Universidad de las Ciencias Informáticas

Instituto Superior Politécnico "José Antonio Echeverría"
Facultad de Ingeniería Industrial
Ingeniería Informática

SISTEMA DE GESTIÓN DE SERVICIOS COMUNITARIOS

Trabajo para optar por el título de Ingeniería en Informática

AUTORES

José Fidalgo Hidalgo

Yudiel Alfredo Tamayo Agramonte

TUTOR

Ing. Renier Pérez García

Ciudad de la Habana
Junio 2005

”La formulación de un problema, es más importante que su solución.”

Albert Einstein

DECLARACION DE AUTORIA

Por este medio declaramos que somos los únicos autores de este trabajo y autorizamos a la Universidad de las Ciencias Informáticas (UCI) y al Centro de Estudios de Ingeniería y Sistemas (CEIS) para que hagan el uso que estimen conveniente con este trabajo.

Y para que así conste firmamos la presente a los 20 días del mes Junio del 2005.

_____	_____	_____
José Fidalgo Hidalgo	Yudiel A. Tamayo A.	Ing. Renier Pérez García

DEDICATORIA

A mis padres.

José.

A mis padres y mi hermana.

Yudiel.

AGRADECIMIENTOS

Ante todo deseo expresar mi más sincero agradecimiento a la Revolución Cubana, por haberme dado la oportunidad de realizar mis sueños.

A mis padres, por ser mis padres y haberme dado lo mejor de ellos.

A todas las personas que han significado algo en mi vida, familia o no, que por ser tantos y tan poco espacio, no los pongo aquí.

A mis amigos Yudiel, Julio, Chony, Yanier, Ronny y Yanesky, el “Escuadrón mete-la-pata” por haber hecho estos cinco años de carrera inolvidables.

A la Universidad de Camaguey y a la CUJAE, por contribuir con mi formación profesional.

A todos los profesores que me ayudaron, Geyser, Rigre, Julio y Sandro.

Al grupo Jagger donde aprendí gran parte de lo se hoy en día.

A mi tutor Renier Pérez.

A mis compañeros de cinco de años de estudio por haber compartido tanta cosas buenas y malas, que no se me olvidarán.

A Eminem, Metallica, Avril, Evanscene, Linkin’ Park, Varela, Back Street Boys, Moneda Dura, Buena Fe, por haber musicalizado mis días.

A la serie “Friends”, por haber alegrado mis días.

A todos los que no puse, pero no por no acordarme de ellos, sino por el espacio.

…a todos muchas gracias.

José Fidalgo Hidalgo.

Ante todo deseo expresar mi absoluta e infinita gratitud a la Revolución Cubana y a la persona que día a día hace realidad nuestros sueños y es nuestro comandante en jefe Fidel Castro Ruz.

A mis padres y mi hermana por haber dado lo mejor de ellos para que yo estuviese aquí ahora.

A todas las personas que han significado algo en mi vida…

A mis amigos Jose, Chony, Julio, Ronny, Yanier, Livan, Henry y Yanesky, por haberme dejado compartir con ellos estos cinco años inolvidables.

A la Universidad de Camaguey y a la CUJAE por haber contribuido en mi formación profesional.

A todos los profesores que de una manera u otra me ayudaron…gracias.

A mi tutor Renier Pérez.

A mis compañeros de grupo que hemos compartido tantos momentos inolvidables.

A la Serie “Friends” por haberme relajado las tensiones.

A todas las personas que no he mencionado en este documento y que me han ayudado gracias. Y gracias por ayudarme a ser una persona mejor.

…A todos lo mencionados y los no mencionados gracias.

Yudiel Alfredo Tamayo Agramante.

RESUMEN

La prestación de servicios en la UCI se resume como una secuencia de actividades que van desde la contratación de los servicios, la recepción de la solicitud del servicio, la planificación de las necesidades, la elaboración de los reportes y órdenes de servicios, el aseguramiento de la calidad incluyendo los plazos de garantía así como el pago del servicio brindado.

El presente trabajo está encaminado a desarrollar un sistema a través del cual se automatice el proceso de prestación de servicios en la UCI.

Para lograr los objetivos se hace un estudio del estado del arte de sistemas similares existentes en el mundo y en Cuba específicamente, de las tecnologías y herramientas, así como del entorno del negocio. También un estudio preliminar para determinar las características del sistema, describir las etapas de análisis y diseño del primer ciclo de desarrollo del Sistema de Gestión de Servicios Comunitarios.

De esta forma se automatiza un proceso que en estos momentos tiene grandes problemas a la hora de llevarse a cabo, entre las principales razones encontramos que los clientes no saben precisamente a qué lugar dirigirse ni cómo hacer una solicitud de servicio, incluso ni acerca de la existencia de los servicios que se prestan en la UCI, la información estadística acerca de los problemas que han existido y los historiales de los reportes de trabajo se llevan a cabo manualmente, lo que equivale a una pérdida enorme de tiempo por parte del personal que labora en estos temas.

Como se aprecia existen deficiencias que son producidas tanto por el actuar del ser humano como por las condiciones reales de trabajo y por tanto ameritan la consideración para la realización de este trabajo.

INDICE

INTRODUCCION

En la Universidad de las Ciencias Informáticas (UCI) existe hoy en día un gran volumen de personal. A su vez, existe también una serie de **Servicios Comunitarios** encaminados a resolver los problemas que se presentan en el personal del centro.

En la actualidad, estos servicios son brindados por las áreas correspondientes de forma independiente. Hay áreas en las que existe un software para resolver en cierta medida el problema, mientras que otras no cuentan ni siquiera con eso.

Hoy en día, gestionar los servicios comunitarios en la UCI es una tarea difícil de desarrollar. Por otra parte en el centro se realiza todo el proceso de forma manual, por el **problema** de que no existe actualmente un software capaz de llevar el control de forma eficiente de las solicitudes que se hacen actualmente, y que a su vez, le brinde al usuario de forma fácil y sencilla la posibilidad de realizar una solicitud y seguir el estado de la misma de una forma eficiente. Como consecuencia existe la siguiente **situación problémica**: no existe un mecanismo que le brinde al usuario la posibilidad de solicitar cualquier servicio con una alta eficiencia, obteniendo una respuesta del estado de su solicitud, así como; que sea posible que el cliente pueda evaluar el trabajo finalizado ya, para que de esta forma las entidades puedan conocer el nivel de satisfacción del personal. El proceso de solicitud de cualquier servicio se realiza vía telefónica o directamente en el departamento de servicios técnicos, esto provoca que haya ineficiencia y demora en la atención a los servicios solicitados; tampoco existe un sistema de información para mantener a los usuarios al tanto de los posibles problemas presentados en las áreas de servicios, además la inexistencia de algún mecanismo que permita a los directivos conocer la situación de los servicios en cierto momento, o un historial de los problemas que se han ido presentando.

Por todo esto, se hace necesario desarrollar un sistema que permita darle solución a los problemas aquí mencionados, y de esta forma eliminar el engorroso trabajo que hasta ahora se realiza.

Para ello se creará el **Sistema de Gestión de Servicios Comunitarios** el cual será capaz de satisfacer todas las necesidades anteriormente planteadas.

Para la implementación del mismo se ha decidido desarrollar la aplicación sobre Web, debido a su fácil acceso, y además para que el usuario pueda acceder a la versión más reciente en todo momento.

Para la realización de este trabajo fueron consultados los libros de UML y Patrones:

- Introducción al análisis y diseño orientado a objetos de Craig Larman.
- El Proceso Unificado de Desarrollo de Software de los autores Ivar Jacobson, Grady Booch y James Rumbaugh.
- Además de gran parte de la documentación sobre Ingeniería de Software que se encuentra en la red de la CUJAE, la UCI e Internet.

Con este sistema se espera automatizar todos los procesos que hasta ahora se hacían de forma manual, aumentando la confiabilidad y seguridad de los datos, proporcionando además facilidades a los usuarios del sistema, a través de una interface amigable y sencilla la cual a su vez reduce los costos económicos que hasta ahora se hacían en materiales de oficina y en hojas.

Las personas que realizan las solicitudes de un servicio comunitario en la UCI, son todas aquellas que trabajan en el centro (estudiantes, profesores, dirigentes, trabajadores), ya sean internos ó externos, mientras que las mismas son atendidas por el personal de área a que corresponda dicha solicitud (estos son designados por el jefe del área).

El Sistema de Gestión de Servicios Comunitarios le brindará al usuario la posibilidad de que este especifique los datos del problema que presente, estos pueden ser:

- Medio (medio con problema).
- Problema (problema asociado al medio).
- Local (Ubicación donde se encuentra el medio con problemas).

Además ofrecerá la posibilidad de seguir el estado de la solicitud hecha, así como un historial de todas las solicitudes hechas por el usuario en cualquier momento, permitiéndoles también, hacer reclamaciones en caso de no estar conforme con la solución a cierta solicitud.

El Sistema ofrece reportes acerca del estado de la situación con los servicios comunitarios en todo momento, y de esta forma todas las estadísticas se realizan de

forma automatizada, eliminando gran parte del trabajo que se realiza actualmente de forma manual.

El Sistema de Gestión de Servicios Comunitarios formará parte de la infraestructura interna de la UCI, ayudando en la toma de decisiones importantes dentro del centro.

El **objeto de estudio** de este trabajo es todo lo referente a la gestión de servicios comunitarios, desarrollo de aplicaciones multicapas y diseños de bases de datos.

De esta forma el **campo de acción** de este trabajo son todas las áreas que prestan servicios en la UCI, además de los gestores de bases de datos, la tecnología .NET, y los servicios web.

El **objetivo general** de este trabajo es realizar la primera versión del software con interfaz web, para la realización de solicitudes, además de brindar reportes e informes a varias personas en la UCI.

Y como objetivos específicos tenemos:

- Automatizar el procedimiento de realizar una solicitud.
    - Obtener reportes de:
    - Servicios brindados por áreas.
    - Reclamaciones hechas por áreas.
    - Solicitudes hechas por áreas.
    - Generales destinados a directivos.
- Otros.
    - Confeccionar el manual de usuario.

Para cumplir con los objetivos, se trazaron las siguientes tareas:

- Estudio de las tendencias actuales que brindan solución a problemas similares.
- Fundamentación teórica del sistema.
- Reconocimiento de Requisitos funcionales y Confección de los casos de uso.
- Diseño del sistema
- Implementación del sistema

- Prueba e implantación del sistema.

Se pretende finalmente obtener un producto de software a la altura de las exigencias actuales de producción de software en nuestra sociedad, acorde con los estándares internacionales de catalogación y los estándares de diseño y presentación de aplicaciones Web.

El presente trabajo ha sido organizado de la siguiente forma:

Capítulo I: Fundamentación teórica que contiene los conceptos necesarios para la comprensión plena de los temas tratados en el resto del documento.

Capítulo 2: Trata la situación de las tecnologías a utilizar en el desarrollo de la aplicación, se comparan y seleccionan las mejores propuestas para el trabajo, y se explican los conceptos principales que se van a tratar.

Capítulo 3: Describe el negocio a través de un modelo de Dominio, y se hace el análisis del sistema a desarrollar. Se definen las funcionalidades del sistema y se describen detalladamente, utilizando herramientas de modelación.

Capítulo 4: Enfoca la construcción de la solución mediante diagramas de clases, de datos, y se plantean los principios para el diseño y la implementación. Aquí se construyen las funcionalidades que se definieron en el capítulo anterior.

Capítulo 5: Es un estudio de factibilidad sobre el sistema, obteniendo los beneficios tangibles e intangibles y analizando los costos del desarrollo de esta propuesta.

CAPITULO 1

FUNDAMENTACION DEL TEMA

1.1 Introducción.

En el presente capitulo se brinda una vista global de los temas relacionados con los Servicios Comunitarios, así como los principales conceptos asociados al dominio del problema, que son necesarios para entender el modelo de negocio y la propuesta de la solución. Se realizarán además comparaciones sobre las tecnologías punteras en el área.

Además, se describen los procesos del negocio que se relacionan con el objeto de estudio de este trabajo. Se identifican los principales problemas que fundamentan la propuesta de solución, y se marcan los objetivos generales y específicos.

1.2 Principales conceptos asociados al dominio del problema.

1.2.1 World Wide Web (WWW)

En 1988 algunos investigadores del Instituto Europeo de Física de Partículas (CERN), deseaban desarrollar un método mejorado para que los científicos distribuidos alrededor del mundo pudieran compartir su información.

Debido a que la investigación se llevaba a cabo entre sitios distantes, realizar tareas sencillas (como leer un papel o visualizar una imagen) a menudo requería establecer la comunicación con la computadora donde se encontraba y luego traerlo a la computadora local. Además esta actividad requería la utilización de varios programas (como Telnet, FTP y un visualizador de imágenes, por ejemplo). Lo que buscaban los investigadores era crear un método que permitiera realizar toda esta actividad a través de una única interface.

En el transcurso de un año el proyecto se implementó. Hacia fines de 1990 los investigadores del CERN tenían un browser (navegador). En 1991, la WWW se publica para su utilización general en el CERN. Inicialmente estaba preparada sólo para hipertexto y artículos dentro de las noticias USENET. A medida que fue avanzando el

proyecto se agregaron interfaces a otros servicios de Internet (WAIS; FTP anónimo, Telnet y Gopher).

Durante 1992, el CERN comenzó a publicitar el proyecto WWW. La comunidad Internet se percató rápidamente de que estaban ante una gran idea y comenzó a crear sus propios servidores WWW para publicar información en la red. Incluso, algunos comenzaron a trabajar en crear interfaces simples para la WWW. A fines de 1993 los *browsers* se habían desarrollado para una gran variedad de computadoras y sistemas operativos. A la fecha, la WWW es una de las formas más populares de acceder a los recursos de Internet.

1.3 Objeto de estudio.

1.3.1 Descripción general.

En la UCI se prestan una serie de servicios del tipo comunitario, es decir, servicios que le resuelven problemas a todo el personal existente en el centro. Este personal no es más que el posible cliente. Entre los servicios que se brindan al cliente tenemos:

- Reparación de equipamiento tecnológico.
- Reparación de equipamiento electrodoméstico.
- Mantenimiento constructivo (albañilería, plomería, carpintería, cerrajería, electricidad, etc.)

Otros:

- Acueducto.
- Áreas verdes.
- Comunales.
- Lavandería.
- Telefonía.
- Docentes.
- Información acerca de los Servicios.

Estos servicios se subordinan a una dirección o área en específico, que puede ser Reparación tecnológica, Servicios generales, Mantenimiento y Soporte de software, en caso de que la solución no esté al alcance de las mismas se realiza el trámite por vías de terciarios, que pueden ser COPEXTEL, GRUPO ELECTRÓGENO, TELRED, Industria Ligera, ETECSA.

En la actualidad, estos servicios son brindados por las áreas correspondientes de forma independiente, y cada cuál con sus respectivos problemas. Hay áreas en las que existe un software que le da una solución temporal al problema, y otras en las que no.

1.3.2 Descripción del proceso de negocio actual

Actualmente en la UCI existen una serie de áreas de servicios destinadas a la recepción y solución de las solicitudes de los clientes, estas solicitudes se pueden realizar o por vía telefónica o presentándose personalmente al área. Una vez realizada la solicitud el usuario obtiene el id de su solicitud, en caso de que quiera averiguar por el estado en que se encuentra, con este id puede obtener la información. Mientras que el área se encarga de generar una orden de trabajo para dar solución a la solicitud del cliente.

1.3.3 Situación problémica

No existe en la UCI un mecanismo que le brinde al usuario la posibilidad de solicitar cualquier servicio con una alta eficiencia, obteniendo una respuesta del estado de su solicitud, así como que sea posible que el cliente pueda evaluar el trabajo finalizado ya, para que de esta forma las entidades puedan conocer el nivel de satisfacción del personal. El proceso de solicitud de cualquier servicio se realiza vía telefónica o directamente en el departamento de servicios técnicos, esto provoca que haya ineficiencia y demora en la atención a los servicios solicitados, tampoco existe un sistema de información para mantener a los usuarios al tanto de los posibles problemas presentados en las áreas de servicios, también la inexistencia de algún mecanismo que permita a los directivos conocer la situación de los servicios en cierto momento, así como un historial de los problemas que se han ido presentando.

1.4 Sistemas automatizados existentes vinculados al campo de acción.

1.4.1 Software para la Gestión de Servicios Técnicos: STECNICO.

Este sistema se utiliza para SAT (Servicios de Asistencia Técnica) de todo tipo de equipos electrodomésticos. STECNICO gestiona todo lo referente a talleres de servicios técnicos SAT, gestiona la reparación de electrodomésticos; genera los números de reparación desde el 10.000 hasta 9.999.999[1]; imprime resguardo para el cliente y hoja para el taller al entrar una reparación; busca las reparaciones por ocho campos distintos[1]; permite corregir cualquier dato en cualquier momento; lista las reparaciones pendientes de presupuestos; reporta los totales facturados a un mismo cliente; controla las reparaciones para ser entregadas; lista las facturas emitidas entre dos fechas; lista las reparaciones en espera de presupuestos visibles en unos segundos; devuelve un listado de reparaciones por operario; realiza confecciones de presupuestos y facturas; contiene una base de datos para componentes y/o repuestos, este se añade al presupuesto con una pulsación; contiene una base de datos para clientes; brinda la opción de elegir el Nº de código de cliente, manual o automático; el programa genera un número de factura automáticamente; realiza la impresión en papel de presupuestos y facturas con total claridad.

1.4.2 Solución para técnicos de soporte en servicios externos: EASYWORK.

Permite agilizar la transferencia de información de soporte de campo, y organizar los reportes de las actividades ejecutadas.

Este sistema se aplica para los diferentes tipos de agentes de soporte y asistencia técnica. Los tipos de usuarios que se benefician del sistema EASYWORK son:

Técnico de soporte de manutención: atiende llamados, es decir, se le llaman por móvil, vía voz, datos o por SMS.

Técnico en prevención de manutención: ejecuta labores programadas, ya sea para la sustitución de piezas o para la limpieza, calibración o ajuste de los equipos.

Agente de campo: provee materiales que son vendidos, como máquinas de venta, freezers, post-mix y otros.

EASYWORK está compuesto por dos softwares que controlan las actividades de campo:

EASYWORK-Mobile, para técnicos de campo. Los usuarios de este módulo son los técnicos de campo que dan soporte correctivo y manutención preventiva de los equipos, y otras actividades relacionadas al control de equipos y materiales de repuesto.

EASYWORK-Server, sirve de interfaz entre el módulo EASYWORK-Mobile y los sistemas existentes, en el cliente, tal como el de Control de Equipos y el de Control de las Actividades de Asistencia Técnica. En caso que el cliente no posea estos sistemas, se podrá ejecutar los controles por el EASYWORK-Server.

El sistema EASYWORK tiene como funciones las comentadas a continuación:

Proceso de preparación de datos: Planifica las actividades de soporte y otros servicios externos. En caso de soporte correctivo, abre un llamado para que se envíe al técnico.

Transmisión de datos: La transmisión de llamados de soporte correctivo, se puede hacer por teléfono convencional o por SMS. Los demás datos, se pueden transferir por el mismo medio o por medio de sincronización de dados.

Proceso de selección de tarea ejecutables y registro de las actividades: El sistema registra y controla tanto el tiempo de desplazamiento y ejecución, como los kilómetros recorridos en cada trayecto.

Permite también, registrar datos cualitativos, obtenidos de las mediciones de parámetros y también de ajustes y arreglos, hechos en equipos especializados.

1.5 Objetivos generales y específicos.

Para darle respuesta a la situación problémica planteada, se propone para este trabajo un conjunto de objetivos:

1.5.1 Objetivo general.

El objetivo general de este trabajo es desarrollar la primera versión del sistema *Gestión de Servicios Comunitarios*, el cual le permitirá al personal existente en la UCI atender las solicitudes de manera automatizada, aumentando de esta forma la eficiencia de este servicio.

1.5.2 Objetivos específicos.

Además el sistema debe ser capaz de:

- Realizar una solicitud.
- Ver solicitudes realizadas por el usuario.

- Ver las solicitudes realizadas a un área de servicio.
- Realizar reclamación a una solicitud.
- Emitir orden de trabajo.
- Ver órdenes de trabajo de un área de servicio.
- Generar reportes a directivos

1.6 Conclusiones.

En el presente capítulo se detallaron las condiciones y problemas que rodean el objeto de estudio, y a través de las definiciones y conceptos, se determinaron las condiciones específicas que rodean al problema y en base a esto se obtuvieron los objetivos generales y específicos para este trabajo.

CAPITULO 2

TENDENCIAS Y TECNOLOGIAS ACTUALES A CONSIDERAR

2.1 Introducción.

En el presente capitulo, se hace un estudio de los temas relacionados con este trabajo, tanto a nivel nacional, como internacional, definiendo conceptos importantes de la teoría en que se basa la solución del problema, facilitando de esta forma la comprensión del mismo. También se analizarán las herramientas, los lenguajes de programación y las metodologías para el análisis y diseño del sistema.

2.2 Fundamentación de las tecnologías en que se basa la propuesta.

2.2.1 .NET Framework.

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años, con el objetivo de obtener una plataforma sencilla y potente para distribuir el software en forma de servicios que puedan ser suministrados remotamente y que puedan comunicarse y combinarse unos con otros de manera totalmente independiente de la plataforma, lenguaje de programación y modelo de componentes con los que hayan sido desarrollados. Ésta es la llamada *Pataforma.NET (.NET Framework)*, y a los servicios antes comentados se les denomina *servicios Web (Web Services).*

El .**NET Framework** es un entorno de desarrollo y ejecución, que permite a diferentes lenguajes trabajar en conjunto para crear aplicaciones fáciles de construir, administrar, desplegar e integrar con otros sistemas.

El **.NET Framework** consiste en:

- Common Language Runtime (CLR).
- Framework Class Library (FCL).

2.2.1.1 Common Language Runtime (CLR).

Es el núcleo de la plataforma .NET. Es el motor encargado de gestionar la ejecución de las aplicaciones para ella desarrolladas y a las que ofrece numerosos servicios que

simplifican su desarrollo y favorecen su fiabilidad y seguridad. Las principales características y servicios que ofrece el CLR son:

- Modelo de programación consistente.
- Modelo de programación sencillo.
- Eliminación del "infierno de las DLLs.
- Ejecución multiplataforma.
- Integración de lenguajes.
- Gestión de memoria.
- Seguridad de tipos.
- Aislamiento de procesos.
- Tratamiento de excepciones.
- Seguridad avanzada.
- Interoperabilidad con código antiguo.

2.2.1.2 Framework Class Library (FCL).

El FCL es una librería incluida en el *.NET Framework,* formada por cientos de tipos de datos que permiten acceder a los servicios ofrecidos por el CLR y a las funcionalidades más frecuentemente usadas a la hora de escribir programas. Además, a partir de estas clases prefabricadas el programador puede crear nuevas clases que mediante herencia extiendan su funcionalidad y se integren a la perfección con el resto de clases de la FCL. Por ejemplo, implementando ciertos interfaces podemos crear nuevos tipos de colecciones que serán tratadas exactamente igual que cualquiera de las colecciones incluidas en la FCL.

Esta librería está escrita en MSIL, por lo que puede usarse desde cualquier lenguaje cuyo compilador genere MSIL. A través de las clases suministradas en ella es posible desarrollar cualquier tipo de aplicación, desde las tradicionales aplicaciones de ventanas, consola o servicio de Windows NT hasta los novedosos servicios Web y páginas ASP.NET. Es tal la riqueza de servicios que ofrece que puede crearse lenguajes que carezcan de librería de clases propia y sólo usen la FCL - como C#.

2.2.2 C-Sharp (C#).

C# es un lenguaje de programación que toma las mejores características de lenguajes existentes como *Visual Basic*, *Java* y *C++*, combinándolas en uno solo. El hecho de ser relativamente reciente no implica que sea inmaduro, pues *Microsoft* ha escrito la mayor parte de la *FCL* usándolo, por lo que su compilador es el más depurado y optimizado de los incluidos en el *.NET Framework SDK.*

C# (leído en inglés “C Sharp” y en español “C Almohadilla”) es el nuevo lenguaje de propósito general diseñado por *Microsoft* para su plataforma *.NET*. Sus principales creadores son ***Scott Wiltamuth*** y ***Anders Hejlsberg***, éste último también conocido por haber sido el diseñador del lenguaje Turbo Pascal y la herramienta RAD Delphi.

Aunque es posible escribir código para la plataforma *.NET* en muchos otros lenguajes, *C#* es el único que ha sido diseñado específicamente para ser utilizado en ella, por lo que programarla usando este es mucho más sencillo e intuitivo que hacerlo con cualquiera de los otros lenguajes.

La sintaxis y estructuración de *C#* es muy similar a la *C++*, ya que la intención de *Microsoft* es facilitar la migración de códigos escritos en estos lenguajes a *C#* y facilita su aprendizaje a los desarrolladores habituados a ellos. Sin embargo, su sencillez y el alto nivel de productividad son equiparables a los de *Visual Basic*.

2.2.2.1 Características principales del C#

- Sencillez.
- Modernidad.
- Orientación a componentes.
- Gestión automática de memoria.
- Seguridad de tipos.
- Instrucciones seguras.
- Sistema de tipos unificado.
- Extensibilidad de tipos básicos.
- Extensibilidad de operadores.

- Versionable.
- Eficiente.
- Compatible.

2.2.3 ASP.NET.

ASP.NET es un marco de trabajo de programación generado en CLR que puede utilizarse en un servidor para generar eficaces aplicaciones Web. ASP.NET ofrece varias ventajas importantes acerca de los modelos de programación Web anteriores:

- **Mejor rendimiento.** *ASP.NET* es un código de CLR, compilado que se ejecuta en el servidor. A diferencia de sus predecesores, *ASP.NET* puede aprovechar las ventajas del enlace anticipado, la compilación *just-in-time*, la optimización nativa y los servicios de caché desde el primer momento. Esto supone un incremento espectacular del rendimiento antes de siquiera escribir una línea de código.
- **Compatibilidad con herramientas de primer nivel.** El marco de trabajo de ASP.NET se complementa con un diseñador y una caja de herramientas muy completos en el entorno integrado de desarrollo (*Integrated Development Environment*, IDE) de Visual Studio. La edición WYSIWYG, los controles de servidor de arrastrar y colocar y la implementación automática son sólo algunas de las características que proporciona esta eficaz herramienta.
- **Eficacia y flexibilidad.** Debido a que ASP.NET se basa en CLR, la eficacia y la flexibilidad de toda esa plataforma se encuentra disponible para los programadores de aplicaciones Web. La biblioteca de clases de *.NET* Framework, la Mensajería y las soluciones de Acceso a datos, se encuentran accesibles desde el Web de manera uniforme. ASP.NET es también independiente del lenguaje, por lo que puede elegir el lenguaje que mejor se adapte a la aplicación o dividir la aplicación en varios lenguajes. Además, la interoperabilidad de CLR garantiza que la inversión existente en programación basada en COM se conserva al migrar a ASP.NET.
- **Simplicidad.** ASP.NET facilita la realización de tareas comunes, desde el sencillo envío de formularios y la autenticación del cliente hasta la implementación y la configuración de sitios. Por ejemplo, el marco de trabajo de

página de ASP.NET permite generar interfaces de usuario, que separan claramente la lógica de aplicación del código de presentación, y controlar eventos en un sencillo modelo de procesamiento de formularios de tipo Visual Basic. Además, CLR simplifica la programación, con servicios de código administrado como el recuento de referencia automático y el recolector de elementos no utilizados.

- **Facilidad de uso.** ASP.NET emplea un sistema de configuración jerárquico, basado en texto, que simplifica la aplicación de la configuración al entorno de servidor y las aplicaciones Web. Debido a que la información de configuración se almacena como texto sin formato, se puede aplicar la nueva configuración sin la ayuda de herramientas de administración local. Esta filosofía de "administración local cero" se extiende asimismo a la implementación de las aplicaciones ASP.NET Framework. Una aplicación ASP.NET Framework se implementa en un servidor sencillamente mediante la copia de los archivos necesarios al servidor. No se requiere el reinicio del servidor, ni siquiera para implementar o reemplazar el código compilado en ejecución.
- **Escalabilidad y disponibilidad.** ASP.NET se ha diseñado teniendo en cuenta la escalabilidad, con características diseñadas específicamente a medida, con el fin de mejorar el rendimiento en entornos agrupados y de múltiples procesadores. Además, el motor de tiempo de ejecución de ASP.NET controla y administra los procesos de cerca, por lo que si uno no se comporta adecuadamente (filtraciones, bloqueos), se puede crear un proceso nuevo en su lugar, lo que ayuda a mantener la aplicación disponible constantemente para controlar solicitudes.
- **Posibilidad de personalización y extensibilidad.** ASP.NET presenta una arquitectura bien diseñada que permite a los programadores insertar su código en el nivel adecuado. De hecho, es posible extender o reemplazar cualquier sub-componente del motor de tiempo de ejecución de ASP.NET con su propio componente escrito personalizado. La implementación de la autenticación personalizada o de los servicios de estado nunca ha sido más fácil.
- **Seguridad.** Con la autenticación de Windows integrada y la configuración por aplicación, se puede tener la completa seguridad de que las aplicaciones están a salvo.

2.2.4 XML Web Services.

Los *XML Web Services* permiten que las aplicaciones compartan información y que además invoquen funciones de otras aplicaciones independientemente de cómo se hayan creado las aplicaciones, cuál sea el sistema operativo o la plataforma en que se ejecutan y cuales son los dispositivos utilizados para obtener acceso a ellas. Aunque los *XML Web Services* son independientes entre sí, pueden vincularse y formar grupos de colaboración para realizar tareas determinadas.

Para integración, los "*Servicios Web XML*" sobre *.NET* hacen posible que diferentes piezas de software trabajen en conjunto para:

- Unir aplicaciones. Convertir aplicaciones independientes a constelaciones de aplicaciones para usar datos reales.
- Intercambiar datos. Los datos de clientes residen en aplicaciones aisladas, impidiendo ofrecer nuevos servicios que exploten esos datos.

¿Para qué sirve un *Web Service*? La respuesta puede ser otra pregunta: ¿Para qué sirve en programación una rutina? Todos sabemos que una rutina es como una caja negra, que encierra cierto proceso o algoritmo, y que cumple una función clara. Muchas rutinas y un guión central componen un programa en lo que se llama "programación estructurada". Un *Web Service* viene a ser una rutina en Internet.

¿Por qué se llama "*Web Service*" y no "Rutina en Internet"? Los protocolos que soportan los *Servicio Web* se comunican normalmente por el puerto 80, y basándose en *HTTP*, métodos *GET* y *POST*. Esto hace que podamos acceder a ellos al igual que lo hacemos en una página Web. La diferencia entre una página Web y un Web Service, es que la página la visita cualquier individuo interesado, mientras que el servicio sólo lo visitan programas que lo requieren.

De modo, que el conjunto de *Web Services* en Internet es una *World Wide Web* paralela, de carácter no humano, sino cibernético. Veamos, que los ordenadores ya hablan solos a través de Internet.

Los *Web Services* se actualizan de forma transparente para el programador y para el encargado de mantenimiento de la aplicación. Además, mediante un *Web Services* puedes implementar a tu programa funciones imposibles de contemplar bajo el uso de rutinas de librerías, como por ejemplo, incorporar un buscador de páginas Web. Por otro

lado, la carga de CPU que supone la ejecución de una rutina, desaparece al usar *Web Services*. La carga se reparte por Internet, sobre el servidor del *Web Services*. Esto es un comienzo de "Computación Distribuida".

Los *Web Services* permiten a los usuarios usar aplicaciones que comparten datos con otros programas modulares. Son aplicaciones independientes de la plataforma que pueden ser fácilmente publicadas, localizadas e invocadas mediante protocolos Web estándar, como XML. El objetivo final es la creación de un directorio *on-line* de *Web Services*, que pueda ser localizado de un modo sencillo y que tenga una alta fiabilidad. La integración de aplicaciones hará posible obtener información demandada en tiempo real, acelerando el proceso de toma de decisiones.

2.2.5 SQL Server

SQL Server es el eje principal de la administración y análisis de datos de la siguiente generación de productos y servicios de *Microsoft .NET*.

Es una solución integral de base de datos y análisis. *SQL Server* ofrece el rendimiento, escalabilidad y confiabilidad que requiere la Web y los entornos empresariales de línea de negocios. El nuevo soporte de *XML* y *HTTP* simplifica el acceso a datos y el intercambio, mientras que las poderosas capacidades de análisis mejoran el valor de los datos. Las características de disponibilidad mejoradas maximizan el tiempo de actividad, las funciones de la administración avanzadas automatizan las tareas rutinarias y las herramientas mejoradas de programación y los servicios aceleran el desarrollo.

2.3 Fundamentación de la metodología utilizada.

Para desarrollar la propuesta que presenta este trabajo, se ha decidido utilizar como metodología el Proceso Unificado de Desarrollo (Racional Unified Process), por sus características especiales y las facilidades que aporta a todo el proceso de desarrollo del software.

2.3.1 El proceso unificado de desarrollo (RUP)

Para desarrollar un software se necesita una forma coordinada de trabajo. Un proceso que integre las múltiples facetas del desarrollo. Se necesita un método común, un proceso que:

- Proporcione una guía para ordenar las actividades de un equipo.
- Dirija las tareas de cada desarrollador por separado y del equipo como un todo.
- Especifique los artefactos que deben desarrollarse.
- Ofrezca criterios para el control y la medición de los productos y actividades de proyectos.
- El Proceso Unificado de Desarrollo es una solución al problema del software.

"El Proceso Unificado es un proceso de desarrollo de Software. O sea, es el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema software. Sin embargo, el Proceso Unificado es más que un simple proceso; es un marco de trabajo genérico que puede especializarse para una gran variedad de sistemas de software, para diferentes áreas de aplicación, diferentes tipos de organización, diferentes niveles de aptitud y diferentes tamaños de proyecto. El Proceso Unificado está basado en componentes, lo cual quiere decir que el sistema software en construcción está formado por componentes software interconectados a través de interfaces bien definidas."

"El Proceso Unificado utiliza el Lenguaje Unificado de Modelado (Unified Modeling Language, **UML**) para preparar todos esquemas de un sistema software , De hecho, UML, es una parte esencial del Proceso Unificado – sus desarrollos fueron paralelos."

No obstante, los verdaderos aspectos definitorios del Proceso Unificado se resumen en tres fases claves – dirigido por caso de uso, centrado en la arquitectura e iterativo e incremental. Esto es lo que hace único al Proceso Unificado.

2.3.2 Lenguaje unificado de modelado (UML)

"UML son las siglas de Unified Modeling Language (Lenguaje Unificado de Modelado), notación (esquemática en su mayor parte) con que se construyen sistemas por medio de conceptos orientados a objetos".

"El UML (Lenguaje Unificado para la Construcción de Modelos) se define como un lenguaje que permite especificar, visualizar y construir los artefactos de los sistemas de software..." [BJR971]. "Es un sistema notacional (que, entre otras cosas, incluye el significado de sus notaciones) destinado a los sistemas de modelado que utilizan conceptos orientados a objetos".

EL UML (Lenguaje Unificado de Modelado) es una de las herramientas más atractivas y utilizadas en el mundo del desarrollo de software, esto se debe a que permite a los desarrolladores de sistemas generar diseños que capturen sus ideas en una forma convencional y comunicarlas a terceras personas. Básicamente el UML es un lenguaje gráfico para visualizar, especificar y documentar cada una de las partes que comprende el desarrollo de software. UML entrega una forma de modelar cosas conceptuales como lo son procesos de negocio y funciones de sistema, además de cosas concretas como lo son escribir clases en un lenguaje determinado, esquemas de base de datos y componentes de software reusables.

El UML prescribe un conjunto de notaciones y diagramas estándares para modelar sistemas orientados a objetos, y describe la semántica esencial de lo que estos diagramas y símbolos significan. Mientras que ha habido muchas notaciones y métodos usados para el diseño orientado a objetos, ahora los modeladores sólo tienen que aprender una única notación.

“El UML es un lenguaje para construir modelos; no guía al desarrollador en la forma de realizar el análisis y diseño orientados a objetos ni le indica cual proceso de desarrollo a adoptar.”

UML se puede usar para modelar distintos tipos de sistemas: sistemas de software, sistemas de hardware, y organizaciones del mundo real. Diagramas de Casos de Uso para modelar los procesos del negocio.

Además UML prescribe una notación estándar y semántica esencial para el modelado de un sistema orientado a objetos. Previamente, un diseño orientado a objetos podría haber sido modelado con cualquiera de la docena de metodologías populares, causando a los revisores tener que aprender las semánticas y notaciones de la metodología empleada antes que intentar entender el diseño en sí. Ahora con UML, diseñadores diferentes modelando sistemas diferentes pueden sobradamente entender cada uno los diseños de los otros.

En resumen UML es el resultado de la unión de tres metodologías, *Booch*, *OMT*, y *OOSE*. Estas han tenido una aplicación extensa en el campo del la POO, tienen su historia, y han sido aplicadas en una gran variedad de industrias y problemas, por lo que pueden ser clasificadas como muy maduras.

2.4 Herramientas utilizadas para el diseño y construcción del sistema.

2.4.1 Visual Studio .NET.

Visual Studio .NET es una herramienta completa para generar e integrar con rapidez aplicaciones y servicios Web XML, lo que mejora notablemente la productividad del programador y abre las puertas a nuevas oportunidades empresariales.

Visual Studio .NET y el .NET Framework permiten al desarrollador hacer servicios Web basados en XML además de otro tipo de aplicaciones. El .NET Framework viene incorporado directamente en la nueva línea de sistemas operativos Windows .NET. Para los dispositivos móviles se llama .NET Compact Framework.

Los componentes de la plataforma .NET pueden interactuar de distintas maneras. Esta comunicación es permitida por los servicios Web que integran los distintos tipos de dispositivos y componentes. [6]

La arquitectura abierta permite a los programadores utilizar cualquier lenguaje orientado a Microsoft .NET Framework y aprovechar los conocimientos de programación actuales, ahorrando así los cursos de reciclaje largos y costosos. Visual Studio .NET está basado en la más reciente plataforma de servidor de Microsoft Windows®, lo que incorpora escalabilidad, confiabilidad y seguridad a las aplicaciones. Se han simplificado la administración y la implementación de las aplicaciones en un ambiente de producción, reduciendo así los costes totales del ciclo. [6]

Es importante destacar que Microsoft ha puesto en manos de los desarrolladores un conjunto de herramientas gratis para la confección de aplicaciones en .NET, este paquete de aplicaciones esta conformado por versiones gratis de importantes programas como Visual Studio .NET y SQL Server.

2.4.2 Microsoft SQL Enterprise Manager.

Microsoft SQL Enterprise Manager es la herramienta administrativa fundamental para Microsoft SQL Server 2000. Proporciona una interfaz de usuario la cual permite:

- Definir grupos de servidores que estén corriendo el SQL Server.
- Registrar servidores individuales en un grupo.
- Configurar todas las opciones para cada servidor registrado.

- Crear y administrar todas las base de datos, usuarios, permisos y objetos para cada servidor registrado.
- Definir y ejecutar todas las tareas administrativas en cada servidor registrado.
- Diseñar y probar consultas SQL, batches y scripts interactivamente invocando SQL Query Analyzer.
- Invocar los wizards definidos para SQL Server.

2.4.3 Embarcadero ERStudio.

Es una de las herramientas CASE de diseño de bases de datos que ayuda a generar, mantener alta calidad y gran rendimiento en las aplicaciones de la base de datos desde un modelo lógico de los requerimientos de información y las reglas de negocio que definen la base de datos al modelo físico optimizado por las características especificas de esta. Permite visualizar la estructura, elementos clave y optimizar el diseño de las bases de datos, genera tablas u otras especificaciones en dependencia de la plataforma seleccionada. Tiene como ventajas:

- Facilidades de diseño de diagramas Entidad-Relación y Entidad-Relación extendido y transformación de este al modelo relacional (en tercera forma normal, preservando las dependencias funcionales y sin pérdidas de información).
- Comparación comprensiva entre el modelo de datos y la base de datos.
- Soporta la separación del modelo lógico y del físico.

2.4.4 Rational Rose.

Es la herramienta Case desarrollada por los creadores de UML que cubren todo el ciclo de vida de un proyecto: concepción y formalización del modelo, construcción de los componentes y certificación de las distintas fases. Nos permite una trazabilidad real entre modelo (análisis y diseño) y el código ejecutable.

Rational Rose domina el mercado de herramientas para el análisis, modelación, diseño y construcción orientada a objetos, tiene todas las características que los desarrolladores, analistas, y arquitectos exigen – soporte **UML** incomparable, desarrollo

basado en componentes con soporte para arquitecturas líderes en la industria y modelos de componentes, facilidad de uso e integración optimizada.

La corporación **Rational** ofrece el Proceso Unificado de Rational (**RUP**), que unifica las mejores prácticas de muchas disciplinas en un consistente y completo proceso del ciclo de vida, que permite al equipo de desarrollo disminuir los tiempos de liberación, además de hacer más predecible el software que ellos producen. Este proceso esta basado en el Lenguaje Unificado de Modelación (**UML** – estándar de la industria) y únicamente integrado a herramientas líderes en el desarrollo de software de Rational, el Proceso Unificado de Rational apoya el equipo completo de desarrollo de software con guías detalladas e información crítica aplicable a la mayoría de las aplicaciones de la industria.

Rose es una herramienta con plataforma independiente que ayuda a la comunicación entre los miembros del equipo, a monitorear el tiempo de desarrollo y a entender el entorno de los sistemas. Una de las grandes ventajas de **Rose** es que utiliza la notación estándar en la arquitectura de software (**UML**), la cual permite a los arquitectos de software *y* desarrolladores visualizar el sistema completo utilizando un lenguaje común, además los diseñadores pueden modelar sus componentes e interfaces en forma individual y luego unirlos con otros componentes del proyecto.

2.4.5 Adobe Photoshop 7.0

Como es ya conocido por todos los diseñadores a nivel mundial, el Adobe Photoshop es una herramienta que no puede faltar cuando de gráficos se trata. La misma tiene gran utilidad en el mundo de los diseños de las páginas web.

2.5 La solución.

Basándose en los análisis hechos en este capitulo se propone:

- Desarrollar una aplicación web debido al entorno donde se usará el sistema, ya que esto permitirá que todos los usuarios puedan acceder a la versión más reciente del sistema, con solo usar el navegador.
- Montarla sobre la plataforma **.NET** debido a las características de esta plataforma, la cual es probablemente la mejor opción hoy en día para las aplicaciones web.

- Lenguaje de programación **C#**, por ser fácil de aprender, además de ser el lenguaje hecho especialmente para ser usado con la plataforma **.NET**
- Gestor de bases de datos **SLQ Server 2000**. Uno de los mejores gestores de bases de datos que existen actualmente.

2.6 Conclusiones.

En el presente capítulo se han analizado las tecnologías actuales y se profundizó en algunos conceptos necesarios para la comprensión de la solución de este trabajo. Además se ha fundamentado la elección de cuales herramientas se utilizaran para el desarrollo de la aplicación. Finalmente se ha llegado a la conclusión de que el sistema se desarrollará sobre la plataforma **.NET**, usando como lenguaje de programación **C#**, y como gestor de bases de datos, se usará el **SQL Server 2000**.

CAPITULO 3

DESCRIPCION DE LA SOLUCION PROPUESTA

3.1 Introducción.

En el presente capítulo se describe la solución propuesta para el sistema, para ello se explican los procesos del negocio que tienen que ver con el objeto de estudio del sistema.

Además se exponen los requisitos funcionales y no funcionales del sistema que se propone, lo que permite tener una concepción general del sistema y representar a través del diagrama de caso de uso, las relaciones de los actores del sistema, y las secuencias de acciones con las que interactúan.

3.2 Reglas del negocio a considerar.

3.2.1 Conceptos asociados al modelo de negocios.

Área de servicios: En la UCI para la prestación de los servicios comunitarios existen áreas. Cada área brinda una serie de servicios, además tiene uno o varios jefes del área, supervisores y técnicos.

Medio: Cualquier objeto que exista en la UCI y que pueda presentar problemas.

Problema: Posible situación que presente un medio.

Situación problémica: Situaciones que puedan presentar los Usuarios dentro de la UCI, pero que no se puede definir de la forma medio-problema.

Local: Cualquier lugar dentro de la estructura de la UCI.

Supervisor: Persona que da respuesta a las solicitudes hechas por los usuarios, y además genera órdenes de trabajos.

Técnico: Persona que da solución a una orden de trabajo.

Jefe de área: Persona que dirige un área de servicios.

Solicitud: Petición por parte de un usuario de un servicio a un área de servicios por presentar problema. Ya sea por tener un medio con problemas, ó presentar una situación problémica.

Estado de solicitud: Estado que indica en un momento dado la situación en que se encuentra una solicitud.

Orden de trabajo: Orden definida para dar solución a un problema especifico dentro de un rango de fecha.

Reclamación: Queja por problemas en la solución de una solicitud.

Usuario: Toda persona que trabaje o estudie en la UCI.

3.3 Descripción de los procesos del negocio actual.

Actualmente en la UCI existen una serie de áreas de servicios destinadas a la recepción y solución de las solicitudes de los clientes, estas solicitudes se pueden realizar o por vía telefónica o presentándose personalmente al área. Una vez realizada la solicitud el usuario espera a que le den solución a su problema, mientras que el área se encarga de generar una orden de trabajo para dar solución a la solicitud del cliente. Además existen los reportes acerca del estado de los servicios comunitarios, los cuales se realizan de forma manual.

3.3.1 Actores y trabajadores del negocio actual.

Actores del Negocio	Justificación
Solicitante	**Todo el personal existente en la UCI.**

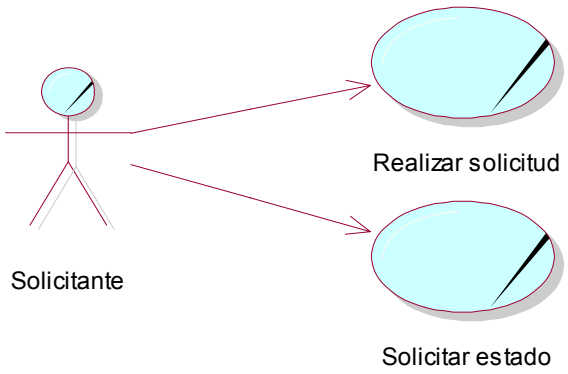
Trabajadores del Negocio	Justificación
Recepcionista	**Personal de las áreas de servicios encargados de registrar las solicitudes de los usuarios.**
Jefe de brigada	**Personal encargado de obtener las solicitudes de su área y de asignar técnicos que den solución a**

	las mismas.
Técnico	**Personal encargado de dar solución a la solicitud del usuario.**

3.3.2 Descripción de los Casos de Usos del Modelo de Negocio Actual.

Los casos de uso le proporcionan a los analistas del sistema un medio intuitivo para capturar los requisitos funcionales para cada usuario o sistema externo. Estos le permiten a los desarrolladores y a los clientes lleguen a tener una visión común sobre el problema en sí.

3.3.2.1 Diagrama de Casos de Uso del modelo de negocio actual.

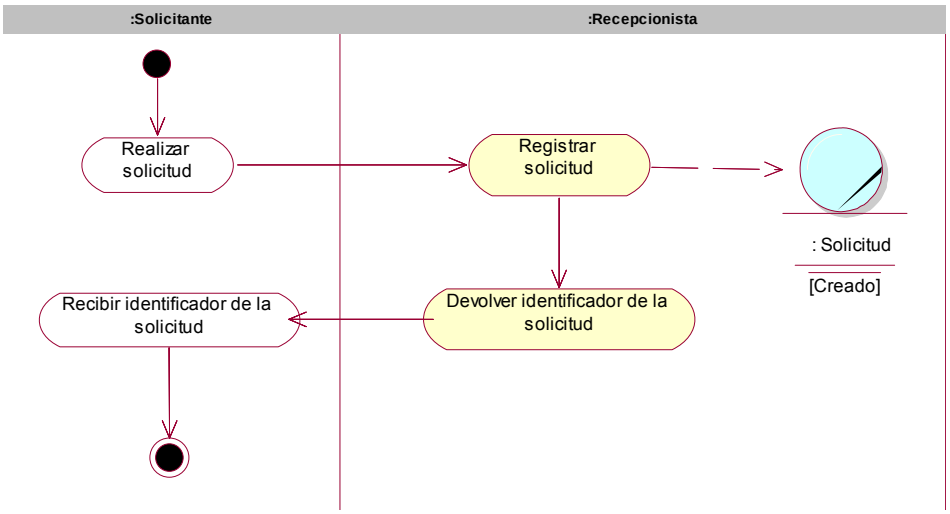


3.3.2.2 Expansión de los casos de uso.

Nombre del caso de uso del negocio:	CU1 – Realizar solicitud.
Actores del negocio:	Usuario.
Propósito:	Solicitar un servicio comunitario.
Resumen: El caso de uso inicia cuando un Usuario se pone en contacto con el Recepcionista de un área de servicios y este registra su solicitud devolviendo un	

identificador de su solicitud.	
Casos de uso asociados:	
Flujo de trabajo	
Acción del actor	**Respuesta del negocio**
1 El Usuario le informa al Recepcionista los datos del problema que presenta. 3 El Usuario recibe identificador de la solicitud.	2 El Recepcionista registra los datos de la solicitud.
Prioridad:	. Alta(De que se realice una solicitud, depende todo lo demás en el sistema)
Mejoras:	La automatización de este proceso de evaluación reducirá el tiempo de respuesta y permitirá a los Usuarios mejorar su gestión. El Usuario no tendrá que interactuar con el Recepcionista, pues este último desaparecerá al automatizarse este proceso.
Cursos alternos:	

Diagrama de actividades

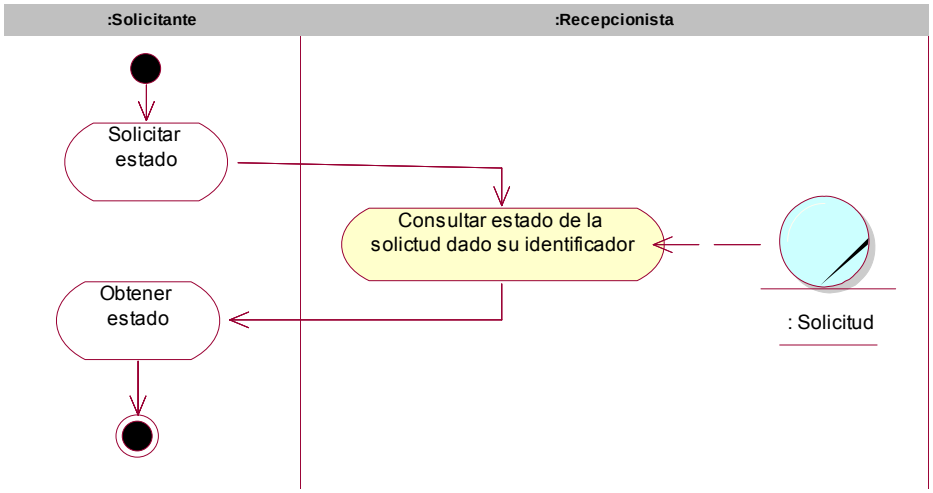


Nombre del caso de uso del negocio:	CU2 – Solicitar estado.
Actores del negocio:	Usuario.
Propósito:	Solicitar es estado en que se encuentra una solicitud hecha anteriormente.
Resumen: El caso de uso inicia cuando un Usuario se pone en contacto con el Recepcionista de un área de servicios para obtener el estado de una solicitud hecha previamente, el Usuario le proporciona el identificador de su solicitud y el Recepcionista le devuelve el estado en que se encuentra su solicitud.	
Casos de uso asociados:	
Flujo de trabajo	

Acción del actor	**Respuesta del negocio**
1 El Usuario le informa al Recepcionista el identificador de la solicitud de la cual desea conocer el estado en que se	2 El Recepcionista con el identificador de solicitud suministrado por el Usuario, busca su estado y se lo devuelve.

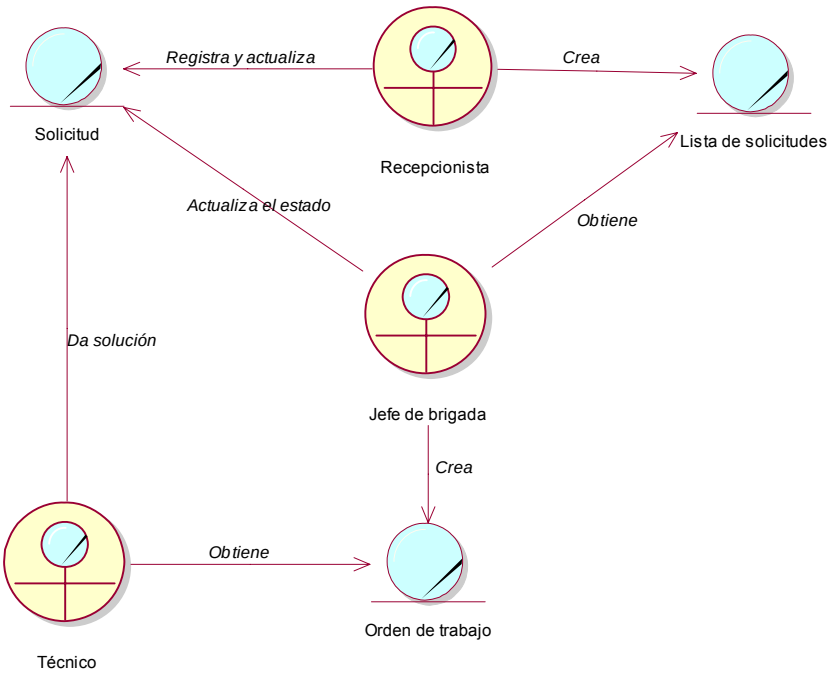
encuentra. 3 El Usuario recibe el estado en que se encuentra la solicitud solicitada.	
Prioridad:	Alta (El conocimiento por parte del Usuario del estado de su solicitud es fundamental)
Mejoras:	La automatización de este proceso de evaluación reducirá el tiempo de respuesta y permitirá a los Usuarios obtener el estado de su solicitud en todo momento.
Cursos alternos:	

Diagrama de actividades



3.3.3 Diagrama de clases del Modelo de Objetos.

El diagrama de clases usado para describir el Modelo de Objetos, muestra la participación de los trabajadores y entidades del negocio y la relación entre ellos.



3.4 Requisitos funcionales.

1. Registrar nueva solicitud de servicio.

    1.1. Datos que posee la solicitud.

        1.1.1. Datos generales de la persona que solicita.

        1.1.2. Local donde se encuentra la situación a reportar.

        1.1.3. Tipo de situación

        1.1.4. Situación problémica.

            1.1.4.1. Nombre de la situación problémica.

            1.1.4.2. Medios asociados a la situación problemática.

        1.1.5. Medio con problema.

            1.1.5.1. Medio que presenta el problema.

            1.1.5.2. Problema asociado al medio escogido.

        1.1.6. Descripción de la solicitud.

1.2. Una vez realizada la solicitud se almacenan además los siguientes datos.

1.3. IP de la máquina de donde se realizó la solicitud.

1.4. Fecha y Hora en que se realizó la solicitud.

1.5. Un servicio puede ser solicitado por varios usuarios.

1.6. Un medio puede tener más de una solicitud.

1.7. Un local puede ser motivo de varias solicitudes.

2. Seguimiento y reclamación de una solicitud.

2.1. Mostrar el estado en que se encuentra la solicitud.

2.1.1. Los estados son definidos por los administradores de sistema.

2.2. Para realizar la reclamación, se necesita la solicitud y la descripción de la reclamación.

2.3. Mostrar el historial de las solicitudes realizadas por un usuario.

2.4. Se dará la posibilidad al usuario de ver el conjunto de solicitudes que ha realizado.

2.5. Estas solicitudes podrán ser ordenadas atendiendo a: fechas, estados, tipos de servicios.

3. Emitir órdenes de trabajo.

3.1. Cada solicitud puede generar una orden de trabajo.

3.2. La orden de trabajo posee los siguientes datos:

3.2.1. Solicitud que la genera.

3.2.2. Fecha de asignación, fecha de inicio y fecha de fin.

3.2.3. A una orden de trabajo se le asigna un grupo de técnicos.

3.3. Actualizar solicitud de servicio.

3.4. Teniendo en cuenta el estado en el que se encuentra la solicitud, este se modifica.

4. Actualizar las áreas de servicios.

4.1. Añadir nueva área de servicios.

4.1.1. Datos del área de servicios: nombre del área y descripción.

4.2. Modificar los datos del área de servicios.

4.2.1. Datos modificables: nombre y descripción.

4.3. Eliminar área de servicios.

5. Actualizar los jefes de áreas de servicios.

5.1. Adicionar nuevo jefe de área.

5.2. Eliminar jefe de área.

6. Actualizar estados de las solicitudes.

6.1. Añadir nuevo estado de solicitud.

6.1.1. Datos del nuevo estado: nombre y descripción.

6.2. Modificar los datos del estado de solicitud.

6.2.1. Datos modificables: nombre y descripción.

6.3. Eliminar estado de solicitud.

7. Actualizar los servicios de un área de servicios.

7.1. Adicionar nuevo servicio.

7.1.1. Datos del servicio: nombre y descripción.

7.2. Modificar servicio.

7.2.1. Datos modificables: nombre y descripción.

7.3. Eliminar servicio.

8. Actualizar medios de un servicio.

8.1. Adicionar nuevo medio.

8.1.1. De los medios se conoce el id. Los demás datos se obtienen a través de un Web Services.

8.2. Eliminar medio.

9. Actualizar los problemas de un medio.

9.1. Añadir un nuevo problema.

9.1.1. De los problemas se conoce el nombre y la descripción.

9.2. Modificar los datos de un problema.

9.2.1. Datos modificables de un problema: nombre y descripción.

9.3. Eliminar problema.

10. Actualizar las situaciones problémicas.

10.1. Añadir nueva situación problémica.

10.1.1. Datos de la situación problémica: nombre, descripción e ID del área de servicios.

10.2. Modificar una situación problémica.

10.2.1. Datos modificables: nombre y descripción.

10.3. Eliminar situación problémica.

11. Actualizar supervisores de un área de servicios.

11.1. Adicionar nuevo supervisor.

11.2. Eliminar supervisor.

12. Actualizar técnicos de un área de servicios.

12.1. Adicionar nuevo técnico.

12.2. Eliminar técnico.

13. Mostrar solicitudes realizadas al área.

13.1. Se dará la posibilidad a los jefes de áreas y los supervisores, ver las solicitudes hechas al área.

13.2. Las solicitudes podrán ser ordenas por fecha, estado, supervisor, tipo de situación, etc.

14. Ver reportes.

3.5 Requisitos no funcionales.

Interfaz de usuario:

Se aplicará el estándar del proyecto UCI Ciudad Digital para el diseño de interfaz de usuario. Esta será legible, simple de usar, e interactiva.

Usabilidad:

Facilidad de uso para todo tipo de clientes, incluyendo personas con pocos conocimientos en el uso de las PCs.

Manual de usuario. Material de entrenamiento. Ayuda en Línea, soportada por páginas Web, que estará disponible al usuario en todo momento.

Soporte:

El sistema debe ser de fácil instalación, adaptable a numerosas plataformas y de fácil mantenimiento.

Portabilidad:

Facilidad para adaptarlos a diferentes ambientes sin necesidad de usar otros medios que los previstos. Se requiere de Windows como plataforma.

Seguridad:

Se deben implementar varios niveles de usuarios con permisos que correspondan con el rol que desempeñan en la aplicación.

Evaluar mecanismos de tolerancias a faltas. Predicción de fallos.

Protección contra los fallos. El sistema debe ser capaz de en pocos segundos recuperarse de un fallo de una operación.

El sistema permite que los usuarios pocos familiarizados con el sistema perciban sin problemas las salidas del mismo. Las salidas del sistema tienen que tener un 100 % de veracidad y precisión.

Podrá ser usado las 24 horas del día.

Legales:

Debe cumplir con lo establecido en las leyes del sistema de seguridad y protección de nuestro país. El sistema de soporte comunitario y la documentación del mismo pertenecen al proyecto UCI-Ciudad Digital.

Software:

Sistemas Operativos de la familia Windows (Windows NT 4.0, Windows 9.x, Windows 2000). La aplicación se realizará en un ambiente Web, la Base de Datos es independiente de la aplicación.

Hardware:

Compatible con procesadores x486 o superior, con 64 MB de memoria RAM. Un mínimo de 200 MB de espacio disponible en disco.

Soportado por una red hasta 100 Mbps de velocidad.

Restricciones:

Para la documentación del sistema se utilizó para realizar el análisis y el diseño del sistema la metodología RUP, y como herramienta de modelación UML (Unified Modeling Language), como herramienta de apoyo a este se utilizó el Rational Rose. Su desarrollo se llevará a cabo con .Net, y como gestor de base de datos se utilizará el SQL Server.

3.6 Descripción del sistema propuesto.

Con el objetivo de dar solución a los requisitos planteados en este trabajo, se propone un sistema para la Gestión de Servicios Comunitarios.

El sistema define todo lo referente a las solicitudes por parte del personal de la UCI. Para realizar una solicitud el solicitante debe llenar los datos pertinentes, como local donde se encuentra el problema, tipo de problema a resolver, si es un medio con problema o una situación problémica, en caso de ser un medio con problema debe informar el medio y escoger entre una lista de posibles problemas para el medio seleccionado el que más se acomode a su problema en sí. En caso contrario deberá seleccionar entre una lista de situaciones problémicas la que presenta y además debe especificar la lista de medios que se relacionan con el problema en cuestión. Además puede dar una descripción de su problema para mejorar el entendimiento del mismo.

Los Usuarios del sistema pueden ver en todo momento el estado en que se encuentran sus solicitudes, en caso de haber sido atendidas, ver los datos de los técnicos que darán solución a su problema, además en caso de demora o de no estar de acuerdo con los servicios brindados el Usuario tiene la posibilidad de realizar tantas reclamaciones a una solicitud como estime conveniente.

Los Jefes de áreas son los encargados de definir los Supervisores que pertenecen a su área.

Los Supervisores son los encargados de definir los Técnicos que pertenecen a su área, así como los servicios que brinda el área, junto con los medios que pertenecen a cada servicio y los problemas asociados a cada medio en particular. Además puede definir las situaciones problémicas definidas para el área. Igualmente son los encargados de dar respuesta a las solicitudes hechas por los Usuarios del sistema, generar órdenes de trabajo para cada solicitud, así como actualizar los estados de las solicitudes en todo momento.

Los Directivos pueden solicitar reportes del estado de los Servicios Comunitarios en cualquier momento, también definir los reportes que desea recibir, y cada que frecuencia desea recibirlos.

Los Administradores del sistema pueden definir las áreas de servicios, así como los jefes de las áreas de servicios, pueden actualizar los posibles estados por lo que puede pasar una solicitud y definir los accesos de cada tipo de usuario en el sistema.

Además se han definido un conjunto de roles para restringir el acceso a las partes del sistemas según el tipo de usuario que ingrese al sistema.

3.6.1 Actores del sistema.

Actores del SISTEMA	**Justificación**
Usuario	Cualquier persona, estudiante o trabajador de la UCI que solicite un servicio.
Supervisor	Persona que pertenece a un área de servicios, la cual se encarga de las solicitudes que lleguen a su área.
Jefe de área	Persona al mando de un área de servicios de la UCI.
Directivo	Persona con cargo directivo dentro de la UCI, que necesita reportes acerca de los servicios.

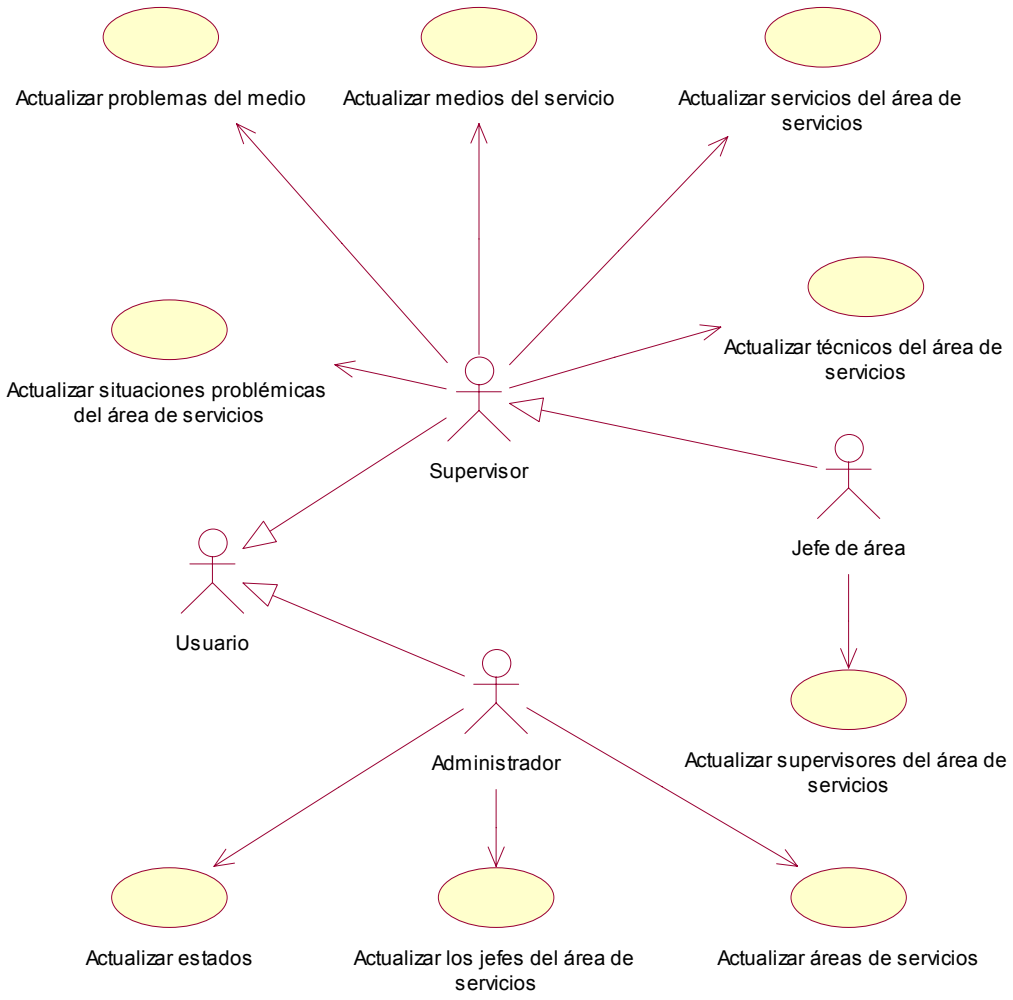
Administrador	Persona designada para darle mantenimiento al sistema.

3.6.2 Modelo de casos de uso del sistema.

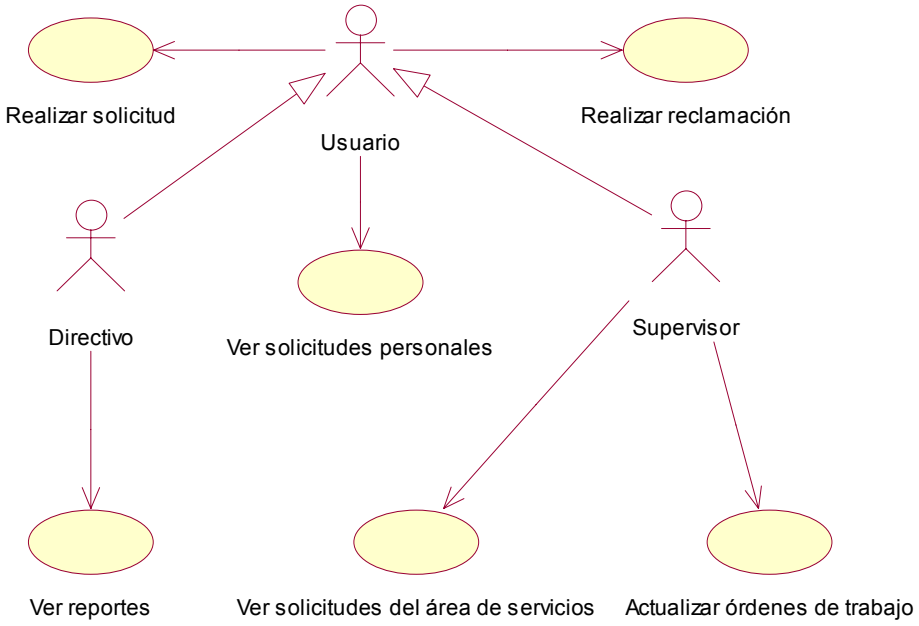
Los modelos de casos de uso del sistema se dividirán en paquetes según la funcionalidad de los casos de uso. Los paquetes son:

- **Paquete de Administración**. Agrupa todos los casos de uso relacionados con la administración del negocio del sistema.
- **Paquete de Gestión de Servicios**. Agrupa los casos de uso relacionados con la gestión de las solicitudes, así como el seguimiento y solución de las mismas.

3.6.2.1 Administración.



3.6.2.2 Gestión de Servicios.

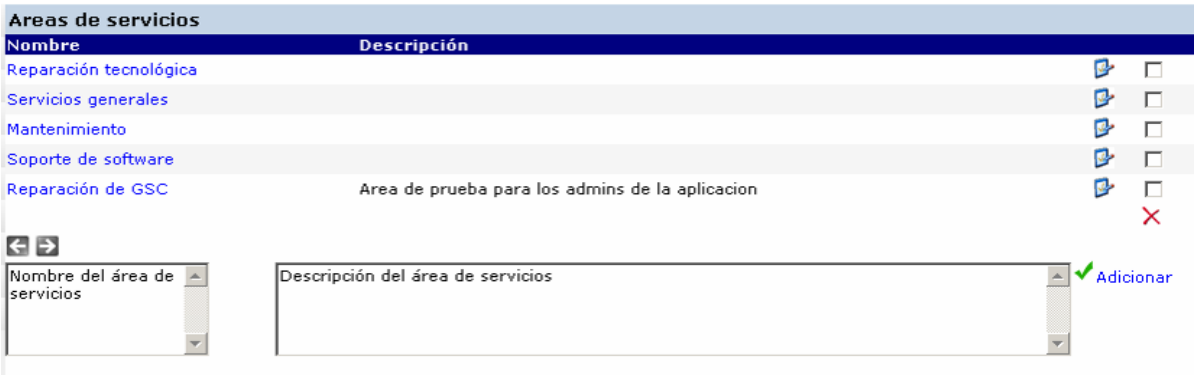


3.6.3 Expansión de los casos de uso del sistema.

Mediante los casos de uso expandidos se describe en detalle la secuencia de eventos que los actores utilizan para completar un proceso a través del sistema.

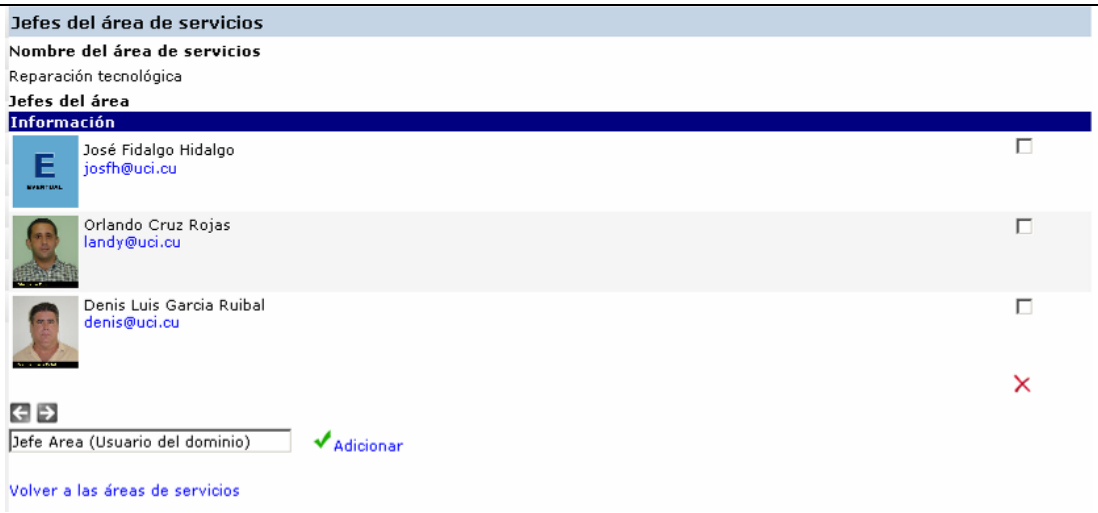
Caso de Uso:	**Actualizar áreas de servicios.**
Actor(es):	Administrador
Propósito:	Permitir a los administrativos actualizar las áreas de servicios.
Resumen:	El caso de uso se inicia cuando el administrador muestra la lista de las áreas de servicios definidas en el sistema, puede agregar, eliminar y modificar las áreas (***Pantalla 1***).
Referencias:	R4
Precondiciones:	
Poscondiciones:	

Requisitos Especiales:	
Prototipo	



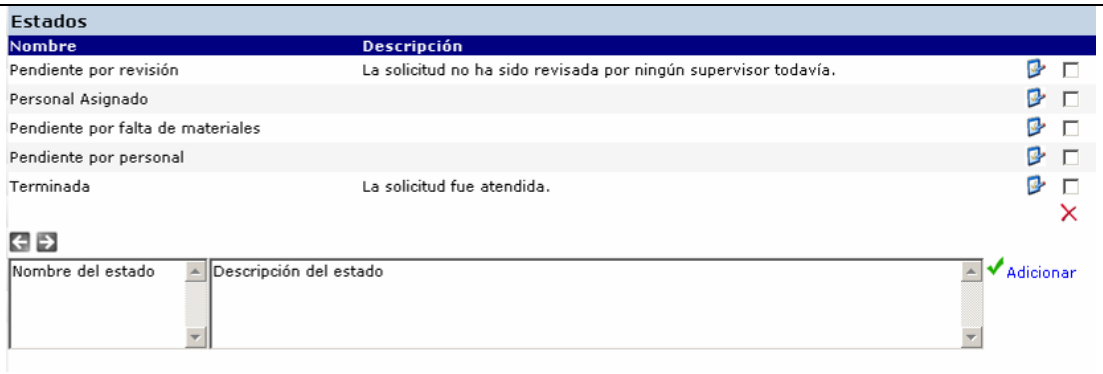
Pantalla 1

Caso de Uso:	**Actualizar los jefes del área de servicios.**
Actor(es):	Administrador
Propósito:	Permitir que los administradores puedan actualizar los jefes de las áreas de servicios.
Resumen:	El caso de uso se inicia cuando el administrador escoge el área de servicios para actualizar sus jefes. El administrador puede agregar o eliminar jefes (***Pantalla 1***).
Referencias:	R5
Precondiciones:	Debe existir al menos un área de servicios para adicionarle jefes (Ver caso de uso Actualizar áreas de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



Pantalla 1

Caso de Uso:	**Actualizar estados.**
Actor(es):	Administrador
Propósito:	Permitir que los administradores actualicen los posibles estados por los que puede pasar una solicitud.
Resumen:	El caso de uso se inicia cuando el administrador revisa los estados, puede agregar, eliminar y modificar los estados (***Pantalla 1***).
Referencias:	R6
Precondiciones:	
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



Pantalla 1

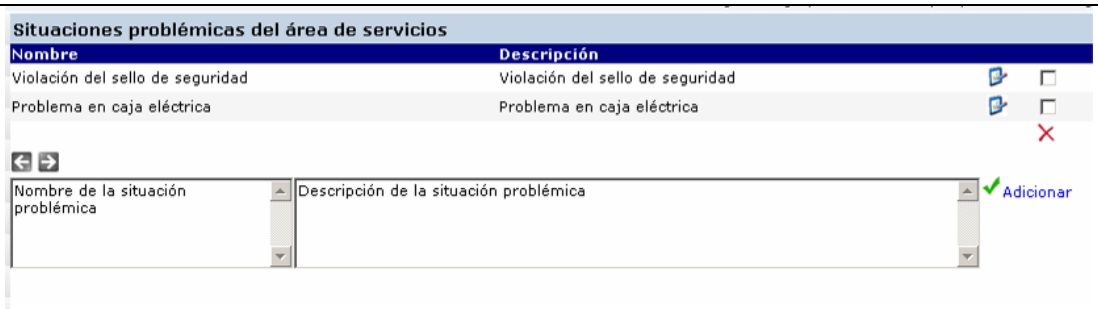
Caso de Uso:	**Actualizar técnicos del área de servicios.**
Actor(es):	Supervisor.
Propósito:	Permitir que los supervisores puedan actualizar los técnicos de su área de servicios.
Resumen:	El caso de uso se inicia cuando el supervisor revisa los técnicos del área, donde puede agregar o eliminar técnicos (***Pantalla 1***).
Referencias:	R12
Precondiciones:	Debe existir al menos un área de servicios para adicionarle técnicos (Ver caso de uso Actualizar áreas de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	

Pantalla 1

Caso de Uso:	**Actualizar supervisores del área de servicios.**
Actor(es):	Jefe de área
Propósito:	Permitir a los jefes de áreas actualizar los supervisores del área de servicios.
Resumen:	El caso de uso se inicia cuando el jefe de área lista los supervisores del área de servicios. Aquí se puede agregar y eliminar supervisores (***Pantalla 1***).
Referencias:	R11
Precondiciones:	Debe existir al menos un área de servicios para adicionarle supervisores (Ver caso de uso Actualizar áreas de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	

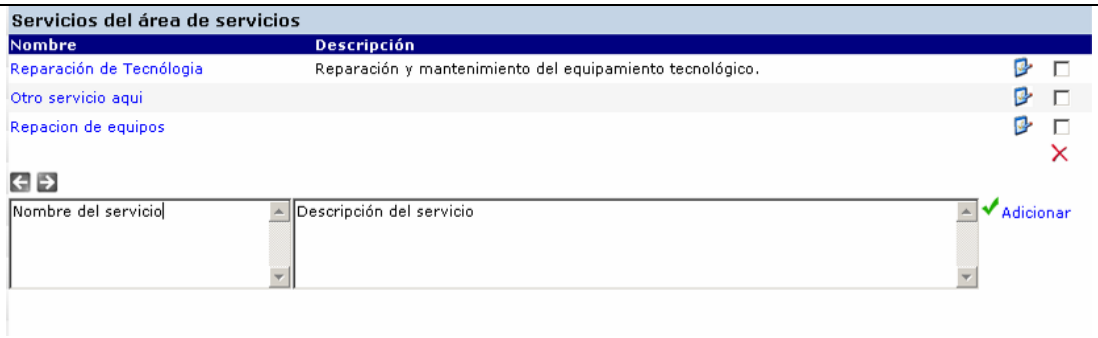
Pantalla 1

Caso de Uso:	**Actualizar situaciones problémicas del área de servicios.**
Actor(es):	Supervisor
Propósito:	Permitir que los supervisores de un área de servicios puedan actualizar las situaciones problémicas.
Resumen:	El caso de uso se inicia cuando el supervisor lista las situaciones problémicas del área de servicios. Aquí se puede agregar, eliminar y modificar las situaciones problémicas (***Pantalla 1***).
Referencias:	R10
Precondiciones:	Debe existir al menos un área de servicios para adicionarle situaciones problémicas (Ver caso de uso Actualizar áreas de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



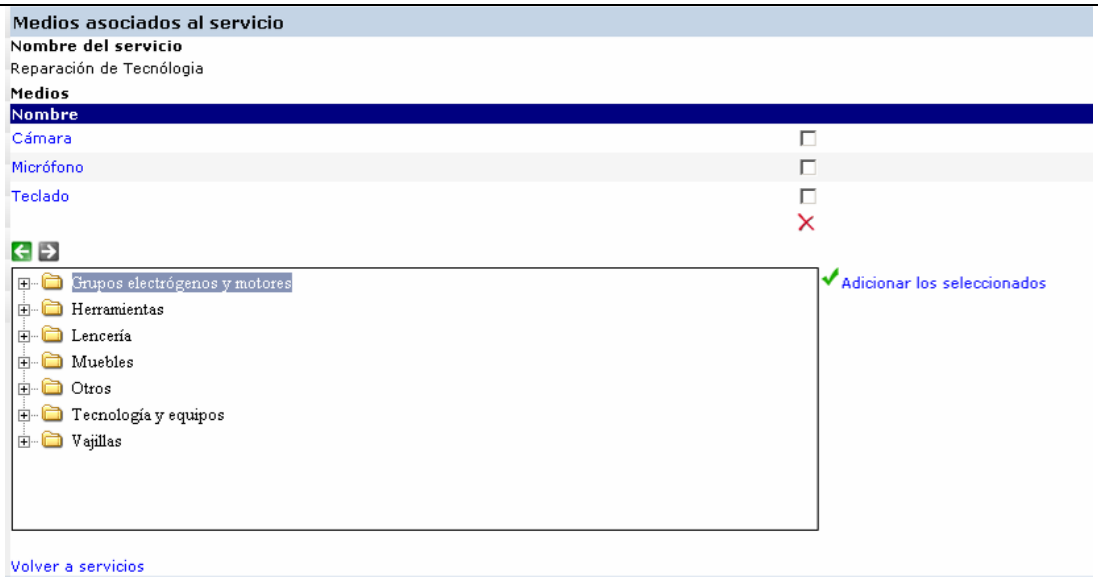
Pantalla 1

Caso de Uso:	**Actualizar servicios del área de servicios.**
Actor(es):	Supervisor.
Propósito:	Permitir a los supervisores de un área de servicios actualizar los servicios del área.
Resumen:	El caso de uso se inicia cuando el supervisor revise los servicios del área, puede agregar, eliminar y modificar los servicios (***Pantalla 1***).
Referencias:	R7
Precondiciones:	Debe existir al menos un área de servicios para adicionarle servicios (Ver caso de uso Actualizar áreas de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



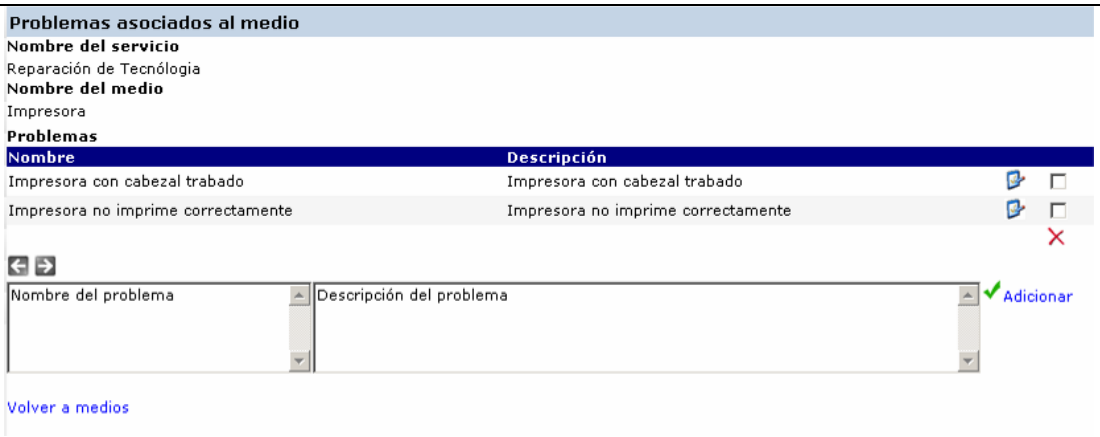
Pantalla 1

Caso de Uso:	**Actualizar medios del servicio.**
Actor(es):	Supervisor
Propósito:	Permitir a los supervisores actualizar los medios asociados a los servicios del área.
Resumen:	El caso de uso se inicia cuando el supervisor escoge un servicio para actualizar sus medios. Se puede agregar y eliminar medios (***Pantalla 1***).
Referencias:	R8
Precondiciones:	Debe existir al menos un servicio para poder adicionarle un medio (ver caso de uso Actualizar servicios del área de servicios).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



Pantalla 1

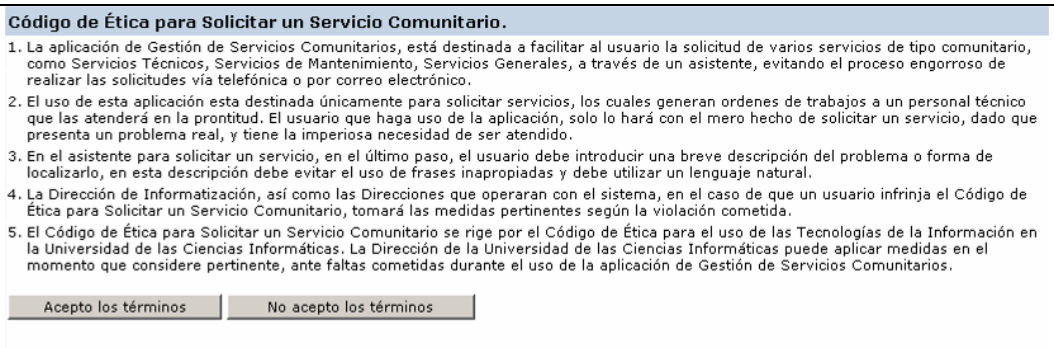
Caso de Uso:	**Actualizar problemas del medio.**
Actor(es):	Supervisor
Propósito:	Permitir a los supervisores actualizar los problemas asociados a los medios.
Resumen:	El caso de uso se inicia cuando el supervisor escoge un medio y ve la lista de problemas asociados al mismo. Se puede agregar, eliminar y modificar los problemas (***Pantalla 1***).
Referencias:	R9
Precondiciones:	Debe existir al menos un medio para asignarle sus posibles problemas (ver caso de uso Actualizar medios del servicio).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



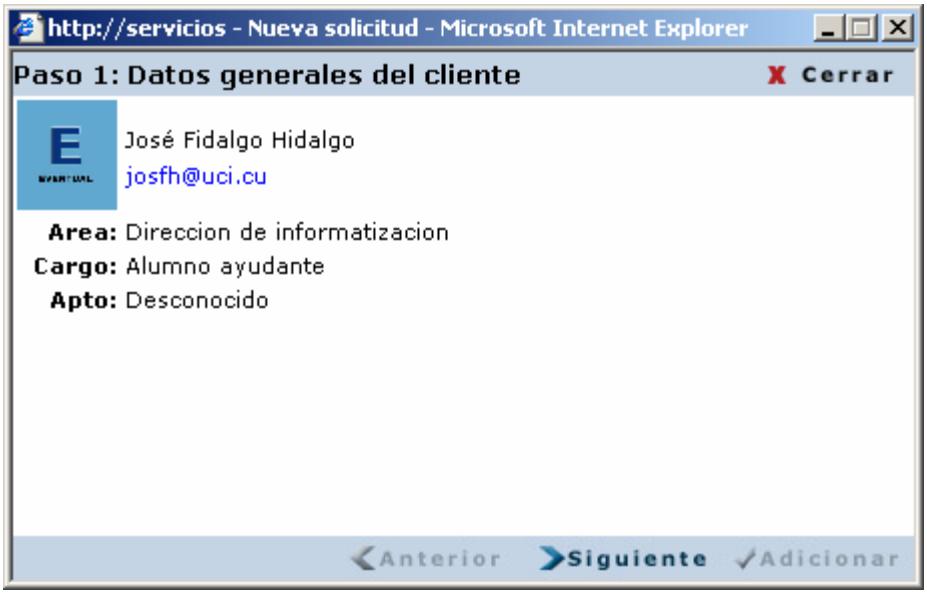
Pantalla 1

Caso de Uso:	**Realizar solicitud.**
Actor(es):	Usuario
Propósito:	Permitir que los Usuarios del sistema puedan realizar solicitudes.
Resumen:	El caso de uso se inicia cuando el Usuario escoge Nueva solicitud, y se le muestra el código de ética. El usuario debe aceptar las condiciones, para poder realizar la solicitud (***Pantalla 1***). La selección de los datos de la solicitud se realizará mediante un asistente, a través de una serie de pasos, para realizar el proceso de una forma más comprensible para el usuario. En el primer paso (***Pantalla 2***) el asistente muestra los datos personales del usuario que está realizando la solicitud. En el paso 2 (***Pantalla 3***), el usuario debe escoger el local donde se presenta el problema, los locales se muestran en forma jerárquica para que el usuario logre ubicarse de una forma fácil en los locales. Paso 3 (***Pantalla 4***), aquí el usuario puede escoger el tipo de situación que presenta, es decir, si es un medio con problema ó

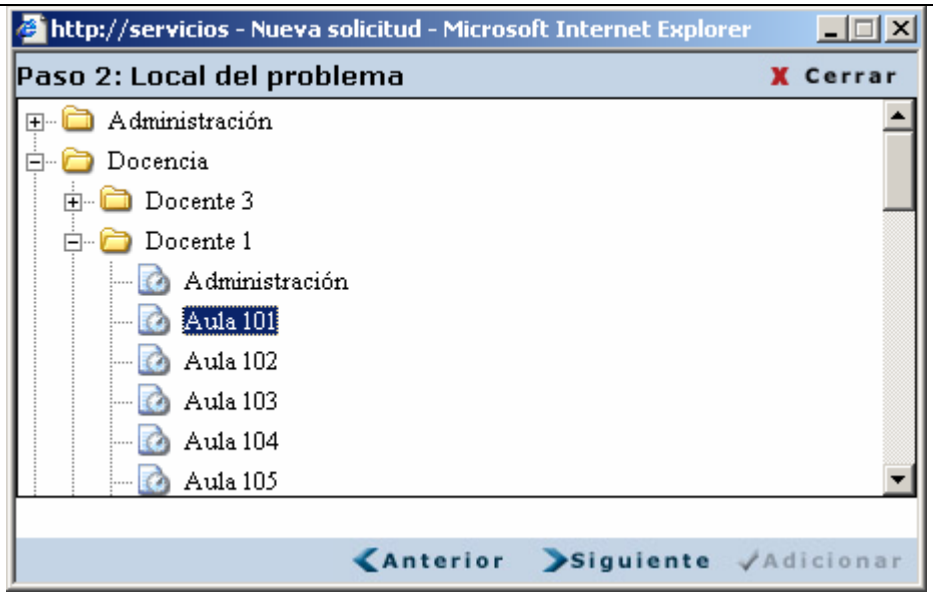
	una situación problémica. **En el caso de haber seleccionado "situación problémica":** Paso 4 (***Pantalla 5***), en este caso se muestran las situaciones problémicas, para que el usuario seleccione la que más se ajuste a su problema. Paso 5 (***Pantalla 6***), aquí se muestran los medios, para que el usuario escoja los medios que puedan estar relacionados a la situación problémica. **En el caso de haber seleccionado un medio con problema:** Paso 4 (***Pantalla 7***), aquí el usuario debe escoger el medio que presenta problema. Paso 5 (***Pantalla 8***), según el medio escogido, en este paso el usuario debe escoger un problemas de los asociados al medio. Paso 6 (***Pantalla 9***), en este paso el usuario debe dar una descripción de la solicitud, en la cual puede poner cualquier dato que estime conveniente. Paso 7 (***Pantalla 10***), en este paso ya los datos de la solicitud han sido entrados, aquí el usuario puede ver una vista previa de la solicitud o aceptar la solicitud. En la vista previa de la solicitud se muestran todos los datos de la solicitud (***Pantalla 11***).
Referencias:	R1
Precondiciones:	
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



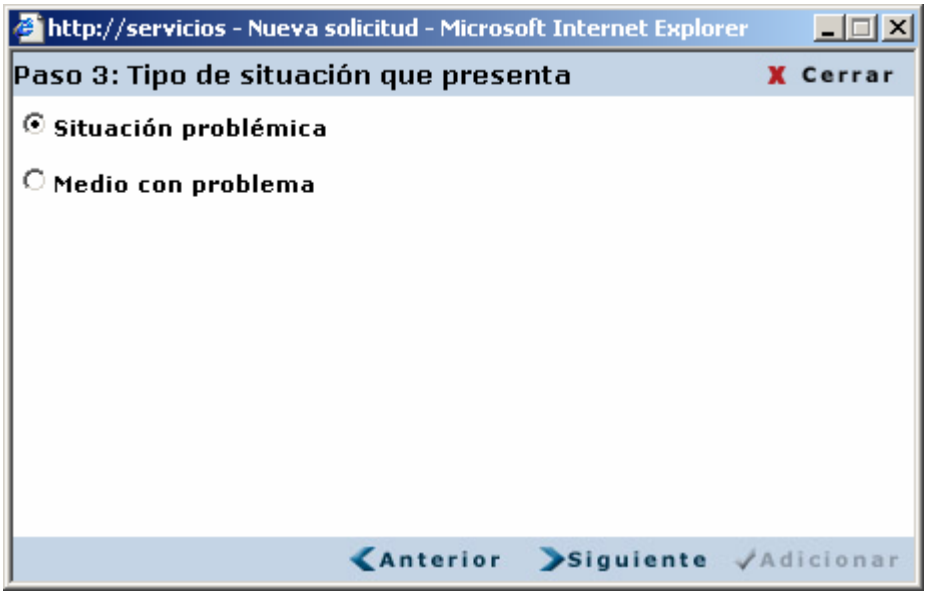
Pantalla 1



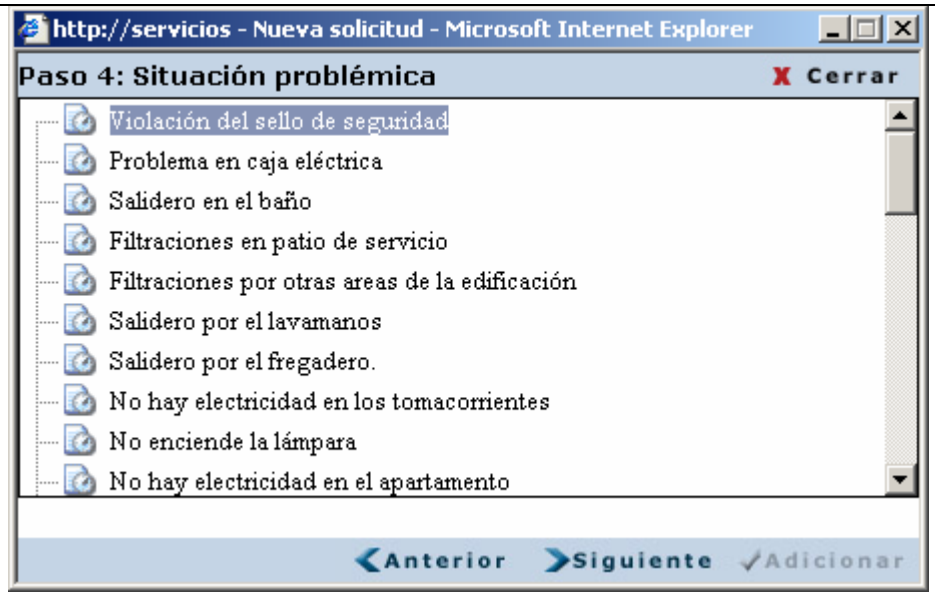
Pantalla 2



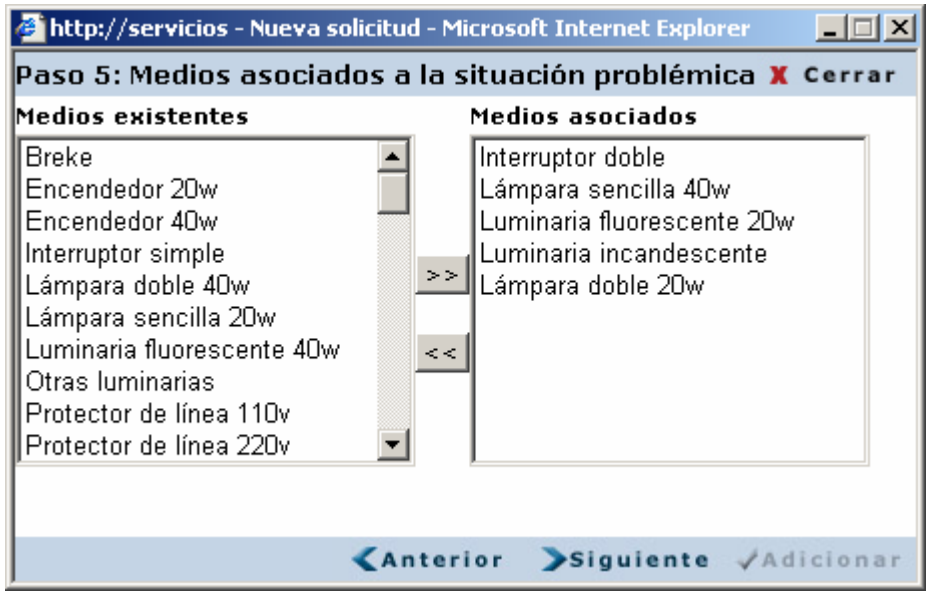
Pantalla 3



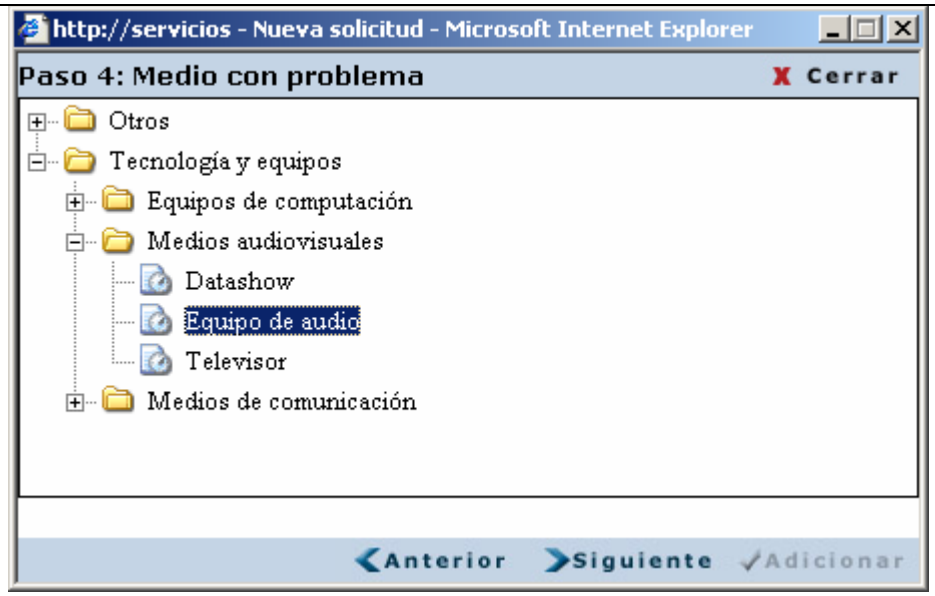
Pantalla 4



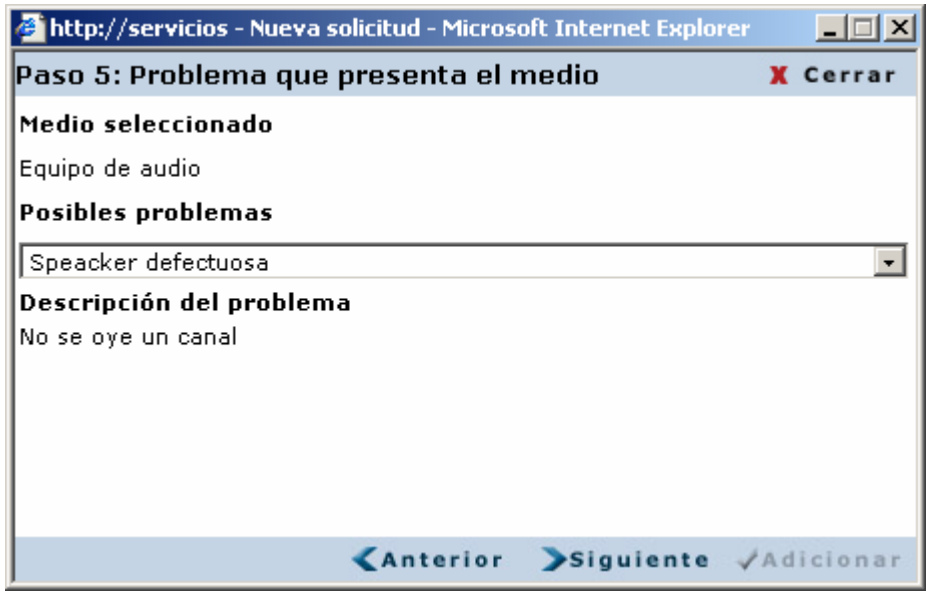
Pantalla 5



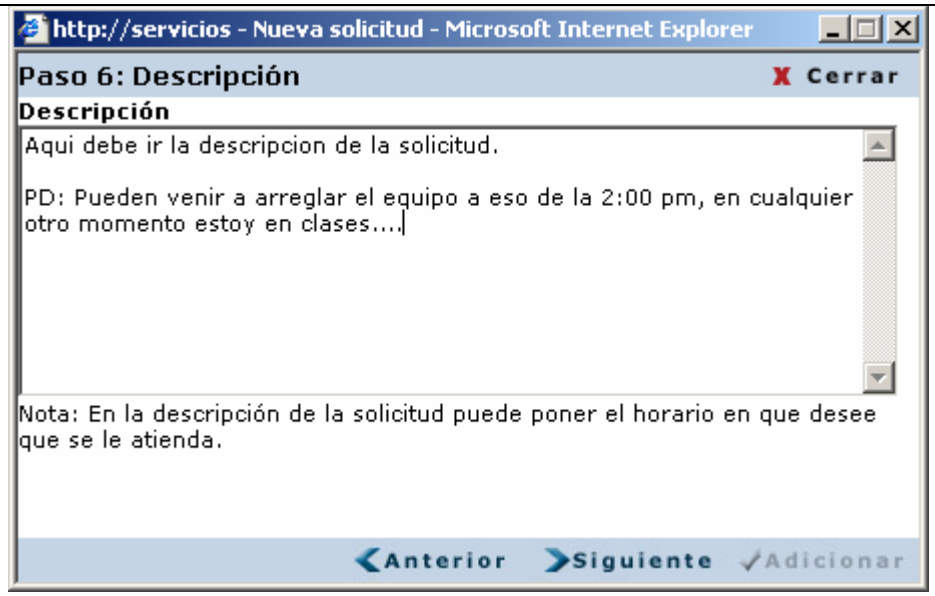
Pantalla 6



Pantalla 7

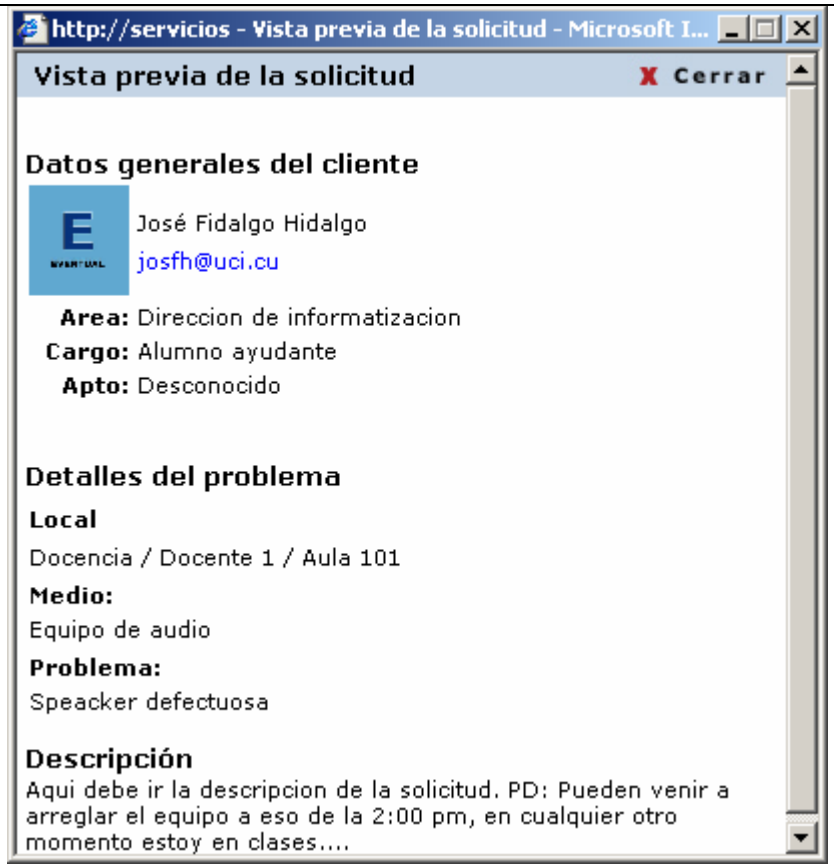


Pantalla 8



Pantalla 9

Pantalla 10

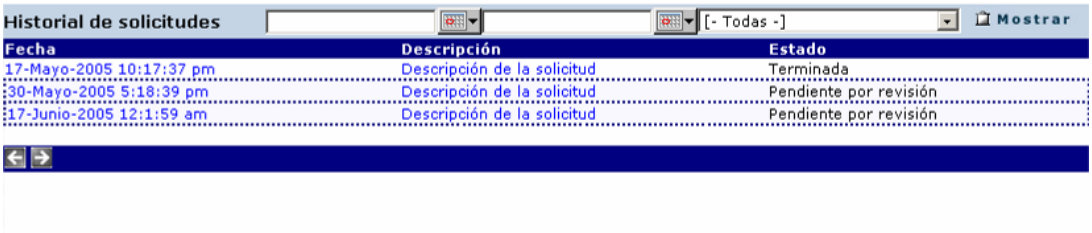


Pantalla 11

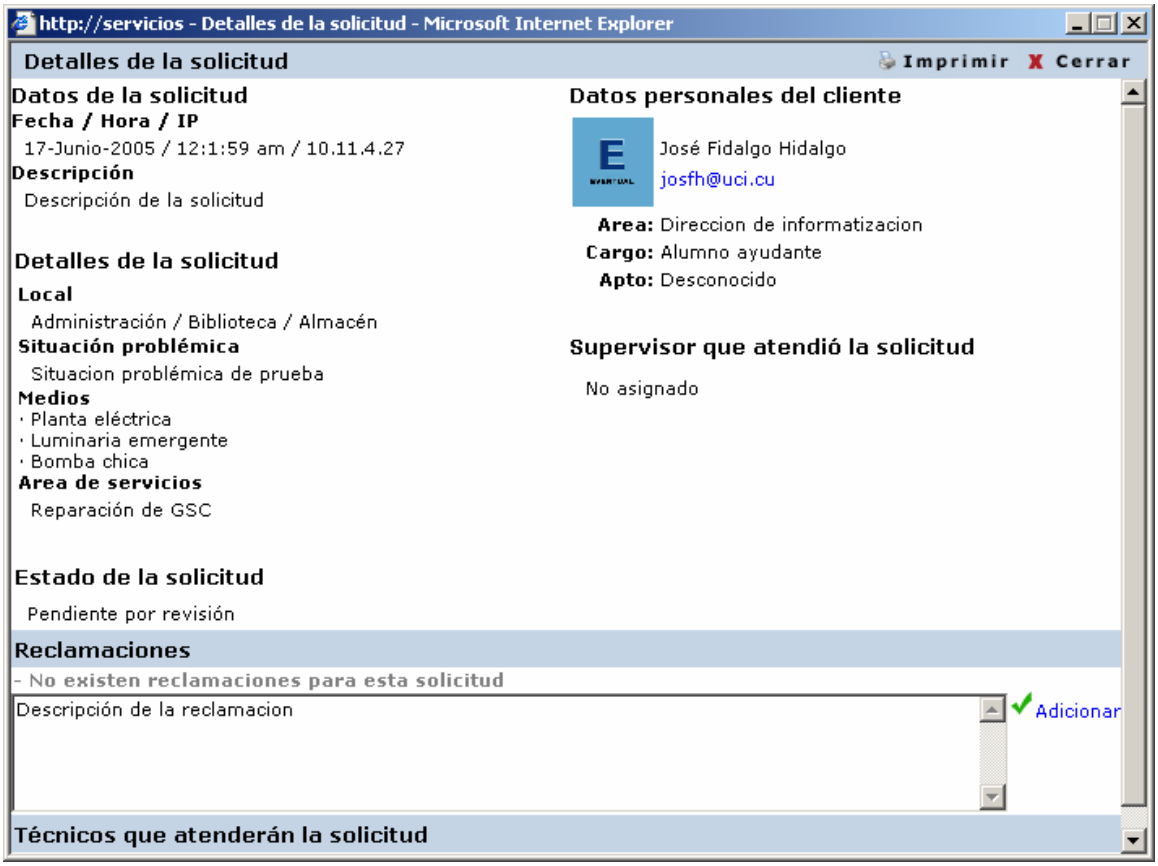
Caso de Uso:	**Ver solicitudes personales.**
Actor(es):	Usuario
Propósito:	Permitir que los usuarios del sistema puedan ver el estado en que se encuentran las solicitudes realizadas por ellos.
Resumen:	El caso de uso se inicia cuando el usuario selecciona la opción de historial de solicitudes, se muestra un listado de las solicitudes realizadas por el usuario y el estado en que se encuentran (***Pantalla 1***). En caso de que el usuario quiera ver los detalles de la solicitud, selecciona descripción y el sistema le muestra los detalles de la

	solicitud realizada (***Pantalla 2***)
Referencias:	R2
Precondiciones:	
Poscondiciones:	
Requisitos Especiales:	

Prototipo

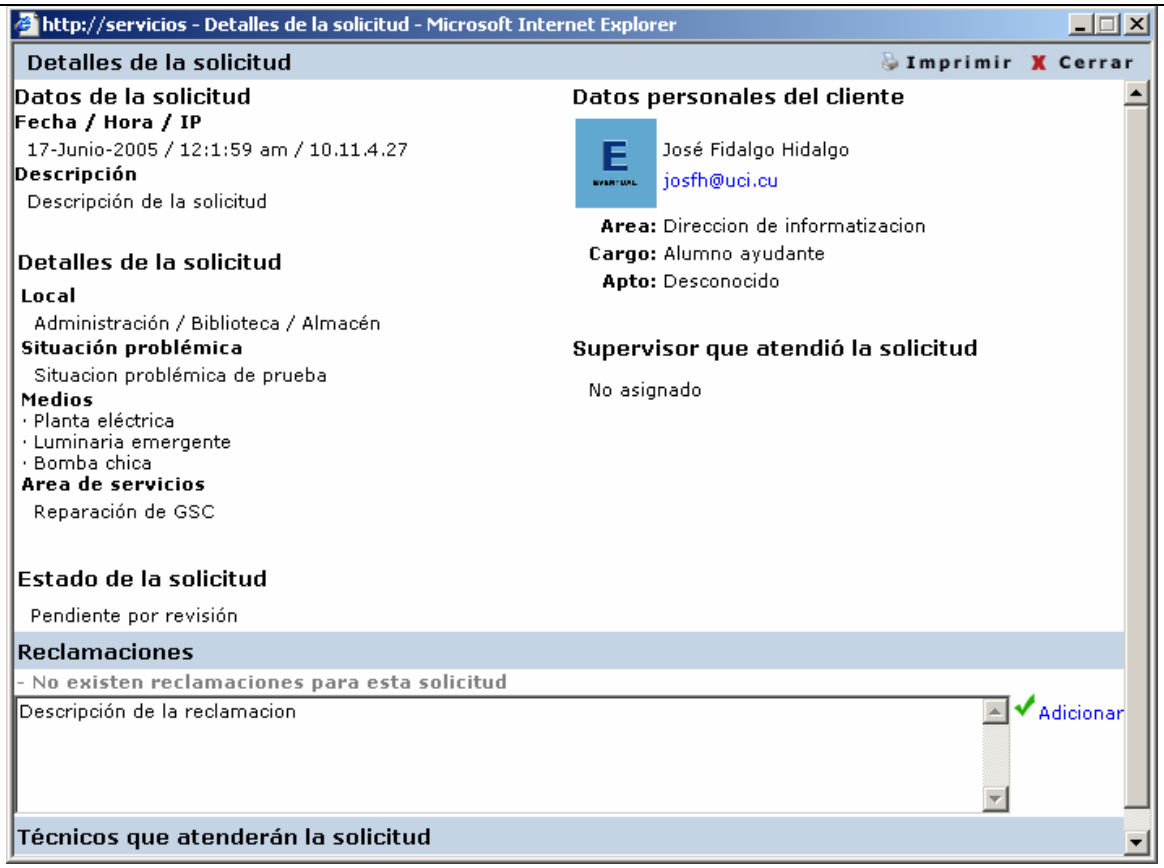


Pantalla 1



Pantalla 2

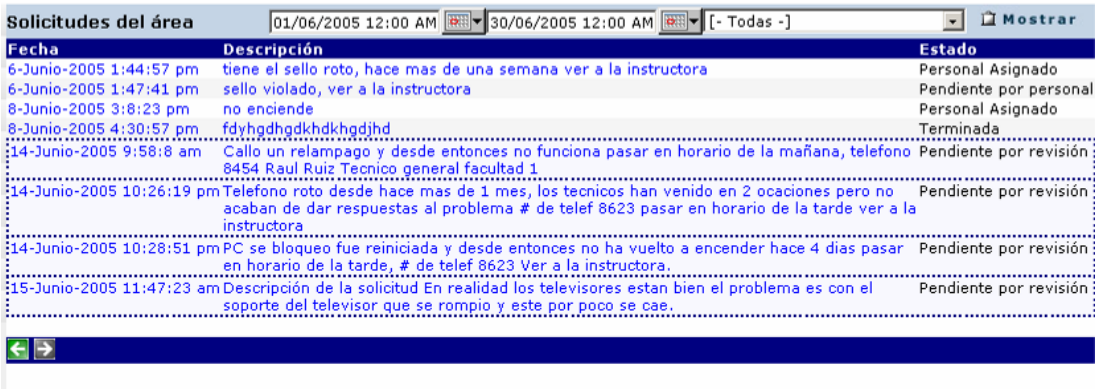
Caso de Uso:	**Realizar reclamación.**
Actor(es):	Usuario
Propósito:	Permite al usuario realizar una reclamación en caso de que la solicitud no haya sido atendida a tiempo o que no haya sido resuelto el problema.
Resumen:	El caso de uso se inicia cuando el usuario revisa los detalles de su solicitud, le puede agregar varias reclamaciones a la solicitud (***Pantalla 1***).
Referencias:	R2
Precondiciones:	Debe existir al menos una solicitud realizada por el usuario para poder realizar una reclamación (Ver caso de uso Realizar solicitud).
Poscondiciones:	
Requisitos Especiales:	
Prototipo	



Pantalla 1

Caso de Uso:	**Ver solicitudes del área de servicios.**
Actor(es):	Supervisores
Propósito:	Permitir a los supervisores ver el listado de las solicitudes que han sido hechas a su área de servicios.
Resumen:	El caso de uso se inicia cuando el supervisor selecciona la opción de responder a las solicitudes (***Pantalla 1***).
Referencias:	R13
Precondiciones:	Debe existir al menos una solicitud hecha al área que pertenece el supervisor (Ver caso de uso Realizar solicitud).

Poscondiciones:	
Requisitos Especiales:	
Prototipo	

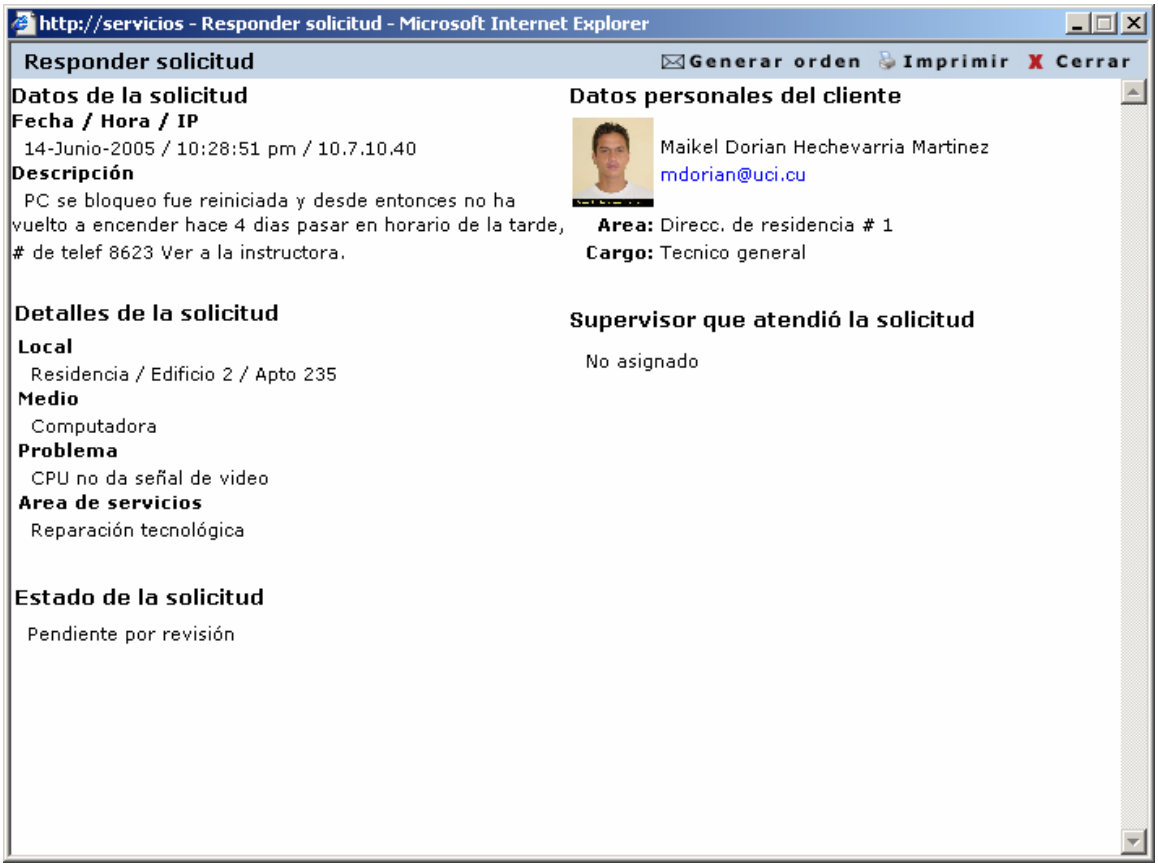


Pantalla 1

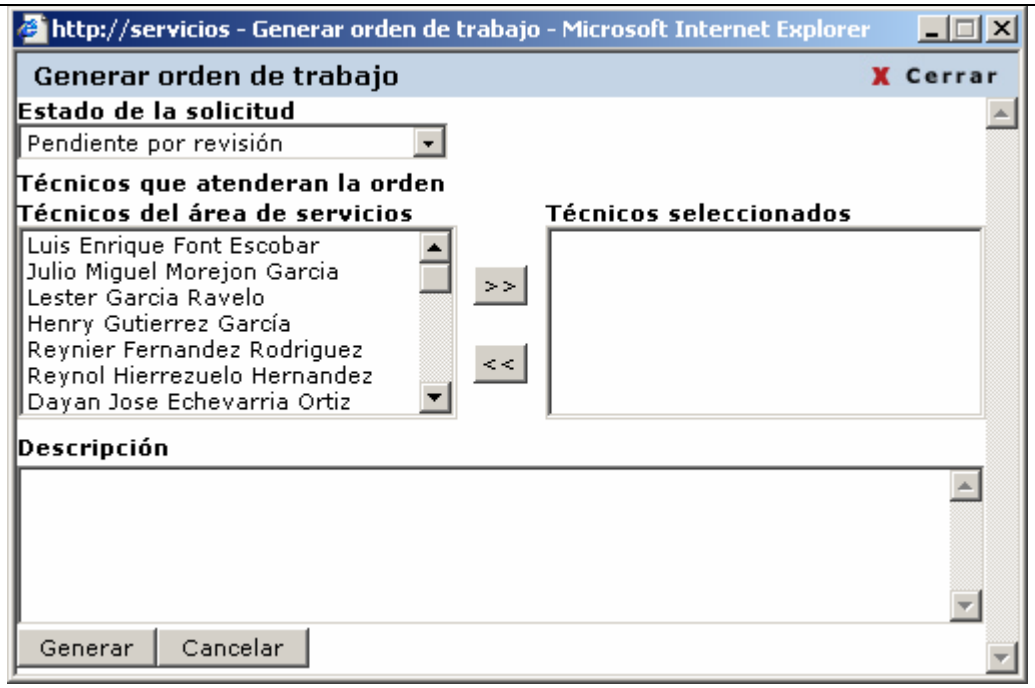
Caso de Uso:	**Actualizar órdenes de trabajo.**
Actor(es):	Supervisor
Propósito:	Permitir a los supervisores generar órdenes de trabajo y actualizarlas además.
Resumen:	El caso de uso se inicia cuando un supervisor ve la lista de solicitudes hechas al área de servicios, aquí para responder una solicitud (***Pantalla 1***), debe generar una orden de trabajo (***Pantalla 2***). Luego el supervisor puede ver la lista de órdenes de trabajo generadas (***Pantalla 3***), de aquí puede imprimirlas (***Pantalla 4***) o editarlas (***Pantalla 5***).
Referencias:	R3
Precondiciones:	Debe existir al menos una solicitud hecha al área que pertenece el supervisor (Ver caso de uso Realizar solicitud).

Poscondiciones:	
Requisitos Especiales:	

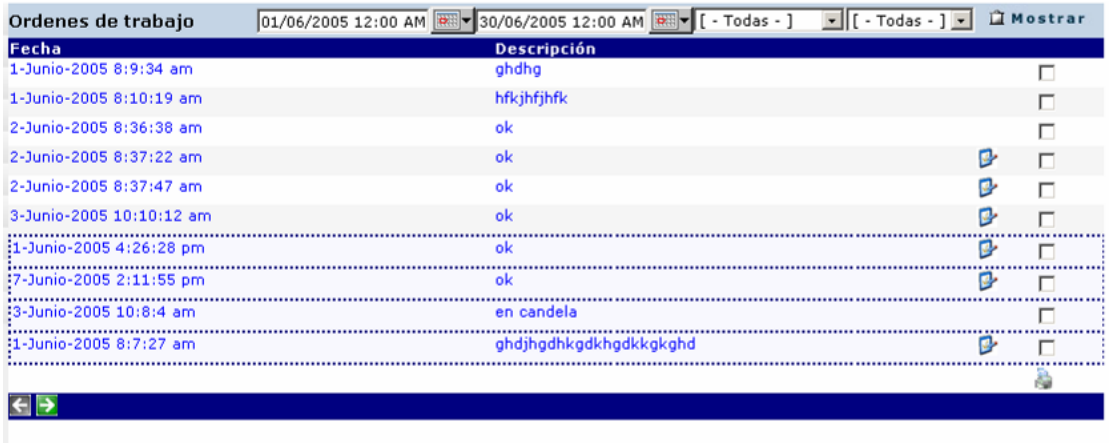
Prototipo



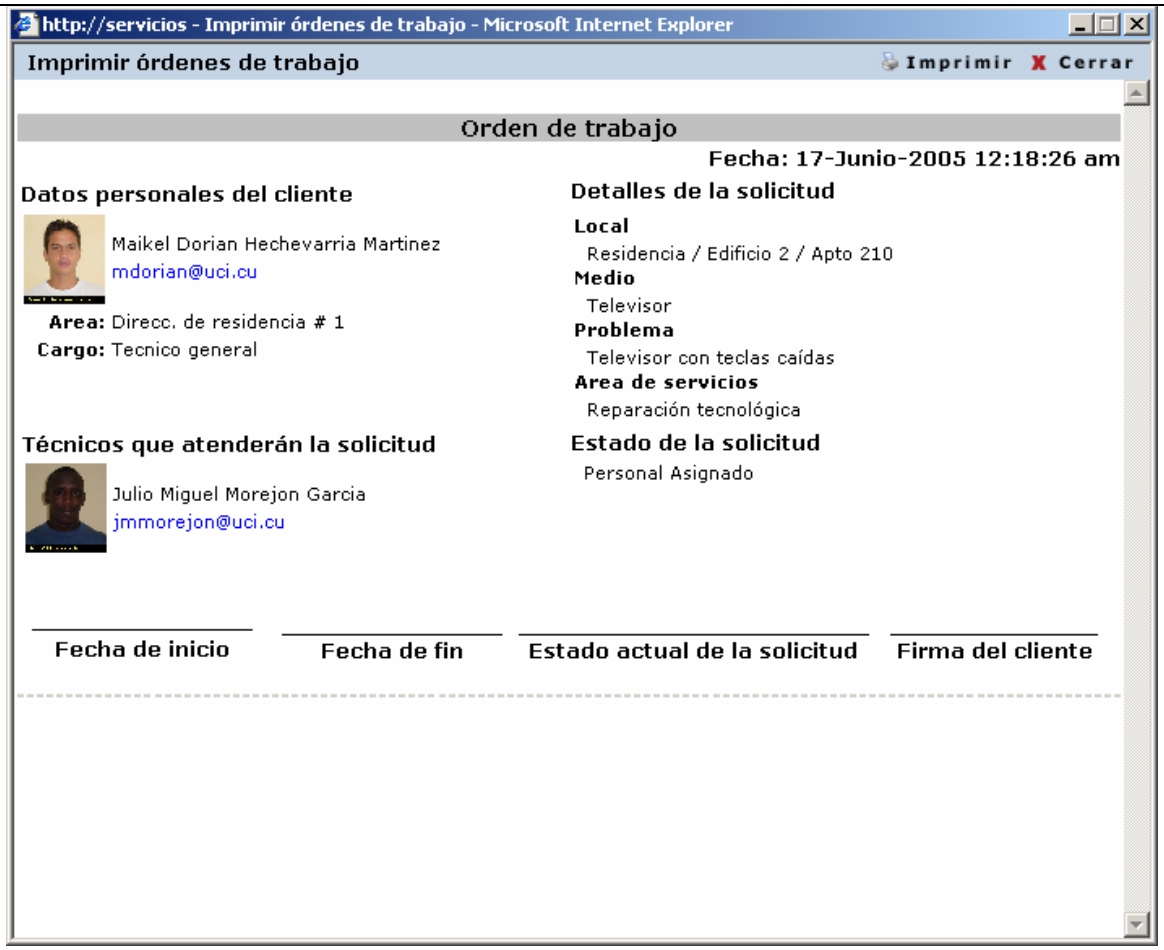
Pantalla 1



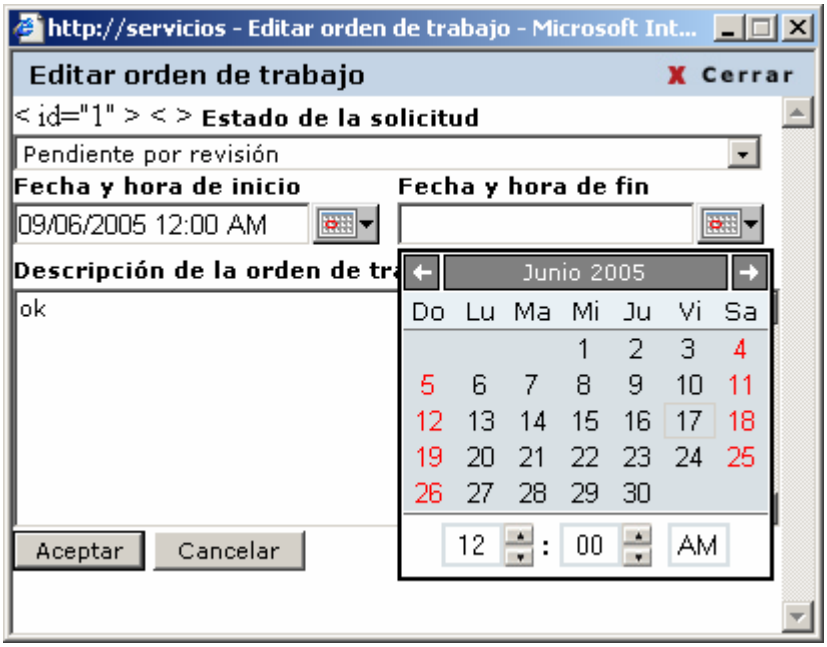
Pantalla 2



Pantalla 3


http://servicios - Imprimir órdenes de trabajo - Microsoft Internet Explorer
Imprimir órdenes de trabajo
Imprimir
Cerrar
Orden de trabajo
Fecha: 17-Junio-2005 12:18:26 am
Datos personales del cliente
Maikel Dorian Hechevarria Martinez
mdorian@uci.cu
Area: Direcc. de residencia # 1
Cargo: Tecnico general
Detalles de la solicitud
Local
Residencia / Edificio 2 / Apto 210
Medio
Televisor
Problema
Televisor con teclas caídas
Area de servicios
Reparación tecnológica
Técnicos que atenderán la solicitud
Julio Miguel Morejon Garcia
jmmorejon@uci.cu
Estado de la solicitud
Personal Asignado
Fecha de inicio
Fecha de fin
Estado actual de la solicitud
Firma del cliente

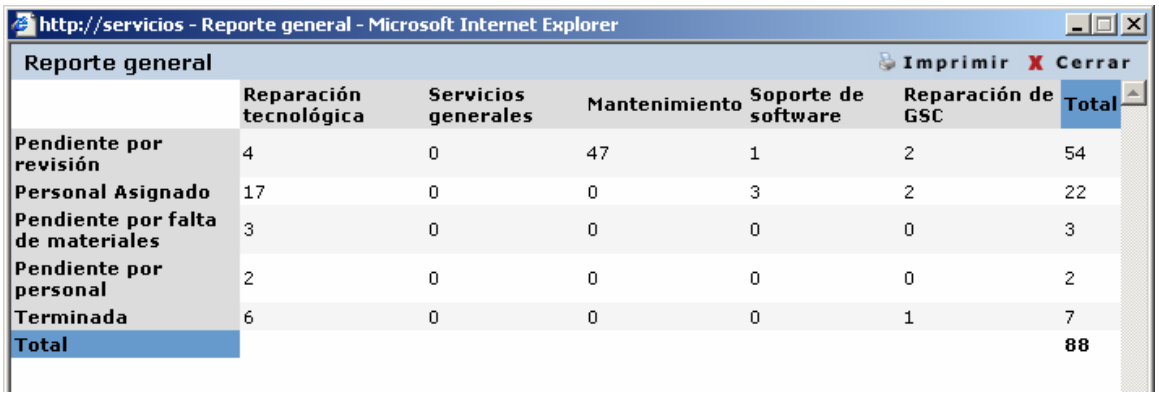
Pantalla 4


http://servicios - Editar orden de trabajo - Microsoft Int...
Editar orden de trabajo
Cerrar
< id="1" > < > Estado de la solicitud
Pendiente por revisión
Fecha y hora de inicio
09/06/2005 12:00 AM
Fecha y hora de fin
Descripción de la orden de tr
ok
Junio 2005
Do Lu Ma Mi Ju Vi Sa
1 2 3 4
5 6 7 8 9 10 11
12 13 14 15 16 17 18
19 20 21 22 23 24 25
26 27 28 29 30
12 : 00 AM
Aceptar
Cancelar

Pantalla 5

Caso de Uso:	**Ver reportes.**
Actor(es):	Directivos
Propósito:	Permitir a los directivos ver y recibir en todo momento reportes que informan el estado general del sistema y de los servicios comunitarios.
Resumen:	El caso de uso se inicia cuando un directivo escoge un reporte para verlo (***Pantalla 1***).
Referencias:	R14
Precondiciones:	Para poder ver los reportes, deben existir solicitudes (Ver caso de uso Realizar solicitud).
Poscondiciones:	
Requisitos Especiales:	

Prototipo



	Reparación tecnológica	Servicios generales	Mantenimiento	Soporte de software	Reparación de GSC	Total
Pendiente por revisión	4	0	47	1	2	54
Personal Asignado	17	0	0	3	2	22
Pendiente por falta de materiales	3	0	0	0	0	3
Pendiente por personal	2	0	0	0	0	2
Terminada	6	0	0	0	1	7
Total						88

Pantalla 1

3.7 Conclusiones.

En el presente capítulo se comenzó a desarrollar la propuesta de solución, obteniéndose a partir del análisis de los procesos del negocio, un listado con las funciones que debe tener el sistema, que se representaron mediante un Diagrama de Casos de Uso, y finalmente se describieron paso a paso todas las acciones de los actores del sistema y los casos de uso con los que estos interactúan. Pasada esta etapa, ahora se puede comenzar a construir la solución propuesta, tratando de que se cumpla con todos los requerimientos y las funciones que han sido consideradas necesarias en este capítulo.

CAPITULO 4

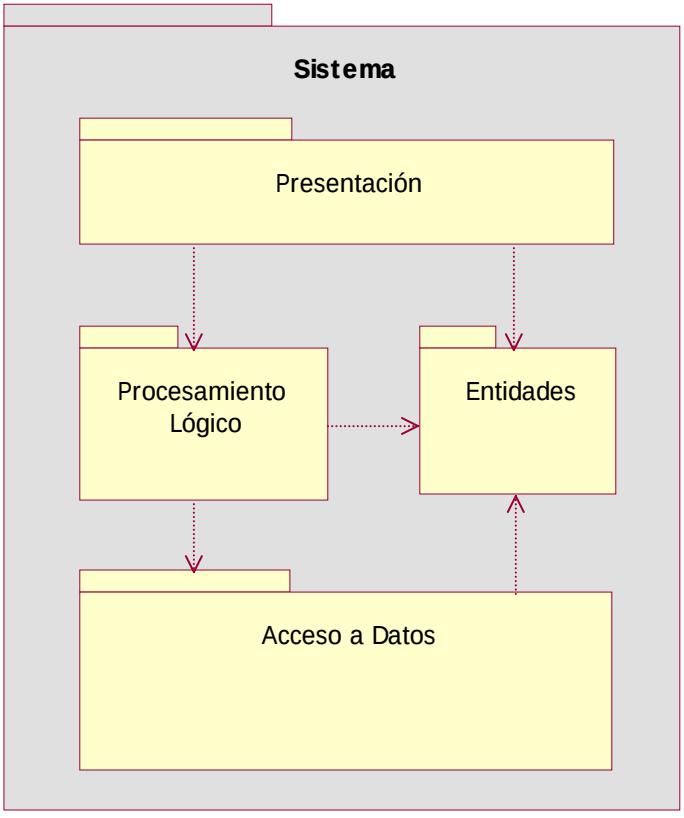
COSTRUCCION DE LA SOLUCION PROPUESTA

4.1 Introducción.

En este capítulo se modelan los artefactos que ayudan a manejar las complicaciones que implican la construcción de aplicaciones Web. Para ello los componentes de la aplicación se tratan como clases, y utilizando el lenguaje de modelación UML, se pueden presentar a través de diagramas de clases Web. Además se presenta el modelo de datos que es la base para construir finalmente la base de datos que soportará el trabajo del sistema. Finalmente después de modelar la lógica del negocio a través de las clases Web, se tratan los principios del diseño de la aplicación.

4.2 Diagrama de clases.

Para un mejor entendimiento de los diagramas de clases, estos se han dividido en paquetes, los cuales también han sido divididos en sub-paquetes.



Relación entre los paquetes de clases del sistema.

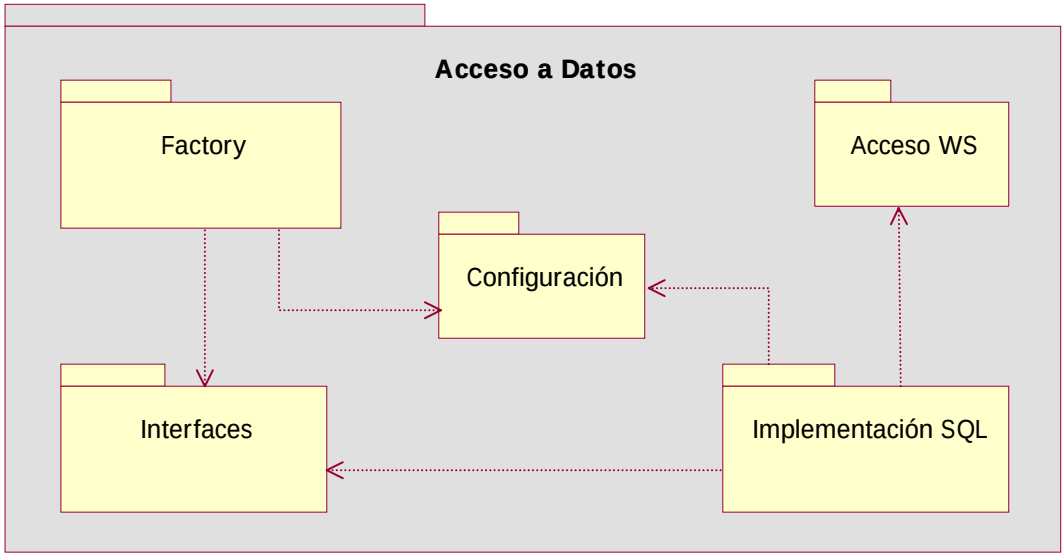
El paquete **Acceso a Datos** contiene las clases para hacer posible el acceso a todo tipo de datos manejados por la aplicación.

El paquete **Procesamiento Lógico** contiene las clases encargadas de manejar la lógica de negocio de la aplicación.

El paquete **Entidades** contiene las clases que representan entidades reales del dominio, no presentan comportamiento, solo propiedades, estas clases son usadas para pasar información entre las capas de la aplicación, por lo que es usada por todas.

El paquete **Presentación** contiene las clases controladoras y de presentación del sistema, en nuestro caso una interfaz web.

4.2.1 Paquete *Acceso a Datos.*



Relación entre los sub-paquetes del paquete Acceso a Datos.

El paquete **Acceso a Datos** se dividió en cinco subpaquetes: **Factory**, **Interfaces**, **Configuración**, **Implementación SQL** y **Acceso WS**.

El paquete **Configuración** contiene datos esenciales para la capa de ***Acceso a Datos***.

El paquete **Factory** contiene las clases puentes entre la capa de ***Acceso a Datos*** y las capas que usen esta.

El paquete **Interfaces** contiene las interfaces que definen los métodos que serán exportados por la capa de ***Acceso a Datos*** hacia otras capas que usen esta.

El paquete **Implementación SQL** contiene las clases para el acceso a la base de datos en *SQL Server*.

El paquete **Acceso WS** contiene las clases encargadas del acceso a los Servicios Web (Web Services), que son usados por el sistema en la recuperación de datos.

4.2.1.1 Paquete *Factory.*

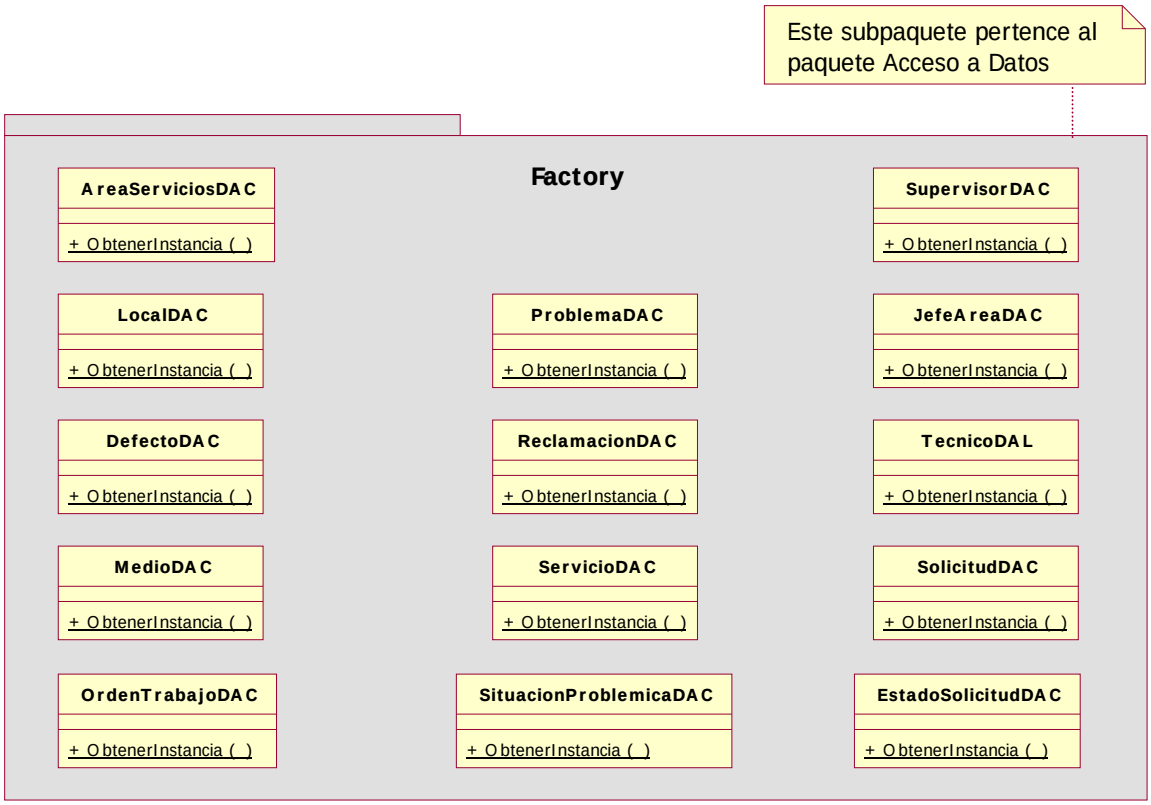


Diagrama de clases del paquete Factory.

4.2.1.2 Paquete *Interfaces.*

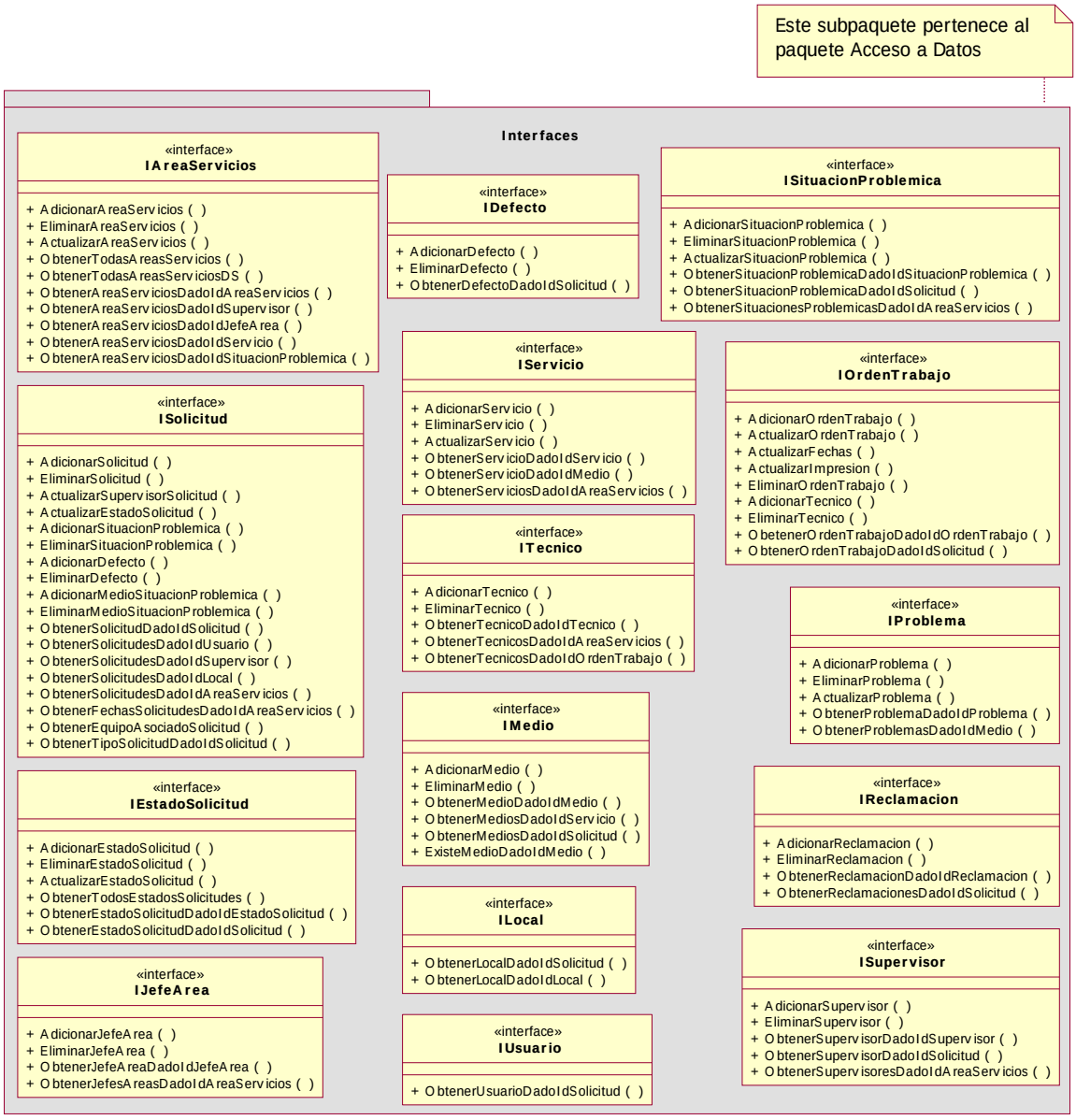


Diagrama de clases del paquete Interfaces.

4.2.1.3 Paquete *Configuración.*

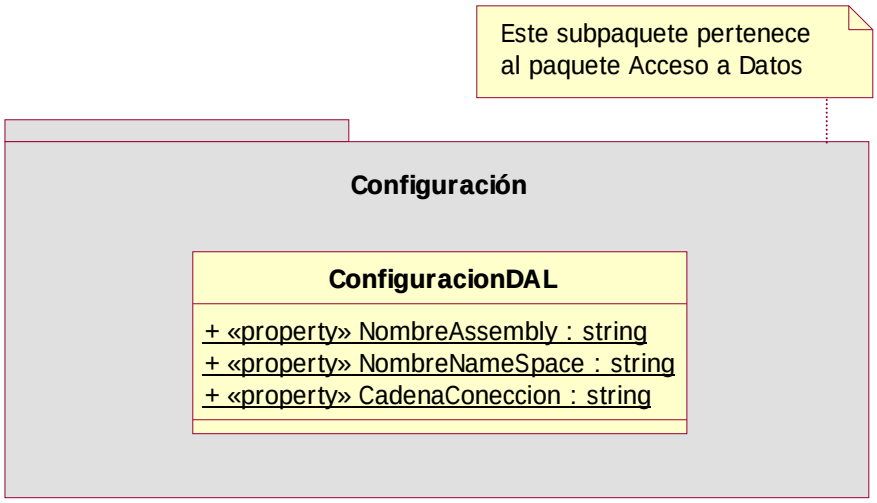


Diagrama de clases del paquete Configuración.

4.2.1.4 Paquete *Implementación SQL.*

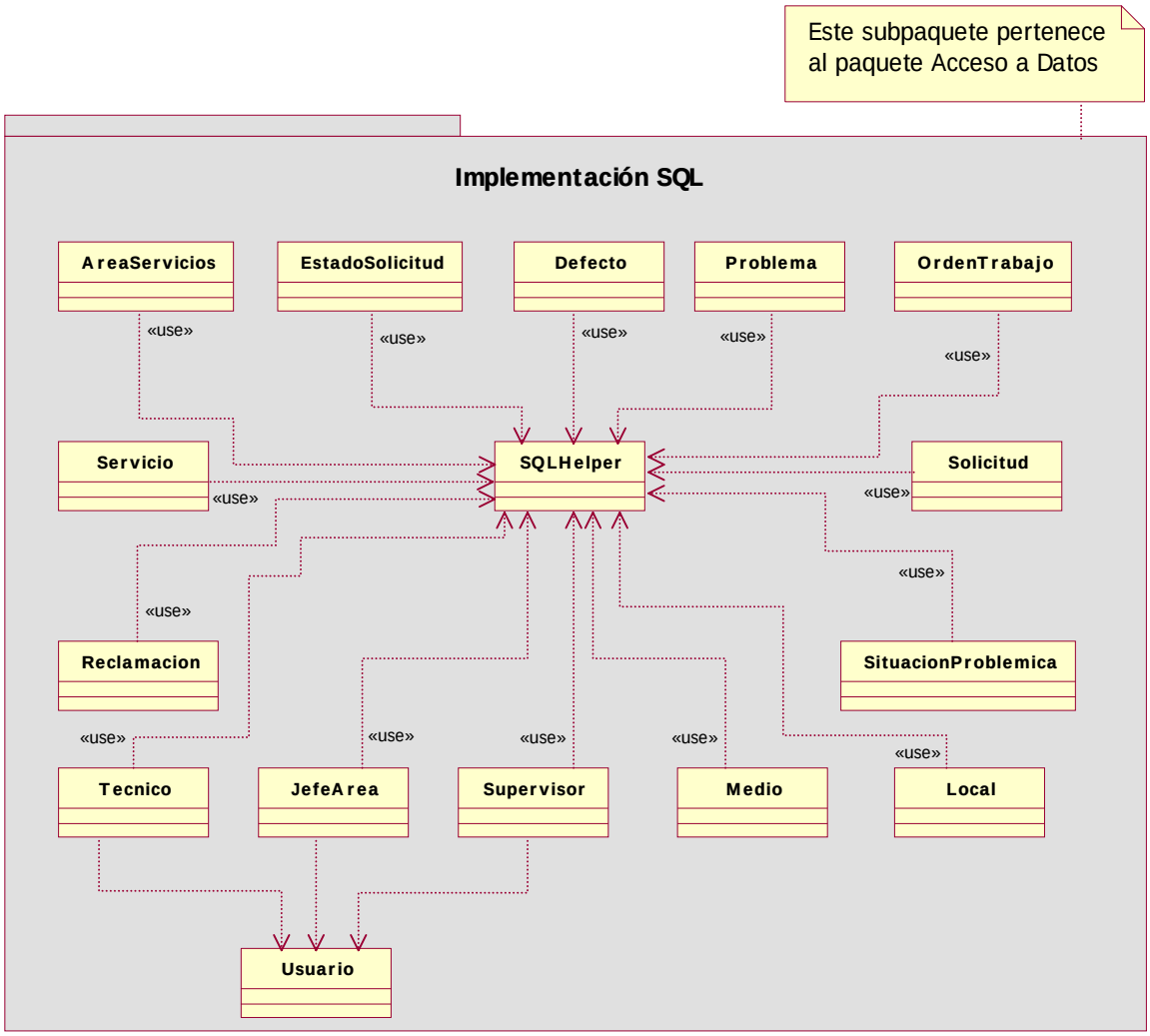
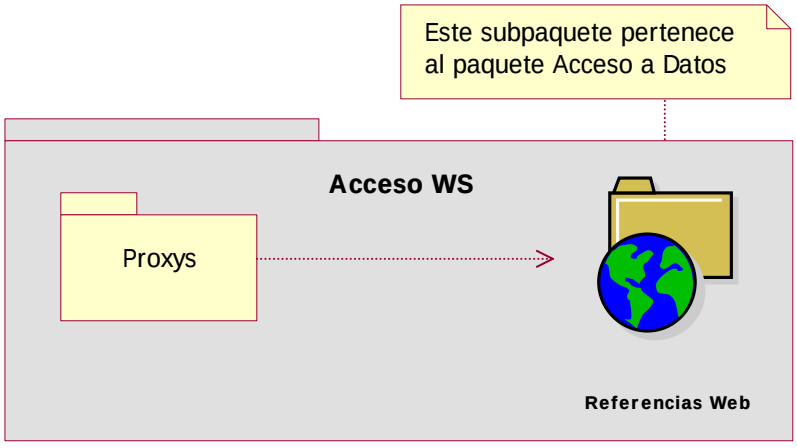


Diagrama de clases del paquete Implementación SQL.

4.2.1.5 Paquete *Acceso WS.*



Relación entre los sub-paquetes del paquete Acceso WS.

EL paquete **Acceso WS** ha sido dividido en dos subpaquetes; el paquete **Proxys** y el paquete **Referencias Web**.

El paquete **Proxys** contiene las clases puentes entre el sistema y las clases encargadas de acceder a los Servicios Web (Web Services).

El paquete **Referencias Web** contiene las clases encargadas de acceder a los Servicios Web (Web Services).

4.2.1.5.1 Paquete *Proxys*.

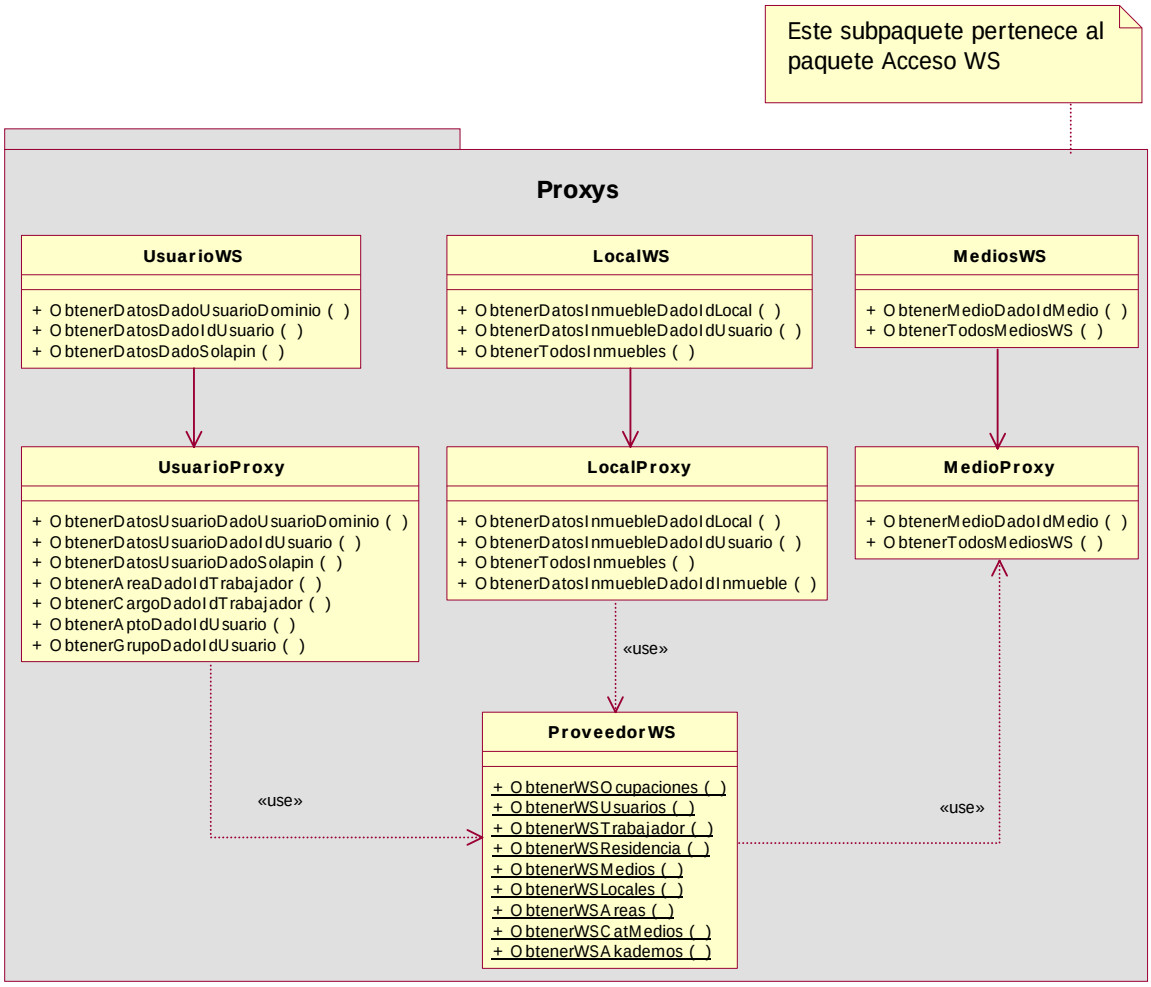


Diagrama de clases del paquete Proxys.

4.2.1.5.2 Paquete *Referencias Web.*

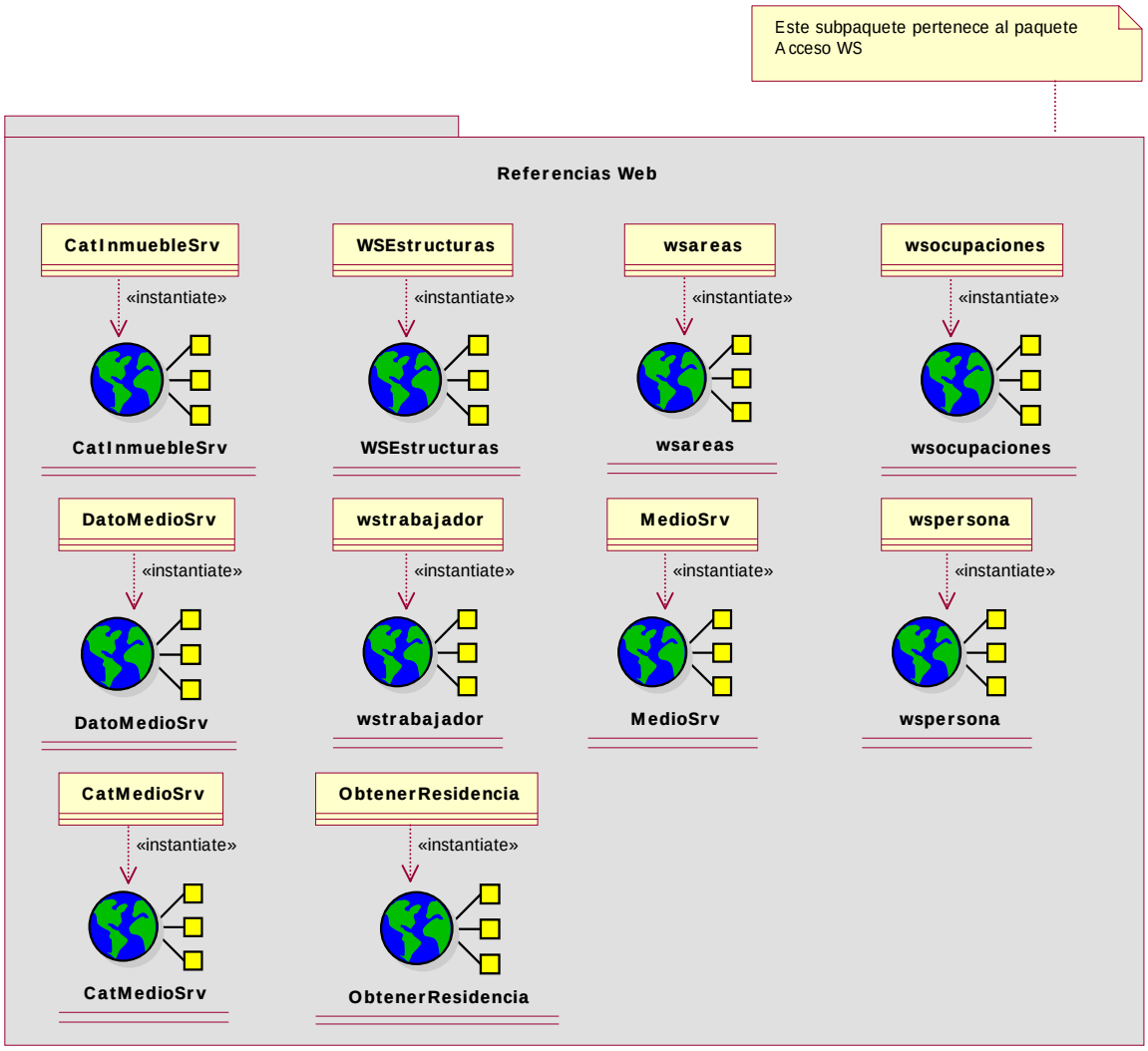


Diagrama de clases del paquete Referencias Web.

4.2.2 Paquete *Entidades.*

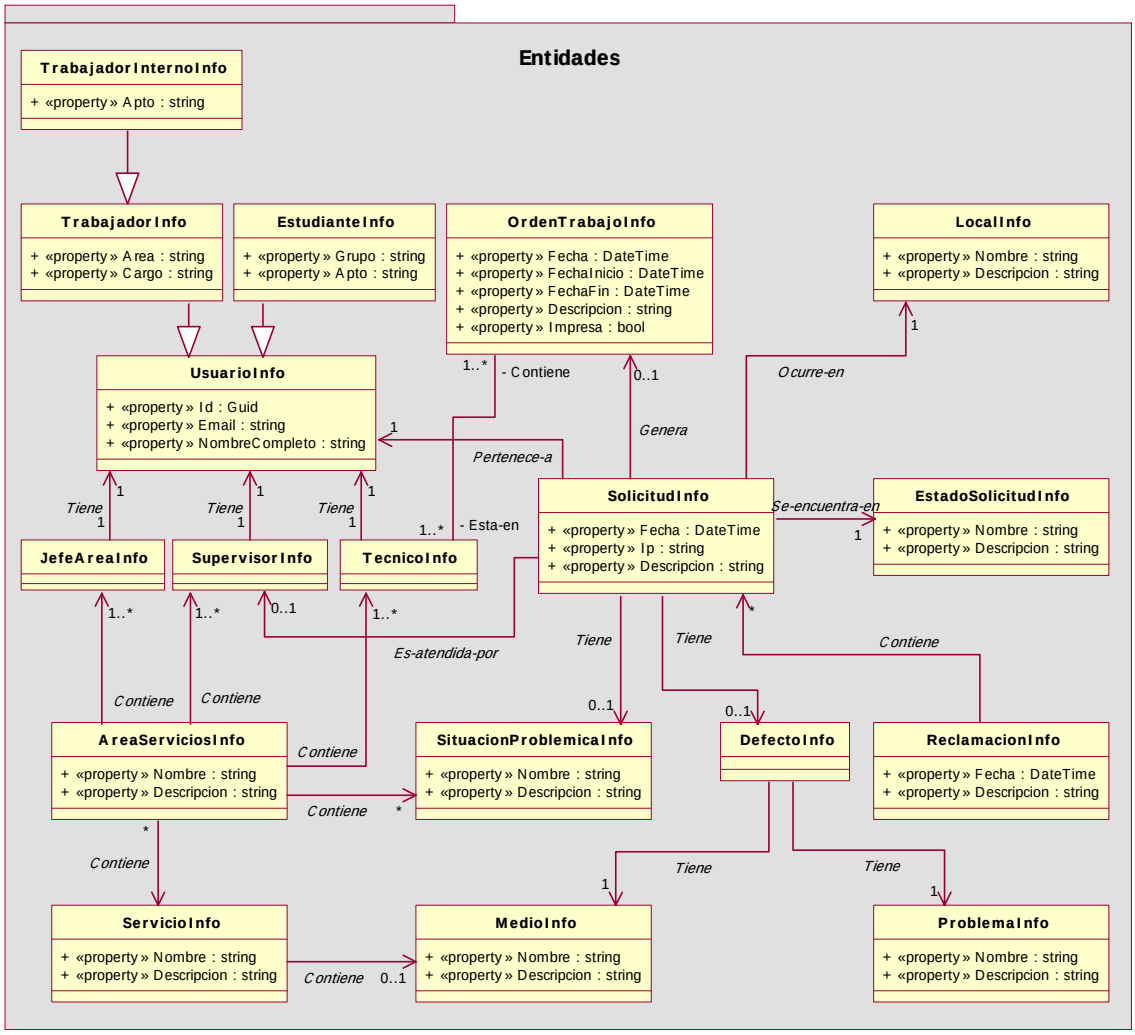


Diagrama de clases del paquete Entidades.

4.2.3 Paquete *Procesamiento Lógico*.

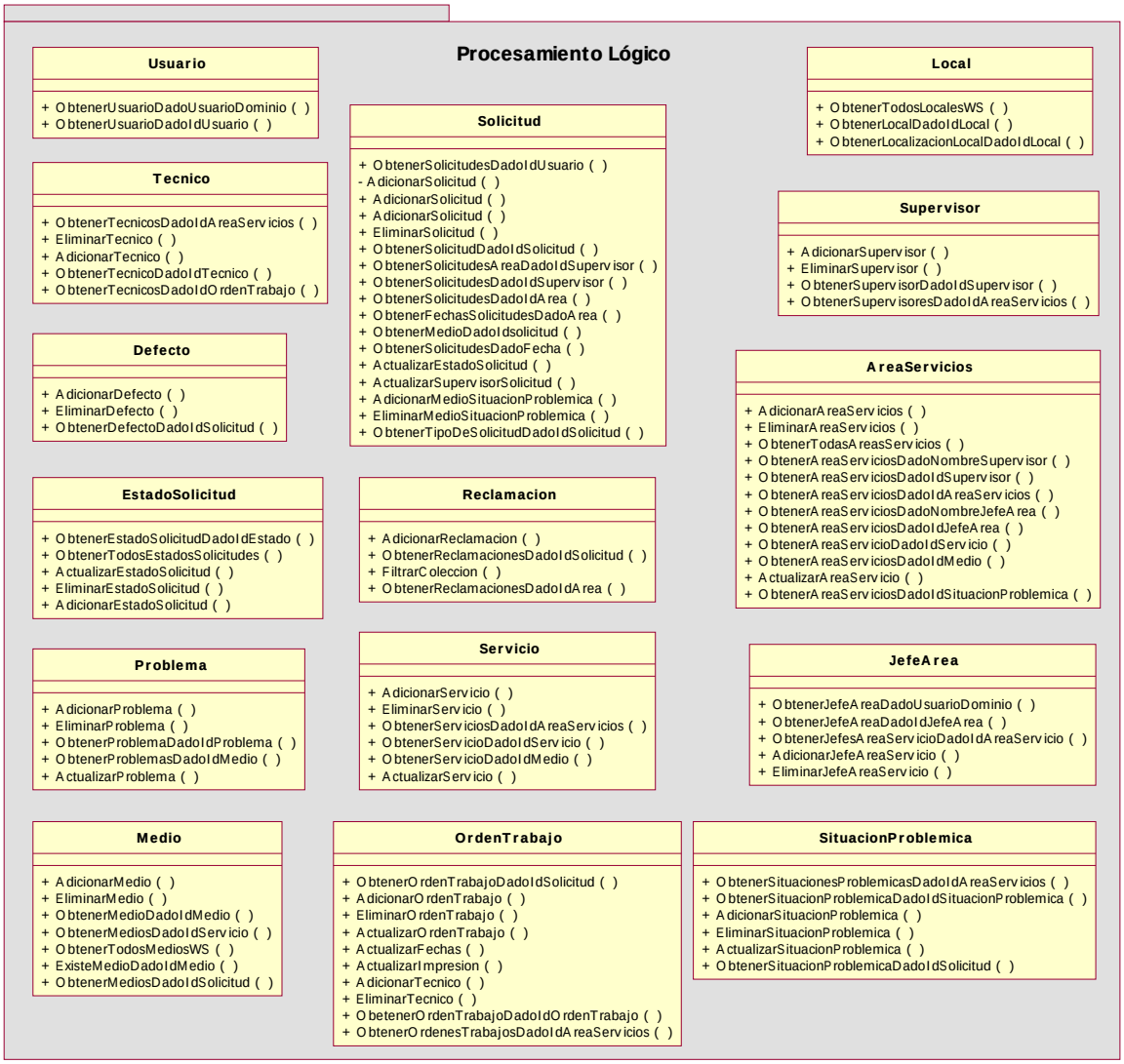
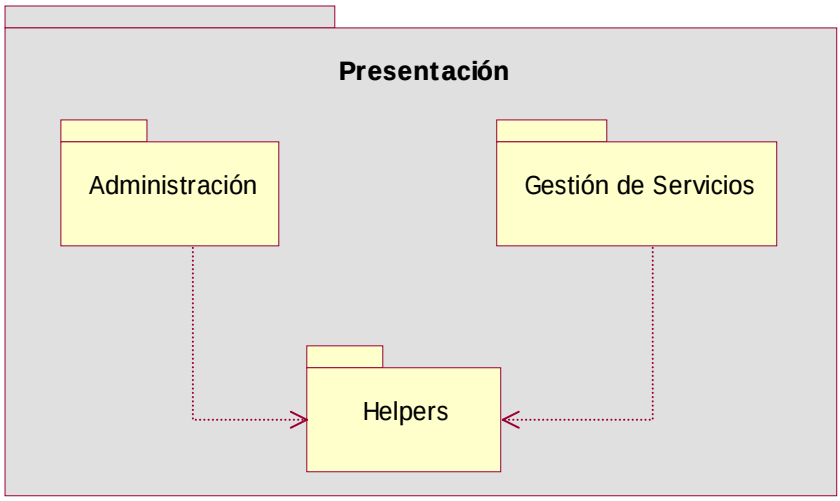


Diagrama de clases del paquete Procesamiento Lógico.

4.2.4 Paquete *Presentación.*



Relación entre los sub-paquetes del paquete Presentación.

El paquete **Presentación** se ha dividido en tres paquetes; el paquete **Administración**, **Gestión de Servicios** y **Helpers**.

El paquete **Administración** contiene las clases y las páginas web encargadas de la parte de administración de los datos esenciales del sistema.

El paquete **Gestión de Servicios** contiene las clases y las páginas web encargadas de la gestión de los servicios de la aplicación.

El paquete **Helpers** contiene las clases encargadas de realizar tareas importantes y que son usadas en gran parte de la capa de ***Presentación***.

4.2.4.1 Paquete *Administración*.

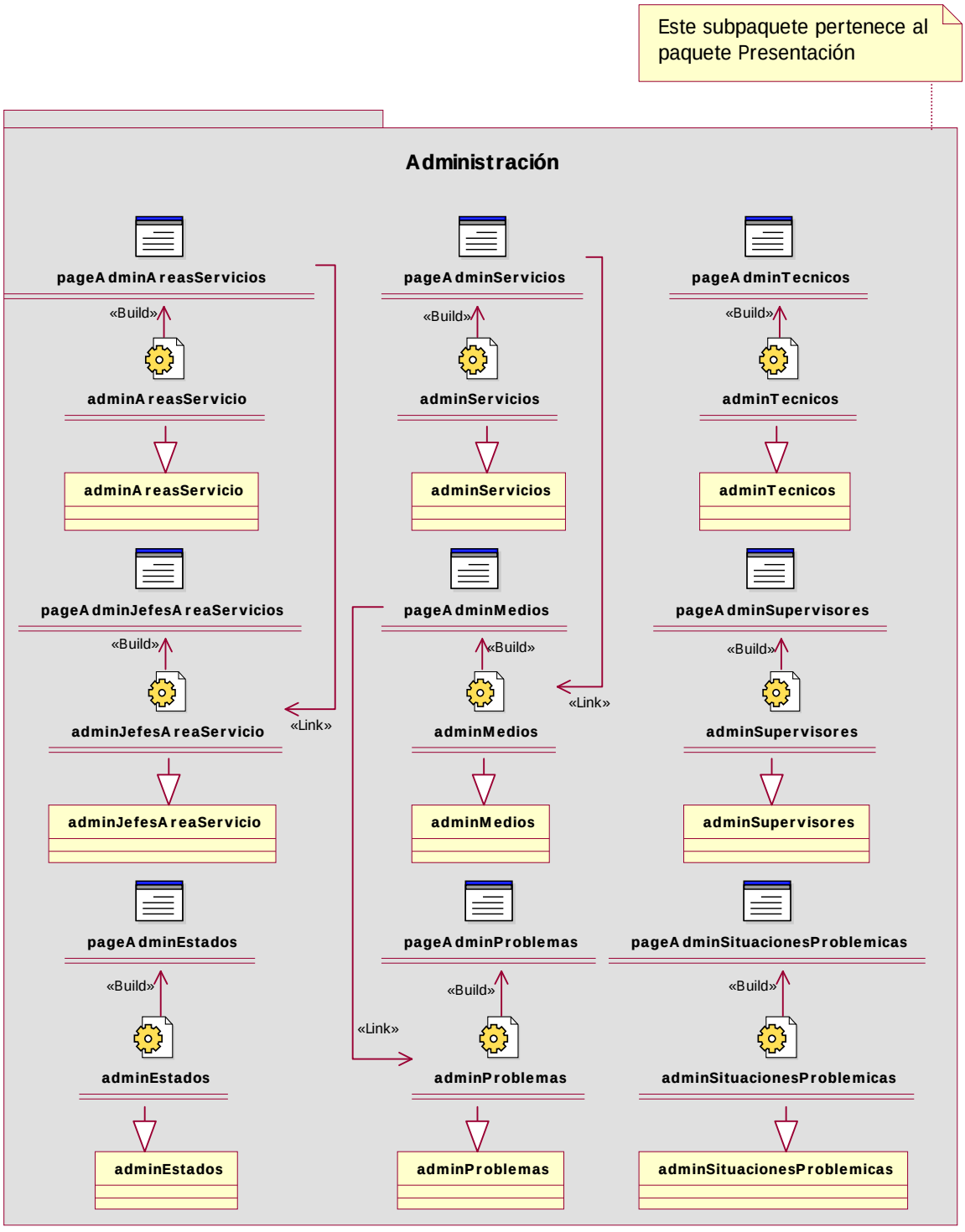


Diagrama de clase web del paquete de Administración.

4.2.4.2 Paquete *Gestión de Servicios*.

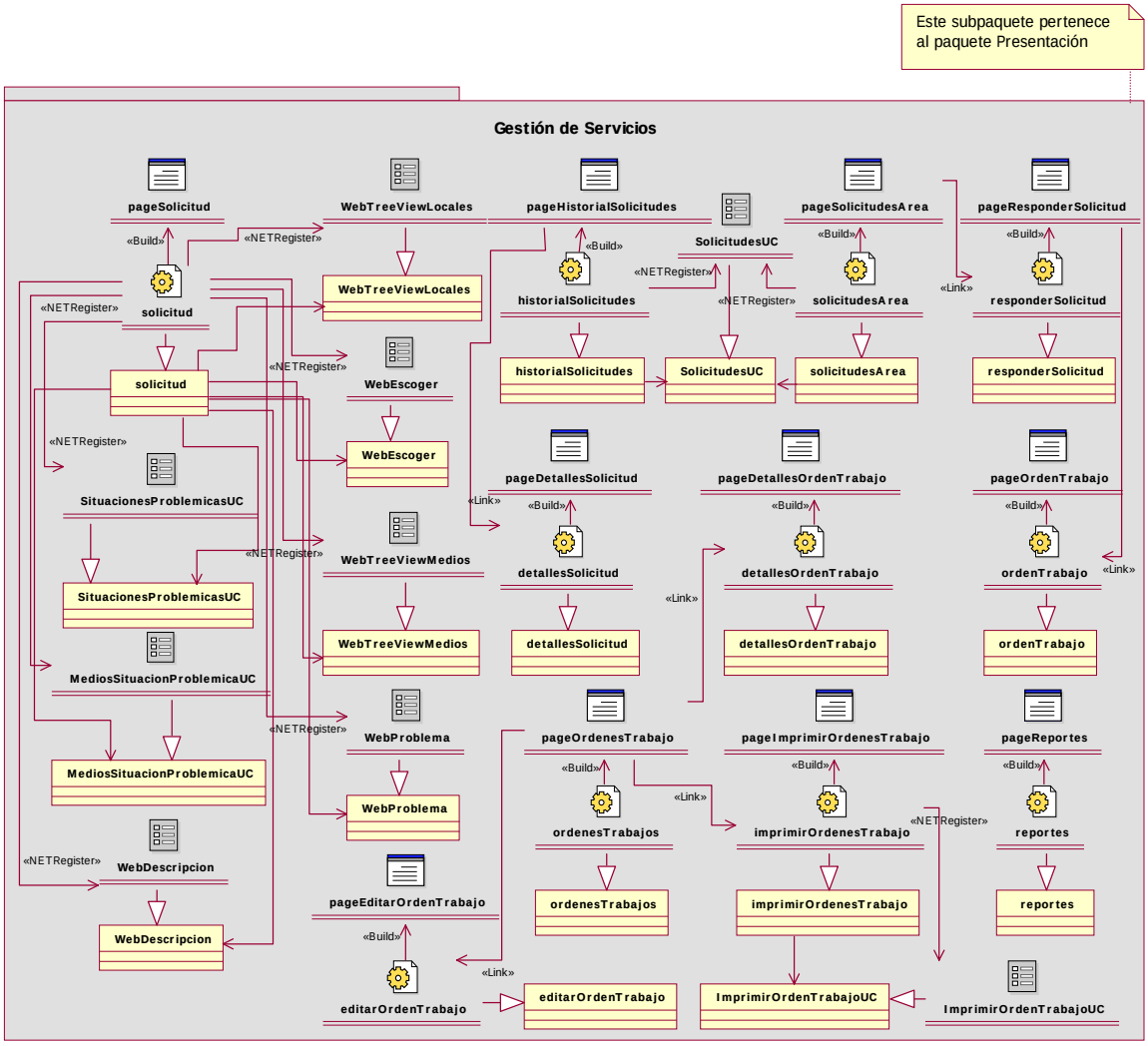


Diagrama de clases web del paquete Gestión de Servicios.

4.2.4.3 Paquete *Helpers*.

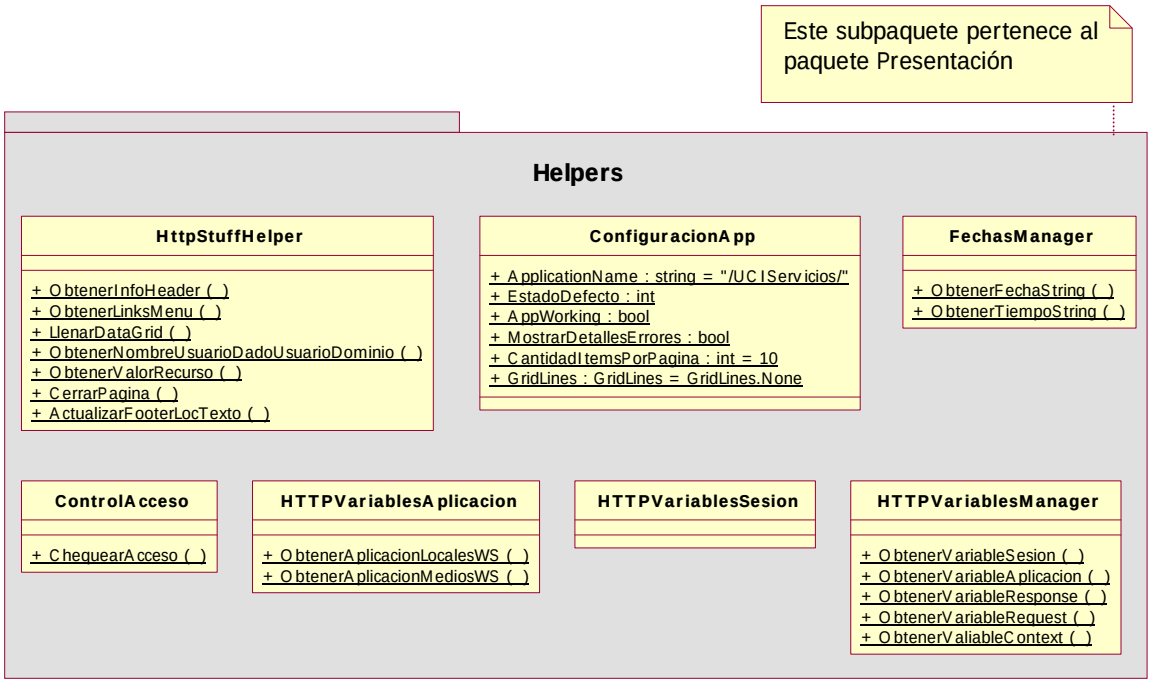


Diagrama de clases del paquete Helpers.

4.3 Diseño de la base de datos.

Para el diseño de la base de datos del sistema, se toman como bases el *Diagrama de clases persistente* y el *Modelo de datos,* los cuales están diseñados a partir de los diagramas de clases vistos en epígrafes anteriores.

4.3.1 Diagrama de clases persistente.

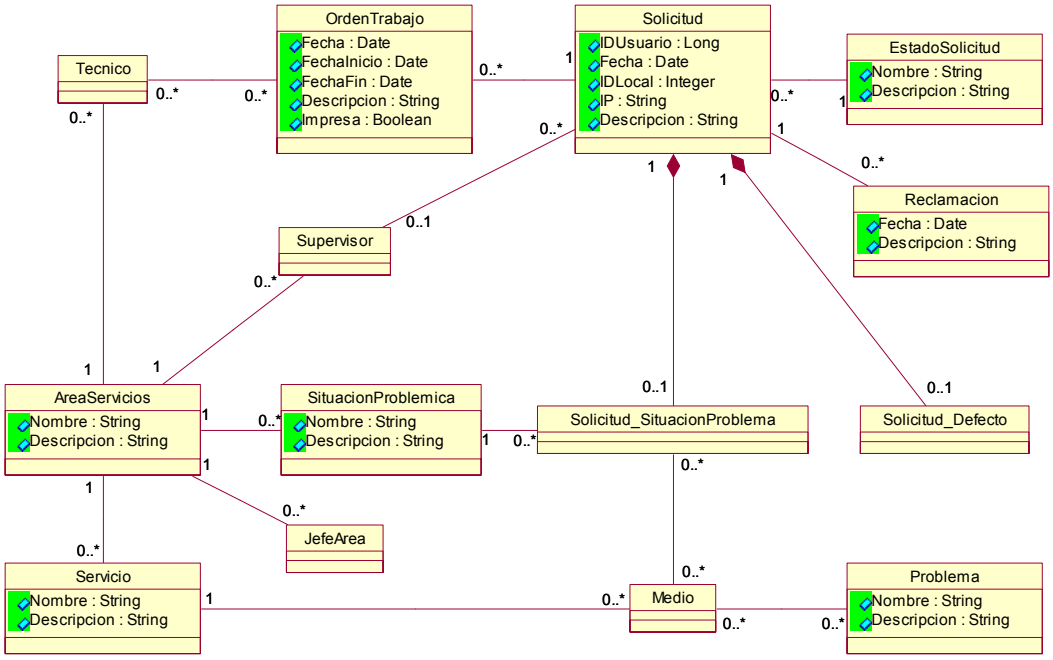
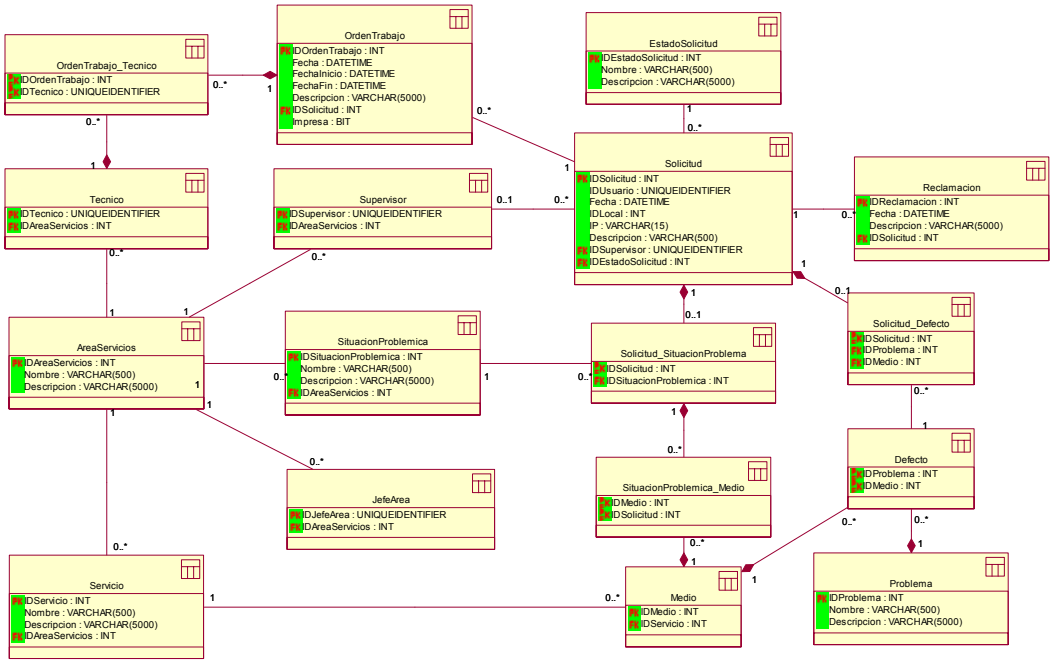


Diagrama de Clases Persistente.

4.3.2 Modelo de datos.



Modelo de datos.

4.4 Principios de diseño.

El diseño ha sido elaborado pensando en los usuarios finales, que serán: estudiantes, profesores y trabajadores, los cuales, no en todos los casos poseen conocimientos sobre computación, por tanto, se ha elegido una interfaz amigable e intuitiva. Se ha mantenido un diseño consistente en todas las páginas, para lograr que el usuario se sienta cómodo y logre acostumbrarse rápidamente a la aplicación.

4.4.1 Estándares en la interfaz de la aplicación.

Para la aplicación se definió un estándar para todas las páginas, existe una principal, la cual tiene opciones comunes para todos los tipos de usuario. Se usaron principalmente colores azules, grises y blancos con el fin de dar claridad al diseño (*Ver anexo I*).

Se utiliza el color rojo para resaltar los errores de campos requeridos y para los mensajes de operaciones no válidas.

De forma general se realizan varias operaciones por página, de forma que el usuario no tenga que moverse tanto dentro de la aplicación, ya que acabaría perdiendo el interés de trabajar con la aplicación.

4.4.2 Formatos de reportes.

El sistema brinda reportes en forma de tablas. En estos se da la posibilidad de filtrar los resultados mostrados por uno o más campos, de manera que el usuario pueda obtener de forma sencilla y sin dificultad la información que desea ver. Las filas de los reportes son de color alterno con el objetivo de facilitar la lectura del mismo. Los colores de los reportes fueron escogidos igualmente para que permitan una cómoda lectura de este. Los reportes además permiten paginado, por lo que se muestran un número limitado de registros, permitiendo moverse adelante y hacia atrás (*Ver figura 4.2*).

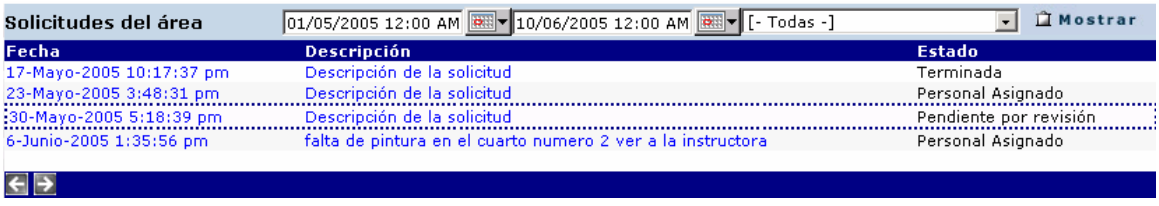


Figura 4.1 *Ejemplo de reporte en forma de tabla*.

Además el sistema de reportes se mostrará en forma de gráfico, para un mejor entendimiento de los reportes, y con el objetivo de se usados en la toma de decisiones por los directivos de la universidad.

4.4.3 Concepción general de la ayuda.

La ayuda se muestra como parte del menú principal de cada página de la aplicación. Además existe un manual de usuario, disponible para descargar desde cualquier lugar de la universidad, el cual contiene la lista de preguntas frecuentes con sus respuestas.

4.4.4 Tratamiento de excepciones.

Con el objetivo de prevenir los errores por parte del usuario, solo se le piden los datos necesarios, además se verifica la integridad de los mismos antes de ser almacenados para evitar la inconsistencia. Los campos obligatorios son verificados, para ello el sistema usa validaciones, tanto en la parte del cliente, como en la parte del servidor. Cuando ocurren errores, estos se muestran en color rojo según el diseño original de la aplicación.

En el caso de las excepciones en tiempo de ejecución, estas son capturadas por el sistema y almacenadas en una base de datos, para su posterior revisión por parte de los desarrolladores del sistema, con el objetivo de mejorar la aplicación.

4.5 Estándares de codificación.

Una parte fundamental cuando se va a escribir el código de una aplicación es el estándar de codificación a usar. La buena selección de los estándares de codificación trae consigo una serie de ventajas, como son:

- Reducción de errores.
- Código claro y comprensible.
- Buena comunicación con el equipo de desarrollo.
- Fácil mantenimiento del software.

En esta aplicación se ha utilizado el estándar de codificación “Pascal Case” que principalmente tiene que ver con la capitalización de los caracteres. Así mismo, se ha

seguido el estilo de codificación propuesto por Microsoft para programar con C# (*Ver figura 4.3*).

```
...
public void AdicionarServicio ( string nombre, string descripcion, int idAreaServicios )
{
    SqlParameter [] parameters = new SqlParameter []
        {
            new SqlParameter ( Parametro.DadoColumna ( ServicioCSP.COL_NOMBRE ), SqlDbType.VarChar, 500 ),
            new SqlParameter ( Parametro.DadoColumna ( ServicioCSP.COL_DESCRIPCION ), SqlDbType.VarChar, 5000 ),
            new SqlParameter ( Parametro.DadoColumna ( AreaServiciosCSP.COL_ID_AREA_SERVICIOS ), SqlDbType.Int )
        };
    parameters [ 0 ].Value = nombre;
    parameters [ 1 ].Value = descripcion;
    parameters [ 2 ].Value = idAreaServicios;
    SQLHelper.ExecuteNonQuery ( ServicioCSP.SERVICIO_INS, CommandType.StoredProcedure, parameters );
}
...
```

Figura 4.2 *Ejemplo de codificación.*

En el diseño de la base de datos se ha seguido el mismo estándar de codificación, las tablas se han nombrado igual que la entidad que almacenan y los campos igual que las propiedades de las entidades (*Ver figura 4.4*).

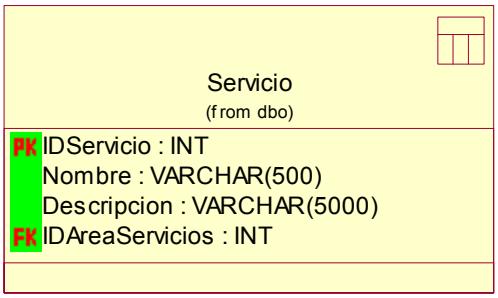


Figura 4.3 *Ejemplo de la codificación utilizada en las tablas.*

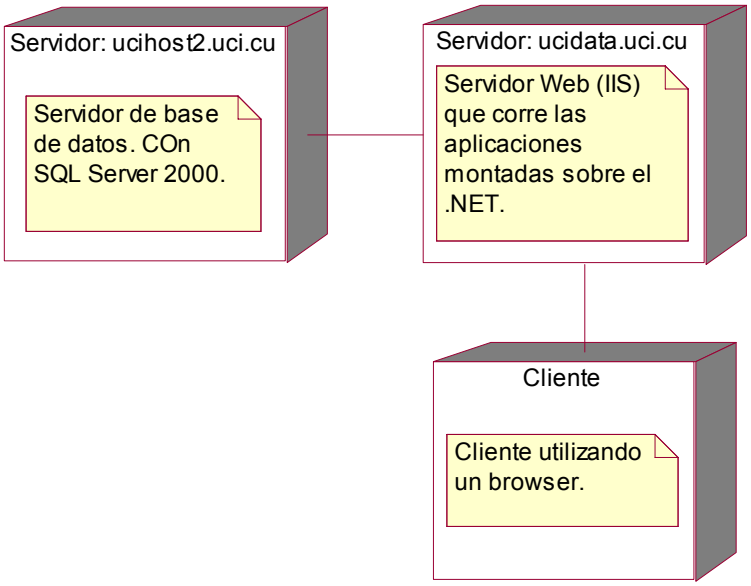
En el área del diseño también se ha tenido especial cuidado al nombrar los componentes usados:

- Campos de edición : **TextBox**Nombre
- Campos de textos : **Label**Nombre
- Botones de acción : **Button**Nombre

4.6 Modelo de despliegue.

Un diagrama de despliegue muestra la configuración de los nodos que participan en la ejecución y de los componentes que residen en ellos. En nuestro caso la base de datos en SQL Server 2000 se encuentra en el servidor **ucihost2.uci.cu**, por otro lado el servidor web se encuentra en **ucidata.uci.cu**; mediante el diagrama de despliegue

podemos ver como se encuentran relacionados físicamente los componentes de la aplicación.



Modelo de despliegue.

4.7 Modelo de implementación.

Durante la fase de implementación, entre los modelos que se describen, tenemos el Modelo de implementación. Este modelo describe los componentes y la organización de acuerdo a los nodos, así como las dependencias de compilación entre ellos.

4.7.1 Diagramas de implementación por paquetes.

4.7.1.1 Paquete *Acceso a Datos*.

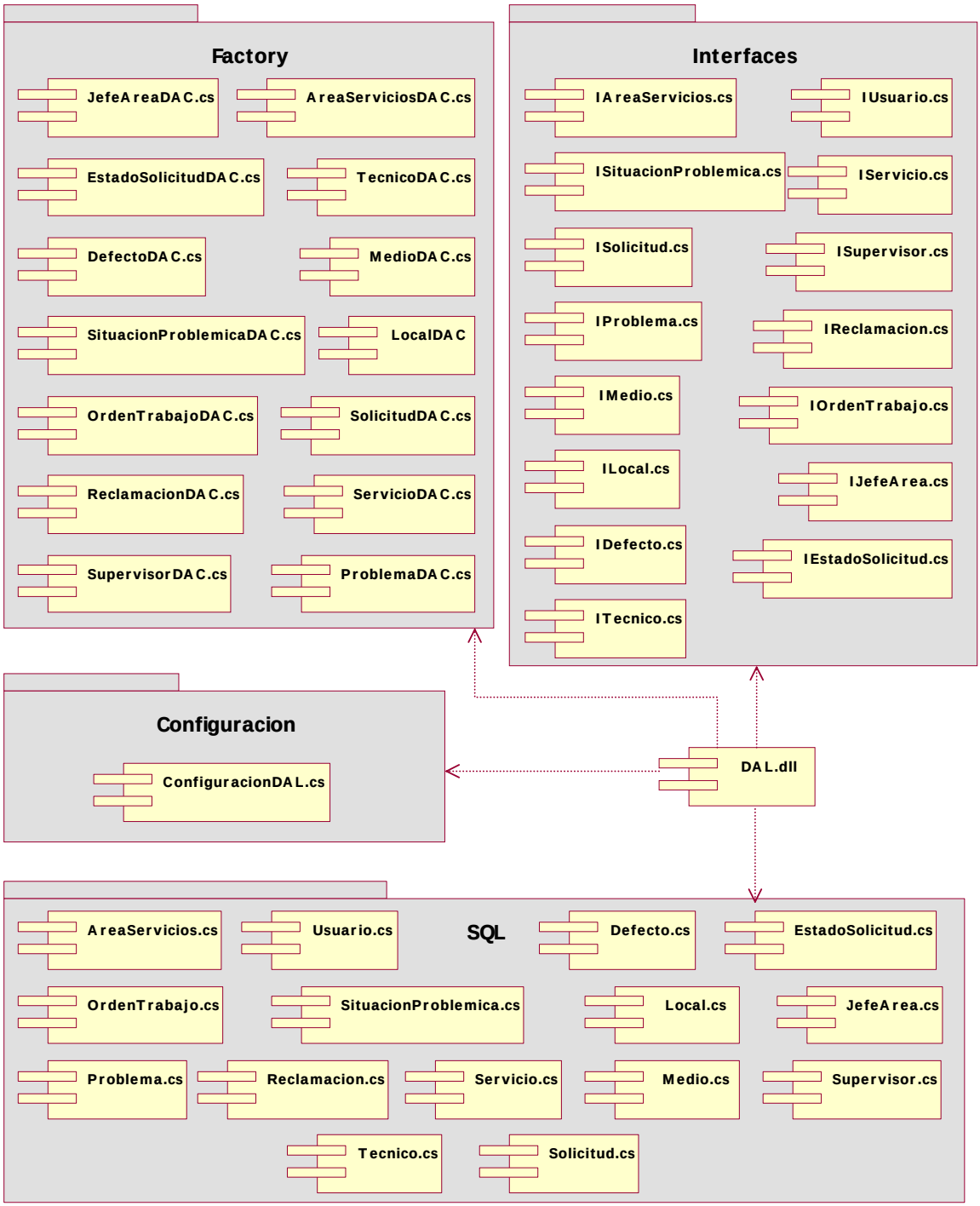


Diagrama de componentes del paquete Acceso a Datos.

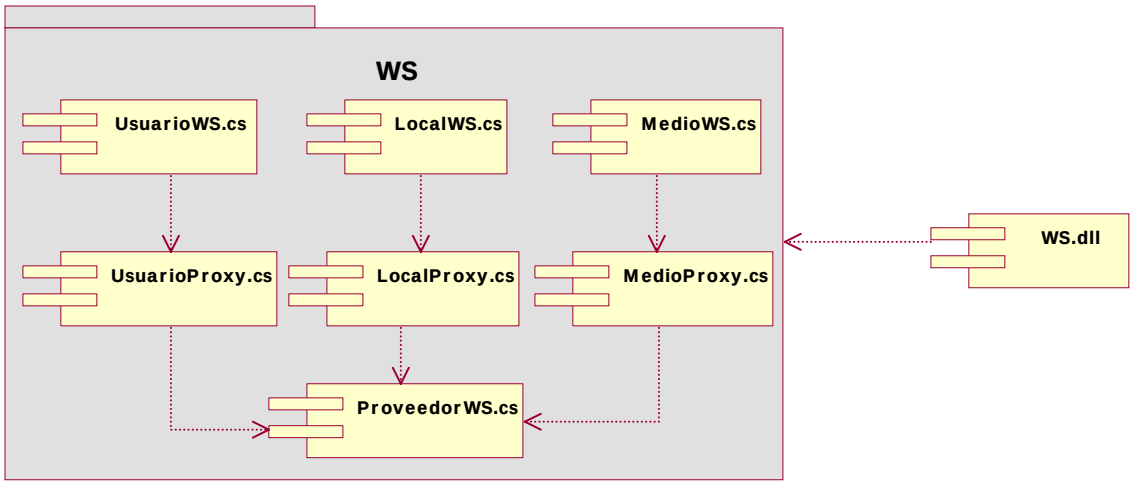


Diagrama de componentes del paquete WS.

4.7.1.2 Paquete *Entidades.*

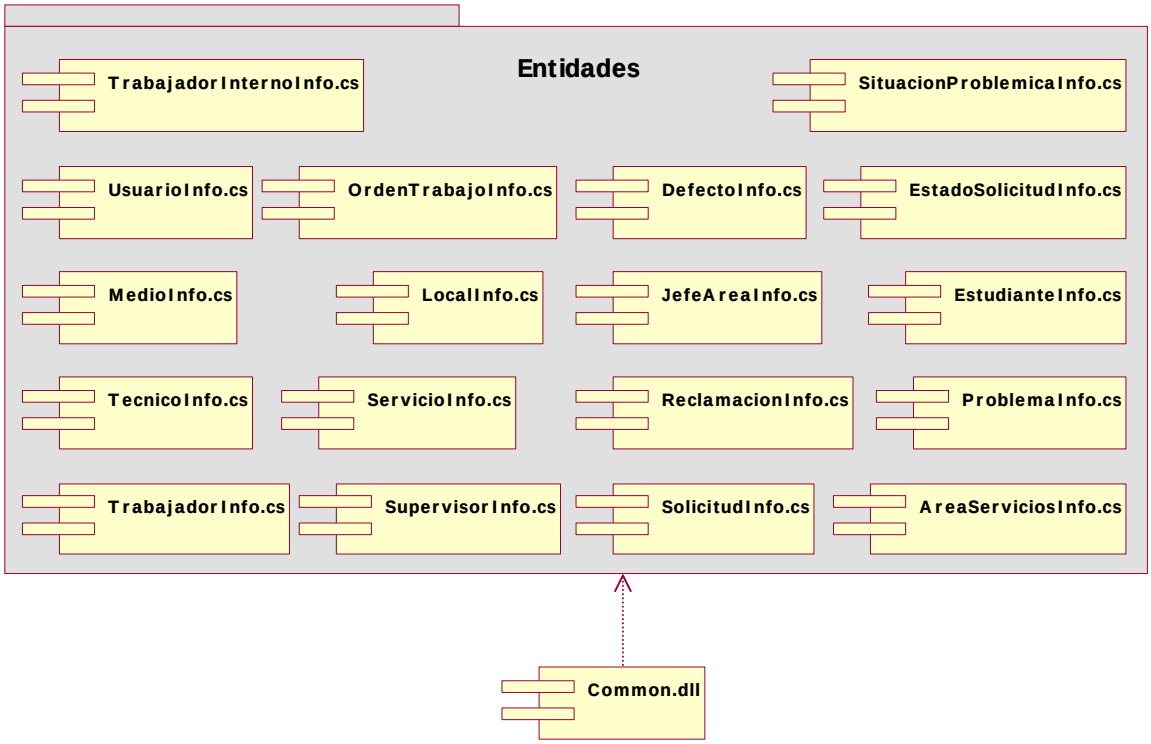


Diagrama de componentes del paquete Entidades.

4.7.1.3 Paquete *Procesamiento Lógico*.

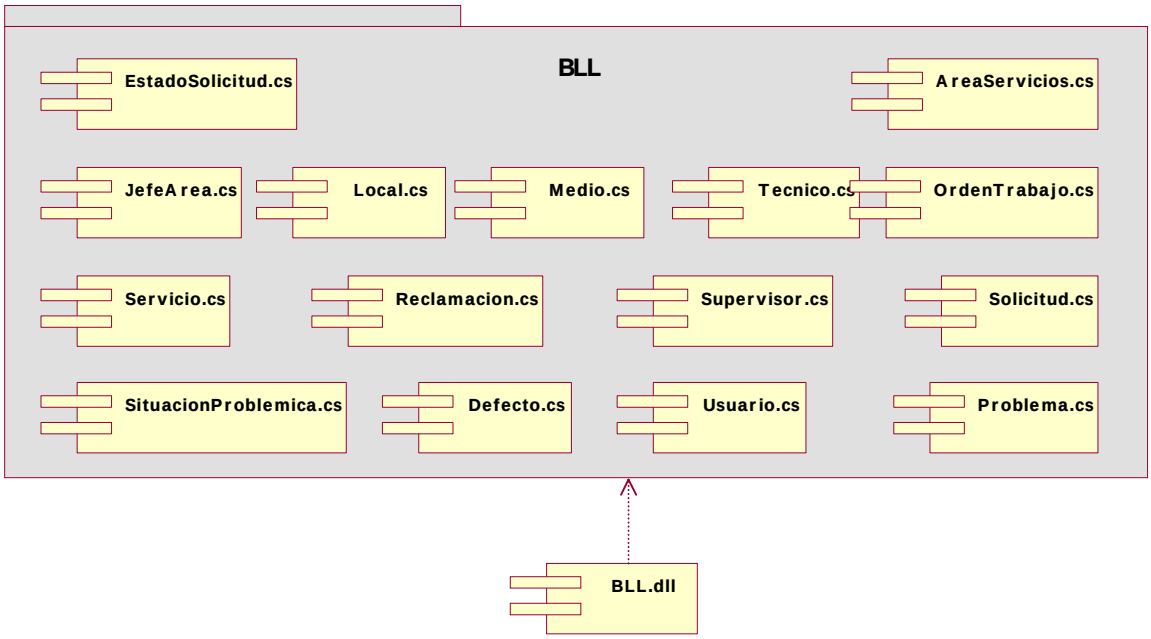


Diagrama de componentes del paquete Procesamiento Lógico.

4.7.1.4 Paquete *Presentación*.

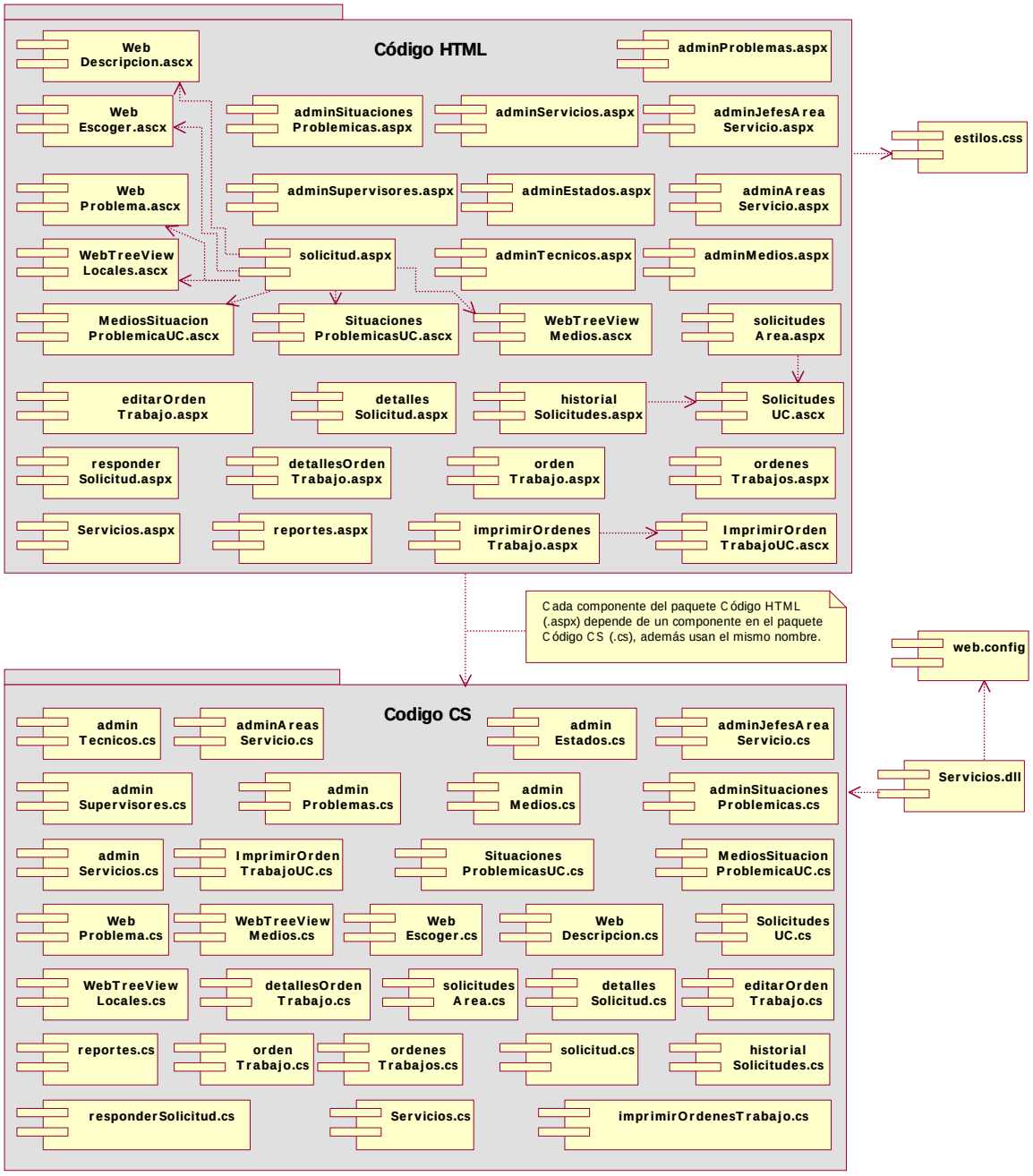


Diagrama de componentes del paquete de Presentación.

4.7.2 Explicación de los componentes.

Con el objetivo de lograr una mejor claridad, por cada componente solo existe una clase o interface, las cuales tienen el mismo nombre que el componente. Además las interfaces que cada componente expone son los mismos métodos públicos de las clases que estos contienen, si desea información sobre el contenido o las interfaces de

los componentes, remítase al diagrama de clases del paquete correspondiente. Los componentes que cumplen con esta regla son los siguientes:

- Paquete **Acceso a Datos**.
    - Paquete **Factory**.
    - Paquete **Interfaces**.
    - Paquete **Configuración**.
    - Paquete **Implementación SQL**.
    - Paquete **WS**.
- Paquete **Entidades**.
- Paquete **Procesamiento Lógico**.
- Paquete **Presentación**.

Los componentes que no cumplen con la regla antes mencionadas, se describen a continuación:

Componente	**Propósito**	**Contenido**	**Interfaces**
DAL.dll	Librería necesaria para el acceso a datos desde una BD.	Implementación de las clases de los paquetes Factory, Interfaces, Configuración e Implementación SQL.	Ver Diagrama de Clases del paquete Acceso a Datos.
WS.dll	Librería necesaria para la recuperación de datos desde Web Services.	Implementación de las clases del paquete Acceso WS.	Ver Diagrama de Clases del paquete Acceso WS.
BLL.dll	Librería necesaria para controlar la lógica de negocio del	Implementación de las clases del paquete	Ver Diagrama de Clases del paquete Procesamiento

	sistema.	Procesamiento Lógico.	Lógico.
Common.dll	Librería que contiene las entidades del sistema.	Implementación de las clases del paquete Entidades.	Ver Diagrama de Clases del paquete Entidades.
Servicios.dll	Librería necesaria para el funcionamiento de las páginas del sitio Web.	Contiene las la implementación de las clases del paquete Presentación.	Ver Diagrama de Clases del paquete Presentación.
Web.config	Archivo de configuración del sitio Web.	Este componente es un XML con valores globales necesarios en diversas páginas.	-
estilos.css	Hoja de estilos del sitio	Contiene los estilos definidos para el sitio Web.	-

Páginas Web.

Componente	**Propósito**	**Contenido**
Servicios.aspx	Página principal de la aplicación.	Contiene el acceso a todas las tareas que el sistema permite realizar al usuario conectado, así como otras informaciones.
solicitud.aspx	Página para realizar una nueva solicitud.	Ver CU Realizar Solicitud

historialSolicitudes.aspx	Página para ver el historial de solicitudes hechas por un usuario.	Ver CU Ver Solicitudes Personales.
solicitudesArea.aspx	Página para ver las solicitudes hechas a un área de servicios.	Ver CU Ver solicitudes del área de servicios.
detallesSolicitud.aspx	Página para mostrar los detalles de una solicitud	Contiene los datos de una solicitud, quien la hace, donde se localiza el problema, cuál es el problema, etc.
responderSolicitud.aspx	Página que brinda la opción de generar una orden de trabajo para una solicitud.	Contiene la opción de dar respuesta a una solicitud generando una orden de trabajo.
ordenTrabajo.aspx	Página para generar una orden de trabajo.	Ver CU Actualizar órdenes de trabajo.
ordenesTrabajo.aspx	Página que brinda la lista de órdenes de trabajo generadas para un área.	Ver CU Actualizar órdenes de trabajo.

imprimirOrdenesTrabajo.aspx	Página para imprimir varias órdenes de trabajo.	Contiene una lista de órdenes de trabajos.
editarOrdenTrabajo.aspx	Página que brinda la opción de actualizar una orden de trabajo.	Contiene los datos de una orden de trabajo y de actualizar el tiempo de duración de la misma.
detallesOrdenTrabajo.aspx	Página que brinda los detalles de una orden de trabajo.	Contiene los datos de una orden de trabajo.
adminAreasServicio.aspx	Página que brinda la posibilidad de actualizar las áreas de servicios.	Ver CU Actualizar áreas de servicios.
adminJefesAreaServicios.aspx	Página que brinda la posibilidad de actualizar los jefes de las áreas de servicios.	Ver CU Actualizar los jefes del área de servicios.
adminEstados.aspx	Página que brinda la posibilidad de	Ver CU Actualizar estados.

	actualizar los posibles estados de las solicitudes.	
adminServicios.aspx	Página que brinda la posibilidad de actualizar los servicios de un área de servicios.	Ver CU Actualizar servicios del área de servicios.
adminMedios.aspx	Página que brinda la posibilidad de actualizar los medios de un servicio.	Ver CU Actualizar medios del servicio.
adminProblemas.aspx	Página que brinda la posibilidad de actualizar los problemas asociados a un medio.	Ver CU Actualizar problemas del medio.
adminTecnicos.aspx	Página que brinda la posibilidad de actualizar los técnicos de un área de servicios.	Ver CU Actualizar técnicos del área de servicios.

adminSupervisores.aspx	Página que brinda la posibilidad de actualizar los supervisores de un área de servicios.	Ver CU Actualizar supervisores del área de servicios.
adminSituacionesProblemicas.aspx	Página que brinda la posibilidad de actualizar las situaciones problémicas de un área de servicios.	Ver CU Actualizar situaciones problémicas del área de servicios.
reportes.aspx	Página que brinda la posibilidad de ver los reportes del sistema.	Ver CU Ver reportes.

4.8 Conclusiones.

En este capítulo se mostraron varias vistas para llevar a cabo el proceso de implementación del sistema. Igualmente se identificaron otras funcionalidades que se pueden tener en cuenta para futuras versiones del sistema. Se utilizaron diagramas de clases Web para explicar la lógica del negocio del sistema, y se diseñaron las clases persistentes que permiten hacer el diagrama de entidad-relación, en el sistema de gestión de bases de datos que se utilizará. En este momento, ya se tiene confeccionada completamente la propuesta que trae este trabajo.

CAPITULO 5

ESTUDIO DE FACTIBILIDAD

5.1 Introducción.

Es importante evaluar la factibilidad de un proyecto antes de su elaboración, para conocer si es conveniente llevarlo a cabo. En el presente capitulo se hace un estudio de factibilidad, beneficios y costo del sistema propuesto.

5.2 Planificación.

Nombre de la entrada externa	**Cantidad de Ficheros**	**Cantidad de Elementos de datos**	**Clasificación (Simple, Media y compleja)**
Adicionar Solicitud	3	12	Medio
Actualizar Supervisor de la Solicitud	1	1	Simple
Actualizar Estado de la Solicitud	1	1	Simple
Adicionar Área Servicio	1	3	Simple
Eliminar Área Servicio	1	3	Simple
Actualizar Área de Servicio	1	2	Simple
Adicionar Jefe de Área de Servicio	1	2	Simple
Eliminar Jefe Área de Servicio	1	2	Simple
Adicionar Supervisor	1	2	Simple

Eliminar Supervisor	1	2	Simple
Adicionar Técnico	1	2	Simple
Eliminar Técnico	1	2	Simple
Adicionar Servicio	1	4	Simple
Eliminar Servicio	1	4	Simple
Actualizar Servicio	1	2	Simple
Adicionar Medio	1	2	Simple
Eliminar Medio	1	2	Simple
Adicionar Problema	1	4	Simple
Eliminar Problema	1	4	Simple
Actualizar Problema	1	2	Simple
Adicionar Situación Problémica	1	4	Simple
Eliminar Situación Problémica	1	4	Simple
Actualizar Situación Problémica	1	2	Simple
Adicionar Reclamación	1	4	Simple
Generar orden de trabajo	1	5	Simple
Actualizar orden de trabajo	1	2	Simple
Imprimir orden de trabajo	1	1	Simple

Adicionar Estado	1	3	Simple
Eliminar Estado	1	3	Simple
Actualizar Estado	1	2	Simple

Tabla 1. *Entradas externas.*

Nombre de la salida externa	**Cantidad de Ficheros**	**Cantidad de Elementos de datos**	**Clasificación (Simple, Media y compleja)**
Reporte	8	22	Complejo
Listado de áreas de Servicios	1	3	Simple
Listado de Estados	1	3	Simple

Tabla 2. *Salidas externas.*

Nombre de la petición	**Cantidad de Ficheros**	**Cantidad de Elementos de datos**	**Clasificación (Simple, Media y compleja)**
Listado de Solicitudes del Área	6	22	Complejo
Listado de Solicitudes de una persona	1	8	Simple
Listado de reclamaciones de una persona	2	12	Medio
Listado de reclamaciones de un Área	7	26	Complejo
Listado de ordenes de	7	29	Complejo

trabajo de un área			
Listado de servicios de un área	2	7	Medio
Listado de medios de un servicio.	2	6	Medio
Listado de problemas de un medio	3	7	Medio
Listado de situaciones problémicas de un área	2	7	Medio
Listado de supervisores de un área	2	5	Simple
Listado de técnicos de un área	2	5	Simple
Listado de jefes de áreas de un área	2	5	Simple

Tabla 3. *Peticiones.*

Nombre del fichero interno	**Cantidad de records**	**Cantidad de Elementos de datos**	**Clasificación (Simple, Media y compleja)**
AreasServicios	1	3	Simple
Defecto	1	2	Simple
EstadoSolicitud	1	3	Simple
JefeArea	1	2	Simple
Medio	1	2	Simple

OrdenTrabajo	1	7	Simple
OrdenTrabajo_Técnico	1	2	Simple
Problema	1	3	Simple
Reclamación	1	4	Simple
Servicio	1	4	Simple
SituaciónProblémica	1	4	Simple
SituaciónProblémica_Medio	1	2	Simple
Solicitud	1	8	Simple
Solicitud_Defecto	1	3	Simple
Solicitud_SituaciónProblémica	1	2	Simple
Supervisor	1	2	Simple
Técnico	1	2	Simple

Tabla 4. *Ficheros internos.*

Elementos	**Simples**		**Medios**		**Complejos**		**Subtotal de puntos de función**
	No	X Peso	No	X Peso	No	X Peso	
Ficheros lógicos internos	7	7		10		15	49
Entradas externas	29	3	1	4		6	91
Salidas externas	2	4		5	1	7	15
Peticiones	4	3	5	4	3	6	50

Total (UFP)							205

Tabla 5. *Interfaces externas.*

5.3 Costos.

Características	**Valor**
Puntos de función desajustados	235
Lenguaje	C# (95 %) JavaScript (1 %) SQL(4 %)
Instrucciones fuentes por puntos de función	(59) (56) (39)
Instrucciones fuentes por lenguaje (miles de instrucciones)	(11.4902) (0.1148) (0.3198)
Instrucciones fuentes (miles de instrucciones)	11.9248

Tabla 6. *Cantidad de instrucciones fuentes*.

Cálculo de:	**Valor**	**Justificación**
Esfuerzo	7,42	ΠEMi ≈ 1.7557 E ≈ 1 ∑SFi = 11.15 PREC = 2.48 FLEX = 3,04

		RESL = 1.41 TEAM = 1,1 EPML = 3.12
Tiempo de desarrollo	12.96 meses	PM ≈ 64.89 F ≈ 0.30
Cantidad de personas	5	Se cuenta con 2 personas para la realización del sistema.
Costo	73001.25	C =CHM *PM CHM = 5*225=1125 PM ≈ 64.89
Salario medio	225	
RCPX	1.00	RELY = NOMINAL DATA = NOMINAL CPLX = NOMINAL DOCU = MODERADO
RUSE	1,00	La aplicación da la posibilidad de ser reutilizada.
PDIF	1,00	TIME = 50% STOR = 50% PVOL = Poco volatil.
PREX	1.12	APEX = media PLEX= alta LTEX = Alta
FCIL	0.87	TOOL = ALTO

		SITE = NOMINAL.
SCED	1.43	SCED = 100%
PERS	1.26	ACAP = ALTO pcap = ALTO pcon = alto

Tabla 7. *Cálculos finales de la estimación de costos y esfuerzos.*

5.4 Beneficios tangibles e intangibles.

El sistema de Gestión de Servicios Comunitarios no es un producto con fines comerciales aunque puede adjuntarse por sus características a algún sistema de gestión de servicios. Tiene como principal objetivo resolver uno de los grandes problemas que tiene la universidad en estos momentos y es los servicios comunitarios de la UCI.

El principal objetivo de la aplicación es mejorar la gestión de los servicios comunitarios en la UCI.

Beneficios:

- Tener un sistema global para la realización de solicitudes desde cualquier ubicación en la universidad.
- Que los solicitantes puedan ver el estado de las solicitudes en cualquier momento.
- Que los jefes de áreas y supervisores conozcan el estado de las solicitudes realizadas en su área de servicio.
- Que los directivos puedan ver el sistema de reportes, el cual los ayudará en la toma de decisiones.

5.5 Análisis de costos y beneficios.

El desarrollo de este sistema no supone grandes gastos de recursos, ni tampoco de tiempo; la base de datos que contiene la información, puede ser alojada en los servidores existentes en la universidad, ya que los mismos tienen buenas prestaciones

y acceso rápido. La tecnología utilizada para el desarrollo del sistema es .NET, que es gratis. El sistema se ha diseñado pensando en una eventual migración al proyecto Mono por lo que un cambio de plataforma para la implantación del mismo es viable y factible, y no hay que incurrir en muchos cambios; debido a la estructuración en capas de los procesos del negocio que se diseñaron.

5.6 Conclusiones.

En este capítulo se efectuó el estudio de factibilidad correspondiente al desarrollo del proyecto. Este permitió llegar a la conclusión de que resultará factible implementar la aplicación, ya que aunque existe cierto costo total, los beneficios sociales que se alcanzarán son considerables.

CONCLUCIONES

Con el presente trabajo se propone una solución integral al problema de la Gestión de los Servicios Comunitarios en la Universidad de las Ciencias Informáticas. Se presenta una aplicación capaz de controlar las solicitudes del personal de la UCI, brindando además reportes sobre las mismas. El buen uso de esta aplicación, puede convertirla en una poderosa herramienta para el control de las solicitudes en la universidad, permitiendo ser usada además en la toma de decisiones.

El sistema se desarrolló siguiendo la metodología RUP, y se utilizaron representaciones para la modelación de todas las fases del proyecto.

El sistema resultante tiene un ambiente fácil de entender y usar, el cual cumple con los estándares de diseño y utiliza técnicas modernas de la programación orientada a objetos.

Por todo lo anterior expuesto se concluye que los objetivos propuestos para el presente trabajo han sido cumplidos satisfactoriamente. Se incluyen además una serie de recomendaciones que deben tenerse en cuenta para el trabajo futuro.

RECOMENDACIONES

A pesar de haberse cumplido los objetivos generales del trabajo, nuevas ideas han ido surgiendo en el desarrollo del mismo, permitiendo esto que en un futuro se cree un sistema más eficiente, por tanto recomendamos:

1. Continuar con el desarrollo del sistema con el objetivo de adecuarlo mejor a las demandas que presenta la universidad.
2. Emigrar el sistema a software libre, con el objetivo de seguir el movimiento que actualmente se esta llevando a cabo en la universidad. El sistema fue diseñado de manera que al emigrarlo, los cambios serán mínimos.
3. Integrar el sistema con otras aplicaciones que requieran obtener la información almacenada por el mismo.
4. Aplicar el sistema a todas las áreas de la universidad donde pueda ser usado, permitiendo un mejor uso del mismo.
5. Extenderlo de forma que pueda ser utilizado no solo en la universidad, sino también en cualquier lugar o empresa que requiera de un sistema para la Gestión de los Servicios Comunitarios.

GLOSARIO DE TERMINOS

Microsoft: Compañía de software más grande del mundo. Fue fundada en 1975 por Paul Allen y Bill Gates. Aunque también se conoce por sus lenguajes de programación y aplicaciones para computadores personales, el éxito sobresaliente de Microsoft se debe a sus sistemas operativos DOS y Windows.

OMT: (Object Modeling Techniques). Metodología de análisis y diseño orientado a objetos, desarrollada por Jim Rumbaugh; es una de las tres grandes metodologías que contribuyeron al desarrollo del RUP.

Web Services: Aplicación que realiza un cometido y que puede formar parte de otros servicios para formar un servicio más completo. La comunicación hacia y desde el WebService se realiza con XML.

SOAP: Acrónimo de Simple Object Access Protocol (Protocolo de acceso de objeto simple). Es un protocolo elaborado para facilitar la llamada remota de funciones a través de Internet, permitiendo que dos programas se comuniquen de una manera muy similar técnicamente a la invocación de páginas Web.

URL: Acrónimo de Universal Resource Locator (localizador universal de recursos), método de identificación de documentos o lugares en Internet, que se utiliza principalmente en World Wide Web (WWW). Un URL es una cadena de caracteres que identifica el tipo de documento, la computadora, el directorio y los subdirectorios en donde se encuentra el documento y su nombre.

CSS: Siglas en Inglés de Cascading Style Sheets (Hojas de Estilo en Cascada).

XML: Es el acrónimo de eXtensible Markup Language (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C).

UML: Lenguaje Unificado de Modelado (UML, por sus siglas en inglés, Unified Modelling Language) es el lenguaje de modelado de sistemas de software más conocido en la actualidad.

Common Language Runtime (CLR*):* Núcleo de la plataforma *.Net*, motor que gestiona la ejecución de las aplicaciones para ella desarrolladas y a las que ofrece múltiples servicios.

REFERENCIAS BIBLIOGRAFICAS

[1] Pressman, R. “Software Engineering. A Practitioner’s Approach”. Fourth Edition. McGraw – Hill. USA, 1999.

[2] Booch, G., Rumbaugh, J., Jacobson, I. “El Lenguaje Unificado de Modelado”. Addison-Wesley. 1999.

[3] Larman, C. “Applying UML and Patterns. An Introduction to Object-Oriented Analysis and Design”. Prentice-Hall, Inc. 1998.

[4] Joseph, Schmuller. “Aprendiendo UML en 24 horas”, Prentice-Hall, Inc. 2001.

[5] Características de Visual Studio .NET. Microsoft Corp. 2005.

http://www.microsoft.com/latam/vstudio/producto/caracteristicas.asp (18/6/2005)

[6] .NET Framework Fundamentals.

http://msdn.microsoft.com/netframework/programming/fundamentals/default.aspx (18/6/2005)

BIBLIOGRAFIA

1. Fonseca Elías, Fernando y Corrales Guerra, Yudiel. Sistema de Servicio Comunitario. Trabajo para optar por el titulo de Ingeniero Informático. Instituto Superior Politécnico "José Antonio Echeverría", Universidad de La Habana, Ciudad de la Habana, Junio del 2004.
2. Booch, G., Rumbaugh, J., Jacobson, I. "El Lenguaje Unificado de Modelado". Addison-Wesley. 1999.
3. Kerievsky, Joshua. "Refactoring to patterns". Industrial Login Inc. 2002.
4. Fowler, Martin, Beck, Kent, Brant, John, Opdyke, William, Roberts, Don. Refactoring: Improving Design of Existing Code. Addison-Wesley. 2002.
5. Shohoud, Yasser. Real World XML Web Services. Addison-Wesley. 2003.
6. Cockburn, Alistair. Writing Effective Use Cases. Addison-Wesley. 2000.
7. Ferguson, Jeff, Patterson, Brian, Beres, Jason, Boutquin, Pierre, Gupta, Meeta. La bilbia del C#. Anaya. 2003.
8. Microsoft. Introduction to C# Programming for the Microsoft .Net Platform. Microsoft. 2001.
9. Cerami, Ethan. Web Services Essentials. O'Reilly. 2002.
10. Joseph, Schmuller. "Aprendiendo UML en 24 horas", Prentice-Hall, Inc. 2001.
11. Benage, Don, Socha, Jody. .Net e-Business Architecture. Sams. 2002.
12. North, Simon. Teach yourself XML in 21 days. Macmillan Computer Publishing. 1999.
13. Mitchell, Scot, Anders, Bill, Howard, Rob, Seven Doug, Walther, Stephen, Willie, Christop, Wolthuis, Don. ASP. NET: Tips, Tutorials and Code. Sams. 2001.
14. *¿Qué es ASP.NET?* http://www.zonagratuita.com/microsoft/asp/aspnet.htm (18/6/2005)

15. Lenguajes de Programación: "Programación Web". http://lenguajes-de-programacion.com/programacion-web.shtml (18/6/2005)

16. Compare Microsoft .NET to J2EE Technology. http://www.gotdotnet.com/team/compare/ (18/6/2005)

17. Practical UML. A Hands-On Introduction for Developers. http://info.borland.com/techpubs/together/together_guides/umlonlinecourse/#component-and-deployment-diagrans (18/6/2005)

18. Características de Visual Studio .NET. Microsoft Corp. 2005. http://www.microsoft.com/latam/vstudio/producto/caracteristicas.asp (18/6/2005)

19. .NET Framework Fundamentals. http://msdn.microsoft.com/netframework/programming/fundamentals/default.aspx (18/6/2005)

20. URL Rewriting in ASP.NET. http://msdn.microsoft.com/library/default.asp?url=/library/en-us/dnaspp/html/urlrewriting.asp (18/6/2005)

21. Introduction to Web User Controls. http://msdn.microsoft.com/library/en-us/vbcon/html/vbconintroductiontocustomwebcontrols.asp (18/6/2005)

22. Tutorial: Introduction to Web Services. http://www.c-sharpcorner.com/Tutorials/IntroductionToWebServicesT.asp (18/6/2005)

23. Database Programming in C# with ADO.NET. http://www.c-sharpcorner.com/Database.asp (18/6/2005)

24. Core J2EE Patterns. http://java.sun.com/blueprints/corej2eepatterns/Patterns/index.html (18/6/2005)

25. Análisis y Diseño Orientado a Objetos. http://www.dcc.uchile.cl/~luguerre/cc40b/index.html (18/6/2005)

ANEXOS

Anexo I. Diseños de la aplicación.

Página de presentación.