funcionales y no funcionales que debe cumplir el sistema, además se elabora el diagrama de casos de uso y la descripción textual de cada uno de estos. El último capítulo, Análisis y diseño del sistema, aborda cómo será construido el mismo basado en las descripciones del capítulo anterior y con el apoyo de los diagramas de clases del análisis y del diseño. Además en dicho capítulo se plasma el modelo de datos de la base de datos que utiliza el sistema (la del Jurado del Caribe) y se detalla cómo se validó la aplicación.

Capítulo 1

En este capítulo se realiza un estudio de las soluciones semejantes del sistema, haciendo énfasis en sus funcionalidades básicas y determinando las que serán aplicables a la solución. Se abordan diferentes formatos que existen para visualizar documentos y se elige el adecuado para la aplicación, se detallan los protocolos de transferencia de correo que se van a utilizar. También se hace un análisis de las metodologías, herramientas, otras tecnologías y una breve justificación de las seleccionadas.

1.1 Jurados online de programación

Los jurados en línea son aplicaciones web disponibles todo el tiempo, que automatizan la evaluación de las respuestas a los problemas de programación de competencia y poseen un público objetivo muy especializado. Estos sistemas se asocian normalmente con entornos académicos (en especial universidades), debido a que en estos la búsqueda de conocimiento teórico se estimula de forma más activa. Los jurados presentan estas características, porque sus objetivos y funcionamiento no son fáciles de convertir en lucrativo, excepto por las actividades colaterales de capacitación, publicaciones y eventos especiales por invitación. (Ramirez, et al., 2008)

Estos sistemas constan generalmente de dos aplicaciones que funcionan de manera conjunta, una aplicación web para interactuar con los concursantes y otra encargada de la calificación de las soluciones enviadas por los clientes. Tal es el caso de los jurados Xtreme y CPAV, actualmente en uso en la comunidad universitaria de la UCI.

Aunque de estos sistemas existe una amplia documentación no hay un estándar definido para su desarrollo, que básicamente está regido por las necesidades de la organización donde estará presente y/o de las habilidades o visión del equipo de desarrollo que lo implemente. De modo general poseen un alto nivel de disponibilidad, permitiendo el libre acceso al sistema en todo momento a los usuarios.

Entre los principales jurados online de programación a nivel mundial que forman parte del estado del arte de esta investigación, se destacan:

- Jurado Online de la Universidad de Valladolid (UVa) (http://acm.UVa.es)
- Jurado Online de la Universidad de Gdansk en Polonia(SPOJ) (http://www.spoj.pl)
- Jurado Online de la Universidad de Pekín(PKU) (http://poj.org)

En la UCI hace aproximadamente cinco años se comenzaron a desplegar jurados online de programación. Actualmente los más importantes son:

- Jurado Online Xtreme (http://onlinejudgef8.uci.cu:5800/JudgeOnline; http://coj.uci.cu)
- Jurado Online de la Cátedra de Programación Avanzada (CPAV) (http://cpav.uci.cu)

1.1.1 UVa

El jurado online de la Universidad de Valladolid (UVa) es el más antiguo y reconocido sistema de evaluación automática de problemas de programación. Es fuente de problemas de los demás jurados, al contar con un amplio banco de estos organizados en volúmenes que se nutren de las competencias de la ACM. Posee entre sus servicios: una interfaz amigable, un detallado conjunto de estadísticas, rankings y un mecanismo sencillo para recibir sumisiones. (Revilla, et al., 2008)

El responsable de su surgimiento fue un estudiante de informática llamado Ciriaco García de Celis. Aunque este sistema fue desarrollado para el entrenamiento de los competidores de los concursos de programación internacional, se ha convertido en una herramienta utilizada para el autoestudio a nivel mundial. (Revilla, et al., 2008)

El grupo que se encarga de administrar este jurado forma parte de la Universidad de Valladolid y sus miembros gozan de gran prestigio a nivel mundial (UVa, 2005). Un ejemplo lo constituye el autor Miguel Revilla que es reconocido como experto en el tema de la programación de competencia a partir de sus libros publicados. Los integrantes de este equipo han estado involucrados en las competencias de la ACM a escala internacional.

En general, se puede afirmar que el UVa es el jurado más grande y prestigioso de todos los que serán objeto de análisis en esta investigación. Contiene los volúmenes más numerosos de problemas presentes en Internet y para la Universidad de las Ciencias Informáticas constituye la principal referencia.

1.1.2 SPOJ

El Sphere Online Judge (SPOJ) es un proyecto del Sphere Interest Project, entidad afiliada a la Universidad de Tecnología de Gdansk en Polonia. Es una solución técnica muy sofisticada. Su sistema de

puntuación es el más complicado de todos los jurados que serán objeto de análisis, al igual que el sistema de distribución de los problemas, de los cuales reconoce varios tipos distintos. Estos aunque no muy numerosos son mantenidos en varios volúmenes y han sido replicados en ruso, portugués y vietnamita. En los volúmenes se encuentran todos los problemas presentados en las diversas ediciones de la ICPC, así como adiciones de los mismos usuarios y administradores del sitio. (Ramirez, et al., 2008)

El SPOJ cuenta con varios rankings, en los cuales es determinante la puntuación de cada usuario, siendo agrupados por países, por categorías (problemsetter, challenger, normal). Dichos rankings, a diferencia de los de otros sistemas, no cambian solamente cuando el usuario en cuestión realiza una operación, sino cada vez que se realiza una sobre uno de los problemas que tributan a la puntuación del usuario. Esto da como resultado un sistema muy dinámico de puntuación y asegura que los problemas más complejos valgan más y que los usuarios que pasen cierto tiempo sin realizar envíos bajen en el ranking a favor de los usuarios más activos. (Ramirez, et al., 2008)

Por otro lado es importante destacar que este jurado cuenta con un módulo especializado de competencia, muy versátil y eficiente, que ha sido anfitrión de muchas competencias (oficiales y de los propios usuarios). (SPOJ, 2007)

Soporta más de 30 lenguajes, lo que representa la mayor cantidad de estos asumidos por un jurado en Internet. Sin embargo, es importante destacar que la mayoría de estos lenguajes son muy especializados, anticuados o simplemente con una base de usuarios muy pequeña. (Ramirez, et al., 2008)

En resumen, el SPOJ es una solución tecnológicamente elegante, muy flexible y que soporta todas las funcionalidades básicas de un jurado online maduro. Se destaca por tener una adecuada variedad de problemas y una base estable de usuarios y competencias.

1.1.3 PKU

El PKU Judge Online (PKU) además de las funcionalidades estándares que presentan el resto de estos sistemas, ofrece correo interno. Su módulo de estadísticas de sumisión es muy interesante, ya que permite apreciar el devenir del jurado desde que fue creado en 2003. El funcionamiento de este es muy suave, intuitivo y rápido. (Ramirez, et al., 2008)

Es un sitio muy utilizado, según estadísticas publicadas en él, se realizaron 1 634 894 sumisiones en el año 2011, lo cual junto al análisis de los envíos diarios lleva a suponer que este sistema es el más visitado

de todos los analizados. Tiene la interfaz más intuitiva de todos y se basa en una comunidad activa de usuarios. (PKU, 2008)

La mayor diferencia entre este y los demás jurados online es la opción de descargar una versión libre de la aplicación. Esta es mucho más pequeña y restrictiva que el original, además de componerse de ficheros compilados que no proporcionan código fuente. La opción de descarga está diseñada para poner en funcionamiento rápidamente una versión del jurado en cualquier entorno. (Ramirez, et al., 2008)

En resumen el PKU es el más visitado de todos los jurados analizados y el que tiene la interfaz más intuitiva. Alberga muchos servicios adicionales y como punto importante a destacar es que se basa en una comunidad activa de usuarios.

1.1.4 Xtreme

El jurado Xtreme está basado funcionalmente en la versión libre del jurado de Pekín y es muy sencillo. Su desarrollo se hizo en Java y tiene como principios fundamentales: simplicidad en las herramientas utilizadas en su construcción, velocidad de respuesta y estabilidad del sistema. Fue desarrollado en la UCI y ha permanecido en activo durante 5 años, cosechando numerosos logros en este período de tiempo.

Consta de las funcionalidades básicas de cualquier jurado online estándar y otras que se le han ido adicionando, entre las que se destacan: agrupación de ranking, sistema de puntuación dinámico (tanto para volumen público como para competencias), funcionalidad de hora muerta y otras mejoras menores. ()

Al ser un sistema en desarrollo y mantenimiento activo se nutre de las ideas de las demás aplicaciones de su tipo en el mundo, además de constituir la línea oficial para dichas herramientas en el marco de la asociación de ACM-ICPC en Cuba.

El sistema del jurado Xtreme consta actualmente de 2 instancias confirmadas: una para el uso interno de la UCI, (Jurado Xtreme de la facultad 4) y una para el uso de los usuarios de la red nacional y el mundo (Jurado Online del Caribe). Las 2 instancias corren iguales versiones del código base. Para la instancia del Juez del Caribe anteriormente mencionada se va a desarrollar el módulo de comunicación por correo electrónico, ya que es esta la que se encuentra de cara a la red nacional e internet.

1.1.5 CPAV

El jurado de la cátedra avanzada de programación es el más joven de los jurados de la UCI. Rompe con los esquemas de los jurados anteriores, en filosofía, herramientas y objetivos. Está construido usando un

CMS de PHP (más exactamente, el PHP-Fusion) y para evaluar las soluciones no utiliza el mismo software, sino un daemon aparte construido en C++. (Ramirez, et al., 2008)

Entre las competencias que ha organizado se destacan: GrundyPC, She Programmer y la Copa Troya. El CPAV ofrece varios tipos de ranking: por año, por facultades, por lenguajes y general. Soporta los lenguajes clásicos de competencia (Java, C, C++ y Pascal), C#, python3.1 y python2.5. Recientemente (abril/2012) hospedó la copa individual Olimpo, convocada por el colectivo de administración de la CPAV. (CPAV, 2006)

Ofrece una serie de servicios que derivan de su calidad de CMS, como son: la galería, el shoutbox, los anuncios y los últimos usuarios conectados. En contraste, no brinda la funcionalidad básica de ver el estado actual de las sumisiones fácilmente, lo cual es una deficiencia grave. El sistema de clasificación de los problemas en volúmenes por temas puede ser contraproducente dado que ofrece información sobre los conocimientos necesarios para resolver el problema. El uso de un CMS puede resultar poco efectivo a la hora de aumentar las funcionalidades de la aplicación o de evolucionar su interfaz gráfica, dado que no se tiene control completo sobre la fuente, al ser código legado con sus propias funcionalidades y arquitectura. (Ramirez, et al., 2008)

Después de analizar las soluciones existentes de jurados online se decidió que las funcionalidades a incluir en el sistema serían aquellas que respondieran a las operaciones básicas para el uso de los jurados online y la presentación de contenido a un formato estándar de visualización. Esta última funcionalidad juega un papel fundamental en el módulo de comunicación sobre correo electrónico para el Juez en Línea del Caribe. Las funcionalidades seleccionadas como conjunto mínimo imprescindible se modelaron como requisitos funcionales del sistema.

1.2 Metodologías de desarrollo

Para asegurar el éxito del proceso de desarrollo y la producción de una documentación consistente durante la implementación de la solución, es altamente recomendable utilizar una metodología de desarrollo. Las más utilizadas y conocidas a nivel mundial y académico son: XP, Scrum y RUP, las cuales se analizan a continuación.

1.2.1 XP

La Programación Extrema (XP, eXtreme Programming) es una metodología ligera de desarrollo de software que se basa en la simplicidad, la comunicación y la realimentación o reutilización del código desarrollado. (Solís, 2008)

Es la más destacada de los procesos ágiles de desarrollo de software para proyectos de corto plazo y pequeño equipo. Pone más énfasis en la adaptabilidad que en la previsibilidad, los cambios de requisitos sobre la marcha son un aspecto natural, inevitable e incluso deseable del desarrollo de proyectos.

La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto. Los defensores de XP creen que ser capaz de adaptarse a los cambios de requisitos en cualquier punto de la vida del proyecto es una aproximación mejor y más realista que intentar definir todos los requisitos al comienzo del proyecto e invertir esfuerzos después en controlar los cambios en los requisitos.

"XP supone:

- Las personas son claves en los procesos de desarrollo.
- Los programadores son profesionales, no necesitan supervisión.
- Los procesos se aceptan y se acuerdan, no se imponen.
- Desarrolladores y gerentes comparten el liderazgo del proyecto.
- El trabajo de los desarrolladores con las personas que conocen el negocio es regular, no puntual." (Solís, 2008)

1.2.2 Scrum

"Scrum es una metodología de desarrollo muy simple, que requiere trabajo duro porque no se basa en el seguimiento de un plan, sino en la adaptación continua a las circunstancias de la evolución del proyecto.

Es una metodología ágil, y como tal:

- Es un modo de desarrollo de carácter adaptable más que predictivo.
- Orientado a las personas más que a los procesos.
- Emplea la estructura de desarrollo ágil: incremental basada en iteraciones y revisiones." (Palacio, 2006)

Establece un conjunto de prácticas y roles, y que puede tomarse como punto de partida para definir el proceso de desarrollo que se ejecutará durante un proyecto. Los roles principales en Scrum son el ScrumMaster, que mantiene los procesos y trabaja de forma similar al director de proyecto, el ProductOwner, que representa a los stakeholders (clientes externos o internos), y el Team que incluye a los desarrolladores.

Durante cada sprint, un período entre 15 y 30 días, el equipo crea un incremento de software potencialmente entregable. Estas iteraciones son la base del desarrollo ágil, gestionándose su evolución a través de reuniones breves diarias en las que todo el equipo revisa el trabajo realizado hasta la fecha y la previsión para el día siguiente.

Un principio clave de Scrum es el reconocimiento de que durante un proyecto los clientes pueden cambiar de idea sobre lo que quieren y necesitan (a menudo llamado requirements churn), y que los desafíos impredecibles no pueden ser fácilmente enfrentados de una forma predictiva y planificada.

1.2.3 RUP

EL Rational Unified Process (RUP) es un proceso de desarrollo de software que constituye el estándar más usado para el análisis y documentación de sistemas orientados a objetos. Se caracteriza por ser iterativo e incremental, estar centrado en la arquitectura y guiado por los casos de uso (CU). Incluye artefactos (que son los productos tangibles del proceso, por ejemplo, el modelo de casos de uso) y roles. Está basado en componentes, consta de nueve flujos de trabajos fundamentales, de ellos seis denominados “de Ingeniería”, que son: Modelado del Negocio, Requerimientos, Análisis y Diseño, Implementación, Prueba y Despliegue. Los restantes tres flujos de trabajo son considerados de apoyo: Administración de Proyectos, Gestión de Configuración y Cambios y Ambiente. (Shuja, et al., 2008)

Es más apropiada para proyectos grandes (aunque es configurable para proyectos pequeños).

Características

- Forma disciplinada de asignar tareas y responsabilidades (quién hace qué, cuándo y cómo).
- Pretende implementar las mejores prácticas en Ingeniería de Software.
- Desarrollo iterativo.
- Administración de requisitos.
- Uso de arquitectura basada en componentes.

- Control de cambios.
- Modelado visual del software.
- Verificación de la calidad del software.

Luego de realizar un estudio de las metodologías anteriores se elige RUP para el sistema porque tiene bien definido y organizado el proceso de desarrollo de software, lo que permite que los involucrados estén claros de los pasos a seguir para obtener un producto con calidad. Provee un entorno de proceso de desarrollo configurable, atendiendo a las características específicas del sistema. Teniendo en cuenta que la aplicación a desarrollar es de nuevo tipo y se espera que tenga continuidad, es importante gestionar una documentación detallada para apoyar el futuro desarrollo. Una de las vías para alcanzar esto es a través de artefactos específicos de RUP. Estas características representan algunas de las razones que justifican la elección para guiar el desarrollo del módulo de comunicación por correo electrónico para el Juez del Caribe con dicha metodología.

1.3 Herramientas CASE

CASE corresponde a las iniciales de: Computer Aided Software Engineering; y en su traducción al Castellano significa Ingeniería de Software Asistida por Computadoras. Se puede definir a las herramientas CASE como un conjunto de programas y ayudas que dan asistencia a los analistas, ingenieros de software y desarrolladores en el modelado visual, durante todos los pasos del ciclo de vida de desarrollo de un software. (Tuesta, 2005)

1.3.1 Rational Rose

Es una herramienta para el modelado visual mediante UML (Unified Modeling Language) de sistemas de software. Proporciona un marco de desarrollo que permite cubrir todo el ciclo de vida de un proyecto, en el que se puede especificar, analizar y diseñar un sistema antes de codificarlo; ayudando a establecer una trazabilidad real entre el modelo (análisis y diseño) y el código ejecutable. Además, permite generar código a partir de los modelos y realizar ingeniería inversa (crear modelo a partir de código). (González Blanco, et al., 2010)

Aunque sus características la convierten en una herramienta muy potente y deseable, la particularidad de ser propietaria hace que sea rechazada por muchas empresas y desarrolladores de software.

1.3.2 Visual Paradigm for UML

Es una herramienta profesional de modelado que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. Permite representar todos los tipos de diagramas de clases, hacer ingeniería inversa, generar código desde diagramas y generar documentación. Brinda soporte para varias versiones de UML. Es una herramienta multiplataforma que ayuda a una rápida construcción de aplicaciones de mejor calidad y a un menor costo. Posibilita la integración con los principales IDEs de desarrollo. (Sierra, 2009)

A partir del análisis de las herramientas anteriores, se decidió que Visual Paradigm for UML es la seleccionada para llevar a cabo el modelado visual. Para esto se tiene en cuenta que sus características la convierten en una de las herramientas más potentes y utilizadas en la actualidad. El hecho de que en el entorno donde se está desarrollando la solución se posee su licencia comprada y que se integre con los principales IDEs de programación constituyen también razones de peso.

1.4 Herramientas de desarrollo

Debido a la naturaleza de la aplicación que se quiere lograr y su relación estrecha con el software ya existente (Jurado Xtreme) la mayoría de las herramientas a utilizar tienen su selección bien fundamentada en el trabajo de diploma: “Análisis, diseño e implementación de un Jurado Online”, de los autores Jorge A. Soria Ramírez y Enrique José Altuna Castillo.

En las versiones anteriores de la aplicación se usó Windows XP como sistema operativo. Sin embargo, producto de que en la universidad donde se va a desarrollar la solución se está llevando a cabo un proceso de migración de software propietario a software libre, se decide adecuar el desarrollo a esta política. Quedando seleccionados Linux, particularmente Ubuntu 10.10, como sistema operativo.

El perfil tecnológico completo queda especificado en detalle en las conclusiones del presente capítulo.

1.5 Protocolos de correo electrónico

El módulo de comunicación por correo electrónico es para el Jurado en Línea del Caribe que está desplegado actualmente en la UCI. Este sistema de evaluación automática está puesto a disposición de los usuarios cubanos y del mundo a través de internet. La cuenta de correo que se va a utilizar para el trabajo con dicho módulo es la asignada para este jurado, por tanto, los protocolos a usar para la comunicación son el Protocolo Simple de Transferencia de Correo (SMTP, Simple Mail Transfer Protocol)

y el Protocolo de Acceso a Mensajes de Internet, versión 4 (IMAP4, Internet Message Access Protocol), que son los utilizados por el cliente de correo Zimbra vigente en la UCI. SMTP para el envío de mensajes e IMAP4 para la descarga de los mismos desde el servidor de correo.

1.5.1 SMTP

El objetivo de SMTP es transferir correo de forma fiable y eficiente. Dicho protocolo es independiente del subsistema de transporte particular y requiere sólo un canal confiable para la transmisión del flujo de datos. (Klensin, 2007)

Una característica importante de SMTP es su capacidad de transportar correos a través de redes distintas, lo que usualmente se conoce como “mail relaying”. Una red en este contexto está constituida de servidores públicos en internet, mutuamente accesibles a través de TCP; de servidores conectados por TCP dentro de una red aislada y protegida por un firewall o router, o servidores en LANs o WANs diferentes que usan otro protocolo de transporte distinto de TCP. A través de SMTP, un proceso puede transferir un correo a otro proceso en la misma red o en alguna otra a través de una puerta de enlace accesible a ambas redes. (Klensin, 2007)

SMTP está ampliamente difundido y existen implementaciones de alta calidad que han demostrado ser muy robustas.

1.5.2 IMAP4

El protocolo IMAP4 permite a un cliente acceder y manipular mensajes de correo electrónico en un servidor. Así como la manipulación de buzones de correo de una manera que es funcionalmente equivalente a carpetas locales de correo. IMAP4 también proporciona la capacidad de una línea cliente para volver a sincronizar con el servidor. (Crispin, 2003)

Este protocolo incluye operaciones para crear, eliminar y cambiar el nombre de carpetas de buzones de correo, comprobar si hay mensajes nuevos, eliminar mensajes de forma permanente, manipular banderas y brinda criterios que pueden utilizarse para ordenar y obtener los correos electrónicos. (Crispin, 2003)

IMAP sobrecarga mucho el servidor de correo, requiriendo que este reciba los nuevos mensajes y los entregue a los usuarios cuando sean solicitados, manteniéndolos en el servidor. Aunque esto centraliza las copias de seguridad, también hace que las carpetas de correo a largo plazo de los usuarios se hagan más grandes. Para usar el IMAP, el servidor de correo debe soportar este protocolo. (Crispin, 2003)

Para el sistema no se corre el riesgo de que el buzón asignado se haga muy grande pues los correos serán eliminados inmediatamente que sean respondidos.

1.6 Tecnología para el envío y recepción de correos

El API JavaMail es un conjunto de funcionalidades y clases que modelan un sistema de correo. Proporciona un framework independiente de la plataforma o el protocolo para construir aplicaciones clientes de correo electrónico basadas en tecnología Java. Brinda los medios para leer y enviar correo electrónico. Los proveedores de servicios implementan protocolos en particular. Varios proveedores de servicios se incluyen con el paquete del API JavaMail, mientras que otros están disponibles por separado. El API JavaMail se implementa como un paquete opcional de Java que puede ser utilizado en el JDK 1.4 y versiones posteriores en cualquier sistema operativo. (ORACLE, 2010)

Se selecciona esta tecnología debido a su gratuidad y por las ventajas que brinda para el programador su fácil uso. JavaMail permite, al implementar desde cero las funcionalidades de correos, el dominio a fondo del código fuente. Esto flexibiliza el proceso de agregar nuevas prestaciones si así se requiere y que solamente tenga las que necesita, haciendo más ligera la solución, en contraste con la situación que se crea al usar una herramienta previamente construida, la cual brindaría funcionalidades innecesarias y sería más compleja de extender. El hecho de que Java sea la tecnología escogida por otros aplicativos de la Iniciativa Xtreme, en cuyo marco se encuentra el resultado esperado de este trabajo, también es un punto a su favor.

1.7 Formatos estándar para la visualización de documentos

Existen varios formatos para estandarizar la información que se muestra en los documentos. La existencia de estos formatos permite algo que en la actualidad parece básico pero que hace muy pocos años no lo era: la portabilidad de los documentos de una impresora o filmadora a otra. Seguidamente se analizan algunos de los formatos para la visualización estándar de documentos y se explica el por qué de la elección de PDF (Portable File Document) para la aplicación.

1.7.1 PDF

PDF es un formato de almacenamiento de documentos, desarrollado por la empresa Adobe Systems que se considera de tipo compuesto (imagen vectorial, mapa de bits y texto). Está especialmente ideado para documentos susceptibles de ser impresos, ya que especifica la información necesaria para la presentación

final del mismo, determinando los detalles de cómo va a quedar, no requiriéndose procesos anteriores de ajuste ni de maquetación. Gracias a su nivel de compresión, accesibilidad y sencillez, se estandarizó dentro del mundo virtual, como el formato de mayor uso. (Leurs, 2000)

Algunas de sus peculiaridades muy destacables son:

- Es multiplataforma, es decir, puede ser presentado por los principales sistemas operativos (Windows, Unix/Linux o Mac), sin que se modifiquen ni el aspecto ni la estructura del documento original.
- Los ficheros PDF son compactos. El formato incluye una serie de complejos y avanzados algoritmos de compresión y una ingeniosa estructura de ficheros que hace que el tamaño de los PDFs se reduzca al mínimo.
- Los PDFs pueden contener elementos multimedia (vídeos o sonido) y elementos de hipertexto (vínculos y marcadores), enlaces a direcciones de correo y pequeñas miniaturas de las distintas páginas.
- Es una especificación abierta, para la que se han generado herramientas de software libre que permiten crear, visualizar o modificar documentos en formato PDF. Un ejemplo es la suite ofimática OpenOffice.org y el procesador de textos LaTeX.
- Puede cifrarse para proteger su contenido e incluso firmarlo digitalmente. (Leurs, 2000)

1.7.2 XPS

El formato XPS, que viene de las siglas XML Paper Specification es un archivo electrónico desarrollado por Microsoft y basado en el estándar XML. Este formato, muy similar al PDF de Adobe, garantiza que cuando el archivo se ve en línea o se imprime, conserva exactamente el formato deseado y los datos del archivo no se pueden cambiar fácilmente, por lo que resultan muy útiles en currículos, documentos jurídicos, boletines informativos y otros archivos concebidos para ser leídos, impresos y aptos para compartir. XPS es un formato independiente de plataforma (que se podría leer en cualquier sistema operativo), abierto y sin royalties (se puede utilizar en cualquier caso sin tener que pagar ningún derecho al creador). (Alvarez, 2009)

Microsoft ha lanzado este formato recientemente y lo han integrado con Windows Vista y Office 2007. Es una apuesta para competir con el formato PDF, aunque todavía tiene bastante por delante para que se

convierta en un verdadero estándar abierto. En Windows XP el visor de documentos XPS habría que instalarlo aparte, porque de entrada no tiene compatibilidad con este formato. Aunque para ello se tiene que instalar primero el .NET Framework 3.0, lo que supone un buen tiempo para, al fin, poder ver los documentos XPS. (Alvarez, 2009)

A pesar de que XPS brinda muchas de las facilidades que ofrece PDF habrá que esperar un tiempo hasta que la distribución de contenidos y documentos de XPS sea tan habitual como la del formato PDF.

1.7.3 PostScript

PostScript es un formato de documentos, creado por la empresa Adobe, para describir documentos listos para imprimir, es decir, documentos que ya están maquetados y preparados para ser impresos, pero que ya no se pueden editar. Hay impresoras que entienden directamente este formato (impresoras PostScript). El formato PostScript se ha convertido, junto con el PDF, en un estándar para el almacenamiento de documentos listos para imprimir. Los archivos PostScript suelen tener extensión PS. Para visualizar un documento PostScript en el ordenador, se necesita un programa visor. Tanto para UNIX como para Windows existe un programa gratuito llamado GhostView, que además permite imprimir el documento aunque la impresora no sea PostScript. (Espino, 2006)

Un documento o fichero PostScript, como programa que es, debe atenerse a unas reglas de construcción muy precisas. Aparte de esto, puede contener dentro datos de todo tipo: textos, imágenes y descripciones matemáticas de gráficos. (Gusgsm, 2003)

La razón de peso que primó para escoger PDF como formato de visualización de documentos fue su reconocimiento a nivel mundial como estándar de mayor uso, ya que todos los estudiados agrupan en general un conjunto de ventajas que harían factible su uso en el sistema. Al centrar la elección en este aspecto se garantiza que los usuarios se sientan familiarizados con los documentos que enviará la aplicación. Además en el caso de PostScript tiene como desventajas que en ocasiones pueden ocurrir los temidos errores PostScript, generados frecuentemente con las fuentes tipográficas y los problemas de “banding” (saltos o cortes) en los degradados.

1.8 Tecnologías para la generación de documentos en formato PDF

A continuación se analizan 3 de las tecnologías para la generación de documentos en formato PDF y se explica el por qué se escoge iText para la aplicación.

1.8.1 iText

iText es un API de código abierto creada por Bruno Lowagie y Paulo Soares que permite a los desarrolladores producir archivos PDF sobre la marcha y/o manipular documentos existentes de ese formato. Puede ser fácilmente integrado en una aplicación web para servir a los navegadores de documentos PDF de una manera dinámica. La premisa de iText es que no se precisa ser un especialista en PDF para generar documentos. (Lowagie, 2007)

Hoy en día, iText se utiliza en muchos servicios en línea y productos de software, directa o indirectamente. Los informes creados con una de las herramientas de generación de reportes más importantes del momento, JasperReports o el Eclipse/BIRT-ll usan iText pues este se utiliza como su motor PDF. Se puede encontrar gran cantidad de documentación y ejemplos en internet sobre su funcionamiento, lo que facilita su uso y comprensión. Una de sus principales ventajas es que puede convertir contenido HTML a formato PDF soportando la mayoría de las etiquetas. (Lowagie, 2007)

1.8.2 LaTeX

Es un sistema de composición tipográfica de documentos formado por un conjunto de macros de TeX (sistema de tipografía escrito por Donald E. Knuth). Es muy adecuado para la producción de materiales científicos y matemáticos de alta calidad tipográfica por su capacidad de representar ecuaciones, fórmulas complicadas, notación científica y las facilidades aportadas para estructurar el documento (con capítulos, secciones, notas, bibliografía e índices analíticos). Por estas razones hoy en día es muy popular en dichas comunidades, también es ampliamente utilizado en la industria. Los científicos envían sus documentos por vía electrónica a sus colegas de todo el mundo en el formato de entrada de LaTeX. (Lamport, 2010)

A diferencia de otros procesadores de texto, LaTeX permite concentrar la atención en el contenido y dejar a un lado las preocupaciones por el formato. Está basado en comandos, cuestión que es considerada una desventaja de dicha tecnología. Es software libre bajo licencia LPPL (Licencia Pública del Proyecto LaTeX).

1.8.3 JasperReports

JasperReports es el motor de informes de código abierto más popular del mundo. Está escrito completamente en Java y es capaz de utilizar los datos procedentes de cualquier tipo de fuente de datos y producir documentos de alta calidad que se pueden ver, imprimir o exportar en una variedad de formatos de documentos incluyendo HTML, PDF, Excel, OpenOffice y Word. (JasperForge, 2008)

JasperReports genera reportes a partir de archivos XML que especifican exactamente el formato que deben llevar los mismos. La construcción de este archivo sin el auxilio de una herramienta visual no es práctica. Para hacer esto se usa iReport, herramienta para obtener archivos XML que proporciona un entorno WYSIWYG (what you see is what you get) para diseñar informes. (Derin, 2005)

Debido a que no permite añadir imágenes de una forma dinámica (solo permite hacerlo si se ha predefinido con anterioridad en el archivo XML), no se selecciona esta potente librería. Esta limitación es un inconveniente en el contexto del presente trabajo pues la base de datos a utilizar posee mucha información en forma de contenido HTML con imágenes embebidas. Queda de esta forma como mejor opción iText, ya que LaTeX, a pesar de ser una de las tecnologías más usadas en la actualidad, no es una opción viable debido a su complejidad de uso.

1.9 Conclusiones

En este capítulo se realizó un estudio de las soluciones similares de sistemas de evaluación automática haciendo énfasis en sus principales funcionalidades y características; los protocolos de transferencia de correo, los formatos estándar de visualización de documentos y las tecnologías de generación de documentos a formato PDF. Para el posterior desarrollo de la solución se investigaron las herramientas disponibles, seleccionando las más adecuadas según diversos criterios.

Las herramientas libres de generación de archivos PDF existentes en el mundo Java son de alta complejidad, requiriendo un conocimiento profundo de las mismas. No existe en ninguna de estas herramientas una forma sencilla y rápida de transformar contenido HTML a contenido PDF manteniendo el formato original. iText es la herramienta libre más sencilla de manipular para lograr una conversión de calidad entre estos formatos.

Al estudiar las metodologías de desarrollo de software se pudo observar que RUP es el proceso más adecuado para generar documentación suficiente que permita continuar desarrollando versiones superiores del producto con personal distinto del equipo inicial.

Finalmente, el lenguaje de programación, las herramientas y tecnologías quedan especificados en el perfil tecnológico de desarrollo (**Tabla 1**).

Clasificación	Herramienta	Versión
Lenguaje de programación	Java	**6.0**
Gestor de datos	MySQL	**5.1.x**
Control de versiones	SUBVERSION	**1.6.12**
Cliente suvbersion	RapidSVN	**0.12.0**
Tecnología para generar documentos en formato PDF	itext	**5.0.6**
Tecnología para el envío y recepción de correos	JavaMail	**1.4.3**
Herramienta CASE	Visual Paradigm for UML Enterprise Edition	**6.4**
SO utilizado	ubuntu	**10.10**

Tabla 1 Perfil tecnológico de desarrollo

Capítulo 2

En el siguiente capítulo se hace una descripción del Modelo de dominio y sus conceptos fundamentales. Se plasman los requerimientos funcionales y no funcionales. Se definen los actores involucrados en el sistema y su justificación. Se identifican y describen los casos de uso. Por último se conforma el artefacto Modelo de casos de uso del sistema a partir de los actores, los casos de uso y sus relaciones.

2.1 Descripción del modelo de dominio

RUP como metodología de desarrollo de software propone realizar un modelo de dominio (modelo conceptual) antes de comenzar el desarrollo, en especial cuando los procesos de negocio no están claramente definidos.

El modelo de dominio es una representación visual estática del entorno real del proyecto. Es decir, un diagrama con los objetos que existen (reales) relacionados con el proyecto que se va a acometer y las relaciones que hay entre ellos. El modelo de dominio ayuda a comprender los conceptos que utilizan los usuarios, con los que trabajan y con los que deberá trabajar la aplicación. (Hall., 2011)

Los conceptos del dominio identificados son:

Jurado: Aplicación generalmente web que automatiza la evaluación de respuestas a problemas de programación y que ofrece otras funcionalidades.

Correo: El correo electrónico o email es un mensaje enviado por el usuario al jurado o viceversa, que contiene la información referida a cada una de las peticiones o funcionalidades del sistema.

Usuario: Individuo que utiliza el sistema mediante el envío y recepción de correos electrónicos. El público objetivo de la aplicación se espera esté compuesto principalmente por estudiantes universitarios de Informática.

Ayuda: Especificación del micro lenguaje utilizado para permitir la comunicación vía correo electrónico entre el usuario y módulo de comunicación.

FAQ (Frequently asked questions): Constituyen las preguntas frecuentes dentro del contexto del jurado online.

Noticia: Noticia o aviso de algún tema relacionado con el jurado, comúnmente competencias, período de mantenimiento o cambio de versiones o compiladores.

Problema: Ejercicio de programación propuesto a los usuarios del jurado para su resolución.

Solución: Código de programación escrito en un lenguaje válido y que da solución a un problema determinado de los dispuestos en los volúmenes del jurado.

Información: Es la información del usuario, incluye: los puntos que acumula en el jurado, la cantidad de soluciones que tiene aceptadas, el por ciento de soluciones aceptadas con respecto al total de soluciones y la cantidad total de soluciones.

Luego de detallar los conceptos del dominio se muestra la imagen correspondiente al Diagrama de objetos del dominio.

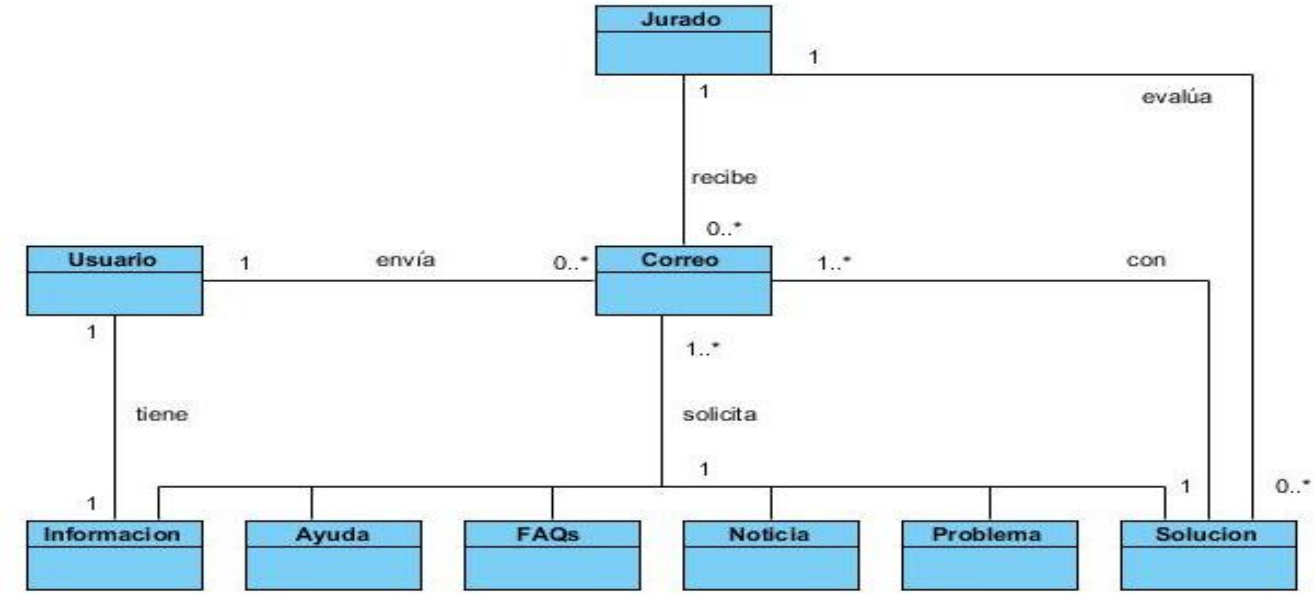


Figura 1 Diagrama de objetos del dominio

2.2 Especificación de requerimientos

Después de hacer un análisis del dominio del problema resulta fácil identificar los requerimientos funcionales y no funcionales que tendrá el módulo de comunicación por correo electrónico para el Jurado del Caribe. Los requerimientos funcionales son capacidades o condiciones que el sistema debe cumplir y los no funcionales son propiedades o cualidades que el producto debe tener. A continuación se presentan los requerimientos funcionales y no funcionales extraídos.

2.2.1 Requerimientos funcionales

RF1: Convertir a PDF información seleccionada.

RF1.1: Convertir a PDF las FAQs.

RF1.2: Convertir a PDF la ayuda.

RF1.3: Convertir a PDF un volumen de problemas.

RF1.4: Convertir a PDF un problema.

RF2: Ver ayuda.

RF3: Consultar el estado de un problema.

RF4: Ver noticias.

RF5: Someter solución de un problema.

RF6: Consultar el estado del servidor.

RF7: Ver FAQs.

RF8: Consultar el ranking de una cantidad n de usuarios especificada.

RF9: Consultar el listado de problemas de un volumen determinado.

RF10: Ver un problema.

RF11: Ver solución de un problema determinado.

RF12: Enviar correo.

2.2.2 Requerimientos no funcionales

Hardware

RNF1: Funcionar en una estación de trabajo estándar: 512 Mb de RAM, conexión a red local o Internet.

Software

RNF2: Tener acceso a un servidor de correo y a una cuenta en el mismo.

RNF3: Las estaciones de trabajo clientes deben tener acceso a un servidor de correo.

Usabilidad

RNF4: Debe proporcionar ayuda sobre cada funcionalidad disponible.

RNF5: El servidor de correo de las estaciones de trabajo clientes debe poseer acceso al servidor de correo del sistema.

Seguridad

RNF6: Las soluciones de los problemas solo pueden ser vistas por el propietario de las mismas.

RNF7: Solo se permitirá someter soluciones a los usuarios registrados.

Apariencia

RNF8: Debe presentar la información al usuario de manera clara y comprensible.

Soporte

RNF9: El correo y otros datos de contacto de los desarrolladores del sistema deben quedar claros para los usuarios.

Portabilidad

RNF10: Podrá ser utilizado bajo cualquier sistema operativo.

RNF11: Los sistemas clientes podrán interactuar con el contenido producido utilizando cualquier cliente de correo estándar.

Rendimiento

RNF12: Los adjuntos de los correos enviados no deben exceder el límite de tamaño establecido por el servidor de correo del sistema.

Restricciones de diseño e implementación

RNF13: Java 6.0 como lenguaje de programación.

RNF14: MySQL 5.1.x como sistema gestor de base de datos.

RNF15: VisualParadigm 6.4 como herramienta CASE para el modelado UML.

RNF16: JavaMail 1.4.3 como API para implementar las funcionalidades del correo.

RNF17: iText 5.0.6 como API para la implementación de las funcionalidades de la generación de documentos.

Estándares

RNF18: La comunicación entre los usuarios y el sistema debe estar regida por el micro lenguaje establecido en los anexos del 1 al 12.

2.3 Modelo de casos de uso de sistema

El modelo de casos de uso describe las funcionalidades propuestas del sistema. Un caso de uso representa una unidad discreta de interacción entre un usuario (humano o máquina) y el sistema. Un diagrama de casos de uso muestra las distintas operaciones que se esperan de una aplicación o sistema y cómo se relaciona con su entorno (usuarios u otras aplicaciones).

2.3.1 Actores

El actor representa un rol de un individuo, grupo de individuos, sistema automatizado o máquina e intercambia información con el sistema. En la siguiente figura se puede observar la representación UML de los actores del sistema propuesto.

Figura 2 Relación entre actores

A continuación se muestra una tabla con los actores del sistema y su justificación.

Actor	Justificación
Invitado	Es todo individuo que interactúa con la aplicación, o sea que utiliza el sistema mediante el envío y recepción de correos electrónicos. El invitado no precisa necesariamente una cuenta de usuario válida registrada en el Jurado Online Xtreme.
Usuario	Es todo individuo que interactúa con la aplicación del mismo modo que el invitado pero además puede enviar soluciones y ver las que tiene enviadas. Para realizar estas funciones, el usuario debe poseer una cuenta de usuario válida en el Jurado Online Xtreme.

Tabla 2 Actores

La siguiente imagen muestra el Diagrama de casos de uso del sistema.

Figura 3 Diagrama de casos de uso del sistema

2.4 Descripciones de casos de uso

Las descripciones de los casos de uso del sistema detallan las acciones que se producen a través de la interacción actor-sistema. Describen el flujo de actividades que se genera al ser utilizada la aplicación por el actor y las respuestas que le brinda el sistema. A continuación se muestran las descripciones de los casos de uso arquitectónicamente significativos y el resto quedan plasmadas en los anexos del documento.

2.4.1 Descripción del CUS "Consultar el estado del servidor"

Caso de uso	Consultar el estado del servidor.
Actores	Invitado
Propósito	Permitir consultar el estado del servidor.
Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema y en el asunto especifica los criterios que desea consultar del estado del servidor. El sistema valida que la petición del usuario esté en el formato correcto. Construye el estado del servidor atendiendo al rango y los criterios de consulta especificados por el usuario y le envía a este un mensaje de correo con los criterios del estado del servidor que solicitó consultar. Finaliza así el caso de uso.
Precondiciones	El usuario debe haber accedido al correo electrónico.
Referencias	RF6,RF12
Prioridad	Crítico

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto los criterios que desea consultar del estado del servidor, con el siguiente formato: mj: status <criterio><valor>**(Ver anexo 1)**	2. Valida el formato del asunto del correo. 3. Construye el estado del servidor atendiendo al rango y los criterios de consulta especificados: ❖ Usuario ❖ Resultado de la submisión ❖ Lenguaje

Ej. (mj:status -t server –u arosa –r ac –n 1-500 –l java)	4. Envía un mensaje de correo donde muestra los criterios del estado del servidor que se solicitó consultar.
	5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1del flujo básico.
Poscondiciones	Se consultó el estado del servidor.

Tabla 3 Descripción del CUS Consultar el estado del servidor

2.4.2 Descripción del CUS "Someter solución de un problema"

Caso de uso	Someter solución de un problema
Actores	Usuario
Propósito	Someter código de un problema determinado.
Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema especificando en el cuerpo del mismo el código que desea sea evaluado por el jurado. El sistema valida el formato del correo y que el usuario esté registrado. Guarda la solución y le envía al usuario un mensaje de confirmación con el id de la misma. Finaliza así el caso de uso.
Precondiciones	El usuario debe estar registrado en el sistema y haber accedido al correo electrónico.
Referencias	RF5,RF12
Prioridad	Crítico

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:submit -l <lenguaje> -i <# de problema>**(Ver**	2. Valida el formato del correo y que el usuario tenga permiso para someter la solución.
	3. Extrae el código enviado por el usuario delimitado por los separadores.
	4. Guarda el código en la base de datos.

Anexo 5) Ej. (mj:submit –l java –i 1234) Luego en el cuerpo del correo especifica el siguiente formato: <delimitador de inicio><código de la solución><delimitador de fin> **(Ver Anexo 7)**	5. Envía un mensaje de confirmación informando que la solución fue recibida y el identificador de la misma para si se desea verla.
	2. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido o no tiene permisos para someter soluciones.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1 del flujo básico.
Poscondiciones	Se envió la solución al jurado de un problema determinado.

Tabla 4 Descripción del CUS Someter solución de un problema

2.4.3 Descripción del CUS "Ver problema"

Caso de uso	Ver problema
Actores	Invitado
Propósito	Permitir ver información de un problema.
Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema y en el asunto pide que le muestre la información de un problema en específico. El sistema valida el formato del asunto del correo, busca el problema y le envía un mensaje de correo con la información del mismo. Finaliza así el caso de uso.
Precondiciones	El usuario debe haber accedido al correo electrónico.
Referencias	RF10,RF1.4, RF12
Prioridad	Crítico
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor	2. Valida el formato del asunto del correo.

envía un mensaje de correo especificando en el asunto el siguiente formato: mj:problem –i <id del problema >**(Ver Anexo 8)** Ej. (mj:problem –i 1256)	3. Busca el problema.
	4. Envía un mensaje de correo mostrando en el cuerpo del mismo: ❖ El ID del problema. ❖ El título del problema. ❖ La descripción del problema. ❖ El formato de entrada del problema. ❖ El formato de salida del problema. ❖ La entrada de ejemplo del problema. ❖ La salida de ejemplo del problema.
	5. Finaliza el caso de uso.
1. a Envía un mensaje de correo especificando en el asunto el siguiente formato: mj:problema<# de problema que desea ver> –f pdf	
	1. a.1Valida el formato del asunto del correo.
	1. a.2 Busca el pdf correspondiente al problema especificado en caso de que exista, sino lo construye.
	1. a.3 Envía mensaje de correo con el problema adjunto en formato PDF.
	1. a.4Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1Envía mensaje de correo con la ayuda.
	2. a.2Ejecuta el punto 1 del flujo básico.
Poscondiciones	Se vio la información de un problema.

Tabla 5 Descripción del CUS Ver problema

2.5 Conclusiones

En este capítulo se realizó el modelo de dominio, se identificaron los conceptos principales que se asocian al entorno del problema. Se extrajeron los requisitos funcionales y no funcionales. Se diseñó el Diagrama de casos de uso del sistema y se plasmaron las descripciones de los casos de uso relevantes.

Capítulo 3

En el presente capítulo se expone el análisis y diseño del sistema propuesto, modelándose los artefactos necesarios que contribuyen a su implementación. Dichos artefactos son: Modelo de análisis, Modelo de diseño, Diagrama de clases persistentes y Modelo de datos. Además se explica por qué es factible el análisis y diseño del módulo de comunicación por correo electrónico.

3.1 Modelo de análisis

El análisis posibilita que se refinen y estructuren los requisitos obtenidos con anterioridad para lograr un mejor entendimiento de los mismos. En el modelo de análisis están contenidas las clases del análisis que se centran en los requisitos funcionales. RUP propone clasificar dichas clases en:

Entidad: Modela información que posee larga vida. Refleja entidades del mundo real que resultan necesarias para realizar tareas internas del sistema.

Interfaz: Modela la interacción entre el sistema y sus actores.

Control: Coordina los eventos necesarios para implementar el comportamiento especificado en el caso de uso.

3.1.2 Diagramas de clases del análisis

El diagrama de clases del análisis constituye una vista estática de las clases que conforman el modelo del análisis y las asociaciones entre las mismas. Está compuesto por las clases interfaz, control y entidad. Constituye una abstracción del modelo de diseño.

Seguidamente se ilustran los diagramas de clases del análisis de los casos de uso arquitectónicamente significativos, el resto de los diagramas están plasmados en los anexos de este documento.

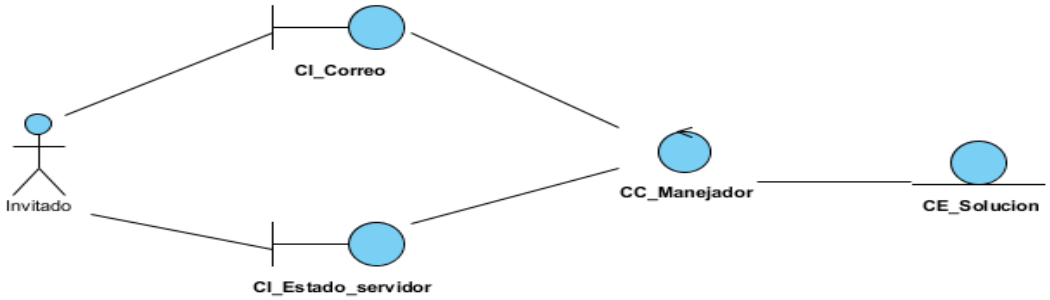


Figura 4 Diagrama de clases del análisis CU Consultar el estado del servidor

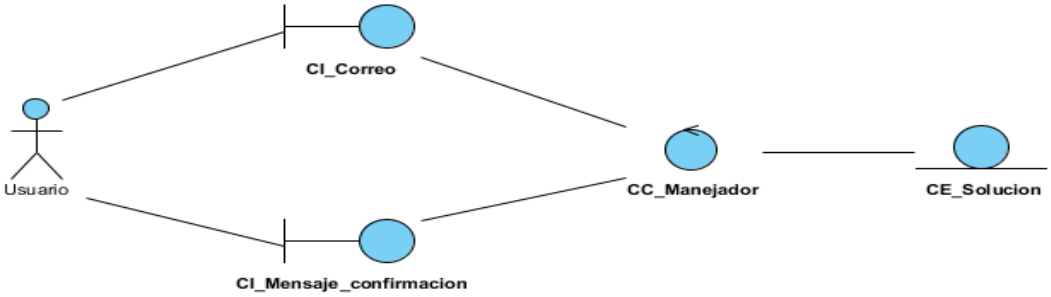


Figura 5 Diagrama de clases del análisis CU Someter solución de problema

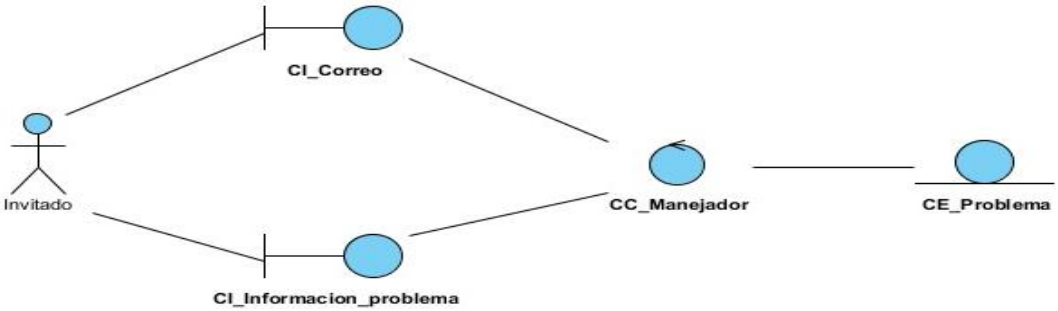


Figura 6 Diagrama de clases del análisis CU Ver problema

3.2 Patrones utilizados

Command

Command es un patrón de diseño GoF (grupo Gang of Four, compuesto por Erich Gamma, Richard Helm, Ralph Johnson y John Vlisides) que encapsula la acción que satisface una petición dentro de una clase particular. Normalmente las clases están organizadas dentro de una jerarquía donde cada una

corresponde a una funcionalidad determinada. Se relaciona con otros patrones de comportamiento o creacionales como: Builder, Template Method o Factory Method o Abstract Factory. (ErichGamma, 1994)

DomainModel

Patrón arquitectónico que permite modelar un sistema centrado en un modelo de dominio. Se entiende como modelo de dominio en este caso el conjunto de clases de la aplicación que se utilizan primariamente para contener datos y que constituyen la realización de las entidades detectadas en el negocio. Por ejemplo: Problema, Solución, Usuario, etc.

El Domain Model constituye el centro del sistema. Es frecuentemente detectado debido a que los objetos del dominio, además de corresponder a conceptos del negocio, se utilizan para mover información entre los componentes o las capas del sistema en el rol de DTOs.

DTO (Data Transfer Object)

Un DTO es una abstracción del sistema que no existe en el dominio. Se utiliza con el único propósito de encapsular una unidad de información que pueda ser transmitida entre componentes diferentes del sistema. Este objeto no pertenece al dominio de la aplicación pues no corresponde a un concepto identificado del negocio. Del mismo modo no pertenece a los estereotipos activos del sistema dado que normalmente no realiza ninguna acción con los datos que transporta o almacena. En ciertas ocasiones puede ser difícil distinguir entre DTOs y objetos del dominio si no se tiene el alcance del negocio claramente definido.

Intérprete

El intérprete es un patrón GoF de comportamiento que consiste en definir una representación de un lenguaje y un componente (clase o clases) para interpretar sentencias escritas en dicho lenguaje. La complejidad del lenguaje puede ser mayor o menor, pero el patrón está presente siempre que el sistema sea capaz de interpretar sentencias escritas por el usuario (o generadas por otros sistemas) y actúe en consecuencia. (ErichGamma, 1994)

Mediador

El mediador es un patrón de diseño que enfatiza la distribución de responsabilidad entre objetos. Este patrón define un objeto que encapsula la forma en que interactúan un grupo de objetos. Promueve un

acoplamiento débil al evitar referencias explicitas entre los objetos y permitiendo por tanto que su interacción se modifique de forma independiente.

El patrón mediador se utiliza para separar la aplicación en varios componentes o subsistemas que pueden funcionar con poca o ninguna relación entre ellos.

TransactionScript

El Transaction Script es un patrón de acceso a datos, derivado del patrón Facade. Consiste en concentrar todas las operaciones de interacción con la fuente de datos en un solo componente o pequeño grupo de componentes. Lo anterior posibilita que las dependencias necesarias para la interacción no se desperdiguen por todo el sistema sino que estén controladas en un solo lugar.

3.3 Modelo de diseño

El modelo de diseño amplía y detalla el modelo de análisis tomando en cuenta todas las implicaciones y restricciones técnicas. Su propósito es especificar una solución que trabaje y pueda ser fácilmente convertida en código fuente y construir una arquitectura simple. Las clases definidas en el análisis se detallan y se añaden otras para manejar áreas específicas de la solución.

3.3.1 Diagrama de clases del diseño

Un diagrama de clases del diseño presenta las clases del sistema con sus relaciones estructurales y de herencia. Las diferencias entre los diagramas de diseño de los distintos casos de uso son mínimas pues el flujo central de acciones del sistema no varía. Debido a lo anterior se muestra una vista general del mismo con los casos de uso arquitectónicamente significativos y los diagramas del diseño específicos para cada uno de estos casos de uso.

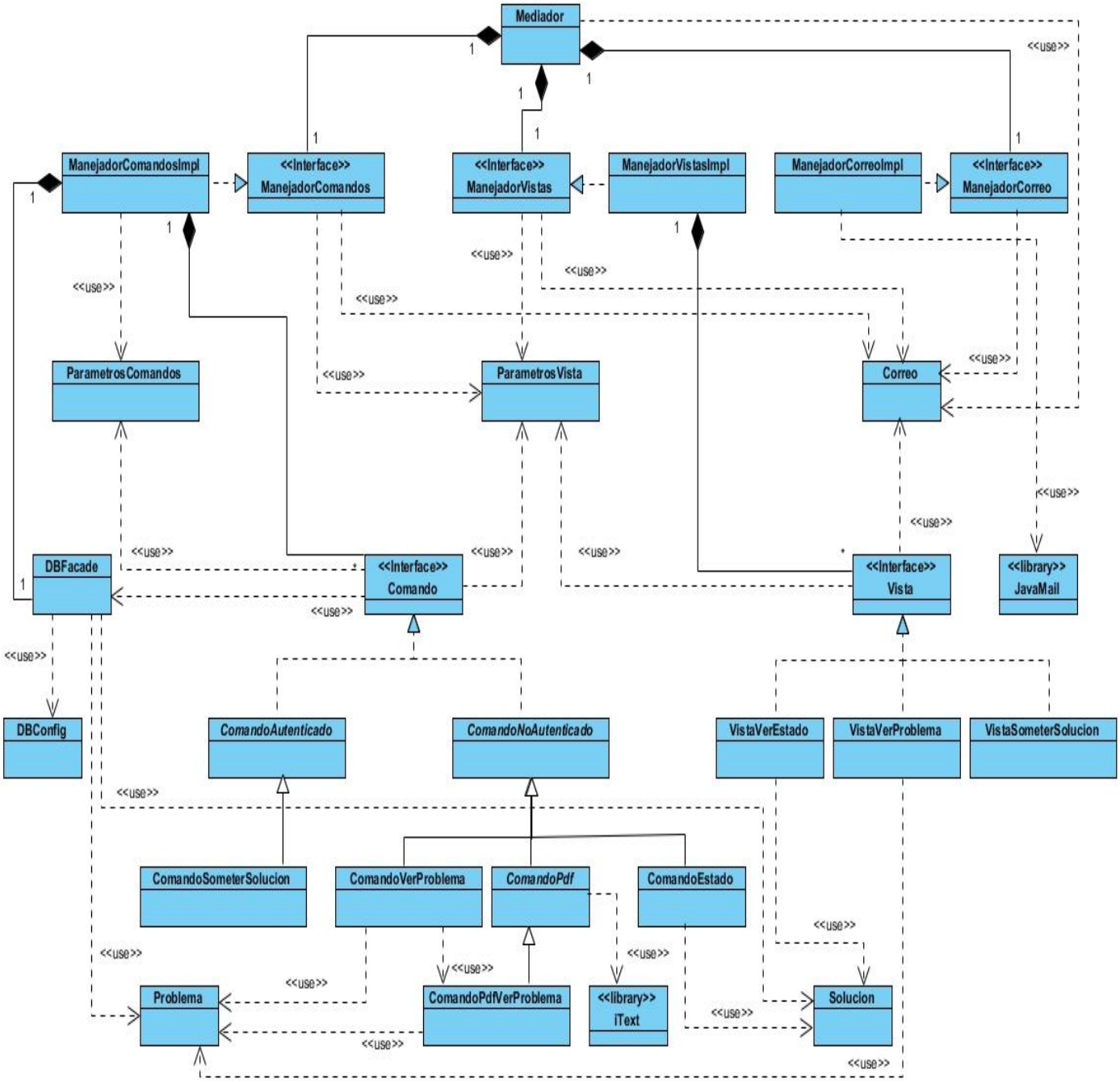


Figura 7 Diagrama de clases del diseño general

A continuación se explican las clases definidas en el diagrama de diseño general para una mejor comprensión del mismo.

Mediador

Esta clase se encarga de encapsular la forma en que interactúan los distintos manejadores (ManejadorCorreos, ManejadorComandos, ManejadorVistas) para darle respuesta a los correos con las peticiones de los usuarios. El flujo central de la aplicación está determinado por la interacción entre las interfaces mencionadas anteriormente. Dicha interacción es controlada por esta clase. De esta forma se logra que la aplicación sea extensible en cuanto a la implementación de cualquiera de sus componentes sin alterar la lógica de la misma.

El flujo es el siguiente: A través de ManejadorCorreos obtiene una lista de los correos que se encuentran en el buzón asignado a la aplicación en una estructura llamada Correo. Por cada uno de esos correos obtiene a través de la clase ManejadorComandos la información que se pide en una estructura llamada ParametrosVista. A partir de esa información se obtiene del ManejadorVistas el correo correspondiente. Finalmente es ese correo el que envía al usuario correspondiente a través de ManejadorCorreos.

ManejadorCorreos

Esta interfaz se encarga de definir el contrato de la clase que se encargará de obtener los correos del buzón asignado y de enviarlos al usuario correspondiente. La forma en que la información de los correos se maneja es a través de la estructura Correo.

Este manejador aísla el resto de la aplicación de las especificidades de implementación de la interacción del sistema con el buzón de correo. Permite potencialmente cambiar la interfaz con el usuario a otro protocolo cualquiera siempre y cuando la información pueda ser transformada en objetos Correo.

ManejadorCorreosImpl

Implementa la interfaz ManejadorCorreos para la interacción con buzones SMTP/POP3 sobre SSL.

ManejadorComandos

Esta interfaz es responsable de definir el comportamiento de la clase que se encargará a partir de un Correo de obtener la información que se pide en él. Esta información se devuelve en una estructura llamada ParametrosVista.

Esta interfaz aísla el componente de procesamiento de la lógica de la aplicación. Por esta razón puede ser extendido tanto el lenguaje que permite la interacción del usuario con el sistema como la manera de implementarlo sin necesidad de modificar los demás componentes de la aplicación.

ManejadorComandosImpl

Implementa la interfaz ManejadorComandos. Contiene un atributo de la estructura DBFacade encargada del acceso a datos. Contiene una tabla con los comandos que pueden ejecutarse.

Su función es a partir de un Correo que contiene una petición devolver lo que se pide en una estructura llamada ParametrosVista. Para ello parsea el Correo con lo que obtiene cada uno de los criterios y valores especificados en el mismo en una estructura llamada ParametrosComando. Identifica el Comando a ejecutar. A través de este valida el ParametrosComando adquirido en el parseo. Por último manda al Comando a ejecutarse brindándole el ParametrosComando y producto de esta acción resulta una estructura llamada ParametrosVista que contiene la petición hecha en el Correo analizado.

ManejadorVistas

Esta interfaz se encarga de definir el comportamiento de la clase que se encargará de: a partir de una estructura ParametrosVista (contiene información a devolver) obtener el correo correspondiente a la petición hecha y la información a devolver.

Este manejador aísla el componente de presentación del sistema, de forma que puedan ser modificadas las respuestas del sistema sin necesidad de modificar los demás componentes de la aplicación.

ManejadorVistasImpl

Implementa la interfaz ManejadorVistas. Contiene una tabla con las Vistas que pueden mostrarse. Su función es a partir de una estructura ParametrosVista que contiene información pedida por un usuario devolver un Correo con esa información organizada. Para ello identifica e instancia la Vista correspondiente al comando ejecutado previamente y la configura utilizando el ParametrosVista con lo que obtiene un Correo.

DBFacade

Constituye la clase de acceso a datos. Utiliza la clase DBConfig que contiene los parámetros de configuración. Contiene todos los métodos de acceso a datos que se utilizan en la aplicación.

La información intercambiada con la fuente persistente de datos se devuelve a la aplicación en forma de entidades del dominio, reforzando el patrón Domain Model. Esta clase abstrae el resto del sistema de la manera en la que se maneja la persistencia de datos.

DBConfig

Contiene los valores de configuración de la base de datos. Esta es una clase auxiliar visible solo para el DBFacade y constituye junto con este la totalidad del componente de acceso a datos de la aplicación. Los valores provistos por el DBConfig son particulares a la implementación y por tanto no deben ser visibles al resto del sistema.

Comando

Es una interfaz que define el comportamiento de cada uno de los comandos presentes en la aplicación. Toda clase que la implemente tiene que redefinir sus tres funcionalidades. La primera es validar a partir de una estructura ParametrosComando, la segunda es construir una estructura ParametrosVista a partir del ParametrosComando provisto y la tercera es responder si un comando requiere autenticación o no.

Implementan el Comando dos clases abstractas: ComandoAutenticado y ComandoNoAutenticado que le dan una implementación a la tercera funcionalidad. De cada uno de ellos heredan los restantes comandos en dependencia de si necesitan autenticación o no. Son estos los que le dan una implementación a las dos primeras funcionalidades en dependencia de sus características y usando las entidades que necesiten.

Los comandos que heredan de ComandoAutenticado son solamente dos ComandoVerSolucion y ComandoSometerSolucion (presente en el diagrama de arriba).

Los comandos que heredan de ComandoNoAutenticado son: ComandoListadoProblemas, ComandoVerProblema, ComandoVerFaq, ComandoVerNoticias, ComandoAyuda, ComandoEstado, ComandoRankingUsuarios y una clase abstracta llamada ComandoPdf de la que heredan 4 comandos, uno por cada uno de los comandos no autenticados que admiten generación a PDF de la información que se pide. Estos comandos son: ComandoPdfAyuda, ComandoPdfFaqs, ComandoPdfListadoProblemas, ComandoPdfVerProblema. Cuando el Comando que se está ejecutando identifica que la información que se le pide es en formato PDF entonces obtiene la información a través del ComandoPdf correspondiente.

Vista

Es una interfaz que define el comportamiento de cada uno de una de las vistas que muestra la aplicación. Toda clase que la implemente tiene que a partir de una estructura ParametrosVista (que contiene la información a mostrar) construir un Correo con la configuración de la vista.

La implementan las clases: VistaAyuda, VistaFaq, VistaNoticias,, VistaEstado, VistaRankingUsuarios, VistaSometerSolucion, VistaListadoProblemas, VistaVerProblema, VistaVerSolucion. Estas en su funcionamiento interno utilizan estructuras que modelan los conceptos del dominio como Solución y Problema en dependencia de sus características especificas.

Correo

Es un concepto del negocio. Contiene la información de un correo: asunto, cuerpo, adjunto y la dirección a la que hay que mandarlo. El correo es la unidad básica de interacción del usuario con el sistema y es también una parte importante del modelo de transporte de datos entre los componentes de la aplicación.

ParametrosComando

Contiene una tabla en la que guarda los pares criterio-valor correspondientes a las opciones del comando especificado definidos por el usuario en un correo. Determina los datos de interés para la ejecución del comando por parte del sistema. Esta clase constituye la entrada del componente de ejecución de lógica del negocio.

ParametrosVista

Contiene una tabla en la que guarda la información que se mostrará. El formato de esta información depende del comando dado por el usuario y los datos que se manejen para dicho comando. Esta entidad determina la vista que se va a utilizar y se convierte en un Correo que se usa para satisfacer la petición hecha.

Solucion, Problema, Noticia, Faq, Usuario

Son conceptos del negocio. Contienen la información referente a los mismos. Los últimos tres no están reflejados en el diagrama por no formar parte de los casos de uso arquitectónicamente significativos.

iText

Biblioteca Open Source para crear y manipular archivos PDF.

JavaMail

API para la comunicación con el servidor de correo.

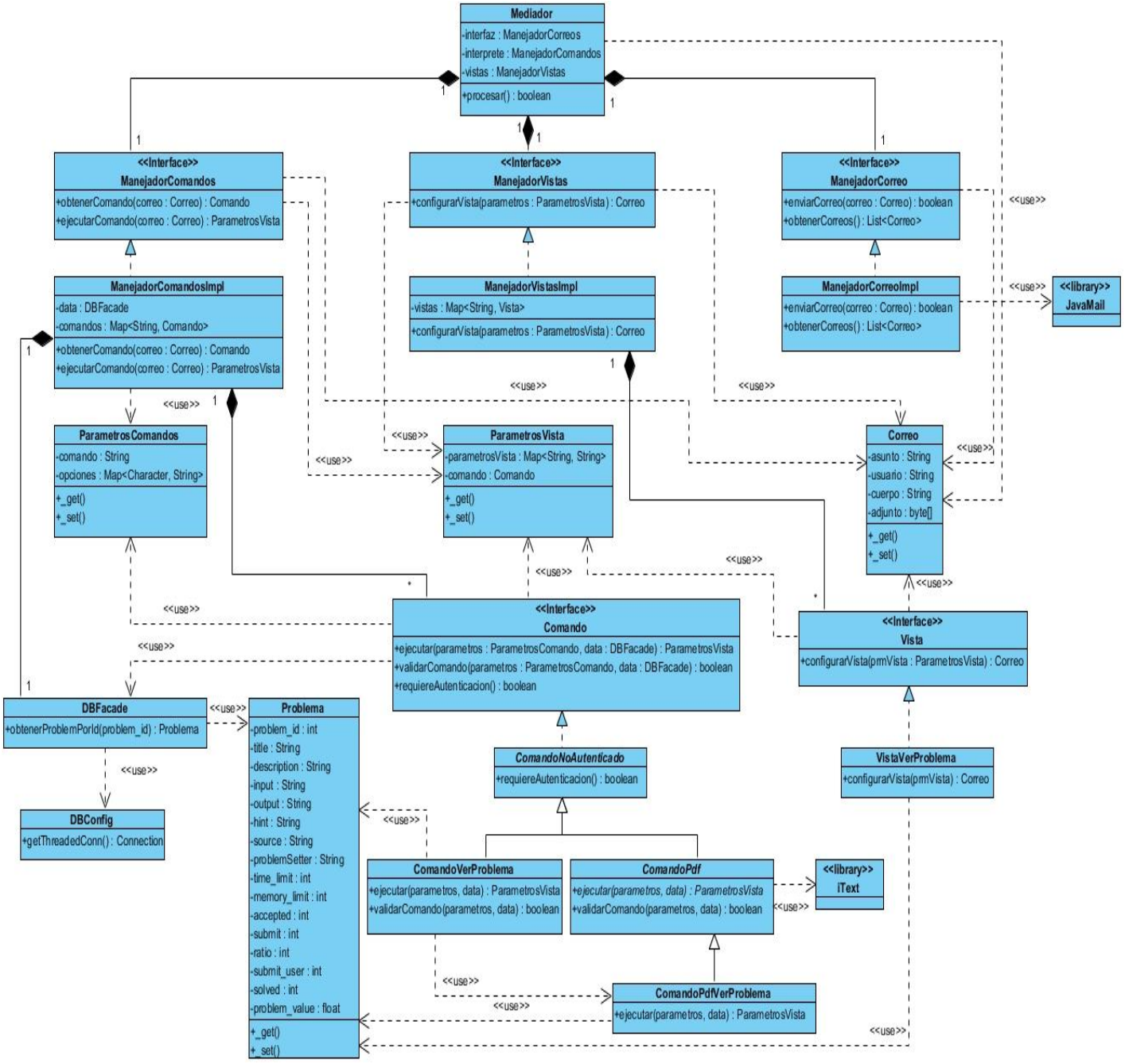


Figura 8 Diagrama de clases del diseño del CU Ver problema

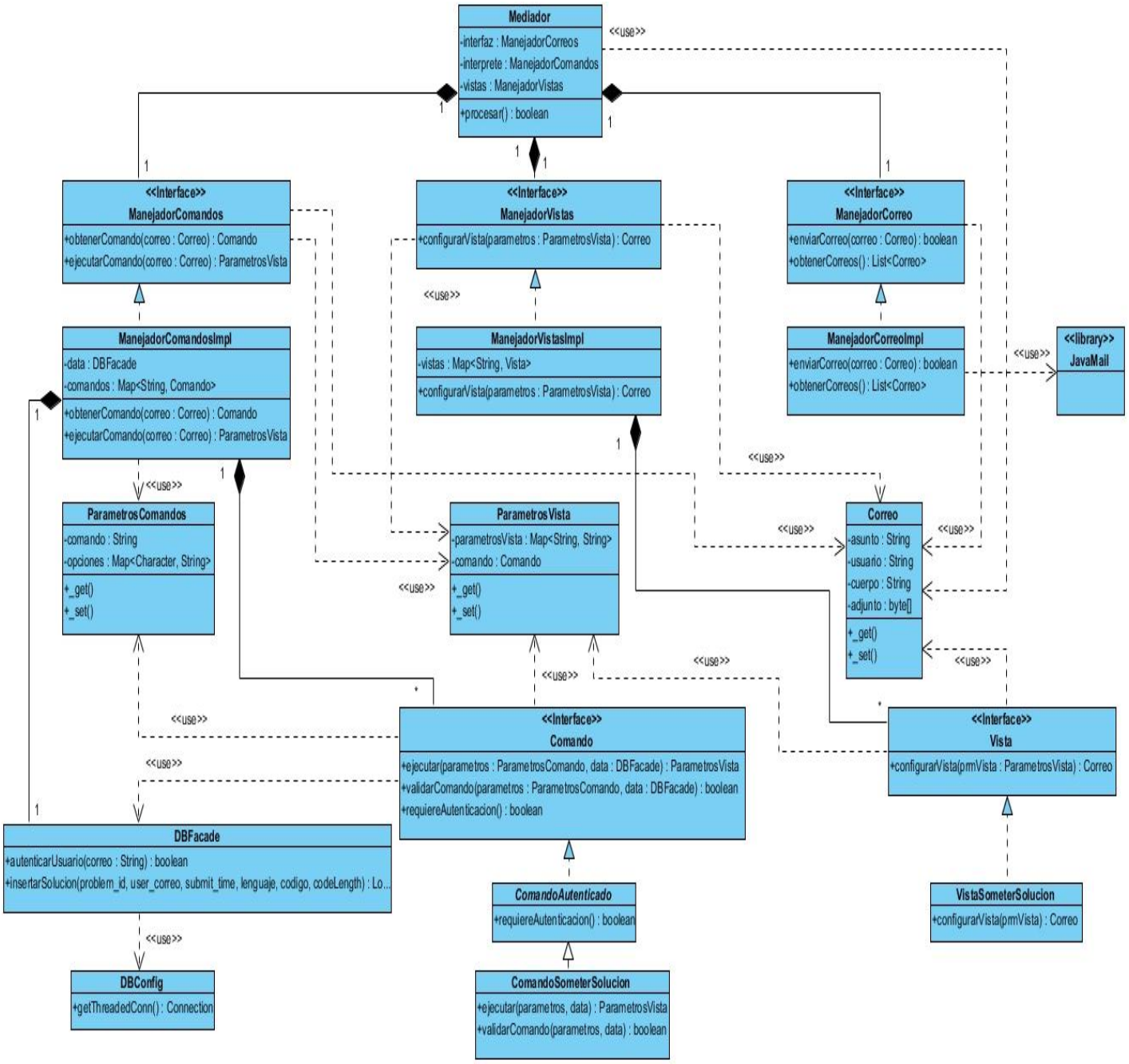


Figura 9 Diagrama de clases del diseño del CU Someter solución

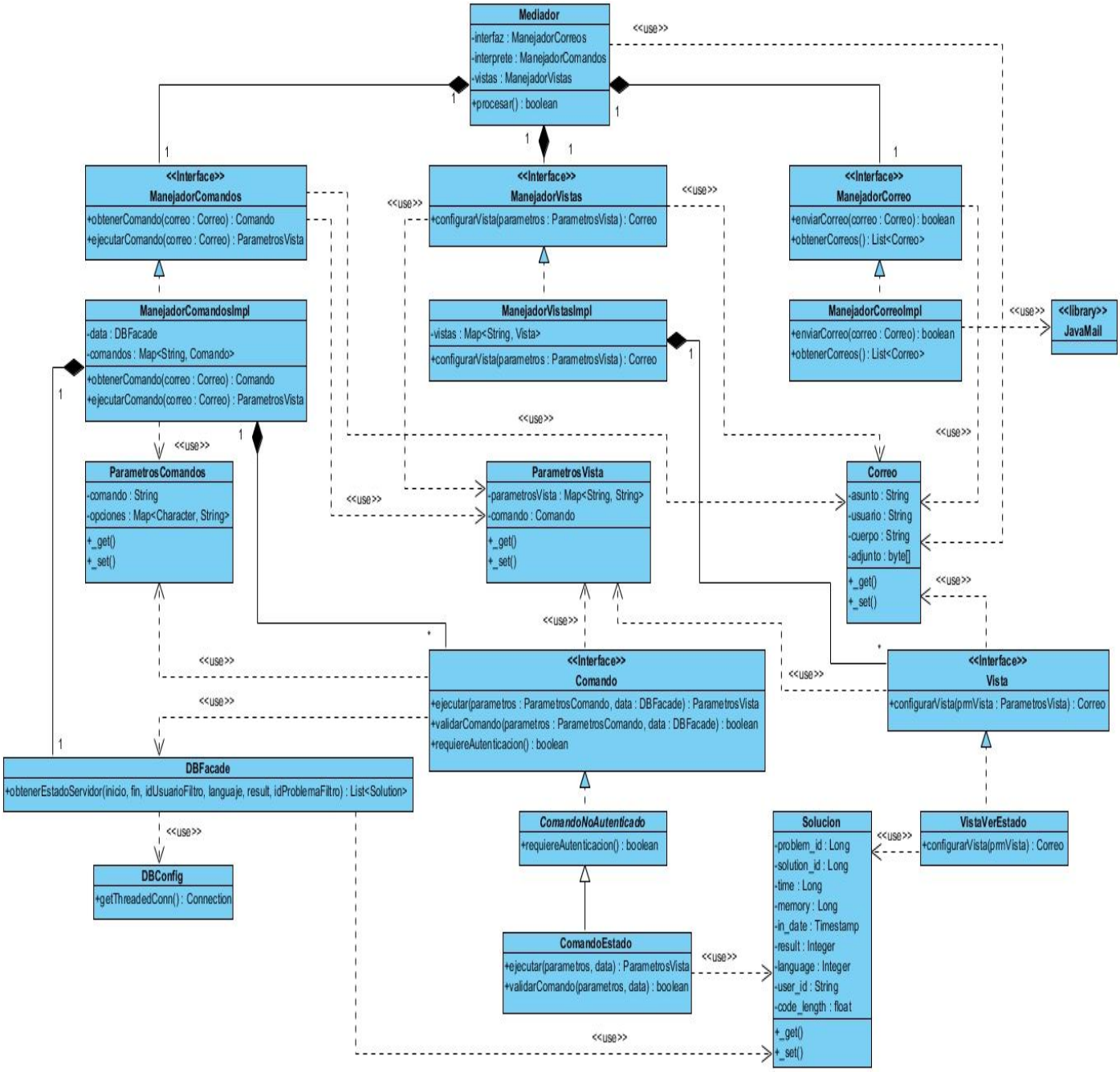


Figura 10 Diagrama de clases del diseño del CU Consultar el estado del servidor

3.4 Diagrama de clases persistentes

Las clases persistentes por lo general tienen como origen las clases clasificadas como entidades porque ellas modelan la información y el comportamiento asociado a algún fenómeno o concepto, como una persona, un objeto del mundo real o un suceso. Todas las clases identificadas en el dominio del análisis no son persistentes. La persistencia es la capacidad de un objeto de mantener su valor en el espacio y en el tiempo (González, 2004)

En la siguiente imagen se muestra el Diagrama de clases persistentes del sistema.

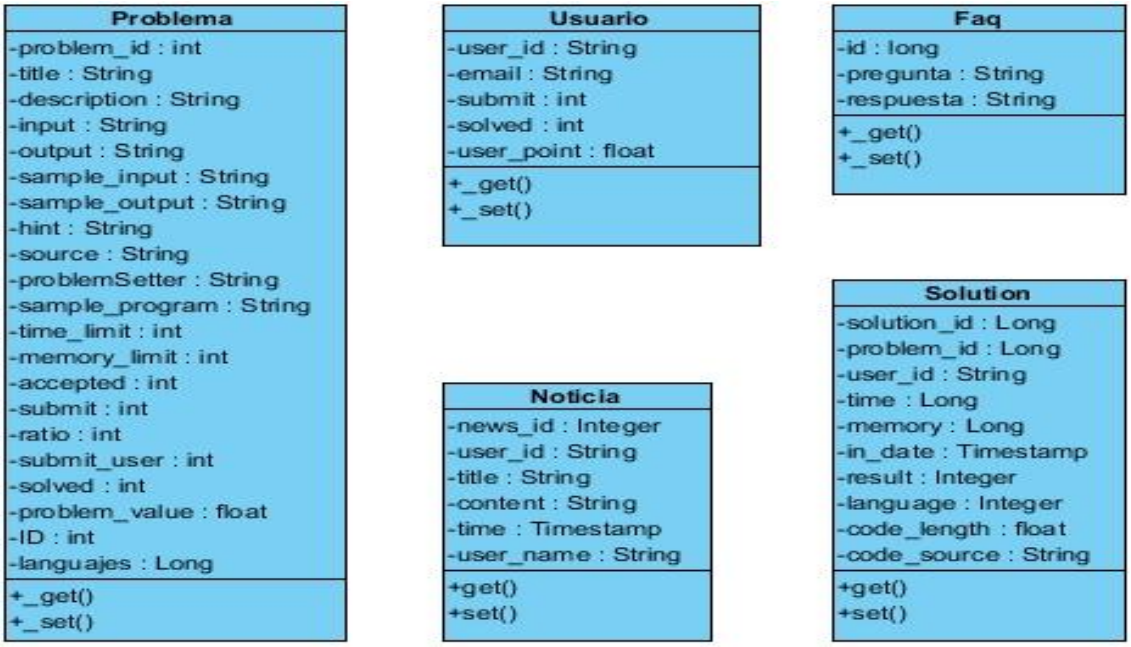


Figura 11 Diagrama de clases persistentes

3.5 Diseño de la base de datos

El Modelo Entidad Relación representa a la realidad a través de entidades que son objetos que existen y se distinguen de otros por sus características. En base de datos estas características se conocen como atributos. A su vez una entidad se puede asociar o relacionar con más entidades a través de relaciones.

A partir del diagrama de clases persistentes se representa el Modelo Entidad Relación referente a la base de datos del sistema. La base de datos que se utiliza en la aplicación es la del Jurado del Caribe, la misma es relacional pero no presenta relaciones en sus tablas. Para el funcionamiento del jurado no es

necesario utilizar todas las tablas, las de mayor importancia por ser las más pesadas en cuanto al número de tuplas son las que se muestran en la siguiente imagen:

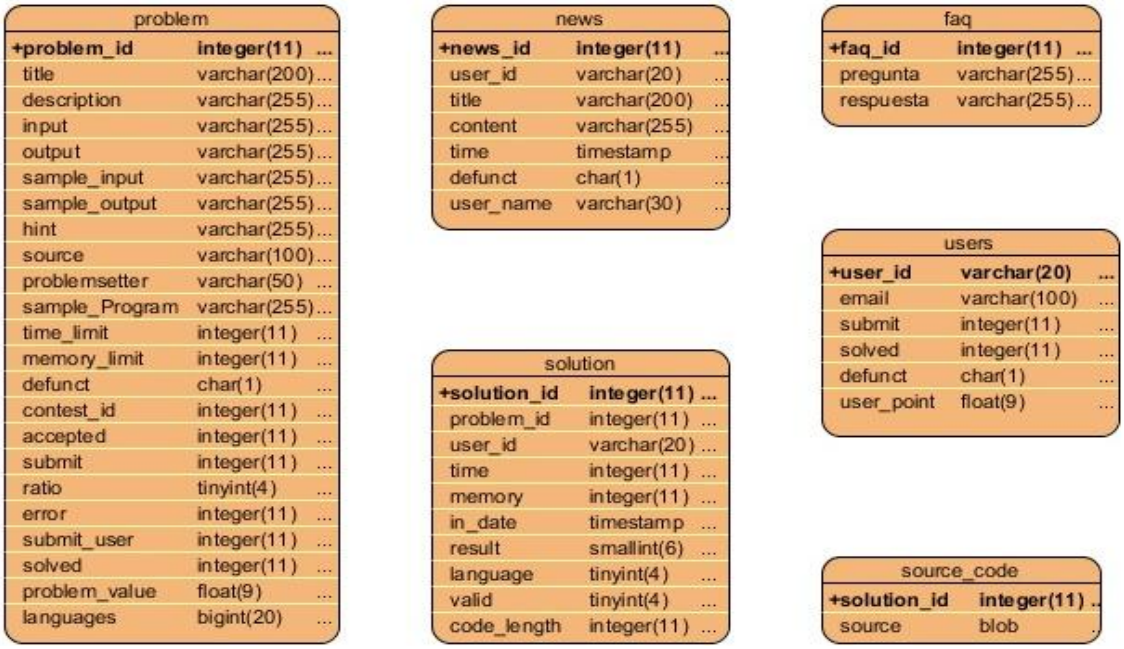


Figura 12 Modelo de datos

3.6 Descripción del funcionamiento del sistema

La aplicación se ejecuta continuamente en el sistema de la máquina en que se despliegue. Su funcionamiento se basa en descargar de manera constante los correos que se encuentren en la bandeja de entrada del buzón asignado. Una vez realizada la descarga, el sistema da respuesta a la petición y elimina el correo del servidor. En caso de que no haya correos para responder, la aplicación espera un tiempo definido por los administradores antes de ejecutar una nueva descarga, de esta forma no se sobrecarga el buzón de forma innecesaria.

El nombre escogido para la aplicación es MailJudge, vocablo que está compuesto por dos palabras del inglés: mail o correo en español y judge o jurado. La comunicación de los usuarios con el MailJudge es a través del micro lenguaje definido para este fin. Para que un usuario pueda interactuar de forma satisfactoria, debe enviar un correo a la dirección establecida para la comunicación. En el asunto debe especificar lo que desea ver, respetando el formato que define el micro lenguaje. Ejemplo: para ver el

problema 1362, en el asunto del correo que debe enviar a la dirección coj@uci.cu , debe escribir "**mj:problem -i 1362**" (**Figura 13**).

Debido a que el buzón se utiliza para otra labores se hace necesario que todos los correos que se envíen a la aplicación contengan en el asunto la cadena "**mj:**". Para facilitar la comunicación con los usuarios, todo correo que sea enviado con un formato incorrecto, es respondido por el sistema con una explicación detallada de cómo se debe utilizar el micro lenguaje.

El proceso de autenticación en el sistema se realiza de la siguiente forma: si existe un usuario en la base de datos que posea la dirección de correo de la cual se está enviando la petición al sistema, entonces ese usuario está autenticado. Dentro de las funcionalidades que se implementan en la aplicación no se incluyen las de administración ni se permite registrar usuarios. Por esta razón, para que un usuario que no está registrado en la base de datos se autentique, debe entrar al jurado Xtreme y hacerlo o escribir a la dirección de contacto que se envía en cada correo de respuesta del sistema.

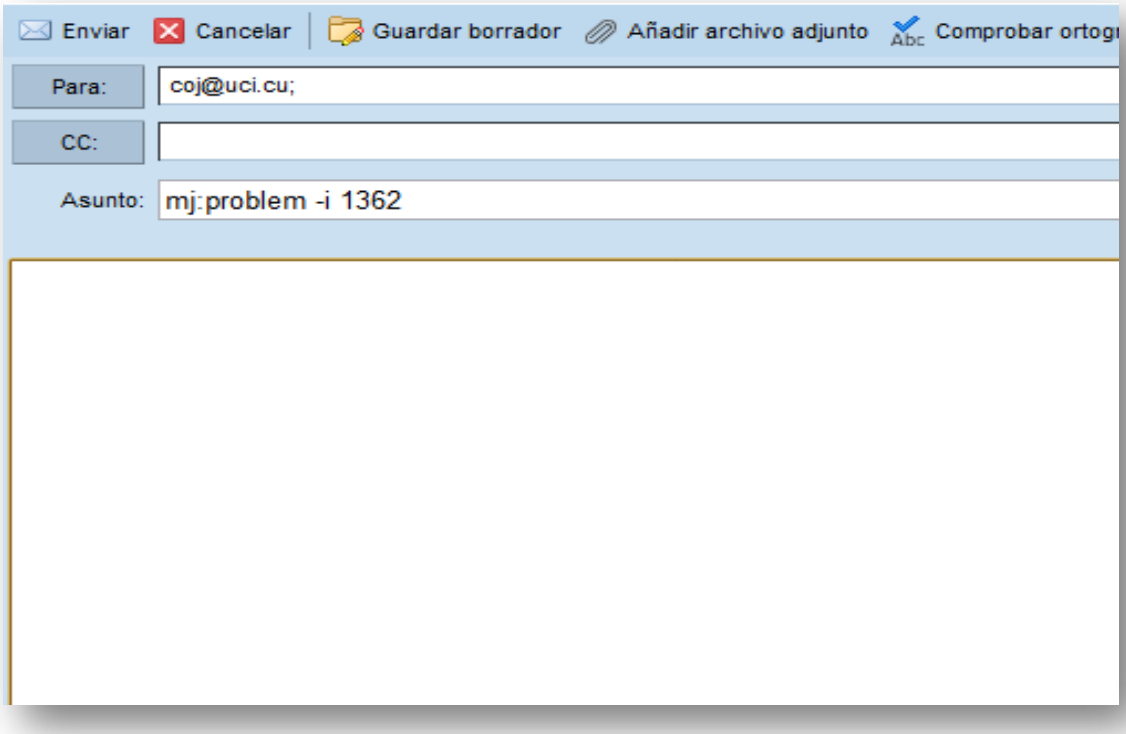


Figura 13 Ejemplo de cómo ver el problema 1362

3.7 Factibilidad de la propuesta de solución

En el trabajo de diploma: Implementación y prueba de un Jurado Online sobre correo electrónico (Labrada, 2011) se desarrolla la implementación y prueba de la propuesta de solución, las pruebas a dicha aplicación demostraron el correcto funcionamiento de la misma. Es importante destacar que la aplicación se desplegó en la UCI a fines de 2011 y estuvo funcionando correctamente durante aproximadamente 6 meses. Con la misma se mantuvieron interactuando usuarios de Cuba y otros países que pusieron a prueba cada una de las funcionalidades inicialmente diseñadas y posteriormente implementadas. Lo antes mencionado demuestra que el análisis y diseño del módulo de comunicación por correo electrónico que constituyó la base de la implementación del sistema, es factible.

El **Anexo 20** muestra un aval otorgado por el ingeniero Dovier Antonio Ripoll Méndez (Director general del ACM-ICPC en el Caribe y líder del equipo de desarrollo del COJ) que da fe de los resultados obtenidos y el beneficio que trajo esta aplicación a la comunidad ACM en Cuba, donde tuvo una excelente acogida. Al convertirse en un éxito el despliegue del módulo de comunicación por correo electrónico se demuestra que se puede acceder a la información dispuesta en un jurado de programación, con el beneficio de las facilidades que brindan protocolos más desconectados y con acceso a documentación en formato PDF para el uso offline.

3.8 Conclusiones

En el presente capítulo se diseñó el modelo del análisis, dentro del cual se modelaron los diferentes diagramas de clases del análisis por cada caso de uso. Quedaron plasmados los diagramas de clases del diseño general y específicos para los tres casos de uso significativos que tiene el sistema. Se elaboró el diagrama de clases persistentes y se especificó la base de datos que se va a utilizar en la aplicación. Se explicó el modo en que se demuestra la factibilidad del análisis y diseño del módulo de comunicación por correo electrónico. Se llegó a la conclusión que es factible el sistema para su utilización en aquellas universidades con problemas de conectividad en calidad de complemento a una instancia tradicional.

Conclusiones generales

El presente trabajo se centró en el análisis y diseño de un módulo de comunicación por correo electrónico para el Juez en Línea del Caribe que genere documentos a un formato estándar de visualización. La finalidad del mismo es dar una solución al problema de accesibilidad a este tipo de software que se presenta en la generalidad de las instituciones cubanas (universidades, joven clubs, centros de trabajo e institutos preuniversitarios). Durante el desarrollo del trabajo se le dio cumplimiento al objetivo propuesto y se arribó a las siguientes conclusiones:

- Todos los jurados online pueden reducirse a un conjunto de funcionalidades básicas. Estas sirven de modelo para crear implementaciones de este tipo de aplicaciones fuera de su paradigma web tradicional.
- Es posible representar todas las funcionalidades básicas de un jurado online sobre los protocolos de correo electrónico SMTP e IMAP4 posibilitando a los usuarios acceder a la misma información disponible sobre HTTP, pero beneficiándose de las características de protocolos más desconectados.
- Existe un conjunto mínimo de información necesaria y suficiente que los usuarios pueden usar de manera desconectada sin necesitar acceder a la aplicación online durante intervalos prolongados de tiempo, dicha información es la que se muestra en formato PDF.

Recomendaciones

Con el objetivo de hacer más eficiente y funcional el sistema se recomienda:

- Fomentar el uso del sistema en instituciones (universidades, joven clubs, centros de trabajo e institutos preuniversitarios) del país que presenten problemas en la disponibilidad de la red.
- Concebir y desarrollar una segunda versión del software con las funcionalidades de competencias y administración.
- Concebir y desarrollar un sistema a utilizar por los usuarios en las máquinas clientes que envíe correos con las peticiones que se seleccionen y muestre las respuestas. De esta forma el usuario no está obligado a memorizar el micro lenguaje establecido para la comunicación, solo debe seleccionar la información que desee.

Referencias bibliográficas

[En línea] [Citado el: 8 de diciembre de 2010.] http://www.visual-paradigm.com/.

Cátedra de programación Avanzada. [En línea] [Citado el: 14 de noviembre de 2010.] http://cpav.uci.cu.

Como abrir archivos XPS. [En línea] [Citado el: 15 de junio de 2012.] http://software.comohacerpara.com/n1159/como-abrir-archivos-xps.html.

Crispin, M. 2003. The Internet Engineering Task Force (IETF). [En línea] marzo de 2003. [Citado el: 15 de diciembre de 2010.] http://www.ietf.org/rfc/rfc3501.txt.

Derin, Onur. 2005. docstoc: Document for Small Bussiness & Professionals. [En línea] 01 de marzo de 2005. [Citado el: 14 de enero de 2011.] http://www.docstoc.com/docs/23951918/A-Tutorial-on-Reporting-in-JAVA-using-JasperReports-iReport.

ErichGamma, RichardHelm, RalphJohnson, JohnVlissides. 1994. *Design Patterns: Elements of Reusable Object-Oriented Software.* s.l. : AddisonWesley Professional, 1994. ISBN 978-0201633610 , ISBN 0-201-63361-2.

Espino, José Miguel Santos. Visualización de archivos PostScript. [En línea] [Citado el: 15 de junio de 2012.] http://labsopa.dis.ulpgc.es/doc/postscript.htm.

Gallego, Juan Pablo Gómez. Scribd Home. [En línea] [Citado el: 16 de diciembre de 2010.] http://www.scribd.com/doc/297224/RUP.

González Blanco, Rubén y Pérez Tobalina, Sergio. LESE-2 Introducción a Rational Rose. Barcelona : s.n.

González, Anaisa Hernández. 2004. *SISTEMA DE INFORMACIÓN CIENTÍFICA REDALYC.* [En línea] 2004. [Citado el: 15 de 03 de 2011.] http://redalyc.uaemex.mx/pdf/615/61570402.pdf. ISSN 1405-5546.

Gusgsm, 2003. Qué es el lenguaje PostScript. [En línea] [Citado el: 15 de junio de 2012.] http://gusgsm.com/que_es_el_lenguaje_postscript.

Hall., "Applying UML and Patterns" Craig Larman. Ed Prentice.

JasperForge. [En línea] [Citado el: 17 de enero de 2011.] http://jasperforge.org/projects/jasperreports.

Klensin, J. The Internet Engineering Task Force (IETF). [En línea] [Citado el: 11 de enero de 2011.] http://www.ietf.org/rfc/rfc2821.txt.

Lamport, Leslie. *Latex: A Document Preparation System(2nd Edition).* ISBN 0-201-52983-1.

Leurs, Laurens. Imagen digital 4.0. [En línea] [Citado el: 15 de junio de 2012.] http://gusgsm.com/que_es_el_formato_pdf.

Lowagie, Bruno. 2007. *iText in Action.* New York : Manning Publications Co., 2007.

Mani, John, y otros. Oracle. [En línea] [Citado el: 15 de enero de 2011.] http://java.sun.com/products/javamail/JavaMail-1.4.pdf.

Myers, Glenford J. 2004. *The art of sotware Testing.* Segunda Edición. s.l. : Wiley, 2004. pág. 234.

Myers, J., Mellon, Carnegie y Rose, M. The Internet Engineering Task Force (IETF). [En línea] [Citado el: 20 de 12 de 2010.] http://www.ietf.org/rfc/rfc1939.txt.

ORACLE. [En línea] [Citado el: 13 de enero de 2011.] http://www.oracle.com/technetwork/java/faq-135477.html#1.

Palacio, Juan. 2006. navegapolis.net. [En línea] 2006. [Citado el: 15 de 12 de 2010.] http://www.navegapolis.net/files/s/NST-010_01.pdf.

Palos, Juan Antonio. *API JavaMail.*

PKU Online Judge. [En línea] [Citado el: 14 de noviembre de 2010.] http://poj.org.

2009. Quality Assurance & Software Testing. [En línea] 2009. [Citado el: 29 de 04 de 2011.] http://www.calidadysoftware.com/testing/pruebas_funcionales.php.

Ramirez, Jorge Amado Soria y Castillo, Enrique José Altuna. 2008. *Análisis, diseño e implementación de un Jurado Online".* Ciudad de la Habana : s.n., 2008.

Revilla, Miguel A., Manzoor, Shahriar y Liu, Rujia. *www.edujudge.eu.* [En línea] [Citado el: 15 de 12 de 2010.] http://www.edujudge.eu/pdfs/revilla_ioi_final.pdf.

Scribd Home. [En línea] [Citado el: 16 de diciembre de 2010.] http://www.scribd.com/doc/7844685/CONCEPTOS-DE-RUP .

Shuja, Ahmad K. y Krebs., Jochen. 2008. *IBM Rational Unified Process Reference and Certification Guide: Solution Designer (RUP).* s.l. : IBM Press, 2008. ISBN-10: 0-13-156292-4;ISBN-13: 978-0-13-156292-9.

Sierra, María. *Trabajando con Visual Paradigm for UML.* Cantabria : s.n.

Solís, Manuel Calero. www.apolosoftware.com. [En línea] [Citado el: 16 de 12 de 2010.] http://www.willydev.net/descargas/prev/explicaxp.pdf.

Sphere Online Judge. [En línea] [Citado el: 14 de noviembre de 2010.] http://www.spoj.pl.

Tuesta, Martín. 2005. El Prisma. [En línea] 2005. [Citado el: 20 de enero de 2010.] http://www.elprisma.com/apuntes/curso.asp?id=13324.

UVa Online Judge. [En línea] [Citado el: 20 de noviembre de 2010.] http://acm.UVa.es.

Visual Paradigm. [En línea] [Citado el: 14 de diciembre de 2010.] http://www.visual-paradigm.com/.

Xtreme Online Judge. [En línea] [Citado el: 14 de noviembre de 2010.] http://onlinejudgef8.uci.cu:5800/JudgeOnline.

Glosario de términos

AC: Código de respuesta aceptada. Estado de una sumisión exitosa.

ACM: Acrónimo de Association for Computing Machinery, organización fundada en 1947 como la primera sociedad científica y educativa acerca de la Computación. Publica varias revistas y periódicos científicos relacionados con la computación; patrocina conferencias y otros eventos relacionados con las ciencias de la computación como por ejemplo el ACM International Collegiate Programming Contest (ICPC).

CMS: Un sistema de manejo de contenido o CMS es un software usado para crear, editar, administrar y publicar contenido de forma consistente. Los CMSs son utilizados frecuentemente para almacenar, controlar, versionar y publicar documentación específica como artículos de noticias, manuales, guías de venta y propaganda de marketing. El contenido que puede manejarse incluye ficheros, imágenes, audio, documentos electrónicos y contenidos Web.

CPAV: Cátedra de programación avanzada, nombre con el que surge un proyecto en la UCI en el curso 2006-2007, con el objetivo de agrupar a los estudiantes con interés o conocimiento para formar un grupo que promueva el estudio de la algoritmia así como su aplicación en la producción.

Envío: Ver sumisión.

Estado o estado de un problema: El estado de un problema determinado se compone del conjunto de soluciones que se le ha dado. El estado aporta mucha información al usuario experto sobre el problema, información que se puede inferir a partir de los datos públicos de consumo de memoria, longitud de código y tiempo.

Estado actual o del servidor: El estado actual de la aplicación es el subconjunto más actual de sumisiones que se han hecho, sin tomar en cuenta problema o usuario. El estado actual es una herramienta útil para evaluar la actividad actual de la aplicación, los ejercicios que despiertan más interés y los lenguajes más utilizados, entre otros datos de interés. El tamaño del subconjunto que se considera estado actual varía de una aplicación a otra.

Evaluador automático: Aplicación que alivia el trabajo de calificación de soluciones programáticas, moviendo su responsabilidad del humano a la máquina y tratando de lograr mejoras en cuanto al chequeo de correctitud, robustez y eficiencia.

Framework: es una estructura conceptual y tecnológica de soporte, normalmente con artefactos o módulos de software concretos, con base en la cual otro proyecto de software puede ser organizado y desarrollado.

ICPC: El ACM International Collegiate Programming Contest (abreviado como ACM-ICPC o solo ICPC) es una competición anual de programación en varios niveles que se celebra entre numerosas universidades del mundo. Este concurso es patrocinado por IBM, y tiene su sede en la Universidad de Baylor. El ICPC define regiones autónomas en seis continentes y opera bajo los auspicios de la ACM.

Jurado en línea: Evaluador automático desplegado en entornos generalmente académicos presentado en la Web (o en cualquier otro soporte igualmente masivo). Provee problemas y sirve para evaluar automáticamente las soluciones algorítmicas que los usuarios desarrollan en uno de entre un número determinado de lenguajes de programación. Un jurado online usualmente ofrece además otros servicios de consulta y suele servir como herramienta para la realización de competencias de programación.

Pdf (Portable Document Format**):** Documento en formato PDF que se le envía al usuario conteniendo cierta información de interés, por ejemplo el listado de problemas, un problema determinado o la ayuda del sistema.

Problema: Enunciado preciso de una situación que presenta un problema a solucionar mediante la aplicación de técnicas algorítmicas, lógica, geometría o cualquier otra rama matemática. Los problemas generalmente se dividen en secciones muy específicas, siendo la división más común: descripción, especificación de entrada, especificación de salida, ejemplo de entrada, ejemplo de salida y fuente del problema. Tradicionalmente el enunciado del problema debe proveer cualquier conocimiento que caiga fuera de las áreas algorítmicas o matemáticas y que sea necesario para resolverlo. Se sigue el razonamiento de que lo que se mide no es el nivel de conocimiento o cultura general del usuario, sino su conocimiento y habilidad en la rama de programación de competencias.

Programación de competencia: Se refiere la programación centrada en el análisis y la implementación de programas con el objetivo de medir conocimientos teóricos en aspectos particulares de algoritmia, matemática o lógica. Esta rama de programación presenta características diferentes de la programación empresarial o educativa, en que no presta atención a los principios de orientación a objetos, chequeo de errores, patrones de diseño, etc.; concentrándose exclusivamente en aspectos de eficiencia y correctitud de los programas. Una aplicación para competencias suele ser de código ofuscado, sin diseño claro y con

poco mantenimiento, con poca o ninguna extensibilidad ni modularidad debido a que el interés principal está puesto en la eficiencia en el uso de memoria y tiempo y en la implementación correcta del algoritmo correspondiente. Opcionalmente algunas personas tienden a desarrollar sus propios estilos de programación, buscando elegancia en sus soluciones lo cual puede contrarrestar las propiedades negativas anteriormente expuestas.

Sumisión: Intento de solución de un problema. Una sumisión generalmente se identifica con un número secuencial, y corresponde a un intento realizado por un usuario para resolver un problema determinado. Constituye la unidad básica de interacción del usuario con la aplicación, y puede tener asociados distintos estados en dependencia de si el intento es exitoso o no, y si no, del error que provocó el fracaso.

Volumen de problemas: Conjunto de problemas que se encuentran en un jurado online o en un evento competitivo. Los volúmenes pueden ser creados siguiendo distintos criterios, como complejidad, asunto, tamaño o simplemente ordenados cronológicamente, dependiendo el criterio del jurado o evento donde se encuentre.

WA: Código de respuesta incorrecta. Estado de una sumisión desacertada.

Zimbra: Cliente / servidor de correo y calendario open source disponible en los repositorios de Ubuntu, que proporciona acceso offline y online a Yahoo, Gmail, AOL y cualquier otra cuenta IMAP y POP. Ofrece diversas herramientas como email, contactos, calendario, documentos y agenda de tareas.

Anexo 1

Para recibir el estado de las submisiones en el servidor se debe especificar en el asunto del correo el siguiente comando:

mj:status <criterio><valor>

<criterio>	<valor>	Cuando no se especifica el criterio el sistema por defecto devuelve
-t (tipo de estado)	**server**	El estado del servidor
-u (usuario)	Un usuario registrado. Ej. (**arosa**)	El estado de las sumisiones de todos los usuarios.
-r (resultado de la submisin)	**ac**(aceptado) **wa**(respuesta incorrecta) **tle** (tiempo límite excedido) **mle**(memoria límite excedida) **ce**(error de compilación) **pe**(error de presentación) **rte**(error en tiempo de ejecución) **ole**(límite de salida excedido)	El estado de las sumisiones de todos los tipos de respuesta.
-n (rango)	rango de números con el formato #-# Ej.(**1-500**)	El estado de las últimas 500 sumisiones.
-l (lenguaje)	lenguaje de programación válido Ej. (**java**)	El estado de las sumisiones en todos los lenguajes

(Nota: Si el usuario desea solicitar más de 1 criterio del estado del servidor puede hacerlo separando por espacios el par: <criterio><valor>. Por ejemplo: **mj:status –n 5-20 –u arosa –l g++)**

Anexo 2

Para recibir el listado de problemas se debe especificar en el asunto del correo el siguiente comando:

mj:volume –v <# de volumen> -c <criterio de ordenamiento>

<criterio de ordenamiento>	<# de volumen>	Cuando no se especifica criterio el sistema por defecto devuelve
points (ordena por los puntos de los problemas) **ac** (ordena por la cantidad de soluciones aceptadas a los problemas) **submissions** (ordena por el total de submisiones enviadas a los problemas.) **%ac** (ordena por el por ciento de soluciones aceptadas a los problemas)	Número que representa a cada volumen de problemas. El número de volumen debe ser un entero entre 1 y 6 Ej. (**2**)	Cuando no se especifica **<criterio de ordenamiento>** el sistema devuelve el listado de problemas del volumen 1 organizados por el identificador. Cuando no se especifica **<# de volumen>** el sistema devuelve los problemas del volumen 1.

(Por ejemplo: **mj:volume -v 2 -c points)**

Anexo 3

Para recibir el ranking de usuarios se debe especificar en el asunto del correo el siguiente comando:

mj:ranking -n <rango> -c <criterio>

<criterio de ordenamiento>	<rango>	Cuando no se especifica criterio el sistema por defecto devuelve

points (ordena por los puntos de los usuarios). **ac** (ordena por la cantidad de soluciones aceptadas de los usuarios.) **submissions** (ordena por el total de submisiones enviadas por los usuarios.) **%ac (o**rdena por el por ciento de soluciones aceptadas de los usuarios)	Es el rango de usuarios que se desea ver del ranking. Tiene el formato #-# y no debe exceder los 500. Ejemplo (**1-500**).	Cuando no se especifica **<criterio de ordenamiento>** el sistema devuelve los usuarios ordenados por points. Cuando no se especifica **<rango>**el sistema devuelve los 500 primeros usuarios ordenados según el criterio especificado.

(Por ejemplo: mj:ranking -n 50-100 -c ac)

Anexo 4

Para recibir el listado de problemas en formato PDF se debe especificar en el asunto del correo el siguiente comando:

mj:volume –v <# de volumen> -f pdf.

<# de volumen>	Especificaciones
Número que representa a cada volumen de problemas. El número de volumen debe ser un entero entre 1 y 6 Ej. (**2**)	Cuando no se especifica el volumen, sistema devuelve los problemas del volumen 1. La lista siempre se ordena de forma ascendente por el ID de los problemas.

Anexo 5

Para someter una solución a un problema se debe especificar en el asunto del correo el siguiente comando:

mj:submit -l <lenguaje> -i <# de problema>

<lenguaje>	<# de problema>
El lenguaje debe ser: g++, gcc, pascal, java, c#, python, texto, bf. Este parámetro es requerido. Ej.(**java**)	Es un identificador del problema y tiene 4 dígitos. Debe ser un entero del 1000 al 1566. Este parámetro es requerido. Ej.(**1236**)

Anexo 6

Para someter una solución a un problema se debe especificar en el cuerpo del correo el siguiente comando:

<delimitador de inicio><código de la solución><delimitador de fin>

<delimitador de inicio >	<código de la solución >	<delimitador de fin>
Es la cadena: **<--CODE START.** Este parámetro es requerido.	Debe ser el código que resuelve el problema en el lenguaje especificado. Este parámetro es requerido.	Es la cadena: **/CODE END-->**. Este parámetro es requerido.

Anexo 7

Para recibir el estado de las submisiones de un problema se debe especificar en el asunto del correo el siguiente comando:

mj:status –t problem –i <# de problema > -c <criterio de ordenamiento>

<criterio de ordenamiento>	<# de problema>	Especificaciones
memory (memoria utilizada.) **code** (longitud del código de la submision) **time** (tiempo utilizado por la solución para ejecutarse)	Es un identificador de 4 dígitos que distingue a cada da problema. El número de problema debe ser un entero del 1000 al 1566. Este parámetro es requerido Ej. (**1560**)	En caso de que el usuario no especifique **<criterio de ordenamiento>** el sistema ordena el estado del problema especificado por el criterio time. El orden siempre es ascendente.

(Por ejemplo: **mj:status -t problem -i 1000 -c memory)**

Anexo 8

Para recibir la descripción de un problema se debe especificar en el asunto del correo el siguiente comando:

mj:problem –i <id del problema>

<id del problema >	Especificaciones
Identificador del problema y tiene 4 dígitos. Debe ser un entero entre 1000 y 1566. Ej.(**1244**)	Este parámetro es obligatorio

Anexo 9

Para recibir una solución se debe especificar en el asunto del correo:

mj:solution -s <id de la solución>

<id solución>	Especificaciones
Identificador de la solución. Un número que se le asigna a cada solución atendiendo al orden en que son enviadas al jurado. Ej.(**12569**)	Es obligatorio para el usuario especificar el id de la solución que desea ver. Se debe ser el propietario de la solución para poder consultarla.

(Por ejemplo: **mj:solution -s 12569)**

Anexo 10

Para recibir la ayuda en formato PDF se debe especificar en el asunto del correo el siguiente comando:

mj:help -f pdf

Anexo 11

Para recibir las preguntas frecuentes en formato PDF se debe especificar en el asunto del correo:

mj:faqs -f pdf

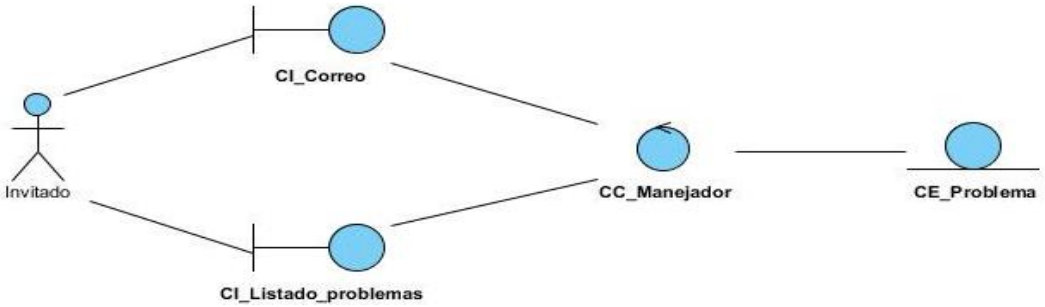
Anexo 12

Para recibir la descripción de un problema en formato PDF se debe especificar en el asunto del correo:

mj:problem -i <id del problema> -f pdf

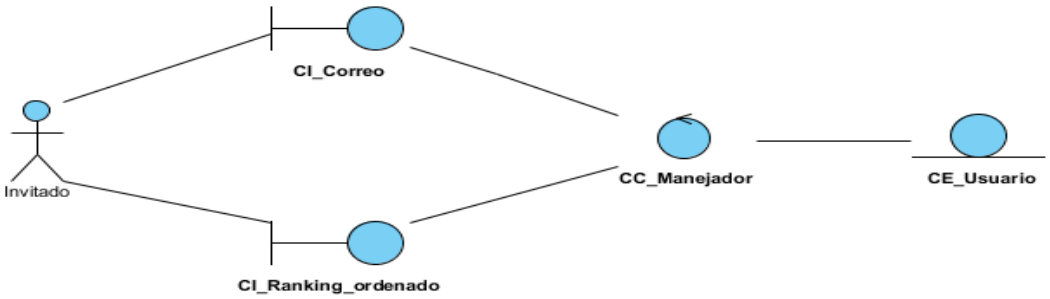
Anexo 13

Diagrama de clases del análisis CU Consultar el listado de problemas



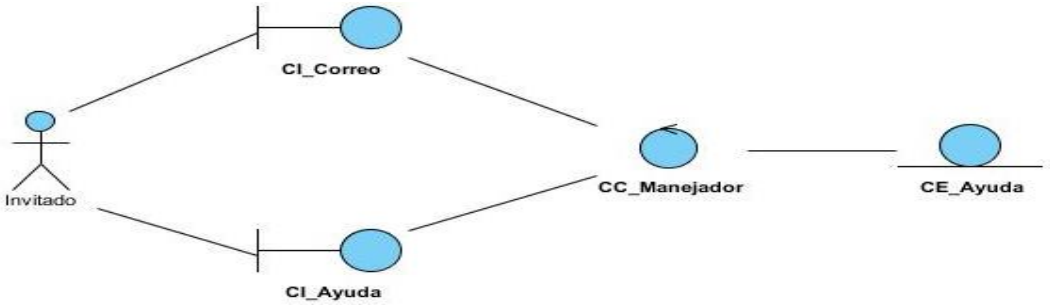
Anexo 14

Diagrama de clases del análisis CU Consultar ranking ordenado



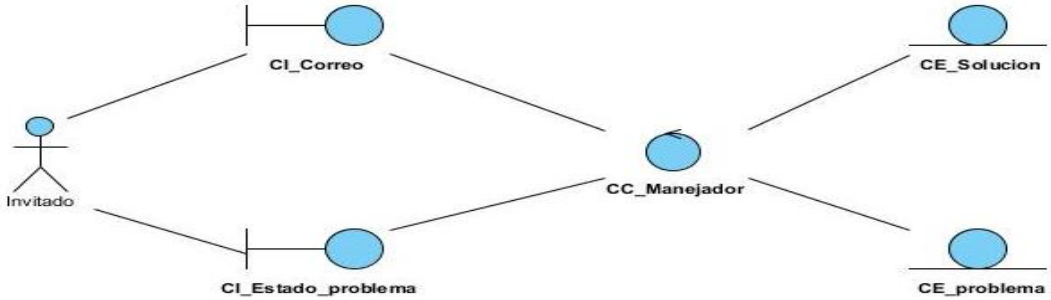
Anexo 15

Diagrama de clases del análisis CU Ver ayuda



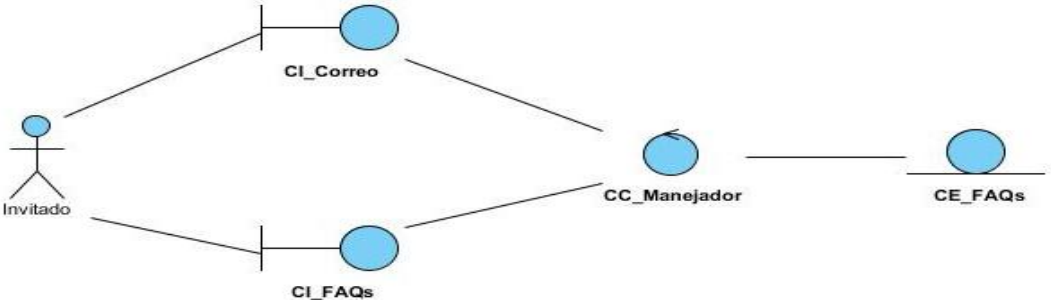
Anexo 16

Diagrama de clases del análisis CU Consultar estado de un problema



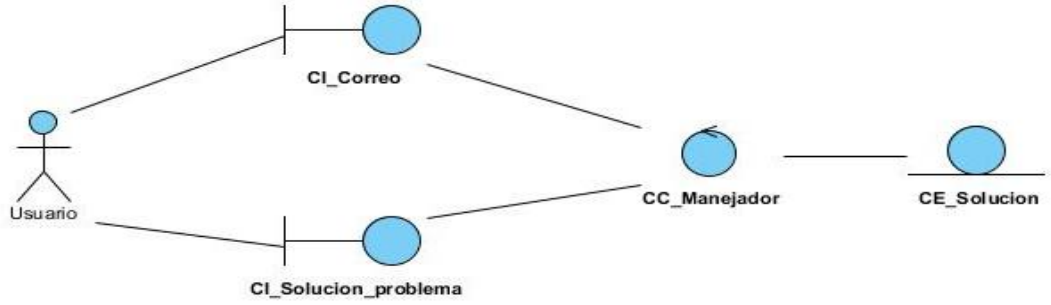
Anexo 17

Diagrama de clases del análisis CU Ver FAQs



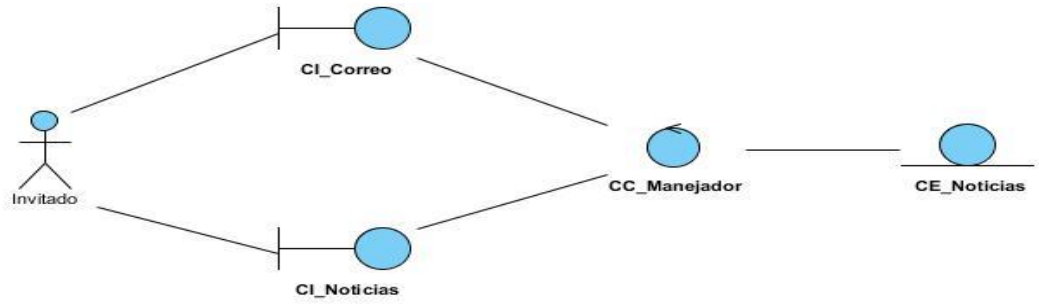
Anexo 18

Diagrama de clases del análisis CU Ver solución de problema



Anexo 19

Diagrama de clases del análisis CU Ver noticias



Anexo 20

Universidad de las Ciencias Informáticas (UCI)

Lunes, 18 de junio de 2012

"Año 54 de la Revolución"

Asunto: **Aval que favorece al módulo de comunicación por correo electrónico para el Juez Caribeño en Línea (COJ, por sus siglas en inglés).**

A quien pueda interesar:

El módulo MailJudge se diseñó, implementó, probó y desplegó satisfactoriamente en la UCI. Entre mayo y octubre de 2011, dicho subsistema estuvo disponible desde la versión 1.0 del COJ (a través de la Red Nacional e Internet), confirmándose su utilización por usuarios de diferentes países y centros de la Educación Superior. Producto de la interacción con el software se pudo constatar lo siguiente: cada petición procesada por el módulo devolvió correcta y exactamente lo que solicitaron los usuarios. Durante la fase de pruebas, se recibieron opiniones muy positivas por parte de administradores y usuarios finales. En octubre de 2011 se actualizó radicalmente la versión del COJ, por lo que desde entonces se revisan/ultiman detalles con vistas a la integración definitiva del módulo MailJudge. Sin lugar a dudas, este módulo le aporta una significativa ventaja al COJ respecto a sus homólogos en todo el mundo; al mismo tiempo que facilita en sobremanera el acceso de los usuarios cubanos al mencionado juez en línea.

Para que así conste, firmo el presente a los 18 días del mes junio del año 2012.

Prof. Ing. Dovier Antonio Ripoll Méndez

Director-General del ACM-ICPC en el Caribe

Líder del Equipo de Desarrollo del COJ

Anexo 21

Descripción del CUS "Consultar listado de problemas"

Caso de uso	Consultar listado de problemas	
Actores	Invitado	
Propósito	Permitir consultar el listado de problemas de un volumen determinado.	
Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema especificando el volumen de problemas y el criterio por el cual desea consultar el listado de problemas organizado o si desea consultarlo en formato PDF. El sistema valida que la petición del usuario esté en el formato correcto. Construye el listado de problemas organizado por el criterio especificado por el usuario y le envía un mensaje de correo con esta información. Finaliza así el caso de uso.	
Precondiciones	El usuario debe haber accedido al correo electrónico.	
Referencias	RF9,RF1.3,RF12	
Prioridad	Secundario	
Flujo normal de eventos		
Acción del actor		**Respuesta del sistema**

1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto si desea ver el listado de problemas de un volumen determinado ordenado por puntos, cantidad de soluciones aceptadas, total de submisiones enviadas, % de soluciones aceptadas o por el id en formato PDF.	2. Si solicita ver el listado de problemas de un volumen determinado organizado por puntos, ir a la **Sección 1: "Points".** Si solicita ver el listado de problemas de un volumen determinado organizado por cantidad de soluciones aceptadas, ir a la **Sección 2: "AC".** Si solicita ver el listado de problemas de un volumen determinado organizado por total de submisiones enviadas, ir a la **Sección 3: "Submissions".** Si solicita ver el listado de problemas de un volumen determinado organizado por % de soluciones aceptadas, ir a la **Sección 4: "% AC".** Si solicita ver el listado de problemas de un volumen determinado organizado por el id en formato PDF, ir a la **Sección 5: "ID".**
	3. Finaliza el caso de uso.
Sección 1: "Points"	
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:volume –v <# de volumen> -c points **(Ver Anexo 2)** Ej. (mj:volume –v 1 –c points)	2. Valida el formato del asunto del correo.
	3. Construye el listado de problemas organizado por puntos.
	4. Envía mensaje de correo y en el cuerpo del mismo muestra el listado de problemas del volumen solicitado organizado por puntos.
	5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1 del flujo básico.
Sección 2: "AC"	

1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:volume –v <# de volumen> -c ac **(Ver Anexo 2)**. Ej. (mj:volume –v 2 –c ac)	2. Valida el formato del asunto del correo. 3. Construye el listado de problemas organizado por soluciones aceptadas. 4. Envía mensaje de correo y en el cuerpo del mismo muestra el listado de problemas del volumen solicitado organizado por soluciones aceptadas. 5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda. 2. a.2 Ejecuta el punto 1 del flujo básico.
Sección 3: "Submissions"	
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:volume –v <# de volumen> -c submissions **(Ver Anexo 2)** Ej. (mj:volume –v 3 –c submissions)	2. Valida el formato del asunto del correo. 3. Construye el listado de problemas organizado por total de submisiones enviadas. 4. Envía mensaje de correo y en el cuerpo del mismo muestra el listado de problemas del volumen solicitado organizados por total de submisiones enviadas. 5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda. 2. a.2 Ejecuta el punto 1 del flujo básico.
Sección 4: "% AC"	
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato:	2. Valida el formato del asunto del correo. 3. Construye el listado de problemas organizado por % de aceptados con respecto al total de envíos.

mj:volume –v <# de volumen> -c %ac (**Ver Anexo 2**) Ej. (mj:volume –v 4 –c %ac)	4. Envía mensaje de correo y en el cuerpo del mismo muestra el listado de problemas del volumen solicitado organizados por % de aceptados con respecto al total de envíos.
	5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1 del flujo básico.
Sección 5: "% ID"	
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:volume –v <# de volumen> -f pdf **(Ver Anexo 4)** Ej. (mj:volume –v 4 –f pdf)	2. Valida el formato del asunto del correo.
	3. Busca el pdf correspondiente a ese volumen en caso de que exista, sino lo construye.
	4. Envía mensaje de correo con el listado de las descripciones de los problemas del volumen especificado organizados por el id de forma ascendente, adjunto en formato PDF.
	5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1 del flujo básico.

Anexo 22

Descripción del CUS "Ver ayuda"

Caso de uso	Ver ayuda
Actores	Invitado
Propósito	Permitir ver la ayuda.

Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema especificando en el asunto que desea ver la ayuda. El sistema construye la ayuda y le envía un mensaje de correo con la misma. Finaliza así el caso de uso.
Precondiciones	El usuario debe haber accedido al correo electrónico.
Referencias	RF2,RF1.2, RF12
Prioridad	Secundario
Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:help	2. Construye la ayuda. 3. Envía un mensaje de correo y en el cuerpo del mismo muestra la ayuda. 4. Finaliza el caso de uso.
Flujo Alternativo	
1. a Envía un mensaje de correo especificando en el asunto el siguiente formato: mj:ayuda –f pdf	
	1. a.1 Busca el pdf correspondiente a la ayuda en caso de que exista, sino lo construye. 1. a.2 Envía mensaje de correo con la ayuda adjunta en formato PDF. 1. a.3 Finaliza el caso de uso.
Poscondiciones	Se vio la ayuda.

Anexo 23

Descripción del CUS "Ver FAQs"

Caso de uso	Ver FAQs
Actores	Invitado
Propósito	Permitir ver las FAQs

Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema y en el asunto pide que le muestre las FAQs. El sistema valida que la petición del usuario esté en el formato correcto. Busca las FAQs y le envía un mensaje de correo con las mismas. Finaliza así el caso de uso.
Precondiciones	El usuario debe haber accedido al correo electrónico.
Referencias	RF7, RF1.1, RF12
Prioridad	Primario

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:faqs	2. Valida el formato del asunto del correo.
	3. Busca las FAQs.
	4. Envía un mensaje de correo mostrando en el cuerpo del mismo las FAQs.
	5. Finaliza el caso de uso.
Flujo Alternativo	
1. a Envía un mensaje de correo especificando en el asunto el siguiente formato: mj:faqs –f pdf	
	1. a.1 Valida el formato del asunto del correo.
	1. a.2 Busca el pdf correspondiente a las FAQs en caso de que exista, sino lo construye.
	1. a.3 Envía mensaje de correo con las FAQs adjuntas en formato PDF.
	1. a.4 Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda.
	2. a.2 Ejecuta el punto 1 del flujo básico.

Poscondiciones	Se vieron las FAQs.

Anexo 24

Descripción del CUS "Ver noticias"

Caso de uso	Ver noticias
Actores	Invitado
Propósito	Permitir ver noticias.
Resumen	El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema y en el asunto pide que le muestre las noticias. El sistema valida que la petición del usuario esté en el formato correcto. Busca las noticias y le envía un mensaje de correo con las mismas. Finaliza así el caso de uso.
Precondiciones	El usuario debe haber accedido al correo electrónico.
Referencias	RF4, RF12
Prioridad	Primario

Flujo normal de eventos	
Acción del actor	**Respuesta del sistema**
1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato: mj:news	2. Valida el formato del asunto del correo. 3. Busca las noticias. 4. Envía un mensaje de correo mostrando en el cuerpo del mismo las noticias. 5. Finaliza el caso de uso.
Flujo Alternativo	
2. a Envía mensaje de correo que no se corresponde con el formato definido.	
	2. a.1 Envía mensaje de correo con la ayuda. 2. a.2Ejecuta el punto 1 del flujo básico.
Poscondiciones	Se vieron las noticias.

Anexo 25

Descripción del CUS "Ver solución de problema"

Caso de uso	Ver solución de problema
Actores	Usuario
Propósito	Permitir ver la solución de un problema determinado.

<table>
<tr><td>Resumen</td><td colspan="2">El caso de uso se inicia cuando el usuario envía un mensaje de correo al sistema y en el asunto especifica el número de la solución que desea ver. El sistema valida el formato del asunto del correo y que el usuario tenga permiso a ver la solución. Busca la información que pide el usuario y le envía un mensaje de correo con la misma. Finaliza así el caso de uso.</td></tr>
<tr><td>Precondiciones</td><td colspan="2">El usuario debe tener permisos para ver la solución y haber accedido al correo electrónico.</td></tr>
<tr><td>Referencias</td><td colspan="2">RF11, RF12</td></tr>
<tr><td>Prioridad</td><td colspan="2">Secundario</td></tr>
<tr><td colspan="3">Flujo normal de eventos</td></tr>
<tr><td colspan="2">Acción del actor</td><td>Respuesta del sistema</td></tr>
<tr><td colspan="2" rowspan="3">1. El caso de uso se inicia cuando el actor envía un mensaje de correo especificando en el asunto el siguiente formato:

mj:solution -s <id solución> (Ver Anexo 9)

Ej. (mj:solution –s 18906)</td><td>2. Valida el formato del asunto del correo y que el usuario tenga permiso a ver la solución.</td></tr>
<tr><td>3. Busca la solución que pide el usuario.</td></tr>
<tr><td>4. Envía un mensaje de correo mostrando en el cuerpo del mismo:
❖ El Id del problema
❖ El usuario que envió la solución
❖ El código de la solución.
❖ La memoria y tiempo consumido.
❖ El lenguaje utilizado en la solución.
❖ El resultado de la solución.</td></tr>
<tr><td colspan="2"></td><td>5. Finaliza el caso de uso.</td></tr>
<tr><td colspan="3">Flujo Alternativo</td></tr>
<tr><td colspan="3">2. a Envía mensaje de correo que no se corresponde con el formato definido o no tiene permisos para ver la solución solicitada.</td></tr>
<tr><td colspan="2" rowspan="2"></td><td>2. a.1 Envía mensaje de correo con la ayuda.</td></tr>
<tr><td>2. a.2 Ejecuta el punto 1 del flujo básico.</td></tr>
<tr><td>Poscondiciones</td><td colspan="2">Se vio el código de la solución.</td></tr>
</table>