



**Universitat Autònoma
de Barcelona**

Interfaz Siebel - Intranet

Memoria del proyecto
de Ingeniería Técnica en
Informática de Gestión
realizado por
David Pérez Ruiz
y dirigido por
Óscar Cubillo Alonso

Escuela de Ingeniería
Sabadell, Septiembre de 2012

El abajo firmante, **Óscar Cubillo Alonso**,
profesor de la Escuela de Ingeniería de la UAB,

CERTIFICA:

Que el trabajo al que corresponde la presente memoria
ha sido realizado bajo su dirección por

David Pérez Ruiz

Y para que conste firma la presente.
Sabadell, ***Septiembre*** de **2012**

Firmado: **Óscar Cubillo Alonso**

HOJA DE RESUMEN – PROYECTO FIN DE CARRERA DE LA ESCUELA DE INGENIERIA

Título del proyecto: Interfaz Siebel - Intranet	
Autor: David Pérez Ruiz	Fecha: Septiembre 2012
Tutora: Óscar Cubillo Alonso	
Titulación: Ingeniería Técnica en Informática de Gestión	
Palabras clave: CRM, Siebel, HTML, XML, XDS, Apache Tomcat, Java, Oracle.	
Resumen del proyecto <ul style="list-style-type: none"> • Castellano: <p>Una empresa que da servicios de asistencia a usuarios para una entidad bancaria gestiona y registra las incidencias que los empleados u oficinas puedan tener mediante un software CRM. Los empleados del banco no tienen acceso a dicho software, con lo que necesitan alguna forma de poder llegar a cierta información. Este proyecto rellena dicho hueco, estableciendo una conexión extremo a extremo de forma segura y eficaz. Por medio de ficheros XML se envían las solicitudes y las respuestas desde la intranet del banco (mediante un formulario de registro de incidencias) hasta los sistemas CRM.</p> • Catalán: <p>Una empresa que presta servei d'assistència als usuaris d'una entitat bancària gestiona i enregistra les incidències que els empleats i les oficines poden tenir mitjançant un software CRM. Els empleats del banc no tenen accés a aquest software, llavors requereixen algun mètode per tal d'accedir a part de la informació. Aquest projecte omple aquest buit, mitjançant la connexió extrem a extrem de forma segura i eficaç. Mitjançant fitxers XML s'envien sol·licituds i respostes des de la Intranet del banc (mitjançant un formulari de registre de incidències) fins al sistema CRM.</p> • Inglés: <p>An enterprise that provides user assistance support for a bank offices manages and record issues that employees or offices can suffer by using CRM software. Those employees don't have access to that CRM software, so they may need some way to reach the information that the CRM stores. This project fills that hole by establishing a connection peer to peer securely and efficiently. Requests and responses are sent through XML files from the bank's intranet (by an issue recording form) to CRM systems.</p> 	

CONTENIDO

1. INTRODUCCIÓN	7
1.1 RESUMEN.....	7
1.2 TIPOLOGÍA Y PALABRAS CLAVE	9
1.3 DESCRIPCIÓN	9
1.4 CRM	9
1.4.1 QUE ES UN CRM?	9
1.4.2 SIEBEL.....	10
1.5 REQUERIMIENTOS DEL CLIENTE	10
1.6 OBJETIVOS DEL PROYECTO	12
1.7 ESTRUCTURA DE LA MEMORIA.....	13
1.8 MOTIVACIONES	15
1.9 ACRÓNIMOS Y ABREVIACIONES	15
1.10 PARTES INTERESADAS.....	17
2. ESTUDIO DE VIABILIDAD	19
2.1 SITUACIÓN INICIAL.....	19
2.2 REQUISITOS DEL PROYECTO	20
2.2.1 REQUISITOS FUNCIONALES	20
2.2.2 REQUISITOS NO FUNCIONALES	21
2.2.3 RESTRICCIONES DEL SISTEMA.....	21
2.2.4 CATALOGACIÓN Y PRIORIZACIÓN DE LOS REQUISITOS.....	22
2.3 PLANIFICACIÓN	23
2.3.1 WBS (WORK BREAKDOWN STRUCTURE).....	23
2.3.2 RECURSOS DEL PROYECTO	24
2.3.3 CALENDARIO DEL PROYECTO	26
2.3.4 EVALUACIÓN DE RIESGOS	29
2.3.5 VIABILIDAD ECONÓMICA.....	31
2.3.6 CONCLUSIONES.....	33
3. ANÁLISIS FUNCIONAL	34
3.1 INTRODUCCIÓN	34
3.2 SITUACIÓN INICIAL.....	34

3.2.1 CONTEXTO	34
3.2.2 SITUACIÓN ACTUAL	35
3.3 SITUACIÓN DESEADA	36
3.4 REQUERIMIENTOS FUNCIONALES	37
3.4.1 OBTENER FORMULARIO	38
3.4.2 CREACIÓN DE INCIDENCIAS	40
3.4.3 LISTAR INCIDENCIAS	41
3.4.4 CONSULTAR INCIDENCIA	43
3.4.5 MODIFICAR INCIDENCIA	44
3.4.6 RECLAMAR INCIDENCIA	45
4. DISEÑO TÉCNICO	47
4.1 FUNDAMENTOS TÉCNICOS	47
4.1.1 JAVA	48
4.1.2 TOMCAT	49
4.1.3 XML	49
4.1.4 SIEBEL	51
4.1.5 BBDD	51
4.2 PERFIL DE USUARIO	52
4.3 DISEÑO TÉCNICO DE LA APLICACIÓN	53
4.3.1 MÉTODOS	53
4.3.2 CARACTERÍSTICAS COMUNES	53
4.3.3 FICHEROS ANEXOS	54
4.3.4 CÓDIGOS DE ERROR	54
4.3.5 PARÁMETROS ENTRADA/SALIDA	56
5. PRUEBAS Y PROBLEMAS DETECTADOS	63
5.1 PRUEBAS UNITARIAS	63
5.2 PRUEBAS INTEGRADAS	65
5.3 PRUEBAS DE VALIDACIÓN	65
5.4 PRUEBA DE ESTRÉS	65
5.5 PROBLEMAS DETECTADOS	66
5.5.1 CODIFICACIÓN DE CARACTERES EN DIRECCIONES URL	66
5.5.2 ADJUNTAR ARCHIVOS EN SIEBEL	67

6. CONCLUSIÓN	67
6.1 CUMPLIMIENTO DE LOS OBJETIVOS.....	67
6.2 VARIACIONES EN LA PLANIFICACIÓN	68
6.3 LINEAS DE AMPLIACIÓN FUTURAS	69
ANEXO. JUEGO DE PRUEBAS Y FICHEROS XML DE ENTRADA/SALIDA	69
BIBLIOGRAFÍA	74
EN PAPEL.....	74
ONLINE.....	75
ÍNDICE DE FIGURAS.....	75

1. INTRODUCCIÓN

En este capítulo se hará una pequeña introducción al proyecto detallando los objetivos y aspectos más importantes del mismo.

1.1 RESUMEN

La empresa SAU¹ presta servicios de asistencia y resolución de incidencias a una red de oficinas de una entidad bancaria. Esta asistencia cubre todas las necesidades que una oficina puede tener, entre las más comunes tenemos la reparación de un PC, dificultades técnicas con algún programa, impresoras y escáneres, problemas para realizar una operación determinada a un cliente, avería de un cajero automático, etc.

Dichos servicios de asistencia se registran en un CRM² usado por el Contact Center³ para hacer el seguimiento y resolución de las incidencias. El CRM utilizado es Siebel⁴. La parte que nos afecta en este proyecto es básicamente la creación de estos registros en Siebel, aunque más tarde veremos que proporciona otras herramientas como la modificación y el seguimiento de incidencias.

Actualmente los empleados de las oficinas dan de alta las incidencias por teléfono o bien mediante la intranet de la entidad bancaria. La segunda opción funciona mediante un sencillo formulario que es enviado por correo electrónico a un buzón de correo perteneciente a SAU. Siebel consulta constantemente dicho buzón y ejecuta un flujo de trabajo para la creación de la incidencia en Siebel.

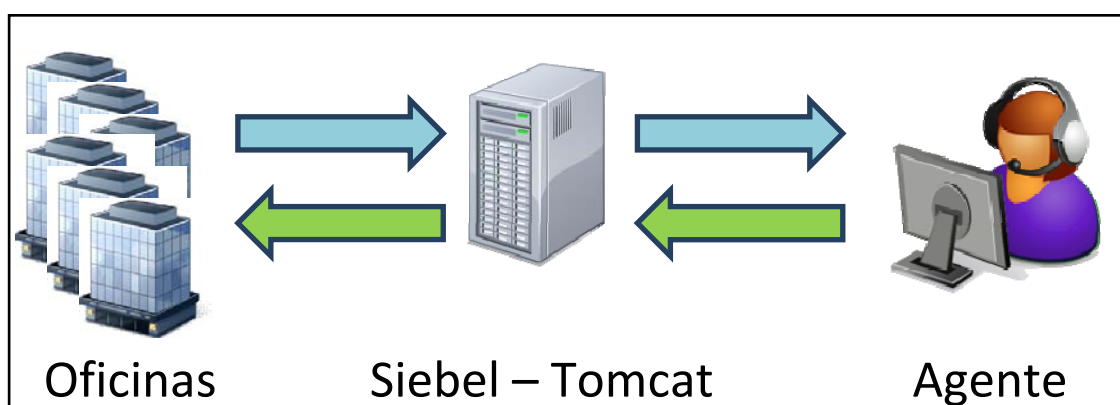


Figura 1: Servidor Siebel - Tomcat

1. Servicio de atención a usuarios.
2. *Customer Relationship Management*. Aplicativo para gestionar la relación con el cliente.
3. Centro de atención de llamadas.
4. Aplicativo CRM de Oracle.

Esta forma de trabajar ha quedado obsoleta con el paso de los años y hoy en día tenemos a nuestro alcance soluciones más fiables. Trabajando de esta manera nos hemos encontrado con diferentes problemas, que pese a no ser muy habituales, suponen grandes problemas para el SAU. Algunos de esos problemas son directamente atribuibles al protocolo que estamos utilizando, el correo electrónico no es fiable al 100%, puede sufrir demoras y no siempre tenemos la garantía de su envío/recepción. Otros problemas pueden deberse a caracteres utilizados en el correo que no son reconocibles por Siebel. También hemos visto problemas con determinados ficheros adjuntos en un correo. Otro problema puede ser la indisponibilidad del buzón de correo, bien sea por mantenimiento o incidencia.

Debido a estos problemas y a unas nuevas necesidades del cliente, se ha propuesto este proyecto. La solución consiste en ofrecer a los servidores de intranet que albergan el servicio actual de formulario de envío de incidencias una interfaz estandarizada de transmisión de datos en formato XML, que permite interactuar directamente con la lógica de negocio residente en Siebel.

Para acceder, se podrá usar un cliente de servicios http-XML con el que se podrán enviar mensajes XML que contendrán las peticiones que se deseen realizar. Un cliente de servicios http-XML se define según la mensajería de cada servicio o funcionalidad. A cada uno le corresponde un documento XML de entrada según el formato especificado en los XSD (XML Schema Definition Language) y una respuesta con el código que indica el resultado de la operación. Las ventajas de la solución propuesta son:

- Se sustituye la entrada de formularios a través de correo electrónico por la entrada de servicios http entre el formulario de Intranet y el servidor Siebel, que aporta robustez y permite la confirmación del procesado.
- Aporta robustez a la operativa de formularios:
 - Solución extremo a extremo.
 - Mayor control: permite que desde la aplicación de formularios se tenga confirmación de entrega y procesado del formulario.
 - Reducción del impacto por indisponibilidad de los servicios de correo.

Todos los mensajes recibidos se registrarán en la BBDD de Siebel para dar soporte a eventuales auditorías sobre el uso de datos confidenciales y a la vez permitir generar informes para generar estadísticas de uso por centro, áreas geográficas, tipos de petición, etc.

Con este documento se pretende analizar tanto la viabilidad como los diseños funcionales y técnicos, así como explicar todo el proceso de desarrollo de la aplicación.

1.2 TIPOLOGÍA Y PALABRAS CLAVE

Tipología: Desarrollo de aplicación.

Palabras clave: Java, Tomcat, XML, XDS, CRM, Siebel, HTML, Oracle.

1.3 DESCRIPCIÓN

Actualmente cuando un empleado de una oficina bancaria da de alta una incidencia no vuelve a tener más información de esta hasta que se soluciona y es informado mediante un correo electrónico. En el caso que el empleado desee consultar el estado de resolución lo ha de hacer telefónicamente. Cuando el nuevo proyecto esté implantado, el empleado podrá consultar el estado de su petición y podrá realizar modificaciones o incluso hacer reclamaciones.

1.4 CRM

A menudo hablaremos de conceptos como CRM y Siebel en este documento, por lo que es necesario explicar qué son.

1.4.1 QUE ES UN CRM?

Lo primero es saber a qué nos referimos cuando hablamos de CRM (Customer Relationship Management o Gestión de las Relaciones con el Cliente).

CRM es una estrategia corporativa compuesta de tecnología, procedimientos y planteamientos que tiene como objetivo común lograr las siguientes metas:

1. Unificar y maximizar la información que se obtiene del cliente a través de los distintos canales de atención que la empresa pueda tener.

2. Utilizar la información para aumentar las ventas, fidelizar a los clientes y mejorar la atención del cliente.

1.4.2 SIEBEL

Siebel es una compañía que desde sus inicios se dedicó a hacer CRMs, pero que en el 2006 pasó a formar parte del grupo Oracle.

Siebel ha sido líder indiscutible durante muchos años en el mercado de los CRMs.

Siebel tiene productos para muchos ámbitos de mercado, en nuestro caso la opción que nos interesa es Siebel Call Center. Este producto viene preparado con las funcionalidades mínimas que todo contact center puede necesitar. Cuando los requerimientos se salen de esta funcionalidad mínima, es cuando entran en juego los desarrolladores Siebel, que parametrizan y añaden la funcionalidad requerida a la aplicación.

1.5 REQUERIMIENTOS DEL CLIENTE

A nivel funcional, el cliente requiere:

- Los empleados de las oficinas deben tener acceso a:
 - Apertura de nuevas incidencias.
 - Consulta de incidencias creadas por ellos mismos o empleados de su oficina.
 - Consultar el estado de las incidencias y saber qué equipo/unidad las está gestionando.
 - Añadir información adicional a una incidencia previamente abierta.
 - Realizar reclamaciones (durante un tiempo definido) sobre incidencias resueltas si la respuesta no es acorde a las expectativas.

- La operación de creación de incidencias:
 - Recibe como entrada los datos que actualmente se envían mediante correo y devuelve un identificador de la incidencia que es generado aleatoriamente por Siebel.
 - Permite el envío de documentos anexados.
 - Valida que la tipología (clasificación de la incidencia) introducida en el formulario de la intranet es correcto.

- La operación de listar incidencias de un usuario:
 - Recibe como parámetro de entrada el Usuario, un rango de fechas y un código de oficina.
 - Retorna una lista de incidencias abiertas, cerradas con opción a reclamación y cerradas sin opción a reclamación.

- La operación de consultar una incidencia:
 - Recibe como parámetro de entrada un número de incidencia (generado previamente por Siebel).
 - Retorna la información más relevante de la incidencia.

- La operación de añadir información a una incidencia:
 - Recibe como parámetro de entrada un número de incidencia (generado previamente por Siebel) y un campo de texto libre con la información a añadir y opcionalmente uno o más documentos anexos.
 - Añade los datos a la incidencia en Siebel y retorna un código informando del resultado de la operación.

- La operación de reclamación de una incidencia:
 - Recibe como parámetro de entrada un número de incidencia (generado previamente por Siebel) y un campo de texto libre con la información a añadir y opcionalmente uno o más documentos anexos.
 - Añade los datos a la incidencia en Siebel y retorna un código informando del resultado de la operación.
 - Cambia el estado de una incidencia que está cerrada, pasando de Cerrado a Abierto, siempre y cuando se encuentre dentro del periodo máximo de reapertura. Dicho periodo es parametrizable por parte del cliente.
- Validación de control de acceso:
 - Un usuario no puede consultar información acerca de incidencias de otro usuario a excepción que ambos pertenezcan a la misma oficina.
- El servicio debe estar disponible de 7 a 22 horas de lunes a viernes.

1.6 OBJETIVOS DEL PROYECTO

Los principales objetivos del proyecto son:

1. Evitar el factor humano, que puede provocar errores.

Cuando el empleado llama al contact center puede haber un malentendido entre la operadora y él. Sobre todo en numeraciones y códigos como un NIF, un CCC¹, etc.

2. Automatización del proceso de generación de incidencias.

Resulta muy ventajoso que el empleado pueda generar o consultar una incidencia sin tener que depender de una persona al otro lado de circuito.

1. *Código Cuenta Cliente*. Es un identificador único de una cuenta bancaria.

3. Auditar la vida de una incidencia.

Este proyecto permite auditar de forma sencilla la vida de una incidencia. Con una sola consulta sabremos la fecha de generación, que equipo la trató, si fue escalada a otro grupo, cuando se cerró, etc.

4. Transparencia para el usuario.

Se transmite transparencia para el usuario, pues no hay lugar para la objetividad y los malentendidos. Este proyecto es una capa intermedia entre Siebel y el empleado.

Por otro lado tenemos los objetivos personales:

5. Adquirir experiencia en la tecnología Java.

6. Aprender cómo funciona la transmisión de información mediante ficheros XML.

7. Realizar un proyecto desde cero. Hasta ahora siempre he trabajado en proyectos que están en fase de mantenimiento, este proyecto sin embargo es un reto para mí al poder desarrollarlo completamente.

1.7 ESTRUCTURA DE LA MEMORIA

La memoria se dividirá en 11 capítulos, donde veremos las diferentes fases del proyecto y explicaremos las necesidades del mismo. Hay cuatro capítulos fundamentales, donde se detallarán las diferentes partes que constituyen este proyecto.

- Capítulo 1: Presentación.

Se explicarán los objetivos del proyecto, las motivaciones para llevarlo a cabo y la organización de la memoria. Se explica qué es un CRM y en concreto se habla de Siebel.

- Capítulo 2: Estudio de viabilidad.

Se plantea el estado del arte, la situación actual, se definen los objetivos y requerimientos necesarios para conseguirlos, se plantean las posibles alternativas y finalmente se opta por la más conveniente valorándola, planificándola en el tiempo y definiendo los posibles riesgos que implica.

- Capítulo 3: Análisis funcional.

Un análisis funcional es un estudio de la situación de la empresa o del área en la que influirá un proyecto y de la situación a la que se quiere llegar.

- Capítulo 4: Diseño técnico.

Se describen las herramientas utilizadas para desarrollar el proyecto. Se detallan los parámetros de entrada/salida de los métodos y se muestran algunos ejemplos de ficheros XML.

- Capítulo 5: Pruebas y problemas detectados.

Se describen las pruebas realizadas durante el desarrollo del proyecto y la batería de test finales. Se describen también las dificultades que han ido apareciendo con el desarrollo del proyecto y cómo se han solucionado.

- Capítulo 6: Conclusiones.

Veremos si se han cumplido los objetivos del proyecto.

- Capítulo 7: Bibliografía.

Se mostrarán las fuentes de conocimiento o consulta que se han utilizado para la realización del proyecto.

1.8 MOTIVACIONES

Antes de iniciar este proyecto, mis conocimientos en el lenguaje de programación Java son muy básicos. Por eso veo este proyecto como una forma de adentrarme en este lenguaje. Obviamente he tendré muchas dificultades que se irán resolviendo con la consulta de libros e internet. En mi equipo de trabajo, nadie sabe programar en Java, así que he tendré que aprender por mí mismo.

Además de aprender Java, tendré que familiarizarme con el lenguaje XML, que pese a ser más sencillo que Java, también requiere tiempo de aprendizaje.

Además de Java y XML, tendré que programar funciones en Siebel y crear nuevos flujos de trabajo.

Personalmente este proyecto será fructuoso para mí, pues aprenderé nuevas tecnologías, me enfrentaré a retos difíciles, etc.

1.9 ACRÓNIMOS Y ABREVIACIONES

En este apartado se definen todos aquellos acrónimos y abreviaciones que aparecen en este documento.

BBDD:	Base de datos. Conjunto de datos pertenecientes a un mismo contexto y almacenados sistemáticamente para su posterior uso.
Contact Center:	Centro de atención de llamadas. Es un área donde agentes especialmente formados atienden a clientes de una empresa.
CPU:	Central Processing Unit, significa unidad central de procesamiento. Es básicamente el procesador de la máquina.
CRM:	Customer relationship management. Significa gestión de la relación con el cliente. Siebel es uno de los muchos CRMs existentes en el mercado.

HDD:	Hard Disk Drive, significa unidad de disco duro. Es el dispositivo de almacenamiento de la máquina.
HTTP:	HyperText Transfer Protocol, protocolo de transferencia de hipertexto. Es el método más común de transferencia de información en Internet.
Interfaz:	Conexión entre 2 ordenadores dando una comunicación a distintos niveles.
JDK:	Java Development Kit, es un software que provee herramientas de desarrollo para la creación de programas en Java.
PC:	Ordenador personal, proviene del término inglés Personal Computer.
RAM:	Random Acces Memory, significa memoria de acceso aleatorio. Es la memoria volátil del sistema.
SAU:	Servicio de atención al usuario, suele ser un servicio de soporte telefónico. En este documento también haremos referencia al término Contact Center, que es un grupo de agentes tele operadores que dan algún tipo de soporte o venta telefónica.
Siebel:	Es una aplicación CRM que actualmente pertenece al grupo Oracle. Provee la infraestructura necesaria para implantar contact centers, entre otras cosas.

Spam:	Se llama spam, correo basura o mensaje basura a los mensajes de correo electrónico no solicitados, no deseados o de remitente no conocido (correo anónimo), habitualmente de tipo publicitario.
Tomcat:	Servidor web desarrollado en Java, por lo que permite ser utilizado en cualquier plataforma que disponga de la máquina virtual de Java.
VPN:	Significa Red Privada Virtual. Es una tecnología de red que permite una extensión de la red local sobre una red pública. En otras palabras, permite conectarnos al PC del trabajo desde casa.
WBS:	Work Breakdown Structure. Significa estructura de descomposición del trabajo. Es una descomposición jerárquica del trabajo a realizar por un equipo de proyectos.
XML:	Proviene de las siglas en inglés, eXtensible Markup Language. Lenguaje de marcas extensible. Es un sistema que permite definir lenguajes de acuerdo a las necesidades.
XSD:	XML Schema Definition Language. Lenguaje que sirve para definir la estructura de los elementos del XML.

1.10 PARTES INTERESADAS

Las partes interesadas son aquellas que influirán, usarán o supervisarán alguna parte o todo un proyecto. Podemos dividir las partes interesadas en 3 grupos, la gerencia del proyecto, los usuarios y el equipo de proyecto.

- Gerencia: grupo de personas a las que les interesa que este proyecto salga adelante.
- Usuarios: personas que utilizarán la aplicación.

- Equipo de proyecto: grupo de personas que intervienen en cada una de las partes del desarrollo del proyecto.
 - Jefe de Proyecto: Define, gestiona, planifica y controla el proyecto.
 - Coordinador: Colabora con el jefe de proyecto en el estudio de viabilidad y la planificación. Vincula cada parte del proyecto participando en cada una de ellas.
 - Analista: Analiza la arquitectura, metodología, especificaciones y estándares. Participa en el diseño y validación.
 - Programador: Diseña y desarrolla la aplicación de acuerdo con el análisis. Participa en el proceso de validación e implementación.

2. ESTUDIO DE VIABILIDAD

En este capítulo estudiaremos si es viable crear un interfaz que comunique los sistemas Siebel con la intranet del banco.

2.1 SITUACIÓN INICIAL

Tras generar una incidencia, es habitual que un empleado desee consultar el estado en el que se encuentra o saber a qué grupo de trabajo está asignada su petición. Para estas consultas o posibles modificaciones han de ponerse en contacto con el contact center de SAU vía telefónica en un horario determinado. Como sucede con la mayoría de contact centers, el tiempo de espera para ser atendido depende de la franja horaria pero habitualmente suele requerir unos minutos de espera como mínimo, y en el peor de los casos se termina la llamada mediante una locución indicando que todos los operadores están ocupados e invitando a llamar pasados unos minutos.

Esta situación no es agradable para el cliente y requiere que el SAU disponga de muchos agentes para atender el volumen de llamadas. Uno de los principales objetivos de este proyecto es reducir el impacto del gran volumen de llamadas recibidas diariamente. El canal telefónico seguirá siendo imprescindible, pues gran parte de los usuarios prefiere tratar sus incidencias con una persona al otro lado del teléfono, pero se van a reducir drásticamente llamadas para hacer consultas sobre peticiones creadas anteriormente, pues a partir de ahora cada empleado podrá ver qué peticiones tiene abiertas. Podrá hacer búsquedas, consultas, modificaciones, añadir ficheros adjuntos y por último podrá hacer reclamaciones.

Podrá realizar todas estas operaciones con un solo clic de ratón, sin perder tiempo de su trabajo y sin tener que esperar para ser atendido, además lo podrá hacer a cualquier hora del día e incluso desde casa mediante un acceso VPN.

Todas estas ventajas se traducen en un ahorro económico además de la satisfacción de mejora que pueda aportar al cliente. Aproximadamente un 10%¹ de las llamadas atendidas son para gestionar una incidencia existente, es decir, nuestro proyecto va a suponer aproximadamente, tras un periodo de aceptación de los empleados del banco, un descenso de un 10% de las llamadas, lo que se puede traducir a grosso modo en una necesidad de personal un 10% menor de la actual.

1. Porcentaje calculado por el cliente según un estudio interno.

2.2 REQUISITOS DEL PROYECTO

En este punto se detallarán las funciones y restricciones solicitadas por el cliente.

2.2.1 REQUISITOS FUNCIONALES

Los requisitos funcionales definen el comportamiento y funcionalidad de la aplicación.

- RF1 → Crear incidencia: ha de permitir al usuario generar una incidencia nueva en el sistema.
- RF2 → Consultar incidencia: ha de permitir ver todos los datos introducidos al generar la incidencia y además permite ver en que estado se encuentra y a qué grupo está asignado.
- RF3 → Añadir información adicional o un fichero adjunto a una incidencia previamente creada.
- RF4 → Realizar reclamaciones: ha de permitir realizar una reclamación sobre una incidencia. Si dicha incidencia se ha cerrado en un periodo de tiempo determinado, la propia reclamación cambia el estado de la incidencia para que vuelva a estar abierta. Dicho periodo es un parámetro de la aplicación y lo define el cliente.
- RF5 → La incidencia se asigna a un equipo determinado según la tipología de la incidencia. Dicha tipología es una clasificación de 5 niveles que el empleado escoge cuando da de alta la incidencia.
- RF6 → Se debe enviar un correo electrónico de confirmación al empleado cuando se crea una incidencia y otro correo electrónico cuando se soluciona.
- RF7 → Un usuario sólo podrá consultar y/o modificar incidencias creadas por él mismo u otro empleado de la misma oficina.
- RF8 → El servicio debe estar disponible de 7 a 22 horas de lunes a viernes.

2.2.2 REQUISITOS NO FUNCIONALES

Los requisitos no funcionales son las cualidades que debe tener una aplicación, hablamos de términos como rendimiento, seguridad, accesibilidad, etc.

- RNF1 → Rendimiento: se han de desarrollar los procesos aprovechando los recursos necesarios y no haciendo un mal uso de ellos.
- RNF2 → Disponibilidad: se ha de garantizar el buen funcionamiento y disponibilidad de la aplicación dentro del horario requerido por el cliente.
- RNF3 → Seguridad: El sistema tiene que ser robusto en cuanto a seguridad ya que estamos tratando con datos maestros de los clientes. Se debe asegurar que no pueden haber accesos no autorizados y procesos lanzados por error.
- RNF4 → Estabilidad: Se debe tener en cuenta el volumen de peticiones y el volumen de datos a tratar para utilizar los recursos necesarios para garantizar un sistema fluido y estable.
- RNF5 → Coste: Se tratará de minimizar costes usando todos los recursos disponibles en la empresa.
- RNF6 → Mantenibilidad: Se debe garantizar que la aplicación será sencilla de mantener en el futuro y capaz de adaptarse a nuevos requisitos del cliente.

2.2.3 RESTRICCIONES DEL SISTEMA

Se proponen las siguientes restricciones:

- Usar la tecnología Java que funciona con un servidor Tomcat tanto en sistema operativo Windows, Windows Server y Linux.

- Intentar utilizar clases estándar que ya se utilizan en otras aplicaciones de la empresa.
- El modelo de datos usado es el que proporciona Siebel.
- Control de accesos según la normativa establecida por la empresa.
- Guardar registros de todas las peticiones en base de datos.
- La fecha de entrega será en Julio de 2012.

2.2.4 CATALOGACIÓN Y PRIORIZACIÓN DE LOS REQUISITOS

Estableciendo un orden y una preferencia en cada uno de los requisitos a cumplir se obtienen las siguientes tablas:

- Prioridad de los requisitos funcionales.

	Esencial	Condicional	Opcional
RF1	X		
RF2	X		
RF3	X		
RF4	X		
RF5		X	
RF6		X	
RF7			X
RF8	X		

Figura 2: Prioridad requisitos funcionales

- Prioridad de los requisitos no funcionales.

	Esencial	Condicional	Opcional
RNF1	X		
RNF2	X		
RNF3	X		
RNF4	X		
RNF5		X	
RNF6			X

Figura 3: Prioridad requisitos no funcionales

2.3 PLANIFICACIÓN

En este apartado se recogen las tareas, puntos de control, recursos, calendario, evaluación de riesgos y el presupuesto del proyecto.

2.3.1 WBS (WORK BREAKDOWN STRUCTURE)

Para poder organizar el trabajo, saber cuáles son las fases del proyecto, con qué concluye cada una de esas fases y en qué fechas deben finalizarse se utilizará la herramienta WBS.

Se describirá cada una de las partes del proyecto y detallará cuáles son los puntos de control.

2.3.1.1 FASES DEL PROYECTO

Las fases del proyecto son las siguientes:

- Iniciación: Definición del proyecto.
- Planificación: Incluye el estudio de viabilidad y el plan de proyecto.

- Análisis: Análisis de los requisitos funcionales y no funcionales y de la arquitectura del sistema.
- Diseño: Diseño de control y de los test.
- Desarrollo: Fase de desarrollo de la aplicación.
- Test de pruebas: Fase de prueba del sistema. Incluye test unitarios y de integración.
- Implantación: La aplicación se sube al servidor real. Se proporciona formación al usuario.
- Generación de documentos: Documentación del proyecto, manuales y memoria.
- Cierre del proyecto: El director de proyecto da la aceptación y se da por finalizado el proyecto.

2.3.2 RECURSOS DEL PROYECTO

Se ha de hacer distinción entre los recursos materiales y los recursos humanos.

2.3.2.1 RECURSOS MATERIALES

Los recursos materiales son los bienes tangibles que permiten conseguir un objetivo.

2.3.2.1.1 SERVIDORES

El servidor donde se alojará el código Java y el servicio Tomcat tiene las siguientes características:

Descripción	Características
Sistema Operativo	Solaris 9
CPU	4 x Intel Xeon Core 2 5598 MHz
RAM	6144 MB
HDD	400 GB

Figura 4: Características servidor

2.3.2.1.2 PCs PARA PERSONAL EQUIPO DE PROYECTO

Cada persona del equipo de proyecto dispondrá de un PC con la siguiente configuración.

Descripción	Características
Sistema Operativo	Microsoft Windows XP Profesional (64 bit)
CPU	1 x Intel Inside Core i3-2100 3,10 GHz
RAM	4096 MB
HDD	250 GB

Figura 5: Características PC.

2.3.2.2 RECURSOS HUMANOS

Los recursos humanos son todas aquellas personas que intervienen en la realización del proyecto. En este proyecto podemos diferenciar 3 perfiles:

- Jefe de proyecto
- Analista
- Programador

El jefe de proyecto supervisará cada paso que se vaya dando en el proyecto, además de trabajar como analista.

El analista será la persona encargada de tomar los requerimientos tanto funcionales como no funcionales del sistema.

El programador desarrollará la aplicación y estará presente en todos los procesos necesarios para poder alcanzar los objetivos.

2.3.2.3 CALENDARIO DE LOS RECURSOS

Las fases en las que intervienen cada uno de los perfiles anteriores son:

- Jefe de proyecto: Iniciación, planificación, generación de documentos, cierre del proyecto y puntos de control.
- Analista: Análisis, diseño, implantación y puntos de control.

- Programador: Diseño, desarrollo, test e implantación.

	Jefe de Proyecto	Analista	Programador
Iniciación	X	X	
Planificación	X		
Análisis		X	
Diseño		X	X
Desarrollo		X	X
Control	X		
Implementación		X	X
Documentación		X	X
Cierre	X		

Figura 6: Fases de actuación por perfil.

2.3.3 CALENDARIO DEL PROYECTO

2.3.3.1 CALENDARIO GENERAL

- **Calendario del proyecto:**

El proyecto comenzará en Febrero de 2012 y concluirá en mayo de 2012. Tendrá una dedicación aproximada de 20 horas semanales. Tendrá una duración total de 300 horas.

- **Fecha de comienzo:**

6 de Febrero de 2012.

- **Fecha de finalización:**

1 de Mayo de 2012.

- **Tareas de planificación:**

- Microsoft Project.
- Ganttter¹, herramienta online para la gestión de proyectos.

2.3.3.2 DEPENDENCIAS

Se ha utilizado un modelo lineal/secuencial para realizar las diferentes tareas del proyecto de tal manera que cada fase no comienza hasta que no se completa la anterior. El proyecto tiene una duración de 300 horas.

2.3.3.3 CUADRO DE TAREAS

En esta tabla se puede ver cuáles son las tareas a desarrollar en cada fase del proyecto, la duración estimada, las fechas en las que se realizarán, de qué tareas dependen y qué recursos se utilizará.















		Nombre	Duración	Inicio	Fin	Predecesoras	Recursos
1		☐ Interfaz Siebel - Intranet	61d	06/02/2012	01/05/2012		
2		☐ Análisis y definición	8d	06/02/2012	16/02/2012		
7		☐ Documentación del análisis	6d	16/02/2012	24/02/2012	2	
10		☐ Diseño entorno y programación	5d	24/02/2012	02/03/2012	7	
11		Creación y configuración del entorno	2d	24/02/2012	28/02/2012	9	Analista[10%],Programador[90%]
12		Preparación entorno Siebel	2d	28/02/2012	01/03/2012	11	Programador
13		Preparación nuevas tablas de auditoria	1d	01/03/2012	02/03/2012	12	Analista[10%],Programador[90%]
14		☐ Programación	26d	02/03/2012	09/04/2012	10	
15		Pogramación métodos en BS de Siebel	6d	02/03/2012	12/03/2012	13	Programador
16		Programación métodos Java	14d	12/03/2012	30/03/2012	15	Programador
17		Programación añadir ficheros filesystem	5d	30/03/2012	06/04/2012	16	Programador
18		Programación conexión OM Siebel	1d	06/04/2012	09/04/2012	17	Programador
19		☐ Control de calidad	14d	09/04/2012	27/04/2012	14	
20		Pruebas unitarias	4d	09/04/2012	13/04/2012	14	Programador
21		Pruebas integradas	8d	13/04/2012	25/04/2012	14,20	Programador
22		Pruebas de validación	2d	25/04/2012	27/04/2012	21	Analista[50%],Programador[50%]
23		Pruebas de estrés	1d	25/04/2012	26/04/2012	21	Analista[50%],Programador[50%]
24		Puesta en producción	2d	27/04/2012	01/05/2012	19	Analista[50%],Programador[50%]
25		☐ Documentación	48d	06/02/2012	12/04/2012		
26		Elaboración de la memoria	48d	06/02/2012	12/04/2012		Analista

Figura 7: Tareas a desarrollar.

1. Dicha herramienta se puede encontrar en www.ganttter.com

2.3.3.4 DIAGRAMA DE GANTT (GRÁFICO)

En este gráfico se pueden ver la duración y las fechas de cada tarea.

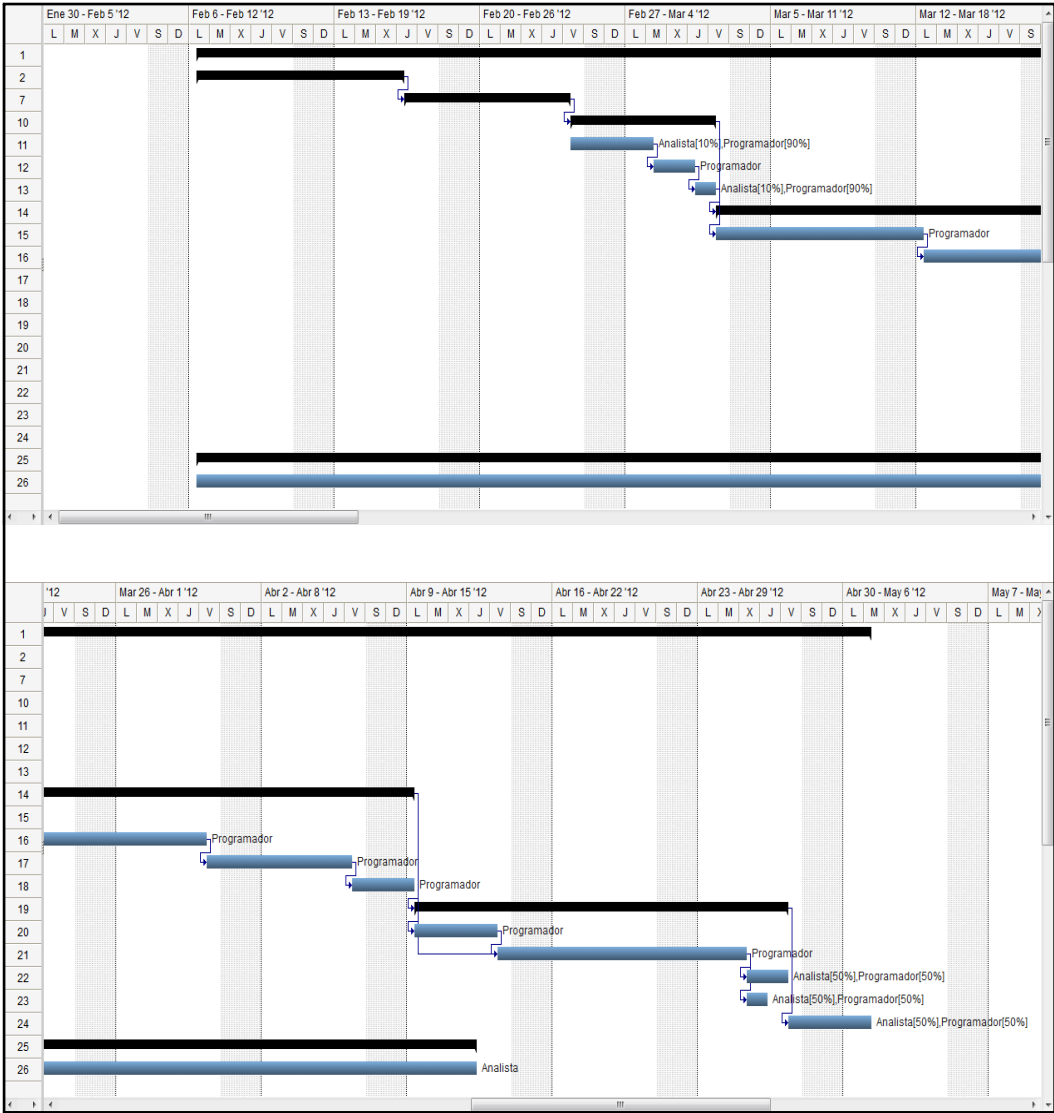


Figura 8: Diagrama de Gantt

2.3.4 EVALUACIÓN DE RIESGOS

Vamos a analizar una lista de riesgos a los que nos podemos enfrentar durante las etapas del proyecto. Viéndolos de antemano podemos tomar medidas para minimizar un posible impacto sobre la planificación del proyecto.

2.3.4.1 LISTA DE RIESGOS

Se han tenido en cuenta los siguientes riesgos:

- Planificación temporal optimista. No se termina en la fecha prevista, se han de incrementar los recursos.
- Falta alguna tarea necesaria. No se cumplen los objetivos del proyecto.
- Cambio de requisitos. Retraso en la finalización del proyecto, no se cumplen los objetivos del proyecto.
- Dificultad para acceder a las personas que deben facilitar información sobre el proyecto. Retraso en la finalización del proyecto, no se cumplen los objetivos del proyecto.
- No se hace correctamente la fase de pruebas. Menor calidad en el resultado.
- Incumplimiento de alguna norma, reglamento o legislación. Repercusiones legales.
- Abandono del proyecto antes de su finalización. Pérdidas económicas. Frustración.
- Presupuesto mal ajustado. Menos calidad, pérdidas económicas.

Por las características específicas de este proyecto, las situaciones que con más probabilidad se podían dar eran que la planificación fuera demasiado optimista o que aparecieran tareas no previstas inicialmente, debido a la falta de experiencia tanto en materia integración de sistemas como de la propia tecnología Java.

Para tener en cuenta este riesgo se dividió el proyecto en diferentes fases que, a medida que se fuera avanzando en ellas, se iría dotando de mayor funcionalidad al producto.

2.3.4.2 CATALOGACIÓN DE RIESGOS

Según el impacto que tienen sobre el proyecto, los riesgos definidos anteriormente se pueden catalogar de la siguiente forma:

	Probabilidad	Impacto
Planificación temporal optimista	Alta	Crítico
Falta alguna tarea necesaria	Alta	Crítico
Cambio de requisitos	Media	Marginal
Dificultad acceso conocimiento	Baja	Crítico
Fase de pruebas incorrecta	Alta	Crítico
Incumplimiento legislación	Baja	Marginal
Abandono proyecto	Baja	Catastrófico
Presupuesto incorrecto	Alta	Crítico

Figura 9: Catalogación de riesgos

2.3.4.3 PLAN DE CONTINGENCIA

A continuación se detallan una serie de medidas a tomar en el caso que algunos de los riesgos mencionados anteriormente tuvieran lugar:

	Solución
Planificación temporal optimista	Aplazar alguna funcionalidad, afrontar posibles pérdidas, hacer un seguro.
Falta alguna tarea necesaria	Renegociar con el cliente, afrontar posibles pérdidas, hacer un seguro.
Cambio de requisitos	Renegociar con el cliente, aplazar funcionalidad, modificar planificación y presupuesto.
Dificultad acceso conocimiento	Fijar un calendario de reuniones de equipo.

Fase de pruebas incorrecta	Diseñar los test con antelación, realizar test automáticos, negociar contrato de mantenimiento, dar garantías, afrontar pérdidas económicas.
Incumplimiento legislación	Revisar las normas y legislación, consultar a un experto, afrontar posibles repercusiones penales.
Abandono proyecto	No tiene solución.
Presupuesto incorrecto	Revisar el Plan de Proyecto, modificar la planificación.

Figura 10: Plan de contingencia

2.3.5 VIABILIDAD ECONÓMICA

Para el presupuesto ofrecido a la empresa se tendrá en cuenta el coste de personal (recursos humanos), el coste material (recursos materiales) y los costes indirectos (alquiler oficinas, luz, agua, teléfono).

2.3.5.1 ESTIMACIÓN DEL COSTE DEL PERSONAL

Según el coste por hora de cada uno de los perfiles y la dedicación total de cada uno de ellos podemos calcular la siguiente tabla:

	Coste/Hora	Horas totales	Coste total
Jefe de Proyecto	54 €	25	1350 €
Analista	45 €	25	1125 €
Programador	38 €	250	9500 €
		Total	11975 €

Figura 11: Coste y dedicación de los recursos humanos.

2.3.5.2 ESTIMACIÓN COSTE DE LOS RECURSOS MATERIALES

Para la realización del proyecto se ha dispuesto de los siguientes recursos materiales:

	Coste	Periodo amortización	Periodo de uso	Coste repercutido
Amortización PC	1200 €	24 meses	4 meses	200 €
Amortización Portátil	1800 €	24 meses	4 meses	300 €
Amortización Servidor	6000 €	24 meses	4 meses	1000 €
Total				1500 €

Figura 12: Coste de los recursos materiales

2.3.5.3 ESTIMACIÓN DE LOS COSTES INDIRECTOS

Son costes indirectos el agua, la luz, el alquiler de las instalaciones, etc., que no tendremos en cuenta al ser insignificantes en proporción al alcance de este proyecto y al ser un gasto que la empresa tiene igualmente aunque no se llevase a cabo este proyecto.

2.3.5.4 AMORTIZACIÓN DE LA INVERSIÓN

Para calcular los beneficios, primero vamos a ver los ingresos que suponemos este proyecto va a conseguir.

- Usuario de contact center:

Como decíamos en el capítulo 2, suponemos que nuestra aplicación va a captar el 10% de las peticiones actuales. De media, un agente atiende 80 peticiones al día, en una jornada de 8 horas, eso supone una media de 1 incidencia cada 6 minutos. Calculamos un ahorro de 48 minutos por empleado. Los empleados cobran la hora a 6 euros, así que supone un ahorro de 4,80 euros por empleado y día. En total hay 30 empleados, así que obtenemos un ahorro de 3168 euros al mes.

- Usuario de oficina:

Usando nuestra aplicación el empleado ahorra 6 minutos en crear o consultar el estado de una incidencia. De media un empleado crea 3 incidencias al mes. El coste de su hora de trabajo es de 15 euros la hora. Por cada usuario estamos obteniendo un ahorro de 4,5 euros al mes, que multiplicado por 300 usuarios da un ahorro mensual de 1350 euros.

Como vemos, obtenemos un ahorro mensual de 4508 euros. En el apartado anterior vimos que los costes materiales y humanos eran de 13475 euros. Con estos datos obtenemos que a partir del cuarto mes la inversión queda amortizada y comenzamos a tener beneficios.

2.3.6 CONCLUSIONES

Después del análisis realizado sobre el desarrollo de este proyecto, se concluye que a nivel técnico, operativo y económico, el proyecto engloba los requisitos para hacerlo viable ya que se mantiene dentro de los márgenes permitidos y esperados.

Además tenemos una ganancia en fiabilidad y garantía de servicio, algo muy importante para el cliente además del aspecto económico

3. ANÁLISIS FUNCIONAL

A continuación se detallará el diseño funcional de este proyecto.

3.1 INTRODUCCIÓN

Un análisis funcional es un estudio de la situación de la empresa o del área en la que influirá un proyecto y de la situación a la que se quiere llegar.

En este capítulo se podrá encontrar:

- Situación actual de la empresa, se hará una contextualización de la empresa explicando cómo se gestionan las incidencias.
- Situación deseada a la que se quiere llegar después de realizar el proyecto.
- Requerimientos funcionales, aquellas funciones que se pretenden conseguir con la aplicación.
- Requerimientos no funcionales, describirán cómo se pide que funcione el sistema. Se propondrán todos aquellos atributos de calidad, o de forma que debe cumplir la aplicación.
- Detalles del sistema, se expondrá de manera general qué es lo que se aspira conseguir y cómo se quiere conseguir.

3.2 SITUACIÓN INICIAL

3.2.1 CONTEXTO

La función principal del departamento de contact center es la de gestionar las incidencias que las oficinas puedan encontrarse. Existen dos canales para crear una incidencia, el teléfono o el correo electrónico. Vamos a centrarnos en el canal correo electrónico, que el que vamos a sustituir con nuestro proyecto, por lo tanto es el que nos interesa.

Los empleados del banco tienen a su disposición un apartado en su intranet que sirve para crear incidencias. No es más que un sencillo formulario donde pueden tipificar el origen de la incidencia y poner algún comentario. Tras enviar el formulario, el cliente pierde la conexión con la incidencia y solamente

es notificado por correo electrónico una vez su incidencia ha sido resuelta. Esto genera incertidumbre y desconfianza al cliente, pues en ningún momento sabe si su petición está siendo atendida, ha sido rechazada, ha sido trasladada, etc. Tampoco tiene a su disposición la fecha propuesta de resolución y otra información de interés.

Por otro lado, no existe manera que el empleado pueda tener un listado de sus incidencias. Las oficinas tampoco pueden tener informes de las incidencias que generan sus empleados.

3.2.2 SITUACIÓN ACTUAL

Actualmente, tras completar todos los campos necesarios, el empleado confirma el formulario y se envía un correo electrónico al servidor de correo de Siebel. Siebel tiene un componente que sabe tratar los correos recibidos (Communications Inbound Manager) y permite crear un flujo de trabajo desde que se recibe un correo, aplicar la lógica de negocio correspondiente y así tener una incidencia pendiente para el agente al que se le ha asignado.

La ventaja de esta forma de trabajar es que tiene un coste muy reducido, puesto que en pocas horas se puede parametrizar Siebel para la correcta recepción de los formularios.

La desventaja es mayormente la fiabilidad. Pueden darse multitud de casos por los que no se envíe o reciba un correo electrónico:

- Problemas en el servidor saliente de correo electrónico del banco.
- Problemas en el servidor entrante de correo electrónico de Siebel.
- Tareas de mantenimiento en cualquiera de los servidores.
- Un correo puede ser descartado si contiene un carácter no admitido por Siebel.
- Colapso en la recepción de correos si alguno de ellos contiene adjuntos y ralentiza la recepción, haciendo que los demás queden en cola de espera.

- Problema de recepción de Spam¹ en la cuenta.
- Problemas con virus o cadenas de correo electrónico.

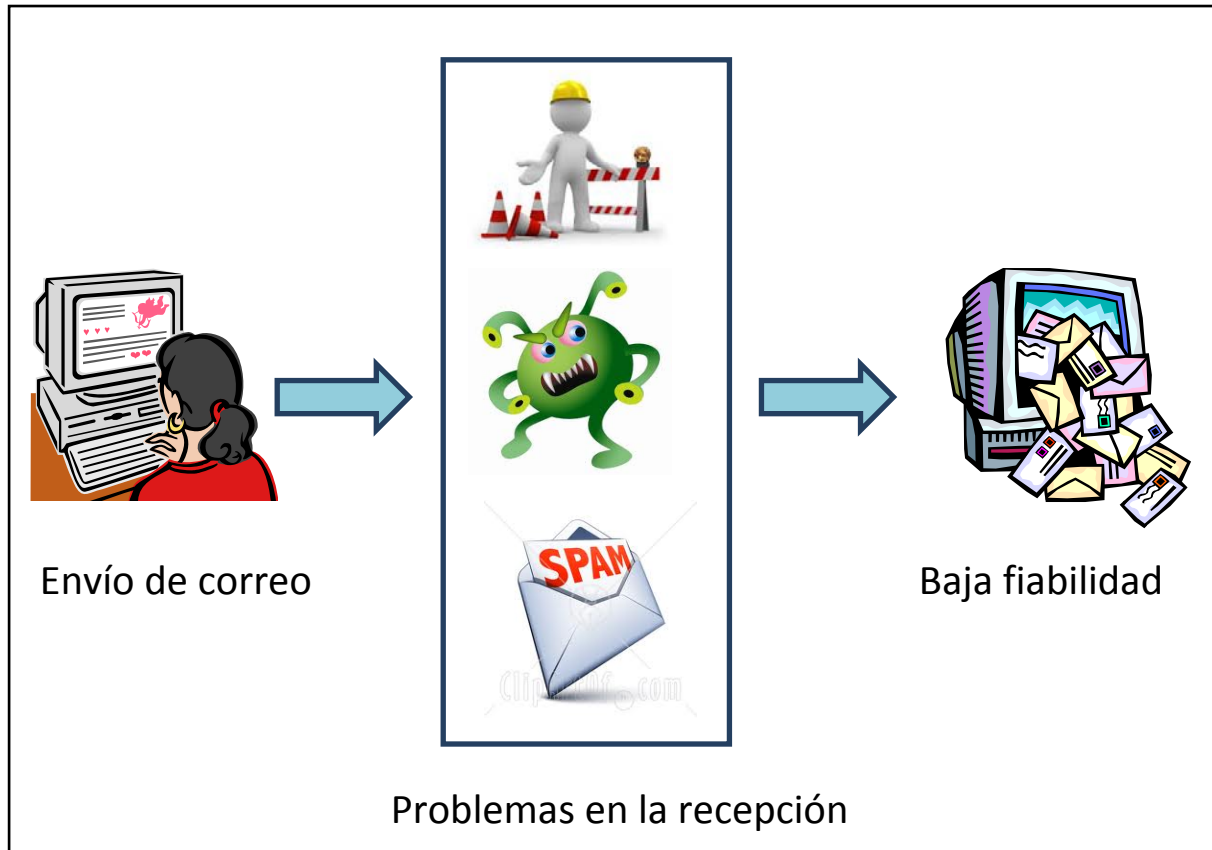


Figura 13: Desventajas envío formulario por correo electrónico.

3.3 SITUACIÓN DESEADA

Vamos a realizar una interfaz entre la intranet y Siebel que va a garantizar el servicio.

En el siguiente gráfico podemos ver las conexiones que realiza esta aplicación entre la intranet y la base de datos de Siebel y el propio sistema Siebel.

1. Correo basura, mensajes no deseados, no solicitados, generalmente publicidad.

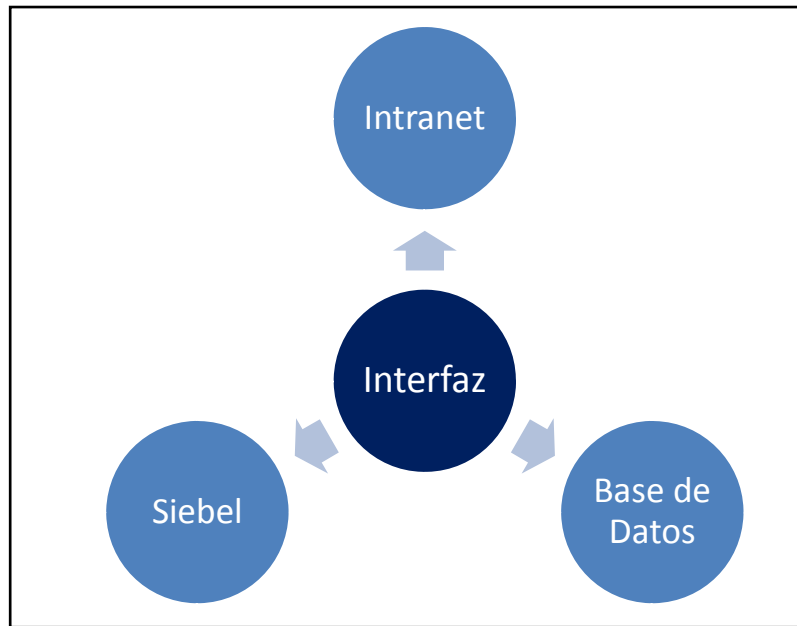


Figura 14: Conexiones Interfaz.

Esta nueva forma de transmitir la información entre la intranet del banco y el sistema Siebel va a reportar grandes beneficios en cuanto a rapidez, fiabilidad y disponibilidad del servicio.

Desde la intranet se van a hacer peticiones vía http con ficheros XML a nuestro servidor tomcat. Más adelante veremos cuáles son estas peticiones. Nuestra aplicación Java va a hacer una petición al servidor Siebel. Cuando este le responda, va a generar un fichero XML con la información de la respuesta para ser devuelta nuevamente vía http a la intranet del banco.

Vamos a tener 2 tipos de peticiones, las que requieren acceder a Siebel y las que no, es decir, aquellas peticiones en las que el servidor tomcat va a acceder directamente a la base de datos para obtener la información solicitada. Este tipo de peticiones son las más sencillas, puesto que no requieren de la actuación de la lógica de negocio de Siebel.

3.4 REQUERIMIENTOS FUNCIONALES

Vamos a dividir el proyecto en métodos o funcionalidades. A continuación se detallan estos métodos:

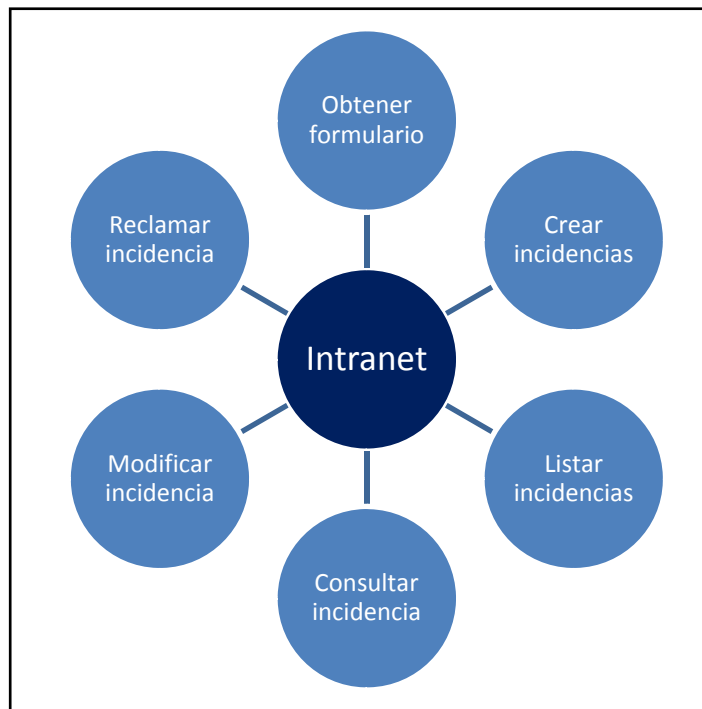


Figura 15: Métodos llamados desde la intranet.

3.4.1 OBTENER FORMULARIO

Las incidencias tienen una tipología de 5 niveles. Según la tipología de una incidencia se requiere una información u otra, es por esto que el método `OBTENER_FORMULARIO` es llamado automáticamente cuando queremos realizar otras operaciones, ya que nos va a proporcionar qué campos vamos a solicitar al empleado y cuáles de estos van a ser obligatorios u opcionales. Es imprescindible saber qué información ha de cumplimentar el empleado dependiendo de la naturaleza de la incidencia. Hay ciertos campos que han de ser cumplimentados siempre, y son comunes a cualquier clasificación, pero hay otros que dependen de esta clasificación.

Por ejemplo, pongamos tres incidencias que pueden darse en una oficina y que están bien diferenciadas por su naturaleza.

- Incidencia A: Se ha fundido una lámpara fluorescente.

- Incidencia B: Tienen problemas haciendo un duplicado de tarjeta de crédito con un cliente.
- Incidencia C: Un cliente les ha hecho una pregunta sobre inversión en Bolsa que no saben responder.

Como vemos, las tres incidencias son muy diferentes. Es obvio que tendrán que ser tratadas por agentes especializados en cada tipología. También es comprensible que los datos a cumplimentar en cada uno de los casos sean diferentes, pues no tendría sentido pedir un número de cuenta del cliente cuando lo único que ha sucedido es que se ha fundido una bombilla en la oficina.

La lista de campos requerida para cada tipología es definida y actualizada por un agente supervisor del contact center. Dicho mantenimiento se hace desde una pantalla de Siebel a la que sólo tienen acceso los supervisores.

Así pues, cuando el empleado vaya a realizar una operación que requiera conocer qué campos se han de cumplimentar, se va a hacer primero una llamada a este método, desde la intranet se va a enviar un fichero XML con la clasificación seleccionada por el empleado, se hará una consulta a la base de datos y se devolverá la respuesta en otro fichero XML con los campos a cumplimentar.

3.4.1.1 DIAGRAMA DE SECUENCIA DE OBTENER_FORMULARIO

En el siguiente diagrama podemos ver cómo funciona el método OBTENER_FORMULARIO.

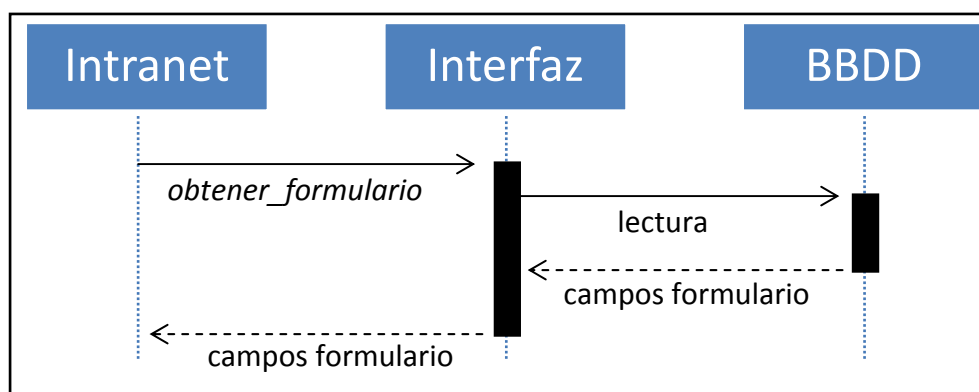


Figura 16: Diagrama del método OBTENER_FORMULARIO.

3.4.2 CREACIÓN DE INCIDENCIAS

El método `CREAR_INCIDENCIA` va a pasar todos los datos introducidos por el empleado a una primitiva de Siebel que genera incidencias.

Tras generar el registro en Siebel, nos toca informarlo y cumplimentarlo debidamente.

1. Asignamos la clasificación establecida por el empleado a la incidencia.
2. Buscamos un grupo de agentes a quienes asignarles la incidencia.
3. Enviamos correo electrónico al empleado informando que su petición ha sido recibida.
4. Si hay ficheros adjuntos, los copiamos al sistema de ficheros de Siebel. Esto requiere llamar a otra función de Siebel que codifica el fichero y relaciona su existencia con un registro en una tabla de la base de datos.
5. Por último informaremos todos los campos requeridos según la clasificación seleccionada.

Respecto este último punto, aclarar que nosotros no realizamos ningún tipo de validación sobre estos campos variables, ya que la validación se hace en la capa de la intranet.

Una vez creada la incidencia en Siebel, se retorna un fichero XML indicando el resultado de la operación, que se espera sea exitoso. En caso contrario, bien porque no se haya podido crear la incidencia o bien porque no se ha recibido correctamente el fichero adjunto, se devuelve un código de error indicando específicamente qué ha sucedido. Dicho error es interpretado por la intranet y muestra al usuario un mensaje por pantalla.

3.4.2.1 DIAGRAMA DE SECUENCIA DE `CREAR_INCIDENCIA`

En el siguiente diagrama podemos ver cómo funciona el método `CREAR_INCIDENCIA`.

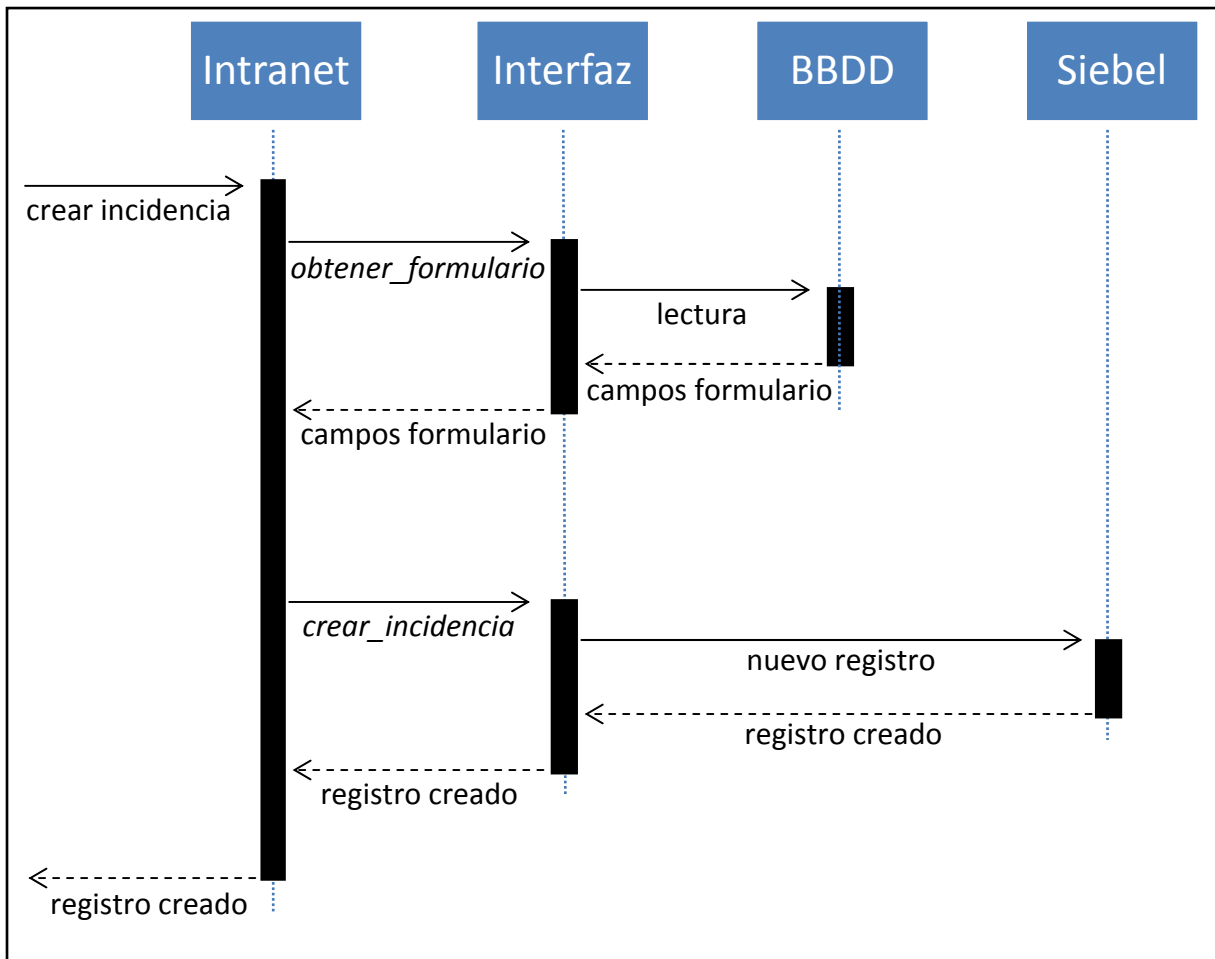


Figura 17: Diagrama del método CREAM_INCIDENCIA.

3.4.3 LISTAR INCIDENCIAS

El método **LISTAR_INCIDENCIA** accede a la base de datos de Siebel para recuperar la lista de incidencias de un usuario. El usuario podrá buscar filtrando por los siguientes elementos:

- Rango de fechas: selecciona las incidencias creadas entre dos fechas (ambas incluidas).
- Oficina: selecciona las incidencias pertenecientes a los empleados de una oficina, siempre y cuando el empleado pertenezca a dicha oficina.
- Estado: selecciona las incidencias con el estado Abierto o Cerrado.

Si el empleado hace uso de alguno de los filtros, sólo se mostrarán las incidencias que cumplan las condiciones de dichos filtros.

A continuación detallamos la información que se mostrará de cada incidencia:

- Número de incidencia: código identificador de la incidencia que es automáticamente asignado por Siebel.
- Usuario: usuario creador de la incidencia.
- Título: asunto o descripción de la incidencia.
- Fecha de creación: fecha en la que se generó la incidencia.
- Fecha de cierre: fecha en la que se cerró la incidencia. No aplica en el caso que la incidencia tenga estado Abierto.
- Estado: estado en el que se encuentra la incidencia, puede ser Abierto o Cerrado.
- Sub-estado: sub-estado de la incidencia. Los valores posibles son:
 - Pendiente de contact center.
 - Pendiente espera datos cliente¹.
- Permite reapertura: este campo nos dirá si la incidencia puede ser reabierta o no. Los valores posibles son Si o No. El motivo para querer reabrir una incidencia puede ser una reclamación.

3.4.3.1 DIAGRAMA DE SECUENCIA DE LISTAR_INCIDENCIA

En el siguiente diagrama podemos ver cómo funciona el método LISTAR_INCIDENCIA.

1. Puede darse el caso que el agente requiera datos adicionales. Tras ponerse en contacto con el cliente, la incidencia toma el sub-estado Pendiente espera datos cliente.

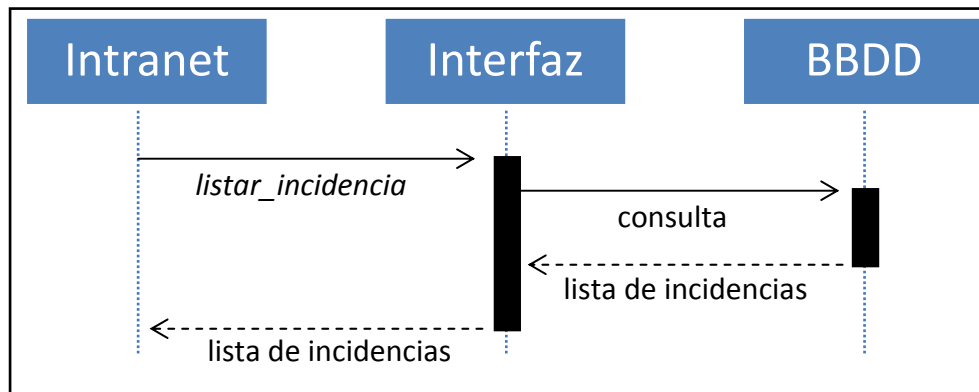


Figura 18: Diagrama del método LISTAR_INCIDENCIA.

3.4.4 CONSULTAR INCIDENCIA

El método `CONSULTAR_INCIDENCIA` accede a Siebel para recuperar toda la información de una incidencia.

Este punto era causante de preocupación para el cliente, puesto que no tenía muy claro qué información de la incidencia desea mostrar. Es por este motivo que se ha decidido hacerlo dinámico, es decir, desde Siebel, un agente administrador puede definir qué campos se van a enviar cuando desde la intranet se haga una llamada al método `CONSULTAR_INCIDENCIA`. Además dichos cambios se pueden hacer en caliente, no hay necesidad de detener el servicio.

Aun así, pasamos a detallar una lista de los campos que se mostrarán con certeza, pues no son los campos que preocupa al cliente:

- Número de incidencia: código identificador de la incidencia que es automáticamente asignado por Siebel.
- Usuario: usuario creador de la incidencia.
- Título: asunto o descripción de la incidencia.
- Fecha de creación: fecha en la que se generó la incidencia.
- Fecha de cierre: fecha en la que se cerró la incidencia. No aplica en el caso que la incidencia tenga estado Abierto.
- Estado: estado en el que se encuentra la incidencia, puede ser Abierto o Cerrado.

- Sub-estado: sub-estado de la incidencia. Los valores posibles son:
 - Pendiente de contact center.
 - Pendiente espera datos cliente.

3.4.4.1 DIAGRAMA DE SECUENCIA DE CONSULTAR_INCIDENCIA

En el siguiente diagrama podemos ver cómo funciona el método CONSULTAR_INCIDENCIA.

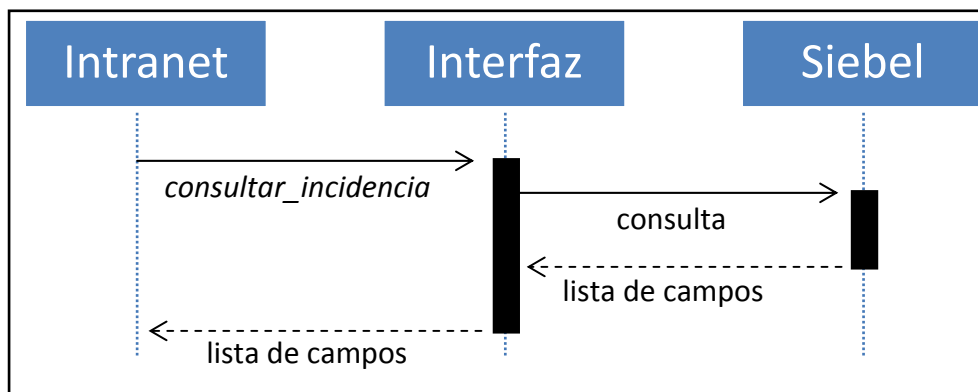


Figura 19: Diagrama del método CONSULTAR_INCIDENCIA.

3.4.5 MODIFICAR INCIDENCIA

El método MODIFICAR_INCIDENCIA accede a Siebel para añadir o modificar la información de una incidencia.

El empleado tiene un campo de texto libre y la opción de anexar archivos adjuntos a la incidencia. Una vez confirmados los cambios, Siebel registra una nueva actividad en la incidencia y anexa el texto y los ficheros adjuntos. Si se ha añadido la información y se han adjuntado correctamente los ficheros, se devolverá un código indicando a la intranet que la operación se ha realizado con éxito. En caso contrario, se devolverá el código de error específico que indica el motivo del error.

3.4.5.1 DIAGRAMA DE SECUENCIA DE MODIFICAR_INCIDENCIA

En el siguiente diagrama podemos ver cómo funciona el método MODIFICAR_INCIDENCIA.

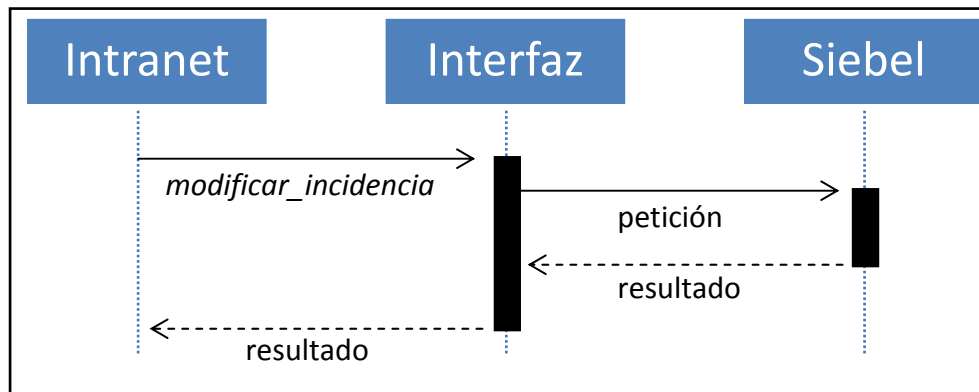


Figura 20: Diagrama del método MODIFICAR_INCIDENCIA.

3.4.6 RECLAMAR INCIDENCIA

El método RECLAMAR_INCIDENCIA accede a Siebel para añadir una reclamación. Normalmente el empleado genera una reclamación cuando cree que la solución a su petición se está demorando más de lo esperado o bien cuando la incidencia ha sido cerrada con una solución no acorde a las expectativas.

Diferenciaremos estos dos casos en función del estado de la incidencia:

- Abierto: el empleado puede realizar la reclamación siempre que lo desee.
- Cerrado: el empleado sólo podrá realizar la reclamación si se encuentra en las siguientes 48 horas hábiles tras haber sido cerrada. Este valor de 48 horas puede ser modificado en caliente por cualquier agente administrador de Siebel.

El empleado tiene un campo de texto libre para exponer los motivos de su reclamación y opcionalmente puede anexar archivos adjuntos a la reclamación. Una vez confirmados los cambios, Siebel registra una nueva actividad de reclamación en la incidencia y anexa el texto y los ficheros adjuntos.

Además, si la incidencia está cerrada, se reabrirá siempre y cuando aún se encuentre dentro del plazo permitido y se registrará el suceso de reapertura. Si se encuentra fuera del plazo establecido, el sistema no le permitirá hacer una reclamación sobre dicha incidencia.

Si se ha añadido la información y se han adjuntado correctamente los ficheros, se devolverá un código indicando a la intranet que la operación se ha realizado con éxito. En caso contrario, se devolverá el código de error específico que indica el motivo del error.

3.4.5.1 DIAGRAMA DE SECUENCIA DE RECLAMAR_INCIDENCIA

En el siguiente diagrama podemos ver cómo funciona el método RECLAMAR_INCIDENCIA.

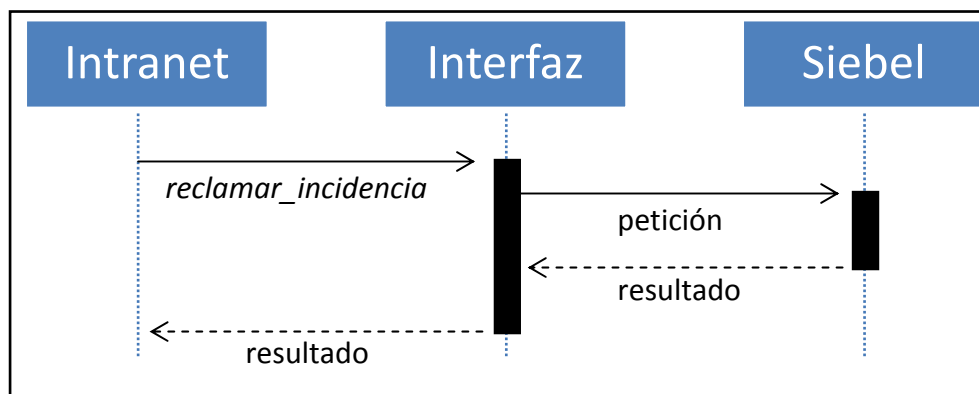


Figura 21: Diagrama del método RECLAMAR_INCIDENCIA.

4. DISEÑO TÉCNICO

En este capítulo se estudiarán los siguientes aspectos:

- Fundamentos técnicos: se hará una descripción de las herramientas utilizadas en el desarrollo de la aplicación.
- Perfil de usuario: se describirán los perfiles de usuario que tendrán acceso a la aplicación.
- Diseño técnico de la aplicación: se explica cómo es la información que se recibe y envía para cada método.
- Ejemplos de ficheros XML: se mostrarán algunos ejemplos para ver cómo son los ficheros que se envían y reciben para cada uno de los métodos.

4.1 FUNDAMENTOS TÉCNICOS

En el siguiente esquema podemos ver la relación entre las tecnologías usadas para realizar este proyecto.

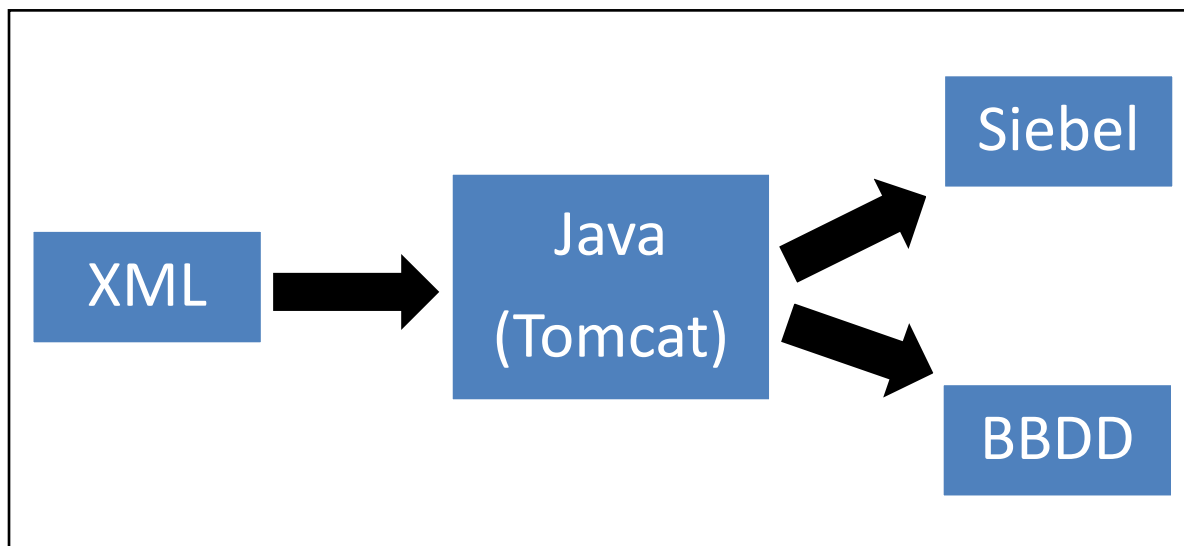


Figura 22: Relación entre las tecnologías usadas.

4.1.1 JAVA

Antes de ver porque se ha escogido Java para desarrollar este proyecto, hagamos una pequeña descripción de sus ventajas en programación web.

- Lenguaje de alto nivel muy potente.
- Multiplataforma.
- Es seguro.
- Orientado a objetos.
- El JDK¹ es gratuito, sin licencias.
- Fácil acceso a bases de datos.
- Muchas librerías.

Entre todas sus ventajas, podemos destacar el hecho que sea multiplataforma. Esto facilita mucho las tareas de desarrollo, puesto que los equipos en los que se ha desarrollado tienen como sistema operativo Windows XP, y el servidor tiene sistema operativo Solaris 9 (Linux).

La aplicación se ha ido desarrollando y testeando a la vez, es decir, cuando se ha terminado de programar un método, se han aplicado los cambios en el servidor y se ha probado como si se tratase del entorno real. Es por este motivo que valoramos que Java sea multiplataforma.

Por otro lado, al escoger Java, tienes a tu alcance mucha documentación y librerías que pueden facilitar las tareas de desarrollo.

Otro motivo por el que se ha escogido Java, es porque se integra perfectamente con el servidor web Tomcat. A continuación veremos porqué hemos elegido este servidor de aplicaciones.

1. *Java Development Kit*, set de herramientas para desarrollar en Java.

4.1.2 TOMCAT

A continuación se detallan algunas de las características de Tomcat

- Servidor Web autónomo.
- Multiplataforma. Ha sido desarrollado en Java, por lo que funcionará en cualquier sistema operativo que disponga de máquina virtual de Java.
- Es gratuito.
- Sencillo de configurar.
- Abundante documentación en internet.

En nuestro caso hemos escogido Tomcat por ser una herramienta gratuita y sobre todo por ser una herramienta que ya conocemos, puesto que ya ha sido utilizada en otros proyectos.

Esto nos ha permitido ahorrar tiempo en conocer la aplicación, instalación, configuración, documentación, etc.

4.1.3 XML

XML (eXtensible Markup Language), significa lenguaje de marcas extensible. Es una tecnología sencilla que tiene a su alrededor otras que la complementan y la hacen mucho más grande y con unas posibilidades mucho mayores. Tiene un papel muy importante en la actualidad ya que permite la compatibilidad entre sistemas para compartir la información de una manera segura, fiable y fácil.

Se compone de etiquetas o tags con una estructura jerárquica en forma de árbol. Cada etiqueta contiene una parte de la información que se desea transmitir.

Algunas de sus muchas ventajas son:

- Es extensible: Después de diseñado y puesto en producción, es posible extenderlo con la adición de nuevas etiquetas.
- Estructura sencilla, fácil de entender.

- Multiplataforma. Podemos comunicar aplicaciones de distintas plataformas sin que importe el origen de datos.
- Abundante documentación en internet.

La estructura de los ficheros XML y sus etiquetas son definidos a partir de otros ficheros llamados XSD. En el siguiente apartado explicaremos brevemente que son estos ficheros. A continuación podemos ver un ejemplo de un fichero XML, más adelante sin embargo, veremos los ficheros XML utilizados en este proyecto.

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE Edit_Mensaje SYSTEM "Edit_Mensaje.dtd">

<Edit_Mensaje>
  <Mensaje>
    <Remitente>
      <Nombre>Nombre del remitente</Nombre>
      <Mail>Correo del remitente </Mail>
    </Remitente>
    <Destinatario>
      <Nombre>Nombre del destinatario</Nombre>
      <Mail>Correo del destinatario</Mail>
    </Destinatario>
    <Texto>
      <Asunto>
        Este es mi documento con una estructura muy sencilla
        no contiene atributos ni entidades...
      </Asunto>
      <Parrafo>
        Este es mi documento con una estructura muy sencilla
        no contiene atributos ni entidades...
      </Parrafo>
    </Texto>
  </Mensaje>
</Edit_Mensaje>
```

Figura 23: Ejemplo de fichero XML.

4.1.3.1 XSD

XSD (XML Schema Definition Language), significa lenguaje de definición de esquema XML. Describe la estructura y las restricciones de los contenidos de los documentos XML. Se consigue así una percepción del tipo de documento con un nivel alto de abstracción.

Este fichero define los tipos de datos que van a contener cada una de las distintas etiquetas del fichero XML.

A continuación podemos ver un ejemplo de un fichero XSD, más adelante sin embargo, veremos los ficheros XSD utilizados en este proyecto.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
  <xsd:element name="Libro">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="Título" type="xsd:string"/>
        <xsd:element name="Autores" type="xsd:string" maxOccurs="10"/>
        <xsd:element name="Editorial" type="xsd:string"/>
      </xsd:sequence>
      <xsd:attribute name="precio" type="xsd:double"/>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

Figura 24: Ejemplo de fichero XSD.

4.1.4 SIEBEL

No es parte de este proyecto alterar o diseñar en Siebel, ya que es la aplicación con la que interactuamos. Desde nuestra aplicación hacemos llamadas a métodos o flujos de trabajo existentes en Siebel.

4.1.5 BBDD

No es parte de este proyecto alterar o diseñar la base de datos de Siebel. Nuestro cometido es interactuar con una serie de tablas que ya vienen predefinidas en un sistema CRM estándar.

Las tablas a las que accedemos son:

- S_SRV_REQ: es la tabla principal de Siebel, guarda las incidencias.
- S_EVT_ACT: esta tabla está relacionada con la anterior, guarda las actividades de cada una de las incidencias.
- S_CONTACT: aquí guardamos los contactos, en nuestro caso es una tabla de empleados.
- S_ORG_EXT: tabla de organizaciones, en nuestro caso podemos decir que es donde guardamos las oficinas.

4.2 PERFIL DE USUARIO

Sólo hay 1 perfil de usuario en la aplicación. Es decir, cada empleado tiene su nombre de usuario y su contraseña, pero todos tienen el mismo perfil, todos pueden hacer lo mismo.

Sin embargo, este apartado de perfiles de usuario está destinado a hablar de los perfiles de usuario en Siebel.

Como hemos ido viendo a lo largo del documento, hay varios parámetros de la aplicación que presumiblemente van a ser modificados en el futuro o van a requerir alternar su valor unos meses al año. Por este motivo se decidió hacer una lista de todos estos parámetros y hacer posible que todos ellos sean modificables desde Siebel.

Esta solución es muy cómoda, pues requiere un pequeño esfuerzo al desarrollar la aplicación, que más tarde es recompensado, pues permite al cliente una gran libertad de movimientos y permite que pueda realizar pequeños cambios de forma autónoma, sin depender de los desarrolladores de la aplicación.

Otro aspecto muy importante es que todos estos cambios se pueden realizar en caliente y por lo tanto resulta muy cómodo para el cliente ya que no detiene la actividad del servicio durante el cambio.

Dichos cambios también pueden ser programados por fechas, por ejemplo, recordemos que el número de horas hábiles para poder realizar una reclamación sobre una incidencia cerrada era un parámetro de los que estamos hablando. Pues bien, el cliente puede desear que durante los meses de verano el periodo máximo de reclamación sea 8 horas, y durante el resto del año desea 14 horas. Para este tipo de situaciones no es necesario que el cambio se haga manualmente, puede ser programado.

Los perfiles de usuario en Siebel son 2:

- Agente: puede acceder a las incidencias y trabajarlas.
- Administrador: además de acceder a las incidencias, tiene unas pantallas donde puede administrar todos los parámetros de la aplicación que eran susceptibles de ser modificados en el tiempo.

4.3 DISEÑO TÉCNICO DE LA APLICACIÓN

A continuación vamos a detallar la estructura del fichero XML de entrada y de salida para cada uno de los métodos.

4.3.1 MÉTODOS

Método	Descripción
OBTENER_FORMULARIO	Envía campos obligatorios para una clasificación dada.
CREAR_INCIDENCIA	Crea una nueva incidencia en Siebel.
LISTAR_INCIDENCIA	Lista las incidencias de un empleado u oficina.
CONSULTAR_INCIDENCIA	Devuelve información sobre una incidencia.
MODIFICAR_INCIDENCIA	Añade información o modifica una incidencia.
RECLAMAR_INCIDENCIA	Reclama una incidencia.

Figura 25: Descripciones breves de los métodos.

4.3.2 CARACTERÍSTICAS COMUNES

Todos los ficheros XML comparten la misma estructura principal.

- Nodo raíz: <interfaz>
- Nodo identificador de la petición o método: <peticion>
- Nodo identificador de usuario: <idUsuario>

Ejemplo:

```
<?xml version="1.0" encoding="UTF-8"?>
<interfaz>
  <peticion>OBTENER_FORMULARIO</peticion>
  <idUsuario>123456A</idUsuario>
</interfaz>
```

La primera línea es el encabezado típico de un fichero XML.

4.3.3 FICHEROS ANEXOS

Anteriormente hemos mencionado que los métodos CREAM_INCIDENCIA, MODIFICAR_INCIDENCIA y RECLAMAR_INCIDENCIA permiten anexar documentos al empleado.

Para conseguir que toda la información viaje conjuntamente una solución es convertir el fichero en cuestión a Base64 (a continuación explicamos qué es Base64). De esta manera estamos transformando un fichero de cualquier formato (como por ejemplo una imagen, un documento office, un fichero PDF, etc.) en una cadena de texto que nos permite poder insertarla en el fichero XML juntamente con el resto de la información.

4.3.3.1 BASE64

Base64 es un sistema de numeración posicional que usa 64 como base. Es un sistema propicio para usar en codificaciones de correos electrónicos y ficheros. Todas las variantes famosas que se conocen con el nombre de Base64 usan el rango de caracteres A-Z, a-z y 0-9.

La codificación Base64 puede ser útil cuando el tamaño de la información de identificación usada en un entorno HTTP es bastante grande.

4.3.4 CÓDIGOS DE ERROR

En caso de error en cualquiera de los métodos, se retorna un fichero XML con una estructura como la siguiente:

```
<?xml version="1.0" encoding="UTF-8"?>
<i9800>
  <peticion>OBTENER_FORMULARIO</peticion>
  <idUserario>123456A</idUserario>
  <error>E001</error >
  <descripcion> Debido a una incidencia técnica, el servicio de
    formularios de SAU no está disponible. Disculpen las molestias.
  </descripcion>
</i9800>
```

Los códigos de respuesta posibles son los siguientes:

Código	Descripción
E001	Debido a una incidencia técnica, el servicio de formularios de SAU no está disponible. Disculpen las molestias.
E002	El usuario no tiene autorización para acceder a la información solicitada.
E003	La operación solicitada está fuera del plazo máximo permitido.
E004	No se ha encontrado la incidencia solicitada.
E005	No se puede añadir información a la incidencia porque está cerrada. Para reabrir la incidencia debe realizar una reclamación.
E006	No se han encontrado registros en la base de datos.

Figura 26: Códigos de error de la aplicación.

Cada uno de estos errores se muestra en una situación de error determinada, algunas de estas situaciones son:

- Error E001: es el error más genérico de la aplicación, ya que puede darse con multitud de errores, como por ejemplo cuando el servicio no esté disponible, bien por estar en tareas de mantenimiento o actualizaciones, o bien por tener complicaciones, inestabilidad, problemas de red, etc.
- Error E002: este código va a aparecer en el caso que un usuario desee acceder a una incidencia que no es suya ni de su oficina. En principio un usuario no tiene visibilidad de otras incidencias que no sean suyas o de su oficina, pero como nosotros no diseñamos la intranet del banco, es mejor cubrir todos los casos posibles de error.
- Error E003: aparecería si un usuario intenta reclamar una incidencia fuera del plazo permitido. Al igual que el error E002, se trata de una situación que debe estar contemplada en la intranet, por lo que hemos definido este error sólo por precaución.
- Error E004: este error solamente aparecería en el caso que una incidencia fuese eliminada. En Siebel no se eliminan las incidencias, así que esto sólo ocurriría si por accidente un usuario Supervisor eliminara una incidencia y posteriormente el empleado quisiera hacer una consulta sobre la incidencia.
- Error E005: este error aparece si un usuario desea modificar o añadir información a una incidencia cerrada. La propia intranet no debería

permitir al empleado modificar una incidencia si su estado es cerrado, pero nuevamente hemos contemplado este caso por precaución.

- Error E006: este error se devuelve cuando un usuario no tiene incidencias en el periodo seleccionado de búsqueda.

Recordemos que estos errores no son mostrados directamente al usuario, sino que es el programador de la intranet el que interpreta nuestros errores para posteriormente tomar la acción oportuna e informar al usuario. Por ejemplo, no tiene sentido informar al usuario de un error por el hecho que no hayan incidencias en un periodo de búsqueda determinado, lo que sucede es que cuando la intranet recibe el código E006, directamente no se muestra nada en la pantalla.

4.3.5 PARÁMETROS ENTRADA/SALIDA

A continuación veremos detalladamente los parámetros de entrada y salida para cada uno de los métodos o peticiones nombrados durante este documento.

4.3.5.1 OBTENER FORMULARIO

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método a invocar.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idMcc	Identificador de la clasificación de la incidencia.	Alfanumérico	10	Sí

Figura 27: Parámetros de entrada para el método OBTENER_FORMULARIO.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idMcc	Identificador de la clasificación de la incidencia.	Alfanumérico	10	Sí
introduccion	Texto introductorio			
formulario	Estructura que contiene elementos de tipo “campo”.			
campo	Estructura que describe un campo del formulario. Contiene elementos de los siguientes tipos.			
Id	Identificador del campo.	Alfanumérico	15	Sí
descripcion	Texto descriptivo del campo.	Alfanumérico	50	Sí
tipo	Valores posibles: - Numérico - Decimal - Alfanumérico - Email - Fecha - Hora	Alfanumérico	15	Sí
requerido	Dice si el campo es obligatorio.	Booleano	N/A	Sí
tamanyo	Tamaño del campo	Numérico	3	Sí

Figura 28: Parámetros de salida para el método OBTENER_FORMULARIO.

La unidad de medida del campo “Longitud” dependerá del valor del campo “tipo”. La relación es la siguiente:

- Numérico: el campo “longitud” indica el número máximo de dígitos.
- Decimal: el campo “longitud” indica el número máximo de dígitos de la parte entera. Los decimales son siempre 3 y se rellenan con ceros por la derecha.
- Alfanumérico: el campo “longitud” indica el número máximo de caracteres aceptados.
- Email: el campo “longitud” indica el número máximo de caracteres aceptados.

- Fecha: el campo “longitud” indica el número máximo de caracteres aceptados.
- Hora: el campo “longitud” indica el número máximo de caracteres aceptados.

4.3.5.2 CREACIÓN DE INCIDENCIAS

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método a invocar.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idMcc	Identificador de la clasificación de la incidencia.	Alfanumérico	10	Sí
formulario	Estructura que contiene elementos de tipo “campo”.			
campo	Estructura que describe un campo de la incidencia. Contiene elementos de los siguientes tipos.			
id	Identificador del campo.	Alfanumérico	15	Sí
valor	Valor con el que se ha de informar el campo.	Alfanumérico	2000	Sí
anexos	Estructura que contiene elementos de tipo “anexo”.			
anexo	Estructura que describe un fichero anexo. Contiene elementos de los siguientes tipos.			
nombre	Nombre del fichero.	Alfanumérico	50	Sí
descripción	Descripción del contenido del fichero.	Alfanumérico	200	Sí
contenido	Contenido del fichero codificado en Base64.	Alfanumérico	-	Sí

Figura 29: Parámetros de entrada para el método CREAR_INCIDENCIA.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí

Figura 30: Parámetros de salida para el método CREAR_INCIDENCIA.

4.3.5.3 LISTAR INCIDENCIAS

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador usuario	Alfanumérico	7	Sí
idMcc	Identificador de la clasificación de la incidencia.	Alfanumérico	10	No
numOfi	Número de oficina.	Numérico	5	No
fecInicio	Fecha de inicio del filtro de búsqueda.	Fecha	10	No
fecFin	Fecha de fin del filtro de búsqueda.	Fecha	10	No
estado	Estado incidencia.	Alfanumérico	7	No

Figura 31: Parámetros de entrada para el método LISTAR_INCIDENCIA.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
incidencias	Estructura que contiene elementos de tipo “incidencia”			
incidencia	Estructura que define una incidencia. Contiene elementos de los siguientes tipos.			
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí

empleado	Nombre y apellidos del empleado creador de la incidencia.	Alfanumérico	100	Sí
titulo	Título o descripción de la incidencia.	Alfanumérico	100	Sí
fecCreacion	Fecha de creación	Fecha	10	Sí
estado	Estado de la incidencia.	Alfanumérico	7	Sí
subestado	Sub-estado de la incidencia.	Alfanumérico	30	Sí
flagReabrir	Indica si se permite la reapertura de la incidencia.	Booleano	N/A	Sí

Figura 32: Parámetros de salida para el método LISTAR_INCIDENCIA.

4.3.5.4 CONSULTAR INCIDENCIA

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método a invocar.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí

Figura 33: Parámetros de entrada para el método CONSULTAR_INCIDENCIA.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí
datos	Estructura que contiene elementos de tipo variable.			
estado	Estado de la incidencia.	Alfanumérico	7	Sí

flagReabrir	Indica si se permite la reapertura de la incidencia.	Booleano	N/A	Sí
-------------	--	----------	-----	----

Figura 34: Parámetros de salida para el método CONSULTAR_INCIDENCIA.

4.3.5.5 MODIFICAR INCIDENCIA

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método a invocar.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí
descripcion	Información para añadir.	Alfanumérico	500	Sí
anexos	Estructura que contiene elementos de tipo “anexo”.			
anexo	Estructura que describe un fichero anexo. Contiene elementos de los siguientes tipos.			
nombre	Nombre del fichero.	Alfanumérico	50	Sí
descripción	Descripción del contenido del fichero.	Alfanumérico	200	Sí
contenido	Contenido del fichero codificado en Base64.	Alfanumérico	-	Sí

Figura 35: Parámetros de entrada para el método MODIFICAR_INCIDENCIA.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí

Figura 36: Parámetros de salida para el método MODIFICAR_INCIDENCIA.

4.3.5.6 RECLAMAR INCIDENCIA

- Parámetros de entrada a la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método a invocar.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí
descripcion	Texto de la reclamación.	Alfanumérico	500	Sí
anexos	Estructura que contiene elementos de tipo “anexo”.			
anexo	Estructura que describe un fichero anexo. Contiene elementos de los siguientes tipos.			
nombre	Nombre del fichero.	Alfanumérico	50	Sí
descripción	Descripción del contenido del fichero.	Alfanumérico	200	Sí
contenido	Contenido del fichero codificado en Base64.	Alfanumérico	-	Sí

Figura 37: Parámetros de entrada para el método RECLAMAR_INCIDENCIA.

- Parámetros de salida de la aplicación:

Tag XML	Descripción	Formato	Longitud	Obligatorio
peticion	Nombre del método.	Alfanumérico	15	Sí
idUsuario	Identificador de usuario.	Alfanumérico	7	Sí
idIncidencia	Identificador de la incidencia.	Alfanumérico	15	Sí

Figura 38: Parámetros de salida para el método RECLAMAR_INCIDENCIA.

5. PRUEBAS Y PROBLEMAS DETECTADOS

En este capítulo vamos a ver las pruebas que se ha ido haciendo a medida que avanzaba el desarrollo.

Veremos también cuáles han sido los problemas más importantes que han aparecido durante el desarrollo del proyecto y cómo se han solucionado.

5.1 PRUEBAS UNITARIAS

Una prueba unitaria es una forma de probar el correcto funcionamiento de un módulo de código. Esto sirve para asegurar que cada uno de los módulos funcione correctamente por separado.

Para realizar las pruebas he pensado qué es lo que puede fallar y he enfatizado las pruebas hacia ese punto.

Vamos a detallar las pruebas unitarias que se han realizado sobre los métodos no triviales:

- **Autentifica:** Sobre este método se han realizado pruebas no sólo para garantizar que un empleado solamente puede acceder a sus incidencias, sino también a las de su oficina. Para probar esto he lanzado peticiones con un empleado y un número de oficina a la que no pertenece dicho empleado y he comprobado cómo el sistema ha denegado la petición devolviendo el código de error pertinente.
- **LecturaXML:** Se han enviado distintos ficheros XML y he comprobado que se reciben y procesan correctamente siempre y cuando el fichero XML no contenga un carácter reservado en direcciones URL. Realizando estas pruebas se ha detectado un problema con un carácter reservado que merece la pena detallar más adelante en el apartado de problemas detectados.
- **Base64Coder:** Se ha comprobado que realiza la codificación de todo tipo de ficheros a Base64 correctamente. De todas formas se trata de una clase reutilizada de otro proyecto, con lo que ya se dispone de una garantía de buen funcionamiento.
- **EscribeXML:** Lo mismo sucede con la clase que contiene los métodos necesarios para generar ficheros XML, es una clase reutilizada y sobradamente testeada en otros proyectos.
- **Listar incidencias:** La clase que contiene el método LISTAR_INCIDENCIA ataca directamente a la base de datos de Siebel, así que lo que he hecho es revisar la consulta SQL y garantizar que está bien hecha y que es óptima. Se ha comprobado que en ningún caso se generan bloqueos en la base de datos.
- **El resto de métodos que pueden ser invocados desde la intranet funcionan de otra manera que el anterior.** Acceden a Siebel abriendo una sesión como si de un usuario de Siebel se tratase. Se ha comprobado que todos los métodos abren y cierran sus respectivas sesiones correctamente.
- **Adjuntar fichero:** se han hecho pruebas con distintos ficheros, tamaños, formatos, etc. Tras algunos problemas para conseguir anexar los ficheros al sistema de archivos de Siebel, se puede decir que funciona correctamente. Dichos problemas se detallan más adelante en el apartado de problemas detectados.

5.2 PRUEBAS INTEGRADAS

Una vez hechas las pruebas unitarias se pasa a probar todo el conjunto a la vez. He definido un flujo de operaciones y se ha ejecutado varias veces para asegurar que todo funciona correctamente. No sólo se trata de ver que el fichero XML obtenido es correcto, sino que más bien es una tarea de ver que no se están dejando sesiones abiertas, objetos sin eliminar tras su uso, etc., cosas que en definitiva podrían resultar desastrosos para el rendimiento de Siebel.

Esta parte de las pruebas ha requerido el seguimiento de muchos logs (trazas) en el servidor Siebel para comprobar que no se ha pasado nada por alto.

5.3 PRUEBAS DE VALIDACIÓN

En el anexo de esta memoria se puede encontrar el documento con las pruebas que se han realizado para validar el correcto funcionamiento de la aplicación.

5.4 PRUEBA DE ESTRÉS

Hasta ahora, el servidor de Siebel atiende aproximadamente entre 40 y 50 conexiones simultáneamente. Con la puesta en marcha de este proyecto, esta cifra se va a incrementar considerablemente. Es por esto que es sumamente importante realizar pruebas de estrés sobre el servidor.

Estas pruebas se realizan con un software específico que simula peticiones igual que lo harían los empleados desde la intranet.

Para realizar estas pruebas me puse en contacto con el departamento de base de datos de la empresa para que monitorizaran la actividad del servidor de base de datos, del servidor de aplicaciones de Siebel y del servidor donde alojamos nuestro proyecto Java. Mi preocupación estaba enfocada en el servidor de aplicaciones Siebel.

Se comenzó probando con 50 sesiones y se fueron añadiendo de 50 en 50 hasta llegar a 500. Afortunadamente los 3 servidores funcionaron correctamente y ninguno de ellos llegó a cifras de ocupación de procesador preocupantes.

Obviamente, si en un futuro se requirieran más conexiones, bien porque se añaden agentes o empleados, se debería realizar de nuevo pruebas de estrés para dictaminar si es necesario incorporar un servidor nuevo y/o un balanceador de carga de peticiones.

5.5 PROBLEMAS DETECTADOS

Durante el desarrollo del proyecto han ido apareciendo numerosas dudas que han requerido consultar documentación a través de Internet. Muchas de estas dudas se deben en gran parte al desconocimiento de la tecnología Java. Creo que puede ser beneficioso exponer algunos de estos problemas.

5.5.1 CODIFICACIÓN DE CARACTERES EN DIRECCIONES URL

Cuando se pasan variables en direcciones URL, a veces es necesario codificar ciertos caracteres. Esto es, reemplazar un carácter por su código ASCII en formato hexadecimal. A continuación, se presentan los caracteres que se codifican más frecuentemente:

Caracter	Código	Caracter	Código	Caracter	Código	Caracter	Código
(espacio)	%20	?	%3f	&	%26	%	%25
#	%23	/	%2f	<	%3c	>	%3e
:	%3a	/	%2f		%7c	;	%3b

Figura 39: Caracteres codificados en direcciones URL.

En mi caso, realizando las pruebas unitarias de lectura de ficheros XML vi que el carácter # presentaba problemas. Toda la parte del fichero que estuviese a continuación de un carácter # no era enviada.

Tras unas cuantas pruebas llegué a la conclusión que se trataba de un problema de carácter reservado. Así que consulté en internet la lista de caracteres prohibidos en URL y comprobé que efectivamente el carácter # es un carácter reservado en el envío URL.

En estos casos, y como se puede ver en la tabla anterior, tan sólo se ha de codificar a ASCII, es decir, tuve que sustituir # por %23. Tras comprobar que funcionaba correctamente se pasó un aviso al diseñador de la intranet para que tuviese en cuenta este problema y no se encontrase con los mismos problemas que yo.

En el anexo de esta memoria se pueden ver ejemplos de ficheros XML tanto de entrada como de salida, en ellos se puede comprobar que en Siebel es bastante común tener el carácter # para identificadores de registros.

5.5.2 ADJUNTAR ARCHIVOS EN SIEBEL

En mi opinión este ha sido el problema que más tiempo me ha hecho perder. Además de tiempo y dolores de cabeza, ha requerido el soporte y consejo del departamento de Sistemas para solucionarlo.

Durante las pruebas unitarias del método para adjuntar a Siebel los archivos recibidos desde la intranet comenzaron los problemas. Tras hacer varios cambios siempre recibía el mismo error, permiso denegado. Tras la frustración de no saber que estaba pasando, pedí ayuda al departamento de sistemas de la empresa. Comenzaron cambiando los permisos del usuario que realiza la operación, en este caso el usuario es Tomcat. Pero no era ese el problema...

Tras un par de días de pruebas, vimos que era imposible que pudiera adjuntar los archivos al sistema de ficheros de Siebel porque yo estaba trabajando en local en mi PC.

Me propusieron probar la aplicación en un servidor con un entorno de test para salir de dudas. Así lo hice, y tras varios problemas de permisos con el usuario, y con la ayuda del departamento de sistemas, conseguí que funcionara.

La conclusión es que no se pueden subir archivos al sistema de Siebel desde un PC en local. Se ha de hacer desde un servidor y se le han de conceder los permisos oportunos.

6. CONCLUSIÓN

Para terminar este documento hare una valoración del cumplimiento de los objetivos, así como de las variaciones sufridas respecto la planificación y las líneas de actuación futuras.

6.1 CUMPLIMIENTO DE LOS OBJETIVOS

Recordemos los objetivos del proyecto, enumerados en el capítulo 1, y veamos si se han cumplido o no.

1. Evitar el factor humano, que puede provocar errores.

Se ha logrado excluir el factor humano en el proceso de generación de incidencias. Objetivo cumplido.

2. Automatización del proceso de generación de incidencias.

Se ha automatizado el proceso de generación de incidencias, esto aporta fiabilidad y garantía de disponibilidad. Objetivo cumplido.

3. Auditar la vida de una incidencia.

Ahora es posible tener la historia de una incidencia con un solo clic. Objetivo cumplido.

4. Transparencia para el usuario.

Ahora no hay lugar a la desconfianza, pues el cliente obtiene los datos de un sistema informático. Objetivo cumplido.

5. Adquirir experiencia en la tecnología Java.

Es complicado cuantificar la experiencia adquirida con este proyecto porque Java es una tecnología muy amplia. Puedo decir que he tomado contacto con el lenguaje de programación y que he adquirido experiencia en la parte de Java orientado a desarrollo web. Objetivo cumplido.

6. Aprender cómo funciona la transmisión de información mediante ficheros XML.

Gracias al desarrollo de este proyecto he conocido y aprendido esta forma de transmitir información por la red. Objetivo cumplido.

7. Realizar un proyecto desde cero.

A pesar de haber requerido asistencia y soporte en algunos tramos del proyecto, he realizado este proyecto desde el inicio hasta el final. Objetivo cumplido.

6.2 VARIACIONES EN LA PLANIFICACIÓN

Este proyecto no ha sufrido variaciones en la planificación debido a que se comenzó tarde respecto a la planificación, y para el cliente era indispensable mantener la fecha de entrega. Así que se ha dedicado la jornada completa de trabajo a este proyecto.

Es por esto que, aparte de los problemas aparecidos durante el desarrollo, se ha hecho un gran esfuerzo para mantener la fecha de entrega tal cual estaba pactada al inicio de redactar este documento.

Finalmente se terminó con 2 semanas de retraso, pero el cliente dio su conformidad, pues gran parte de estas 2 semanas se debieron a cambios en las especificaciones de última hora. Dichos cambios surgieron al hacer las primeras pruebas del sistema conjuntamente con los desarrolladores de la intranet del banco.

6.3 LINEAS DE AMPLIACIÓN FUTURAS

Una de las más que factibles ampliaciones futuras será sin duda la posibilidad de explotar los datos de auditoría de las incidencias con un conjunto de informes.

Actualmente se proporcionan informes bajo demanda, pero son tan sólo extracciones de la base de datos. La ampliación que propongo sería permitir a los empleados (o a directores de zona o región) la extracción de informes desde la intranet. De esta forma podrían ver cuáles son las incidencias más habituales, diferenciarlas por meses, por zona, por oficinas, etc.

ANEXO. JUEGO DE PRUEBAS Y FICHEROS XML DE ENTRADA/SALIDA

• OBTENER FORMULARIO

Entrada	<pre><?xml version="1.0" encoding="UTF-8"?> <interfaz> <peticion>OBTENER_FORMULARIO</peticion> <idUsuario>X737416</idUsuario> <idMcc>1%23LLNB</idMcc> </ interfaz ></pre>
Resultado esperado	Se espera que se devuelva un fichero XML con un texto informativo (opcional) referente al mcc que hemos pedido. Después se presenta una estructura con la siguiente información obligatoria: id del campo, descripción, tipo, requerido y tamaño.
Salida	<pre><?xml version="1.0" encoding="UTF-8"?> < interfaz> <peticion> OBTENER_FORMULARIO</peticion></pre>

	<pre> <idUsuario>X737416</idUsuario> <idMcc>1#LLNB</idMcc> <introduccion>descripcion de prueba</introduccion> <formulario> <campo> <id>NumSR</id> <descripcion>numero que identifica la sr</descripcion> <tipo>Numérico</tipo> <requerido>TRUE</requerido> <tamanyo>12</tamanyo> </campo> <campo> <id>Area</id> <descripcion>Define el área</descripcion> <tipo>AlfaNumérico</tipo> <requerido>FALSE</requerido> <tamanyo>200</tamanyo> </campo> <campo> <id>idUsuario</id> <descripcion>identifica el usuario</descripcion> <tipo>AlfaNumérico</tipo> <requerido>TRUE</requerido> <tamanyo>21</tamanyo> </campo> <campo> <id>Descripcion</id> <descripcion>Descripcion</descripcion> <tipo>AlfaNumérico</tipo> <requerido>FALSE</requerido> <tamanyo>2000</tamanyo> </campo> </formulario> </ interfaz> </pre>
--	---

Figura 40: : Ejemplo ficheros XML entrada/salida para OBTENER_FORMULARIO.

• CREAR INCIDENCIA

Entrada	<pre> <?xml version="1.0" encoding="UTF-8"?> < interfaz > <peticion>CREAR_INCIDENCIA</peticion> <idUsuario>B520505</idUsuario> <idMcc>1-4DY-658</idMcc> <formulario> <campo> <id>Descripcion</id> <valor>prueba%20david</valor> </campo> </formulario> <anexos> <anexo> <nombre>NOMBRE_FICHERO.txt</nombre> <descripcion>Descripción%20del%20fichero%20enviado</descripcion> <contenido>aG9sYSBtZSBsbGFtbyBEYXZpZA==</contenido> </anexo> </anexos> </pre>
----------------	--

	</anexos> </ interfaz >
Resultado esperado	Se espera recibir un XML con el número de incidencia creada. En caso contrario se devolverá un código de error específico para cada caso.
Salida	<?xml version="1.0" encoding="UTF-8"?> < interfaz> <peticion>CREAR_INCIDENCIA</peticion> <idUsuario>B520505</idUsuario> <idIncidencia>1-101833171</ idIncidencia > </ interfaz>

Figura 41: Ejemplo ficheros XML entrada/salida para CREAR_INCIDENCIA.

• LISTAR INCIDENCIAS

Entrada	<?xml version="1.0" encoding="UTF-8"?> < interfaz > <peticion>LISTAR_INCIDENCIAS</peticion> <idUsuario>X737416</idUsuario> <idMcc>1-12WL</idMcc> <numOfi></numOfi> <fecInicio>21/01/2010</fecInicio> <fecFin>25/11/2011</fecFin> <estado>Cerrado</estado> </interfaz>
Resultado esperado	Se espera recibir un XML con un listado de incidencias que cumplan las condiciones de los filtros solicitados.
Salida	<?xml version="1.0" encoding="UTF-8"?>

	<pre> < interfaz> <peticion> LISTAR_INCIDENCIAS </peticion> <idUsuario>X737416</idUsuario> <incidencias> <incidencia> <idIncidencia>1-90455202</idIncidencia> <empleado>NURIA DOMENECH ROVIRA</empleado> <titulo /> <fecCreacion>2011-11-23 13:38:39.0</fecCreacion> <estado>Cerrado</estado> <subestado>Resuelto</subestado> <flagReabrir>FALSE</flagReabrir> </incidencia> <incidencia> < idIncidencia>1-37556322</ idIncidencia> <empleado>NURIA DOMENECH ROVIRA</empleado> <titulo /> <fecCreacion>2009-02-20 11:39:15.0</fecCreacion> <estado>Activo</estado> <subestado>Pendiente Colaborador</subestado> <flagReabrir>FALSE</flagReabrir> </incidencia> </incidencias> </interfaz> </pre>
--	--

Figura 42: Ejemplo ficheros XML entrada/salida para LISTAR_INCIDENCIAS.

• CONSULTAR INCIDENCIA

Entrada	<pre> <?xml version="1.0" encoding="UTF-8"?> < interfaz > <peticion>CONSULTAR_INCIDENCIA</peticion> <idUsuario>X737416</idUsuario> <idIncidencia>1-103588631</ idIncidencia> </interfaz> </pre>
Resultado esperado	Se espera recibir un XML con una serie de campos que nos proporcionan información sobre la incidencia. Como se comentó anteriormente, este listado de campos será variable y lo administrará un agente supervisor.
Salida	<pre> <?xml version="1.0" encoding="UTF-8"?> < interfaz> <peticion>CONSULTAR_INCIDENCIA</peticion> </pre>

	<pre> <idUsuario>X737416</idUsuario> <idIncidencia>1-103588631</idIncidencia> <datos> <estado>Cerrado</estado> <flagReabrir>TRUE</flagReabrir> </datos> </interfaz> </pre>
--	--

Figura 43: Ejemplo ficheros XML entrada/salida para CONSULTAR_INCIDENCIA.

• MODIFICAR INCIDENCIA

Entrada	<pre> <?xml version="1.0" encoding="UTF-8"?> < interfaz > <peticion>MODIFICAR_INCIDENCIA</peticion> <idUsuario>B520505</idUsuario> <idIncidencia>1-88894672</idIncidencia> <descripcion>Reclamación</descripcion> <anexos> <anexo> <nombre>NOMBRE_FICHERO.txt</nombre> <descripcion>Descripción%20del%20fichero%20enviado</descripcion> <contenido>aG9sYSBtZSBsbGFtbyBEYXZpZA==</contenido> <anexo> </anexos> </interfaz> </pre>
Resultado esperado	Se espera recibir un XML con el número de incidencia modificada. En caso contrario se devolverá un código de error específico para cada caso.
Salida	<pre> <?xml version="1.0" encoding="UTF-8"?> <interfaz> <peticion> MODIFICAR_INCIDENCIA</peticion> <idUsuario>B520505</idUsuario> <idIncidencia>1-101833171</ idIncidencia > </interfaz> </pre>

Figura 44: Ejemplo ficheros XML entrada/salida para MODIFICAR_INCIDENCIA.

• RECLAMAR INCIDENCIA

Entrada	<pre> <?xml version="1.0" encoding="UTF-8"?> < interfaz > <peticion>RECLAMAR_INCIDENCIA</peticion> <idUsuario>B520505</idUsuario> <idIncidencia>1-88894672</idIncidencia> <descripcion>Reclamación</descripcion> <anexos> <anexo> <nombre>NOMBRE_FICHERO.txt</nombre> <descripcion>Descripción%20del%20fichero%20enviado</descripcion> <contenido>aG9sYSBtZSBsbGFtbyBEYXZpZA==</contenido> <anexo> </anexos> </interfaz> </pre>
----------------	---

Resultado esperado	Se espera recibir un XML con el número de incidencia reclamada. En caso contrario se devolverá un código de error específico para cada caso.
Salida	<pre><?xml version="1.0" encoding="UTF-8"?> <interfaz> <peticion> RECLAMAR_INCIDENCIA</peticion> <idUserio>B520505</idUserio> <idIncidencia>1-101833171</ idIncidencia > </interfaz></pre>

Figura 45: Ejemplo ficheros XML entrada/salida para RECLAMAR_INCIDENCIA.

BIBLIOGRAFÍA

EN PAPEL

- Guía básica de Siebel proporcionada por el fabricante del CRM. Bookshelf Siebel 7.8
- “Curso de Java”. Darwin, Ian F. (Editorial Anaya) 2005.

ONLINE

- Curso Java orientado a web.
Eduardo. Última visita, 24 de Agosto de 2012.
www.edu4java.com/servlet.html
- Codificación de caracteres en url.
Grupo Caraytech. Última visita, 13 de Agosto de 2012.
http://www.e-planning.net/es/soporte/codificacion_caracteres_en_url.html
- Acrónimos y abreviaciones.
Wikipedia. Última visita, 24 de Agosto de 2012.
<http://es.wikipedia.org>
- Tutorial XML.
DesarrolloWeb. Última visita, 24 de Agosto de 2012.
<http://www.desarrolloweb.com/manuales/manual-introduccion-xml.html>
- Tutorial instalación y configuración Tomcat.
Programación. Última visita, 24 de Agosto de 2012.
http://www.programacion.com/articulo/tomcat_introduccion_134

TABLA DE FIGURAS

Figura 1: Servidor Siebel - Tomcat.....	7
Figura 2: Prioridad requisitos funcionales.....	22
Figura 3: Prioridad requisitos no funcionales.....	23
Figura 4: Características servidor	24
Figura 5: Características PC.	25
Figura 6: Fases de actuación por perfil.....	26
Figura 7: Tareas a desarrollar.	27
Figura 8: Diagrama de Gantt	28

Figura 9: Catalogación de riesgos	30
Figura 10: Plan de contingencia.....	31
Figura 11: Coste y dedicación de los recursos humanos.....	31
Figura 12: Coste de los recursos materiales.....	32
Figura 13: Desventajas envío formulario por correo electrónico.	36
Figura 14: Conexiones Interfaz.	37
Figura 15: Métodos llamados desde la intranet.....	38
Figura 16: Diagrama del método OBTENER_FORMULARIO.	39
Figura 17: Diagrama del método CREAR_INCIDENCIA.	41
Figura 18: Diagrama del método LISTAR_INCIDENCIA.	43
Figura 19: Diagrama del método CONSULTAR_INCIDENCIA.	44
Figura 20: Diagrama del método MODIFICAR_INCIDENCIA.	45
Figura 21: Diagrama del método RECLAMAR_INCIDENCIA.....	46
Figura 22: Relación entre las tecnologías usadas.....	47
Figura 23: Ejemplo de fichero XML.....	50
Figura 24: Ejemplo de fichero XSD.	51
Figura 25: Descripciones breves de los métodos.	53
Figura 26: Códigos de error de la aplicación.	55
Figura 27: Parámetros de entrada para el método OBTENER_FORMULARIO.....	56
Figura 28: Parámetros de salida para el método OBTENER_FORMULARIO.	57
Figura 29: Parámetros de entrada para el método CREAR_INCIDENCIA.....	58
Figura 30: Parámetros de salida para el método CREAR_INCIDENCIA.	58
Figura 31: Parámetros de entrada para el método LISTAR_INCIDENCIA.....	59
Figura 32: Parámetros de salida para el método LISTAR_INCIDENCIA.	60
Figura 33: Parámetros de entrada para el método CONSULTAR_INCIDENCIA.....	60
Figura 34: Parámetros de salida para el método CONSULTAR_INCIDENCIA.	61
Figura 35: Parámetros de entrada para el método MODIFICAR_INCIDENCIA.....	61
Figura 36: Parámetros de salida para el método MODIFICAR_INCIDENCIA.	61
Figura 37: Parámetros de entrada para el método RECLAMAR_INCIDENCIA.....	62
Figura 38: Parámetros de salida para el método RECLAMAR_INCIDENCIA.	62
Figura 39: Caracteres codificados en direcciones URL.....	66
Figura 40: : Ejemplo ficheros XML entrada/salida para OBTENER_FORMULARIO.	70
Figura 41: Ejemplo ficheros XML entrada/salida para CREAR_INCIDENCIA.	71
Figura 42: Ejemplo ficheros XML entrada/salida para LISTAR_INCIDENCIAS.	72
Figura 43: Ejemplo ficheros XML entrada/salida para CONSULTAR_INCIDENCIA.	73
Figura 44: Ejemplo ficheros XML entrada/salida para MODIFICAR_INCIDENCIA.	73
Figura 45: Ejemplo ficheros XML entrada/salida para RECLAMAR_INCIDENCIA.	74