TENDENCIAS EN LA EVOLUCION DE LAS REDES INTELIGENTES: CONVERGENCIA E INTEROPERABILIDAD DE SERVICIOS"

GERARDO CAJAS RUIZ
MAURICIO CONSTAÍN VILLEGAS

Anexo A – Tecnologías facilitadoras de la integración RI/Internet.
Anexo B – Documentación del software.
Anexo C – Manual de usuario y de instalación.

Director Mag. RAFAEL RENGIFO

UNIVERSIDAD DE CAUCA
FACULTAD DE INGENIERIA ELECTRÓNICA Y TELECOMUNICACIONES
GRUPO DE REDES INTELIGENTES Y SERVICIOS AVANZADOS DE TELECOMUNICACIONES
MAYO DE 2003

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

A.1 APIs ESTANDAR DE ACCESO A LA RED

A.1.1 OSA/Parlay

La siguiente Figura 1 muestra la ubicación de las APIs dentro de un esquema de red horizontal:

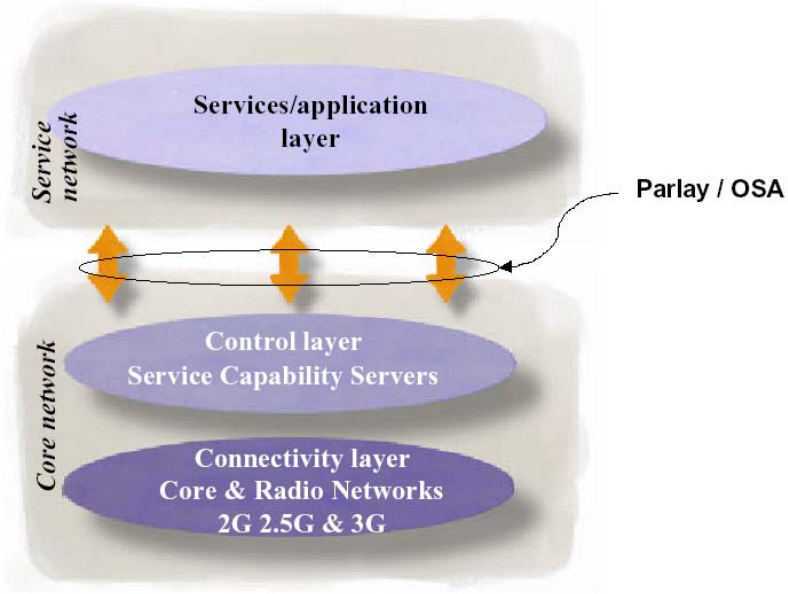

Figura 1: Ubicación de las APIs Parlay/OSA en una red de Telecomunicaciones.

Las APIs Parlay/OSA están divididas en tres partes:

- Interfaces entre las Aplicaciones y el marco de trabajo (Framework) Parlay/OSA, estas proveen los mecanismos básicos para hacer uso de las capacidades de la red a las aplicaciones, por ejemplo, autenticación.

- Interfaces entre las Aplicaciones y las Capacidades Características del Servicio (SCF – Service Capability Feature), las cuales son servicios individuales que pueden ser requeridos por el cliente para permitir la ejecución de aplicaciones de terceros a través de la interfaz, por ejemplo: servicios de mensajería.
- Interfaz entre el marco de trabajo y las Capacidades Características del Servicio, que proveen mecanismos necesarios para el soporte multivendedor.

Estas interfaces están ubicadas en los puntos 1, 2, y 3 de la Figura 2, las demás interfaces no están dentro del alcance de la recomendación.

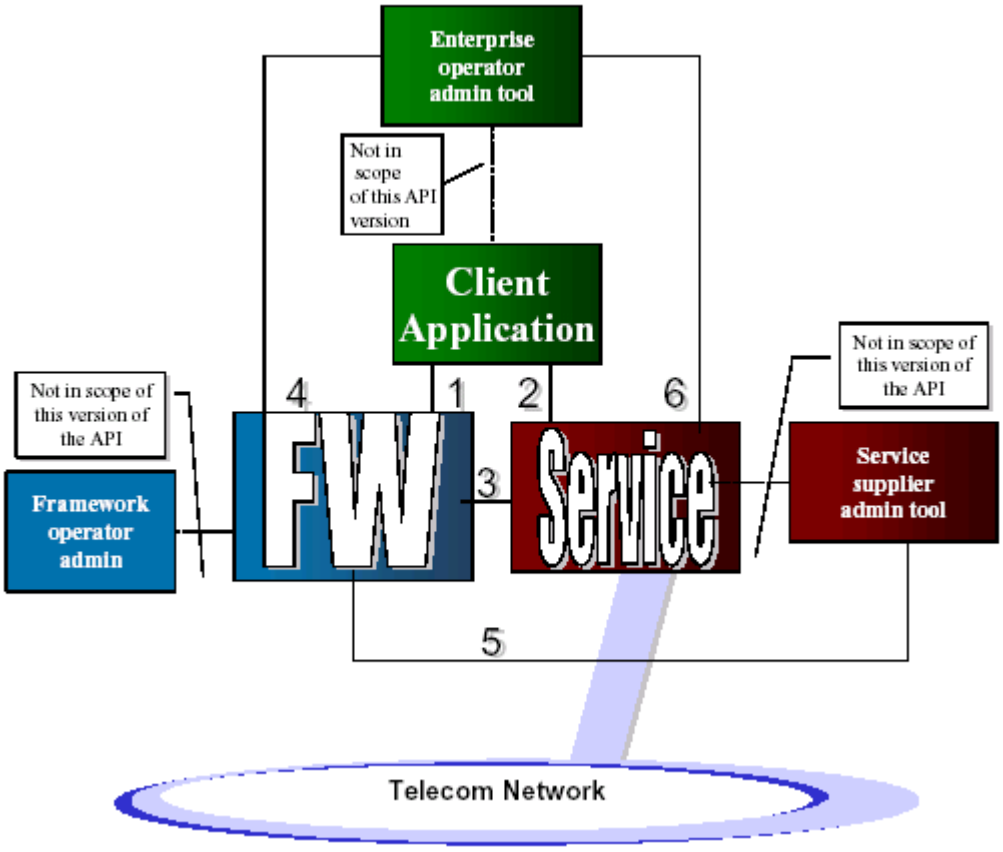

Figura 2: Alcance de la especificación Parlay/OSA

Las especificaciones Parlay 3.0, 3GPP Release 4 y ETSI versión 1 constan de las siguientes 12 partes:

Tabla 1: Partes de la especificación Parlay/OSA

#	SCF	Descripción	Comentarios
1	General	Introducción a la metodología usada	
2	Datos Comunes	Definiciones genéricas de datos usadas en otras partes	
3	Marco de Referencia	Define capacidades de infraestructura como autenticación, descubrimiento de SCF, registro de SCF, manejo de fallas, etc.	
4	Control de llamada	Define la familia de control de llamada con capacidades desde el establecimiento de llamada hasta como manipular llamadas de conferencia multimedia	El control de llamada para conferencias y multimedia no forma parte de 3GPP Rel.4 OSA ya que se considera que esta función reside fuera de su red.
5	Interacción de Usuario	SCF para obtener información del usuario final, enviar anuncios, enviar mensajes de texto corto, etc.	
6	Localización de Usuario/Estado de Usuario	SCF para obtener información de localización y estado del usuario	
7	Capacidades del terminal	SCF para obtener las de capacidades soportadas por un terminal de usuario	
8	Control de datos de sesión	SCF para manejar sesiones de datos	
9	Mensajera genérica	SCF para acceder buzones de mensajes	No forma parte de 3GPP Rel.4 OSA ya que se considera que esta

			función reside fuera de su red.
10	Gestión de conectividad	SCF para proveer QoS	No forma parte de 3GPP Rel.4 OSA ya que se considera que esta función reside fuera de su red.
11	Gestión de cuenta	SCF para acceder las cuentas del usuario final	
12	Tarificación basada en contenido	SCF para cobrar por el uso de aplicaciones o datos	

A.1.1.1 La arquitectura lógica de Parlay:

Los elementos claves a distinguir dentro de la arquitectura lógica son: Aplicaciones, Servidores de Aplicaciones, Servidores de Capacidades de Servicio (SCS - Service Capability Servers), los elementos del marco Parlay/OSA y los elementos del núcleo de la red como se muestra en la Figura 3.

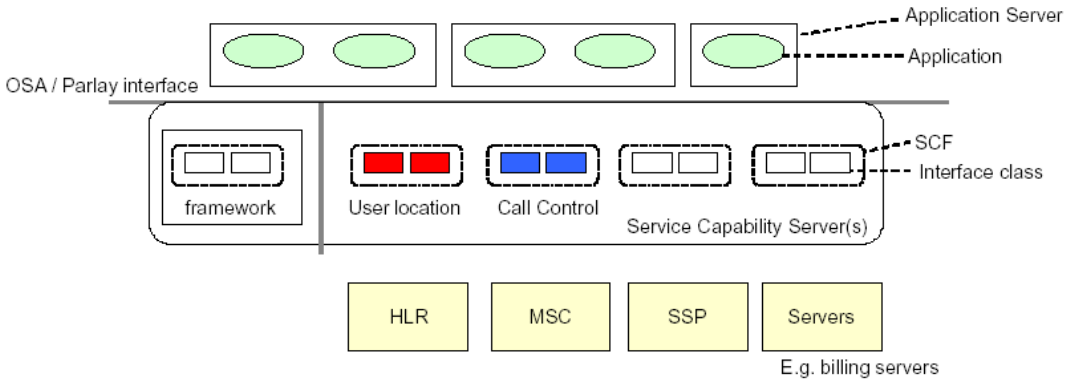

Figura 3: Entidades lógicas involucradas en el marco Parlay/OSA.

Las aplicaciones o servicios, se implantan en **Servidores de Aplicaciones** los cuales son independientes de la arquitectura usada; dichos servidores usan la capacidades definidas por Parlay/OSA que proveen los **Servidores de Capacidades de Servicio** para acceder los recursos de las redes de acceso. De esta forma los servidores de aplicaciones implementan

la parte cliente de las APIs Parlay/OSA y los Servidores de Capacidades del Servicio la parte servidor.

Los SCS son entidades lógicas que implementan las APIs, es decir las interfaces de las **Capacidades Características del Servicio** y posiblemente interactúan con los elementos de la red, por ejemplo el HLR, MSC, SSP, etc. De esta forma los SCS actúan como proxys o pasarelas hacia los elementos del núcleo de la red.

Un Servidor de Capacidades del Servicio puede implementar varias Capacidades Características del Servicio, aunque por lo general se habla de un SCF específico para un SCS.

A.1.1.2 Funciones de los elementos del marco OSA/Parlay

El marco OSA/Parlay controla el acceso a las Capacidades Características del Servicio y a través de una plataforma distribuida provee flexibilidad en la ubicación de las aplicaciones y en la forma como dichas aplicaciones encajan en un modelo de negocio. Además los elementos del marco OSA/Parlay permiten el uso de SCF de diferentes vendedores lo que es crucial para la creación de nuevos servicios innovadores y diferenciados.

El marco OSA/Parlay está formado por una familia de interfaces que consisten principalmente en:

- Gestión de seguridad y confianza: provee una forma de autenticar y formar dominios de confianza.
- Registro de SCF: permite registrar nuevos SCF al sistema.
- Fabrica de SCF: permite crear instancias de un SCF específico.
- Descubrimiento de SCF: descubre los SCF que un operador provee.

Adicionalmente se proveen interfaces para:

- Manejo de integridad: balanceo de carga, gestión de fallas y supervisión de estado de los SCF.
- Notificación de eventos: maneja notificaciones de eventos, por ejemplo, el registro de un nuevo SCS.
- Gestión de contratos de servicio: por ejemplo entre operadores de red y proveedores de servicios.

A.1.2 JAIN

Con la tecnología de JAIN, se establece una entorno de trabajo escalable y entendible que abarca la creación, ejecución y gestión de servicios de telecomunicaciones a la vez que establecen conexiones con redes de datos basadas en IP. La tecnología de JAIN permite a los diferentes servicios de RI correr a través de una variedad de plataformas de conmutación de diferentes proveedores, teniendo en cuenta sus posibles especificaciones e incompatibilidades (proveedores de soluciones, incluyendo proveedores de stack SS7). La Figura 4 representa la estructura de JAIN.

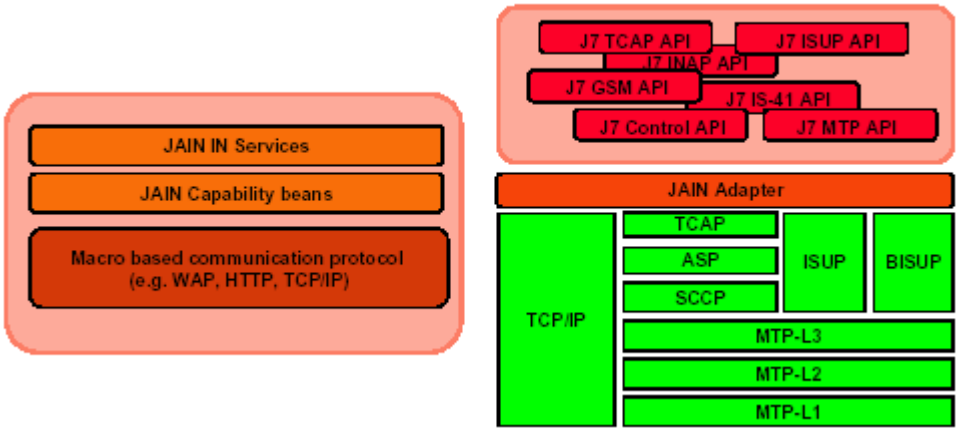

Figura 4: Estructura JAIN

La clave para resolver estos problemas de incompatibilidad fue desarrollar interfaces de programación de aplicaciones estandarizadas (APIs) entre los stacks SS7 y las aplicaciones

que corren sobre ellos, tales como los puntos de control de servicio, registros de ubicación de hogar y directores de estación base, y proporcionar un ambiente operativo que permita a estas aplicaciones ser manejadas de forma segura.

En el núcleo, la arquitectura JAIN define una librería de componentes software, herramientas de desarrollo y ambientes de creación de servicios para construir servicios de RI para operadores de líneas cableadas e inalámbricas. Los componentes para sets de capacidades especificas pueden ser construidos sobre estas librerías.

La función de estas librerías es proveer interfaces que permitan escribir servicios independientes del protocolo, estándares o mecanismos de transporte. Esta clase de portabilidad del servicio puede reducir drásticamente el tiempo de salida al mercado, el costo para el proveedor y el costo al consumidor.

Las aplicaciones resultantes que se espera serán desarrolladas por los proveedores de Stack SS7, proveedores de red y otros, tendrán la capacidad de ser escritas una vez y correr en cualquier stack de protocolos que cumplan con la tecnología JAIN, en cualquier Hardware y a través de cualquier interfaz. Adicionalmente las capacidades de red contenidas dentro de la tecnología JAIN permitirán a los elementos de servicio ser distribuidos a través de la red y dentro de los dispositivos de las Partes Finales, impulsando el desarrollo de teléfonos con pantallas inteligentes y otros periféricos inteligentes.

A.1.2.1 Arquitectura JAIN

Mientras pueden desarrollarse muchos nuevos sistemas usando la arquitectura de JAIN, varias abstracciones del proceso de comunicación básica son soportadas.

Capa de la Red:

- Red de Telecomunicaciones - Redes Inteligentes (AIN/IN) o SS7 - ISUP, INAP, TCAP, etc.

- Red Inalámbrica - SS7 con la capa Parte de la Aplicación Móvil (MAP)
- Internet - SIP, MGCP, Megaco, H.323.

Capa de la Señalización:

- Red de Telecomunicaciones - el Punto de Servicio de Señalización (SSPs) o Conmutadores.
- Red Inalámbrica - los Centros Conmutación Móviles (MSCs)
- Internet - conmutadores de salida o agentes de la llamada, y controladores Gateway o H.323 gatekeepers.

Capa de Servicio:

- Red de Telecomunicaciones - Puntos de control de Servicio (SCPs)
- Red Inalámbrica - cualquier combinación de Controladores de la Estación Base (BSCs), Registros de localización de usuarios locales (HLRs), Registro de localización de visitantes (VLR), y MSCs.
- Internet - el Servidor de Aplicaciones.

Como se ilustra en la Figura 5, la arquitectura de JAIN incluye un Ambiente de Creación de Servicio (SCE) para ambos: **servicios de red confiables y aplicaciones de terceras partes no confiables**. Los servicios confiables (y las políticas) residen dentro del núcleo de las redes públicas. Los servicios no confiables son servicios escritos por terceras partes que accedan funciones dentro del núcleo de las redes públicas. A través de una interfaz de proveedor de servicios segura, estas aplicaciones mantienen el compromiso de fiabilidad e integridad de estas redes.

A.1.2.2 Interfaces estándar para señalización de servicios

La iniciativa JAIN define un conjunto de interfaces estándares para protocolos de señalización y de servicios, con un estándar Java bien definido, sin cambios entre plataformas de diferentes vendedores.

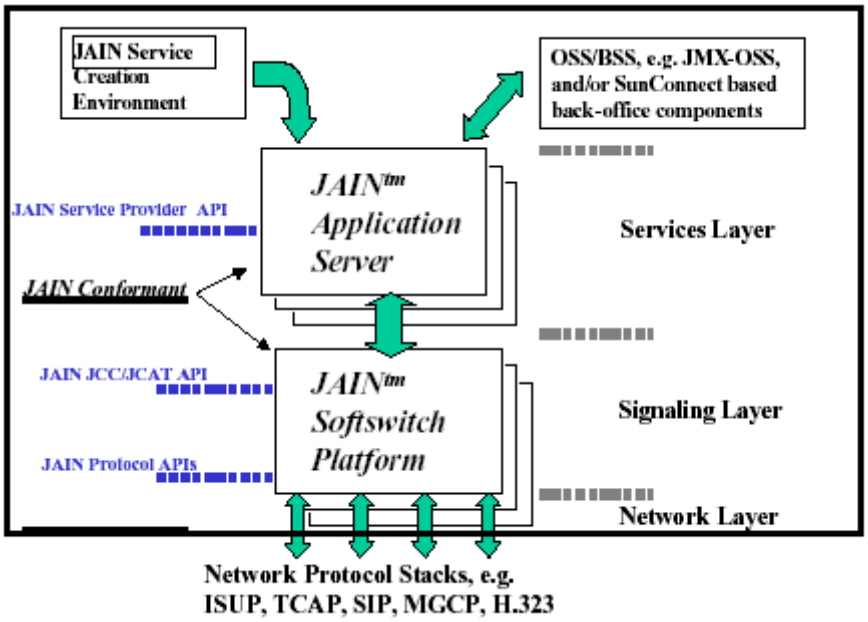

Figura 5: Arquitectura JAIN

Definiendo una interfaz común entre dos o mas protocolos se busca la convergencia de la red. La API de control de llamada de JAIN (JCC) y la de Coordinación y Transacciones (JCAT) dentro de la comunidad JAIN esta definida como una interfaz de sesión y de control de llamadas.

El objetivo de las APIs JCC y JCAT es proveer a las aplicaciones con un mecanismo consistente para control de llamadas y transacciones de procesamiento de llamadas.

Las dos incluyen las facilidades requeridas para observación, inicialización, respuesta, procesamiento y manipulación de llamadas, donde una llamada puede incluir multimedia, sesiones multipartitas, sobre redes integradas(RTPC, paquetes y/o inalámbricas). JCC es un modelo de llamada unificado que incorpora las características básicas de la API de telefonía Java (JTAPI) así como el servicio de control de llamada de Parlay, y provee una estructura Java que soporta las aplicaciones de procesamiento complejo de llamadas. La Coordinación y la Transacción incluyen facilidades para que aplicaciones adicionales sean

llamadas antes, durante o después del procesamiento de la llamada. Generalmente por características de valor agregado, tales como las Redes Privadas Virtuales, sesiones multipartitas, etc.

A.1.2.3 Topología de red

Un aspecto clave de la arquitectura de componentes JAIN es mover la capa de señalización de la conmutación propietaria hacia servidores de control de llamada abiertos, también conocidos como Agentes de Llamada, Controladores de Pasarelas de Medios, o Softswitches.
La señalización que es el protocolo usado para establecer y terminar la conexión de las comunicaciones, es la unión entre conmutadores de telecomunicaciones y softswitches. La habilidad para adaptar los componentes de señalización entre redes es fundamental para el éxito de los portadores y los proveedores de servicios de red.

La Figura 6 es una representación gráfica que muestra donde están definidas las APIs JAIN dentro de una plataforma de comunicaciones. La arquitectura de softswitch esta centrada en el mapeo de las interfaces de control de llamada. Ya que los softswitches trabajan señalizando en redes IP, muchas están equipadas con SIP, MGCP, MEGACO o protocolos H.323. Muchos de los Softswitches también incluyen señalización SS7 para direccionar interfaces en la red telefónica existente.

La iniciativa JAIN ofrece componentes clave para construir arquitecturas y proveer portabilidad de servicios a través de interfaces Java unificadas que pueden ubicar servicios de siguiente generación. La estructura JAIN de las APIs esta dividida en 4 categorías:

- Conexión o Interfaces de Protocolo.
- Control de Llamadas o Interfaces de Sesión.
- Interfaces de Servicios Lógicos.
- Acceso al Proveedor de Servicios.

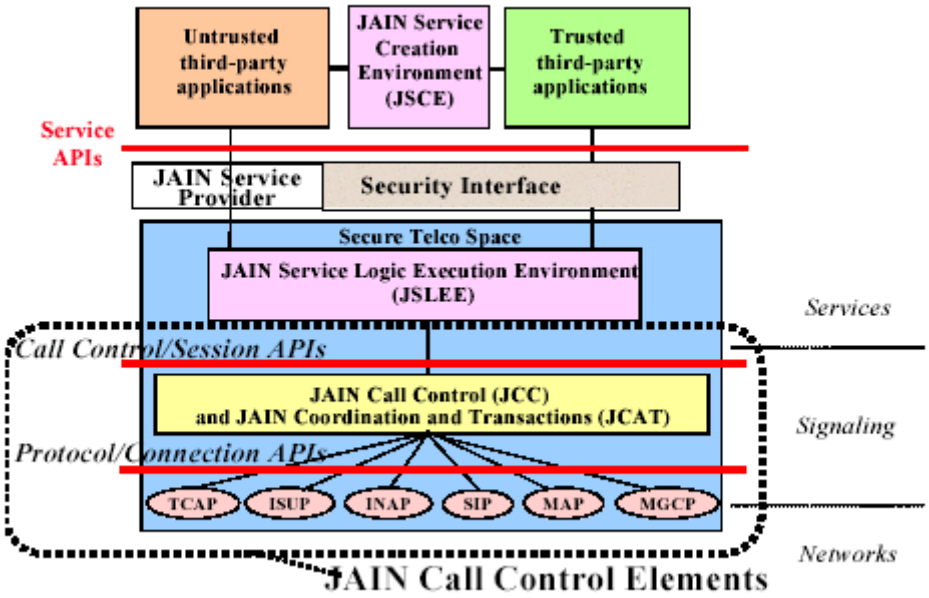

Figura 6: APIs JAIN

Como un ejemplo de un servicio completo, analicemos una llamada en conferencia. El hecho que una conferencia se necesite para aproximadamente 1 hora con 12 participantes se define en la capa de servicios quien también creará el registro de contabilidad global usado por la tarificación. Las participantes actuales (12) de la conferencia son manejados en la capa de la señalización por cualquier participante que llama a un nuevo participantes o por los elementos de control de llamada que agregan algún otro participante, los Miembros de la Llamada podrían estar usando voz sobre IP (VoIP), protocolos alámbricos o inalámbricos. El protocolo de la capa de red maneja las pilas individuales de Miembros de la Llamada Ej. un miembro es desconectado con un hang up.

Deberá hacerse claridad en que los componentes JAIN no necesariamente residen en un solo servidor, desde el Gatekeeper, Pasarela de Control, y la funcionalidad del pasarela de señalización se lleva a cabo típicamente como una aplicación multi-nivel distribuida en todos los elementos de red de señalización. Tal acercamiento mantiene ventajas significantes de estabilidad, escalabilidad, ejecución, fiabilidad, gestión, rehusabilidad, y flexibilidad.

A.2 PROTOCOLOS PARA LA INTEROPERABILIDAD DE SERVICIOS

Esta sección agrupan los protocolos (que son mecanismos que proporcionan el "lenguaje" necesario para la comunicación entre las dos partes involucradas) cuya función principal es la de brindar una mayor eficiencia en la utilización de servicios cuando se implican las Redes de Conmutación de Circuitos y la Red IP, resolviendo así algunos problemas de interconexión que entre estas dos redes se presentan.

A.2.1 MEGACO

Las principales tecnologías de unificación –voz sobre IP (VoIP), voz sobre ATM (VoATM) y voz sobre Frame Relay (VoFR)– poseen fortalezas y debilidades, y no existe una tecnología perfecta e idónea para atender todos los problemas.

Cualquier encargado de una estrategia de redes corporativas necesita entender los sacrificios que hay que hacer con cada enfoque y darse cuenta de que dicha opción es un "blanco móvil". Los pronósticos para cada solución tecnológica han cambiado en los últimos dos años y seguirán evolucionando durante cierto tiempo.

VoFR y VoATM se están utilizando ampliamente para evitar los gastos de larga distancia de las corporaciones (sobre todo entre oficinas internacionales). Sin embargo, la oportunidad para ahorrar en los costos de larga distancia se está acabando.

Si bien son una realidad las perspectivas que se abren ante la telefonía IP, todavía se requiere solucionar diferentes problemas técnicos y de comercialización. Su gran atractivo es sin duda el factor económico.

Para la interconexión entre las redes IP y las redes de circuitos SCN (Switched Circuit Network), específicamente las redes telefónicas (RTPC/ISDN/GSM), de cara al servicio de voz, se requiere dar solución a diferentes problemas técnicos, que por razones de claridad pueden ser enmarcados en dos planos, en el plano de control y en el plano de usuario. En definitiva, de lo que se trata es desarrollar el mapeo o traducción de protocolos, más aún, el mapeo "perfecto" entre los protocolos de las redes a interconectar, tratando, en la medida de lo posible, mantener la "transparencia" entre dichos protocolos.

La problemática de la interconexión en el plano de control se relaciona con los protocolos de señalización que se utilizan en las diferentes redes, se trata del mapeo de señalización entre las redes que se interconectan, lo que presupone la conversión de mensajes, formatos y parámetros de señalización. La interconexión en el plano de usuario trata la adaptación de los datos de voz a las características y propiedades de ambas redes, lo que determina la QoS en términos de la calidad de la voz y de la demora. El aspecto fundamental del plano de usuario es mantener la QoS necesaria para las conexiones de voz.

En general, las pasarelas de interconexión tienen que proporcionar los siguientes "mecanismos" o funciones:

- Adaptación de señalización, básicamente tiene que ver con las funciones de establecimiento y terminación de las llamadas.
- Control de los medios, se relaciona con la identificación, procesamiento e interpretación de eventos relacionados con el servicio generados por usuarios o terminales.
- Adaptación de medios, según requerimientos de las redes.

A.2.1.1 Posibles escenarios

La interconexión entre las redes IP y SCN, en lo que a telefonía se refiere, se puede abordar a partir de los siguientes posibles escenarios:

- Caso 1: solicitante y solicitado pertenecen a diferentes redes, uno a la red IP y el otro a la red modo circuito.
- Caso 2: solicitante y solicitado pertenecen a la misma red, ya sea la red IP o la red modo circuito, pero en ambos casos haciendo tránsito en la otra red.

En el caso 1 la función de interconexión tiene que desarrollar los tres mecanismos antes mencionados: adaptación de señalización, control de los medios y adaptación de los medios. Es decir, el mapeo de interconexión completo.

En el caso 2 se utilizan los mismos protocolos en las interfaces de ambos terminales, no así en las interfaces del backbone entre ambas redes. Vale aclarar que este tránsito por la segunda red a que se hace alusión puede ser para cursar la información de señalización y los medios, sólo la señalización o sólo los medios, y en cada caso se requerirá el mapeo correspondiente.

Dadas las funciones a desarrollar por las pasarelas de interconexión , quizás lo más aconsejable, o lo más evidente, puede ser pensar que son o deben ser entidades físicamente independientes. No obstante, y según el caso, pueden encontrarse incorporadas a otros entes funcionales, como puede ser en el Gatekeeper del modelo H.323 de ITU.

A.2.1.2 Adaptación de señalización

Esta función de interconexión entre redes es necesaria cuando éstas emplean protocolos de señalización diferentes, e incluso aún cuando de emplearse el mismo protocolo de señalización se hace un uso diferente de los mensajes de los mismos, tal es el caso, por ejemplo, de la pasarela del modelo H.323 cuando la interconexión se materializa a través de

accesos RDSI (esto es, cuando la pasarela de interconexión se conecta a la red RDSI como un terminal RDSI), pues en ambos lados de la pasarela H.323 se utiliza en este caso señalización de control de llamada según el protocolo Q.931, pero no hay perfecta correspondencia entre el uso de los mensajes Q.931 a ambos lados de la pasarela como se puede ver en la Figura 7. En cualquier caso, es necesario el correspondiente mapeo de señalización de manera que pueda completarse el establecimiento de las llamadas extremo a extremo.

Las pasarelas de interconexión IP/GSTN pueden ser, en general, conectadas a las redes GSTN en uno de dos posibles modos: modo terminal, modo nodo y, según el modo empleado será el protocolo de señalización que se requiere del lado de la red RTPC, y en consecuencia el mapeo de señalización a desarrollar.

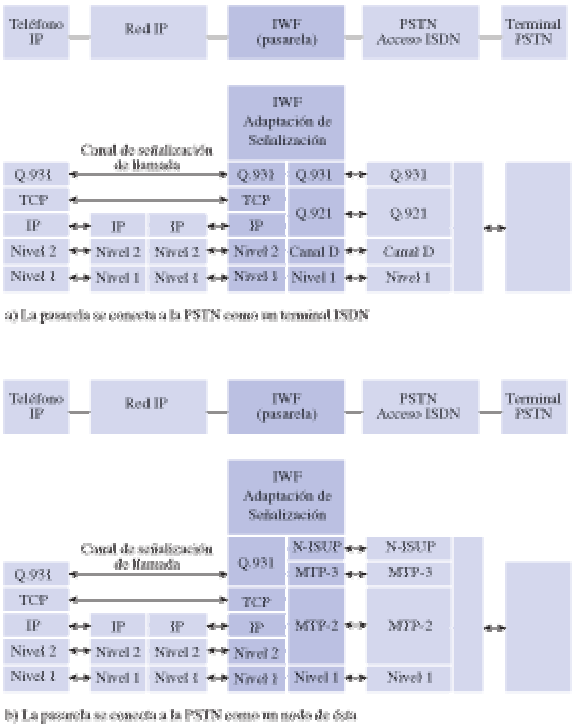

Figura 7: Adaptación de señalización

La conexión de las pasarelas de interconexión a las redes RTPC según el modo terminal no es de manera alguna escalable de cara a la telefonía pública, por cuanto la potencialidad de señalización de los accesos para terminales tienen capacidad muy limitada, aún en el caso del acceso primario RDSI si se compara con la potencialidad de la red SS7. De manera que, dadas las necesidades de la telefonía pública, el modo nodo es el más apropiado, lo que daría acceso a la pasarela, en principio, a la red de señalización SS7, con toda la potencialidad que esto implica.

El Sistema de Señalización SS7 es neurálgico en la operación de las RTPC, de ahí que los operadores de Telecomunicaciones sean muy reacios a exponer sus redes SS7 a la interacción directa con pasarelas de otros propietarios, que por demás serían numerosas. En este sentido la solución del modelo MEGACO está en línea con estas actitudes, es decir, una solución de más complacencia para los operadores de Telecomunicaciones es utilizar una pasarela de señalización relacionada directamente con la red SS7 (muy probablemente de su propiedad) y controladas por ésta varias pasarelas de medios con el empleo de cierto protocolo, que es precisamente lo que promueve el modelo MEGACO con el protocolo MGCP.

A.2.1.3 Resolución de direcciones

Otro problema a solucionar en la interconexión entre redes IP y RTPC, es lo concerniente al direccionamiento de los terminales, pues en ambas redes se utiliza un esquema de direcciones diferente, la red IP emplea un sistema jerárquico basado en la filosofía DNS, y la red RTPC un sistema también jerárquico pero sobre la base de la Recomendación E.164 de ITU.

Entonces, entre las funciones propias de la interconexión, está la resolución de direcciones, esto es, el mapeo de direcciones IP/E.164. Cuando el solicitante pertenece a la red IP, y el solicitado a la red GSTN, no hay mayores complicaciones, pues la dirección de destino E.164 puede ser enviada, en general, fácilmente a la pasarela desde el terminal IP, y de ésta

a la GSTN. El problema resulta algo más complicado en el caso contrario, dada la reducida capacidad de señalización de los aparatos telefónicos convencionales. En cualquier caso resulta un requerimiento importante en el servicio de interconexión que prestan las pasarelas el hecho de que el usuario solicitante debe estar ajeno al tipo de red al que está conectado el usuario solicitado, esta función de mapeo de direcciones debe ser transparente a los usuarios.

Para facilitar y solucionar el problema del necesario mapeo de direcciones IP/E.164, ambos modelos de señalización de telefonía IP, H.323 de ITU y SIP del IETF, han optado por:

- H.323: permite identificar a los terminales H.323 de diferentes formas, por ejemplo, con direcciones E.164, con direcciones del tipo correo electrónico o con H.323-URL de la forma user@host (H.323v4).
- SIP: utiliza los SIP-URL, de la forma user@host.

A.2.1.4 Control de medios

Una vez establecida la conexión, logrado esto por la función de adaptación de señalización, la pasarela de interconexión también desarrolla la función control de medios, que se ocupa de "manejar" toda la información de control generada por el terminal. Para el caso de comunicaciones de voz, la información de control del nivel de usuario más a destacar son los tonos multifrecuenciales (DTMF) que produce un teclado telefónico convencional (por ejemplo, para interactuar con un servidor de voz). En la Figura 8 se muestra este caso en el marco de la Recomendación H.323 de ITU.

Ahora bien, dadas las características de estas señales, en el sentido que están en el rango audible pero no son señales de voz, sino tonos, es necesario prestar particular atención para su trasvase por la conexión híbrida que representa la pasarela de interconexión.

Las técnicas de compresión de voz de baja velocidad introducen considerable distorsión en los tonos DTMF, provocando la recepción y correspondiente decodificación incorrecta en

los receptores. Entonces, esto requiere que las señales de audio y los tonos DTMF sean separados en la pasarela (si no lo ha sido ya en el emisor) y conducidas de forma independiente al receptor.

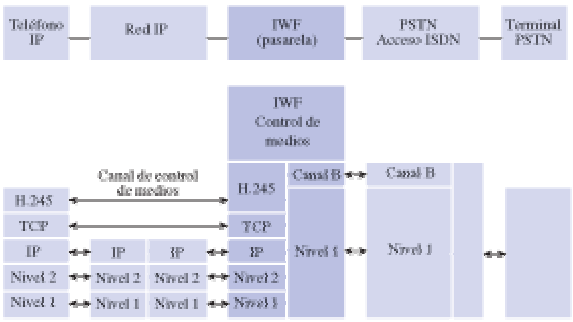

Figura 8: Control de medios

El forum de VoIP ha recomendado dos posibles soluciones para el transporte de los tonos DTMF:

- Transporte dentro de banda: consiste en transportar estos tonos, digitalizados y paquetizados, con los protocolos RTP/UDP, mediante un formato de carga útil dedicado.
- Transporte fuera de banda: conlleva a utilizar un canal de control de medios seguro (no UDP, sino TCP) para el transporte de las señales DTMF.

El transporte de los tonos DTMF dentro de banda se ve afectado por la falta de garantía en la entrega de paquetes que el protocolo UDP ofrece, con nefastas consecuencias para el funcionamiento del servicio en caso de pérdida de un paquete asociado a un tono TDMF. Tiene la ventaja de que los tonos permanecen sincronizados en el tiempo con respecto a la voz.

En cambio, el transporte fuera de banda si bien gana en seguridad respecto a la entrega segura de los paquetes, pierden las señales su referencia exacta en el tiempo en relación con

el flujo de voz. Esta es precisamente la solución adoptada en la Recomendación H.323, mediante el canal H.245.

A.2.1.5 Adaptación de medios

Esta función tiene primordial importancia en relación con la QoS experimentada por el usuario final. En esto influyen dos factores fundamentales:

- la calidad de la voz extremo a extremo, determinada por los sucesivos procesos de codificación – decodificación, y las pérdidas de paquetes en la red.
- el retardo extremo a extremo, debido a las sucesivos procesos de codificación – decodificación, paquetización y encolamiento. Afecta la interactividad en la conversación, y por tanto a la QoS percibida por el usuario.

Las redes IP son redes del tipo mejor-esfuerzo y por tanto no ofrecen garantía de QoS, pero las aplicaciones de telefonía IP si necesitan algún tipo de garantía de QoS en términos de retardo, jitter y pérdida de paquetes. En tal sentido existen dos mecanismos se señalización para QoS , estos son: IntServ y DiffServ. Ambos son mecanismos de cara a la red, sin embargo es una realidad que el objetivo de los proveedores de servicio es incrementar la eficiencia de uso de la red, lo que implica reducir los costos del servicio. Para esto requieren acomodar el número máximo posible de conexiones simultáneas, lo que a su vez conduce al aumento de el retardo, el jitter y la pérdida de paquetes. Por tanto, es necesario buscar QoS no solo en la red, sino también en los terminales, y en los procesos que en los mismos se desarrollan, de ahí que sea necesario también decir que la sensibilidad a la pérdida de paquetes, a los retardos y sus fluctuaciones, que experimentan los servicios de voz sobre IP, dependen en buena medida de los mecanismos implementados en los terminales.

La preparación de los medios en los terminales para ser enviados y transferidos por la red IP involucra varios procesos: digitalización, compresión y empaquetado en el extremo emisor, y los procesos inversos en el extremo receptor. Todo esto se lleva a cabo mediante

un complejo procesamiento que sigue determinado algoritmo, lo cual a su vez se desarrolla en cierto intervalo de tiempo, esto es, implica retardo de procesamiento y retardo de empaquetado:

- retardo de procesamiento: demora producida por la ejecución del algoritmo de codificación, que entrega un flujo de bytes listos para ser empaquetados.
- retardo de paquetización: es el tiempo que se requiere para formar un paquete de voz a partir de los bytes codificados.

Otro aspecto a tener en cuenta es el compromiso entre el retardo de paquetización y la utilización del canal (relación entre bytes de información y bytes de cabecera en cada paquete de voz), es decir, la búsqueda de mayor utilización del canal conduce a mayor retardo de paquetización para cierto estándar de codificación. Claro está, según el estándar de codificación que se utilice será el retardo resultante en relación con la utilización del canal, diferencias que se acentúan cuando la utilización del canal está por encima del 50% , con un crecimiento del retardo en forma exponencial en el caso de los codecs de baja velocidad como el G.723.1. El retardo de paquetización también puede ser reducido mediante multiplexación de varias conexiones de voz en el mismo paquete IP.

A los retardos de procesamiento y empaquetado se suma también el retardo que introduce el proceso de buffering en los terminales, y el retardo de encolamiento en la red. Todo esto da un retardo extremo a extremo que percibe el usuario final en mayor o menor medida. Retardos extremo a extremo por debajo de 400 milisegundos no comprometen la interactividad en la conversación, pero ya por encima de 150 milisegundos se requiere control del eco.

Los retardos antes comentadas son resultado lógico de las características y modo de operación de las redes IP, así como también de la naturaleza de las señales de voz. Ahora bien, otro elemento emana de la necesidad de interconexión entre redes IP y redes GSTN. En las pasarelas de interconexión se debe realizar una transcodificación de los medios, esto

es, un mapeo de codificación de medios, por cuanto en las redes GSTN se ha establecido ya hace un tiempo la norma de codificación de voz ITU G.711, que entrega un flujo de bits a 64 kbps, y no está en línea con los requerimientos de máxima utilización de la red que se persigue en redes IP, dado el elevado consumo de ancho de banda que esto implica, entonces se requiere la conversión a una norma de codificación de voz de baja velocidad (G.723.1, G.729, etc.). Este proceso de transcodificación consume tiempo, es decir, introduce retardo, que se refleja en el retardo extremo a extremo ya antes mencionado. También esto introduce más distorsión en las señales de voz.

En consecuencia, las pasarelas de interconexión entre redes IP y redes GSTN, tan necesarias para la integración de los servicios de voz en ambas redes, y para la propia difusión de la telefonía IP, adicionan factores perturbadores en la QoS de VoIP, por los retardos que introducen y las distorsiones que adicionan a las señales de voz al desarrollar la función adaptación de medios, esto es, la conversión del formato de los medios, que se enmarca en el plano de usuario antes indicado.

A.2.1.6 Funciones genéricas de las pasarelas de interconexión

Como fue apuntado con anterioridad, las funciones de interconexión y mapeo que desarrollan las pasarelas es en relación con los medios (que portan la información de usuario) y la señalización. Entonces, desde el punto de vista funcional y físico, de cara a la implementación, podría decirse, de forma genérica, que las funciones de toda pasarela son las siguientes:

- Funciones de señalización:
    - funciones de señalización relativas a las redes de paquetes.
    - funciones de señalización relativas a las redes de circuitos.
- Funciones de control de recursos (interrelación medios-señalización).
- Funciones de gestión de medios.

Claro está, estas funciones se interrelacionan en aras de poder desarrollar, funcionalmente de conjunto, las tareas propias de la pasarela.

H.248, también conocido como protocolo Megaco, MGCP, como ya se había mencionado anteriormente, es un estándar que posibilita a un MGC controlar uno o varios MG´s (establecer, modificar y terminar "conexiones" en los MG´s)., resultado de un esfuerzo conjunto entre ITU y el IETF. Es un protocolo de control de dispositivos, de control de "conexión", no es un protocolo de señalización de VoIP, es complementario a H.323 y SIP, pues si bien el MGC utiliza H.248 para controlar los MG´s, se comunica con el entorno IP (señalización VoIP) a través de H.323 o SIP.

El MGCP es un protocolo basado en texto (cultura Internet), y soporta un modelo de llamada centralizado. Ha sido una derivación del SGCP (Simple Gateway Control Protocol) y del IPDC (Internet Protocol Device Control). Asume una arquitectura de control donde la "inteligencia" del control de llamada está fuera de los MG´s, controladas éstas por un elemento funcional externo, el MGC.

Es un protocolo stateless, es decir, no requiere una máquina de estados para describir una secuencia de transacciones entre dos entidades de señalización, tampoco mantiene una "memoria" de las transacciones previas entre el MGC y la MG. Esto no debe confundirse con el estado de la llamada, que si es mantenido en el MGC.

MGCP utiliza el protocolo SDP (Session Description Protocol) para describir la sesión, esto es: nombre y propósito de la sesión, tiempo en que la sesión está activa, medios que contempla la sesión, información de destino (puerto, dirección IP), requerimientos de ancho de banda, etc. MGCP se transporta sobre UDP, conformándose la pila MGCP/UDP/IP, de manera que los mensajes MGCP (comandos y respuestas) constituyen el cuerpo de datos de los datagramas UDP.

Dada la arquitectura y concepción del modelo MEGACO, éste presenta características asimétricas:

- Modelo de control centralizado, donde un MGC controla la operación de cierto número de MG´s. Modelo Maestro-Esclavo.
- Casi toda la inteligencia del sistema se encuentra en los MGC´s. Los GW´s casi no tienen autonomía.
- Concebido para terminales sin inteligencia.
- Limitada área de influencia de un MGC, esto es, cada MGC solo puede controlar cierta área geográfica y/o cierto número de terminales y comunicaciones.

En consecuencia, se requiere lograr en la evolución de MEGACO un giro hacia cierta simetría en la señalización, y a la vez se requiere la cooperación entre MGC´s para posibilitar la escalabilidad y lograr la extensibilidad del modelo de acuerdo a los requerimientos de la telefonía.

En el mismo sentido es necesario incorporar al modelo terminales inteligentes, con todo lo que esto representa de cara a la simetría de la señalización.

Señalización MGC-MGC:

La señalización entre MGC´s es una necesidad si se quiere la escalabilidad del modelo MEGACO. Cada MGC controla cierta zona geográfica, es el elemento funcional de mayor jerarquía del modelo. Entonces, en línea con lo apuntado, es necesario que MGC´s de diferentes zonas cooperen entre si mediante un protocolo, esto es, posibilitar el establecimiento de llamadas gestionadas por más de un MGC, e incluso también entre terminales de redes diferentes, por ejemplo, terminal telefónico de la red IP y terminal telefónico de la red RTPC. En fin, se requiere el establecimiento de llamadas entre MGC´s.

También es necesario que este modelo incorpore servicios suplementarios similares a los que se ofrecen en las redes telefónicas, donde indudablemente la señalización MGC-MGC

jugará un rol esencial e imprescindible. En este sentido es posible desarrollar dicha señalización sobre la base de SIP o también con señalización H.323.

A.2.1.7 Arquitectura MEGACO

Desde el punto de vista físico, estas funciones pueden agruparse de maneras diferentes según los requerimientos. En cualquier caso, estas tres funcionalidades, que bien pueden referirse como módulos funcionales básicos, se relacionan mediante interfaces, y a través de los mismos y con determinados protocolos logran la operación de conjunto de la pasarela. Esto puede ser esquematizado de la siguiente manera

En la Figura 9 se emplean denominaciones que están en línea, en buena medida, con lo que establece la Recomendación H.323v4 de ITU, en concreto con las interfaces entre dichos módulos funcionales: A, B, C, D, X, Y, Z. Las interfaces X, Y y Z son externos, es decir, vinculan la pasarela con las redes que ella interconecta. Los interfaces A, B, C y D son internos a la funcionalidad de la pasarela:

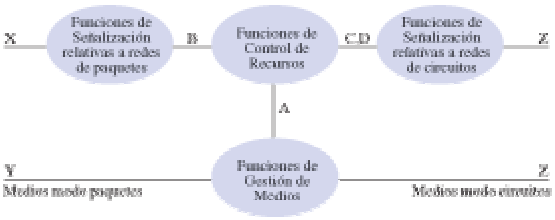

Figura 9: Arquitectura MEGACO

- Interfaz X: es el interfaz de señalización de la pasarela con la red de paquetes, esto es, con la red IP. Puede tratarse de señalización H.323 o señalización SIP.
- Interfaz Y: es el interfaz de medios con la red de paquetes IP, típicamente sobre protocolos RTP/UDP/IP.
- Interfaz Z: se trata del interfaz de la pasarela con la red modo circuito SCN (RTPC, ISDN, GSM). En este caso en la Figura 9 se indica la misma denominación tanto para

el interfaz de señalización como para la interfaz de medios, si bien el tratamiento de ambos por parte de la pasarela es diferente. Es de destacar que la señalización en las redes SCN puede ser del tipo CAS (Channel Associated Signalling), o del tipo CCS (Channel Common Signalling) como es el sistema SS7.

- Interfaz C: interfaz interna, vincula la funcionalidad Control de Recursos con la señalización SCN tipo CAS.
- Interfaz D: interfaz interna, vincula la funcionalidad Control de Recursos con la señalización SCN tipo SS7.
- Interfaz B: interfaz interna, vincula la funcionalidad Control de Recursos con la señalización VoIP.
- Interfaz A: también es un interfaz interna, a través del cual la funcionalidad Control de Recursos controla la funcionalidad Gestión de Medios.

Implementaciones de pasarelas

Las funcionalidades, o bloques funcionales básicos de las pasarelas, pueden agruparse físicamente de diferentes maneras. La escalabilidad de cara a la telefonía pública es un factor a tener muy en cuenta en la implementación de las estas, de ahí que las funcionalidades serán agrupadas, desde el punto de vista físico, o de dispositivos, según los requerimientos específicos en cada caso.

Por tanto, las interfaces A, B, C y D, que se han denominado internas en relación con la propia funcionalidad de la pasarela, muy bien pueden ser externas desde el punto de vista físico.

En este sentido se plantea la descomposición física de la pasarela de interconexión, en un intento de armonizar su funcionalidad y su escalabilidad.

Acorde con lo que establece la Recomendación H.248 de ITU, la funcionalidad Control de Recursos de la pasarela genérica se denomina MGC (Media Gateway Controller), y la

funcionalidad Gestión de Medios se denomina MG (Media Gateway), y entre ambas se establece el protocolo MGCP (Media Gateway Controller Protocol) sobre la interfaz A, que es un protocolo de control de dispositivos. Esto posibilita que un MGC pueda, en principio, controlar varios MGs, e incluso diferentes tipos de MGs, cada tipo optimizado según la aplicación requerida.

Descomposición física de la pasarela

Las funciones genéricas de toda pasarela pueden ser separadas físicamente en aras de satisfacer lo antes señalado, es decir, escalabilidad y funcionalidad. En este sentido las funciones de señalización, o más concretamente, las funciones de la pasarela que la vinculan con la señalización de ambas redes a interconectar (paquetes – circuitos), pueden ser localizadas en el MGC o en el MG, y también de forma independiente en lo que sería una SG (Signalling Gateway).

Media Gateway Control Protocol (MGCP) es un protocolo cliente/servidor que controla el intercambio de información entre MG y MGC. MGCP es el resultado de protocolos anteriormente propuestos y ha sido propuesto en distintos organismos de estandarización como el grupo de trabajo MEGACO del IETF y la ITU-T donde se ha denominado H.248. MGCP utiliza a su vez el Protocolo de Descripción de Sesión-SDP para el intercambio de parámetros entre el MG y MGC (dirección IP, puerto UDP, codificadores a utilizar, etc.).

A.2.2 SIGTRAN

SIGTRAN es un grupo de trabajo de la IETF, conformado en 1999, y se encarga de definir la arquitectura de transporte de la señalización en tiempo real sobre redes IP, así como también los protocolos de transporte de mensajes SS7 e ISDN sobre IP. Este protocolo esta formado por un nuevo protocolo de transporte, el Protocolo de Transmisión de Control Flujo (SCTP – Stream Control Transmición Protocol) y un conjunto de capas de Adaptación de Usuario (UA), las cuales simulan los servicios de las capas de SS7 e ISDN respectivas

A.2.2.1 ARQUITECTURA SIGTRAN

La SG (Signall Gateway), según se define en SIGTRAN (RFC 2719), es un agente de señalización que recibe/envía señalización tipo SCN (Switched Circuits Networks) a la red IP, y su función puede ser retransmitir, traducir o terminar la señalización SS7 en una pasarela SS7-Internet. La SG también puede estar cooresidente en la MG para procesar señalización SCN asociada a líneas o troncales conectados a la MG y controlados por ésta. El hecho de que la señalización asociada a las redes SCN pueda ser del tipo Canal de señalización asociado (CAS) o del tipo CCS (SS7), tiene incidencia en la posible descomposición física de las pasarelas.

En cuanto a la señalización de VoIP, si bien se presentan dos vertientes, H.323 y SIP, en cualquier caso tiene interés desde el punto de vista de la descomposición física de la pasarela, la relativa al control de llamada y al control de canal, que son las propiamente vinculantes con el mapeo de señalización que debe realizar la pasarela. La señalización de VoIP necesaria para control de admisión, control de acceso, localización, etc., no incide en dicha descomposición física.

Para continuar con el tema de transporte de señalización sobre IP, es bueno considerar las carencias o necesidades más importantes, en esencia discutiremos los problemas, después describiremos la solución. De este modo se estudiara la arquitectura conociendo sus debilidades y fortalezas. El protocolo de transmisión de control de secuencia SCTP (Stream Control Transmission Protocol) es una nueva y sólida tecnología de transmisión de señales para las comunicaciones inalámbricas. Diseñado por el grupo de trabajo IETF SIGTRAN, SCTP es el sucesor del protocolo de señalización SS7. La solidez de SCTP es resultado de su capacidad para mantener varias secuencias de datos con una única conexión. De este modo, el protocolo SCTP resulta ideal para conectar y controlar teléfonos móviles inalámbricos y dispositivos de Internet. Con SCTP, se pueden controlar de forma activa las

conexiones y las rutas de las señales al tiempo que se detectan al instante cualquier error o pérdida de las sesiones. Wood implementó una versión de SCTP.

Entorno de VoIP

El espacio de aplicación de SIGTRAN es VoIP, o más exactamente Media sobre IP MoIP, entendiendo por Media cualquier transporte en tiempo real de voz, música o video. Las posibilidades de utilizar Internet para el transporte de voz surgió en 1995, esto se podía lograr digitalizando la voz, proceso que consistía en dividirla en paquetes siendo transmitida a través de la red de Internet, al otro lado en el equipo remoto, los paquetes eran reensamblados en una cadena digital de voz y enviados a los parlantes del PC. Este auge de transmisión de voz sobre IP presentaba algunos defectos (algunos ya mencionados con anterioridad) en su arquitectura como:

- La pésima calidad de voz. Las conexiones de Internet no contaban con el suficiente ancho de banda y estaba sujeto a retardos en la entrega de los paquetes, esto contribuía a introducir jitter en la conversación, con trozos perdidos de voz.
- Estándares mal definidos. Estos no definían como la voz era paquetizada y como las conexiones eran establecidas y gestionadas.

Una simple topología de red (la telefónica), soportaba las conexiones punto a punto entre direcciones de red desconocidas, lo cual conlleva a interfaces no definidas entre las red IP y la red RTPC.

Solamente un servicio era soportado. Básicamente solo los servicios de voz (conversación) era permitidos, ninguno de los servicios normalmente asociados al la red RTPC eran soportados.

Estas limitaciones hacían imposible el desarrollo de VoIP en la escala requerida para negocios o como base de la telefonía para el hogar.

Arquitectura escalable

En la siguiente etapa de la evolución de VoIP se define una arquitectura que permite lograr la integración de las redes IP y RTPC, esta provee la capacidad de señalización para la gestión de la llamada y define los medios o caminos a través de la red IP con buena reserva de ancho de banda.

Varios grupos de trabajan en la investigación de ésta área, algunos de ellos son:

- Proyecto THIPHON de ETSI.
- Grupos de trabajo de H.323 de la ITU.
- El consorcio SOFTSWICH.
- El grupo de trabajo MGCP de la IETF., y muchos otros.

El trabajo emprendido por estos grupos de trabajo establece una arquitectura común la cual define las interfaces entre las redes IP y la RTPC que soporta MoIP. Algunos de estos nodos sus funciones y responsabilidades ya estaban definidos anteriormente y la arquitectura es definida en la Figura 10.

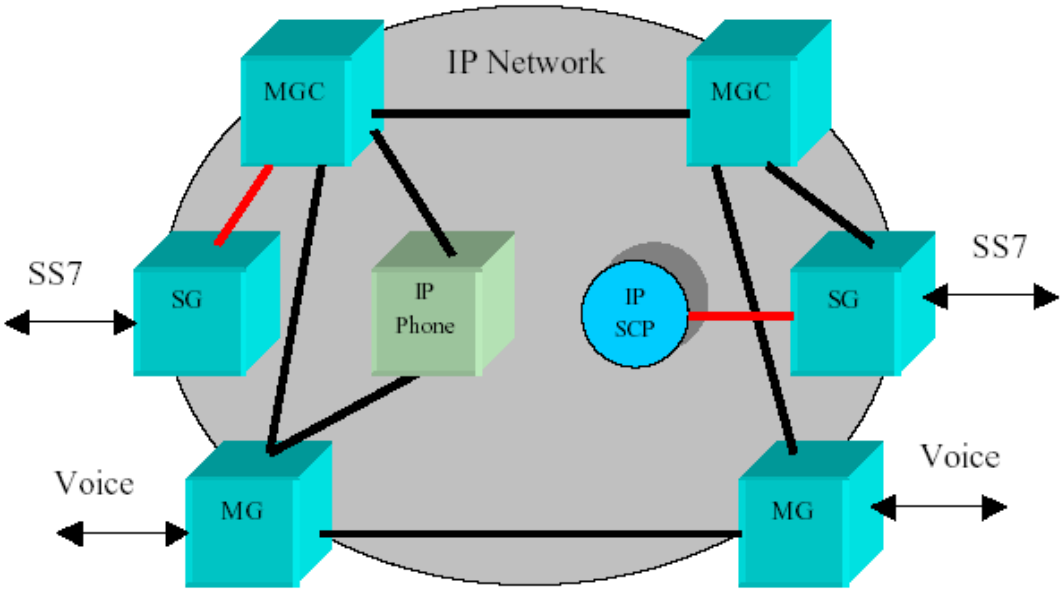

Figura 10: Arquitectura SIGTRAN

La definición de los elementos de red son los siguientes:

- MEDIA GATEWAY CONTROLLER (MGC).
- SIGNALLING GATEWAY (SG).
- MEDIA GATEWAY (MG).
- SERVICE CONTROL POINT- IP (IP – SCP). El punto de control de Servicios para IP, existente solo en la red IP, pero puede ser accedido mediante una dirección por la red SS7.
- IP PHONE. El teléfono IP se refiere genéricamente a un terminal.

A.2.2.2 Modelo IETF

La IETF, a través de sus grupos de trabajo ha establecido un modelo similar al de H.323, (Figura 10) y sobre el cual se han definido los siguientes protocolos:

SCTP (and adaptation layers) – Protocolo de Control de Transporte de Cadenas (y capas de adaptacion). Este es el protocolo de SIGTRAN el cual es utilizado para transportar mensajes SS7 entre la SG y MCG

Megaco – Media Gateway Control. Definido anteriormente, este proporciona el control entre el MGC y las MGs.

SIP – Session Initiation Protocol. Protocolo de Inicio de Sesión. Este protocolo se ejecuta entre MGCs o MGCs y teléfonos basados en IP.

RTP – Protocolo de Tiempo Real. El IETF especifica el mismo protocolo que en H.323 para el transporte de paquetes Media.

Porque es necesario SCTP?

SCTP es un nuevo protocolo diseñado para el de transporte efectivo de señalización. Un interrogante es, el porque el grupo SIGTRAN define una nueva capa de transporte cuando ellos pueden utilizar el protocolo TCP?.

Desde el punto de vista del programador, la mejor forma de transportar señalización SS7 sobre IP es tomar la capa elegida y definir una interfaz de adaptación y conectarla sobre la capa de transporte (TCP). Desafortunadamente la inflexibilidad es uno de los problemas relacionados. Una vez que se consigue ejecutar la Parte de Control de Señalización de Conexión (SCCP) sobre TCP/IP se esta fuera de conseguir una solución para transporte ISUP.

Una forma de acercamiento entre los protocolos es suministrar un adaptador sobre TCP/IP el cual redefina los servicios de transporte en términos de cuales protocolos de señalización superiores cumplan con la expectativa en los siguientes aspectos: en esencia es hacer que TCP/IP y una de las capas bajas de SS7, puedan lograr cierta afinidad, por ejemplo MTP3. este objetivo se puede lograr utilizando una capa de Adaptación de Usuario (UA), el cual se sitúa en la pila de protocolos de TCP.

Los comportamientos primarios de estas capas deben ser:

- Definir una norma o un método estándar mediante el cual las capas superiores como ISUP o SCCP puedan ser encapsuladas dentro de mensajes TCP.
- Proveer una marco de referencia de gestión par a par, como sockets de interfaz, numero de puertos, etc.

Teniendo resueltos los problemas de gestión par a par y de estandarización (ambos finalmente utilizando la misma UA), los aspectos de escalabilidad y reusabilidad quedan bien ubicados, si el mismo modelo es aplicado a otras capas superiores, esto podría ser una UA, para se utilizado por SCCP e ISUP y u otros más fuertes.

El problema de esta arquitectura es que cuenta aun con los servicios de TCP, entonces debemos preguntarnos, es conveniente que una cadena de bytes es la mejor opción para el transporte en tiempo real de mensajes de señalización?. Originalmente TCP fue diseñado para otro propósito, éste protocolo se basa en el transporte de abundante información en una cadena de bytes, y son enviados por segmentos sin orden alguno, ideal para archivos o correos electrónicos, desafortunadamente esto también significa una falla enorme. Como ya lo vimos TCP es particularmente sensitivo a los retardos introducidos por fallas en la red, cuando se pierden segmentos de estas cadenas u ocurre una violación de la cadena, TCP suspende la entrega de la cadena hasta que el error es restaurado.

Consideremos las consecuencias de esto en términos de entrega de mensajes SS7: si en una única cadena de mensajes TCP, que sirve para el transporte de múltiples de señalización ISUP, llegase a perder un paquete correspondiente a una fuente (por ejemplo a una llamada telefónica), esto resultaría presentando un retardo a todos los mensajes ISUP.

Algunos problemas adicionales serían los temporizadores utilizados por TCP los cuales son definidos en términos de muchos segundos. En particular la duración de las conexiones usa temporizadores de retransmisión que pueden resultar en excesivos retardos en la conexión llevando a perdidas de datos de retransmisión. Reconocidos los problemas por expertas industrias en el campo de las telecomunicaciones, surge entonces la solución auspiciada por la IETF, conocida como SIGTRAN.

SCTP

El grupo de trabajo de SIGTRAN, define el protocolo de control de transmisión de cadenas (SCTP) como la aspiración de corregir todos los problemas presentados en TCP con el fin de lograr un transporte mas eficiente de la señalización sobre IP.

SCTP, cuenta con el siguiente conjunto de características:

- Es el único protocolo de intercambio de datos entre dos puntos desconocidos.

- Define los temporizadores de muy corta duración comparados con los de TCP.
- Este provee un confiable transporte de datos del usuario detectando cuando uno de ellos esta defectuoso o fuera de secuencia, ejecutando las respectivas correcciones si son necesarias.
- Mediante una taza adaptable, este protocolo responde a la congestión y embotellamiento de la red, regulando la transmisión de acuerdo con la situación.
- Soporta múltiples búsquedas, cada punto final SCTP puede ser reconocido por múltiples direcciones IP. El enrutamiento de una dirección es independiente de las otras y si alguna ruta esta deshabilitada, ésta dirección podrá utilizar cualquier otra.
- Aprovecha un procedimiento de inicialización que utiliza cookies para prevenir ataques a los servicios.
- Soporta encapsulamiento de múltiples mensajes de señalización (Cuando un mensaje SCTP contiene múltiples paquetes de datos)
- Soporta fragmentación cuando un mensaje de señalización es dividido en múltiples mensajes SCTP para se acomodados dentro del PDU (Protocol Data Unit) principal.
- Es orientado a mensajes, definiendo marcos de estructuras de datos. Por el contrario TCP, no impone estructura para la transmisión de cadenas de bytes.
- Cuenta con capacidad de multi encadenamiento. Una vez son divididos los datos, estos son repartidos en secuencias independientes. TCP no ofrece este servicio.

En cuanto a la transmisión, los marcos estructurados son una característica útil para las UA (así como para otro usuario SCTP). El protocolo de transporte desempeña todo el trabajo de dividir las cadenas de datos en segmentos relevando de la responsabilidad a los usuarios de interpretar una cadena continua de datos, definiendo procedimientos que permitan el soporte de fragmentación y encapsulamiento menos complicados.

El atractivo principal de SCTP es el multi-encadenamiento, sin embargo (aun el nombre del protocolo lo especifica SCTP), esta característica es implícitamente diseñada para permitir a los usuarios particionar una conexión IP simple entre dos puntos finales dentro de cadenas

de datos lógicas separadas y asigna a cada cadena una aplicación o recursos particular. El objetivo es que un error o retraso en alguna de las cadenas no interfiera en el desarrollo normal de las otras.

Consideremos el ejemplo de protocolo ISUP, este transporta señalización para muchas fuentes RTPC, esencialmente troncales de abonados, si por ejemplo se están efectuando tres llamadas simultaneas y una de ellas sufre una perdida de datos, esto causa un retraso, miremos ahora los diferentes casos:

En el la situación ISUP con TCP, una única tubería de datos transporta todos los mensajes ISUP para las tres llamadas. Si ninguna de las llamadas sufre algún percance, las tres llamadas se ejecutan con éxito, pero en el caso en el que alguna de ellas por ejemplo la primera sufriera una perdida de datos todas las demás llamadas posteriores a ella también sufrirían retrasos mientras TCP reconoce el daño y recupera los datos perdidos, ver la Figura 11.

Consideremos ahora el caso de transporte ISUP sobre SCTP.
En este caso los datos perdidos relacionados a la llamada 1, afectan solo a ésta cadena de datos, las llamadas 2 y 3 transcurren normalmente.

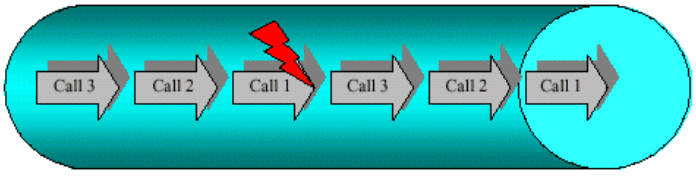

Figura 11: Situación ISUP con TCP

Capas de Adaptación SIGTRAN

La arquitectura SIGTRAN, incluidas SCTP y todas las UAs son mostradas en la Figura 12.

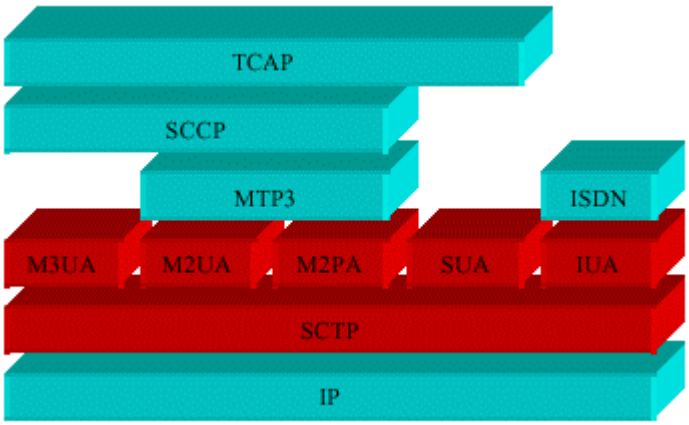

Figura 12: Capas de adaptación SIGTRAN

Algunas notas sobre la Figura 12:

Las organización de los protocolos mostradas en la figura fueron realizadas por SIGTRAN, las áreas azules muestran los protocolos existentes, por simplicidad ISUP no es mostrado pero esto puede ser ejecutado normalmente por MTP3 o M3UA.

Para el transporte confiable de señalización a las capas superiores, SIGTRAN utiliza el transporte basado en IP.

Provee la misma clase de servicios ofrecidos por la interfaz RTPC, por ejemplo, M3UA, en términos de servicios puede hacer similitud con MTP3, pero en la actualidad M3UA no reemplaza las características y operaciones MTP3.

Para el usuario final, es transparente que capa de adaptación reemplaza el protocolo original, aunque este sea ampliamente dependiente de la implementación.

A.2.2.3 Capas de adaptación SIGTRAN

SIGTRAN actualmente define seis capas de adaptación, que son las siguientes:

- **M2UA**, provee los servicios MTP2 en una situación cliente-servidor, y en las situación SG a MGC, sus usuarios serán MTP3.
- **M2PA**, provee los servicios MTP2 en una situación par a par, así como en SG a SG, sus usuarios serán MTP3.
- **M3UA**, provee los servicios de MTP3 en el lado cliente–servidor para SG a MGC, así como también en la arquitectura par a par, sus usuarios serán SCCP y/o ISUP.
- **SUA**, provee los servicios de SCCP en la arquitectura par a par, así como SG a IP SCP. Sus usuarios pueden ser TCAP y otras partes de aplicaciones basadas en transacción.
- **IUA**, provee los servicios de la capa de enlace de datos (LAPD) de ISDN , sus usuarios serán entidades ISDN capa 3 (Q931).
- **V5UA**, provee los servicios para el protocolo V.5.2.

Obsérvese que el marco es bastante flexible para permitir la adición de nuevas capas si son requeridas. Cada capa tiene o desempeña una aplicación en particular las cuales describiremos a continuación:

M2UA

M2UA, es utilizado para el transporte de datos de usuario MTP2 entre una instancia MTP2 en una SG y la instancia MTP3 en una MGC. Este opera en el modelo cliente servidor en donde la MGC es el cliente y el SG es el servidor.

M2UA provee un medio por el cual un servicio MTP2 puede ser provisto en un MGC. En esencia extendiendo SS7 dentro de la red IP.

La arquitectura M2UA es mostrada en la Figura 13:

Efectivamente, la instancia MTP3 en un MGC es el usuario de la instancia MTP2 en la SG. Ni MTP2 ni MTP3, son alertados de que ellos trabajan remotamente el uno con el otro. El

proceso por el cual los mensajes de señalización son pasados sobre una red IP desde una superior de SS7 al fondo de otra se llama backhauling.

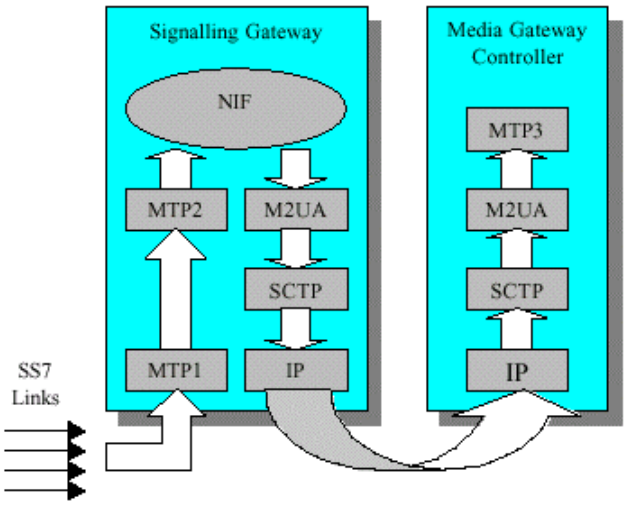

Figura 13: Arquitectura M2UA

El usuario MTP3 de la MGC normalmente será ISUP. Esta arquitectura es mas aplicable en las siguientes circunstancias:

- Cuando la densidad de los enlaces SS7 es baja en un punto fisico en particular de la red.
- Cuando son un gran número de funciones SG separadas físicamente (debido a que los enlaces SS7 están físicamente localizados en forma remota uno del otro.

En este caso, es razonable implementar la capa MTP3 en el MGC. Las direcciones SS7 (de código de punto) de los sistemas residen dentro de la capa MTP3 - si cada SG tiene su propia capa MTP3, un gran numero de direcciones serán requeridas para implementar una sola gateway.

Ésta particular configuración SIGTRAN es muy común en las redes Europeas en donde los enlaces de señalización SS7 son físicamente compartidos con los circuitos de voz.

En el lado de la SG, la descripción de M2UA en el nivel par a MTP2 es ligeramente engañoso. M2UA es en muchos casos un usuario de MTP2, las especificaciones define que M2UA es responsable de iniciar las acciones de protocolos los cuales normalmente serán dados por MTP3, tales como:

- Enlaces de activación y desactivación.
- Secuencia de petición de números.
- MTP2 transmite / retransmite procedimientos actualización de los buffer.
- Limpieza de los buffer.

Estas son implementaciones dadas, sin embargo tales funcionalidades pueden residir adecuadamente dentro de una Función de Nodo de Interconexión (NIF).

M2PA

M2PA es el equivalente par a par de M2UA. Antes de proveer un enlace entre instancias MTP2 y MTP3 remotas, el reemplaza el enlace MTP2 por debajo de MTP3. El usuario de M2PA es MTP3 en ambos lados de la conexión (con M2PA, un usuario es MTP3 y el otro es un SG Interworking Function IWF).

M2PA provee un medio para capas pares de MTP3 entre las SGs comunicadas directamente. En esencia, la red SS7 se extiende sobre la red IP.

Ésta arquitectura es la más aplicada para una conexión SG a SG usado como un puente por dos redes SS7 aisladas. En este caso un cada SG puede conectar muchos otros SGs. De ningún modo estas necesitan conocer algo de las capas superiores que ellas soportan.

MTP3 se presenta en cada SG para suministrar enrutamiento y gestión de los enlaces MTP2/M2PA. Debido a la presencia de MTP3, cada SG requerirá su propia dirección de código de punto.

La diferencia crítica en función de M2UA es que M2PA actualmente provee un servicio de enlace MTP2 asimismo. M2UA simplemente suministra una interfaz a un servicio MTP2 remoto.

De ésta manera M2PA es responsable por:

- Activación y desactivación de enlaces (en respuestas a peticiones de MTP3)
- Mantener los estados de información de los enlaces.
- Mantener números de secuencia y retransmisión de buffer, para retiro de información de MTP3.
- Mantener estadísticas de ociosidad de los procesadores locales y remotos.

M3UA

M3UA es un poco similar a M2UA, este opera como cliente–servidor para suministrar a las capas SS7 superiores un protocolo de acceso remoto a las capas inferiores. M2UA y M3UA operan entre un SG y un MGC.

M3UA provee un medio por el cual un servicio MTP3 se puede suministrar en un MGC (terminando de esta manera la conexión ISUP en el MGC). Otra vez en esencia la red SS7 se extiende dentro de la red IP.La arquitectura que encaja M3UA dentro del modelo SG/MGC es mostrada en la Figura 14.

El MTP3 en el SG ignora que el usuario ISUP está localizado remotamente (el NIF se registrara asimismo como ISUP por medio de MTP3).

Similarmente, la capa ISUP de el MGC será ignorada, así esta no sirve para un MTP3 local, este es otro ejemplo de mensajes de señalización backhauling de una SG.

Esta arquitectura es más apropiada en los siguientes casos:

- Cuando es muy alta la densidad de enlaces SS7 para efectuar un standalone viable.
- Los enlaces SS7 son físicamente accesibles en un único punto.

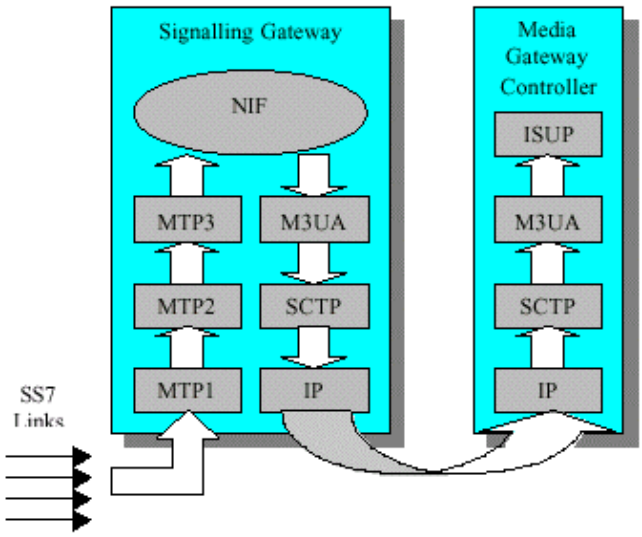

Figura 14: Arquitectura M3UA

Estas condiciones son comunes en redes Norte Americanas, donde los enlaces SS7 son físicamente separados de los circuitos de voz. En este caso, un número de enlaces son reunidos a la vez dentro de un simple medio físico, (por ejemplo una línea T1). Aquí, cada SG tiene una instancia MTP3 local, y así debe tener su propia dirección de código de punto SS7.

La capa M3UA es responsable del mantenimiento de la interfaz MTP3-ISUP de una conexión SCTP. Instrucciones y peticiones para transferencia de datos aprobados en la parte baja por ISUP en la MGC son transportaos (sobre SCTP) por M3UA y presentados a las interfaces altas de MTP3 en la SG. Indicaciones y entradas de datos de mensajes son

aprobados en la parte alta de MTP3 de la SG y transportados (sobre SCTP) por M3UA a la interfaz baja de ISUP en la MGC.

Como con M2UA, la descripción de M3UA es a par a MTP3, puede ser ligeramente engañosa. Lógicamente, M3UA es un usuario MTP3 en la SG.

SUA

SUA provee un medio por el cual una parte de Aplicación (por ejemplo TCAP) en un IP SCP es alcanzable por medio de un SG. La arquitectura de la red asociada con SUA acepta que múltiples IP SCPs sean alcanzables por medio de un simple SG. Los IP SCPs no tienen instancias MTP3 locales, y entonces no requieren ellos de su propia dirección de código de punto SS7, (MTP3 y las direcciones de código de punto, residen en la SG). La arquitectura de SUA usada entre un SG y un IP SCP es mostrada en la Figura 15.

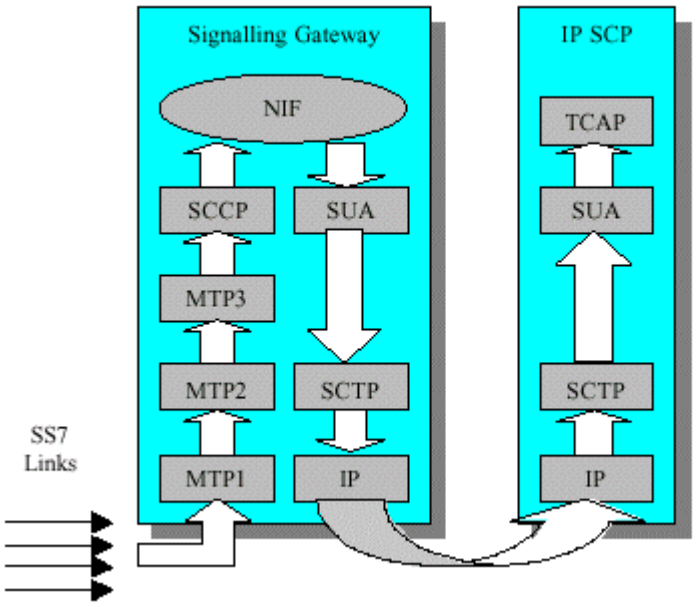

Figura 15: Arquitectura SUA

La funcionalidad de SUA puede ser suministrada por los MTP2 o MTP2 UAs. Sin embargo, SUA provee el mapeo entre direcciones SCCP y direcciones IP en las SG. Sin

dicha función, el SCCP debería estar presente en cada SCP IP y la red externa SS7 necesitaría conocer cada instancia de SCCP. la capa SUA puede abstraer la presencia de cada SCCP IP, permitiendo que un solo SCCP cubra a todos los nodos.

Los servicios de bases de datos individuales son direccionados por medio de un Subsistema de Números (SSN).

SUA es también bastante flexible para soportar ejecuciones de Partes de Aplicaciones entre nodos totalmente dentro de la red IP. Este es particularmente relevante para redes nuevas, cuando estas no necesita de la tradicional red SS7, en este caso, la pila IP SCP será el mismo para ambos nodos (basados en IP).

SUA permitirá fomentar servicios de bases de datos en la red SS7 para ser accedidas desde de la red IP, en éste caso la arquitectura será la misma de la figura anterior.

IUA y V5UA

Estamos interesados en lo concerniente a la adaptación de las capas SS7 sobre SCTP, por lo tanto IUA y V5UA no son relevantes.

A.3 SERVICIOS

Se da paso a algunos servicios que nacen junto a las nuevas características de flexibilidad, rentabilidad y fácil introducción que proporcionan los nuevos modelos de red. Estos servicios han surgido como iniciativa de algunas empresas del sector de las Telecomunicaciones los cuales han aprovechado las ventajas anteriormente mencionadas para convertirse en modelos para otras empresas que están trabajando en el mismo ámbito.

A.3.1 PINT

Por ejemplo, consideremos un usuario que desea que le regresen una llamada a su teléfono, esto puede pasar cuando algún cliente necesite alguien del departamento de soporte de algún negocio y necesite que le devuelva una llamada. Cuando se requiere de una llamada telefónica desde una red IP, tanto la RTPC, como las Redes Inteligentes, PBX's privadas, Redes de telefonía celular, y la RDSI, todas pueden utilizar los servicios PINT para cubrir la petición de la red IP.

Dentro de la arquitectura de Internet, se especifican medios por los cuales se puede establecer una comunicación que son guiadas por protocolos. **SIP** es usado para establecer la asociación de los participantes de la llamada, a esta asociación se le llama una **sesión, SDP** es usado para describir el medio que será utilizado por la sesión. El protocolo PINT utiliza éstos dos protocolos para proveer mejoras y permitir convertir clientes y servidores SIP, se conviertan en clientes y servidores PINT.

Un usuario que requiera un servicio dentro de una red telefónica, utiliza SIP para invitar un servicio PINT dentro de una sesión remota. La invitación contiene una descripción SDP del medio que será utilizado para iniciar la sesión, la cual podría ser "sesión para enviar un fax" o una "sesión para un llamada telefónica". Para PINT el medio de una sesión de un servicio es la red telefónica, mientras que el medio de una sesión SIP es Internet.

Cuando un usuario invoca un servicio PINT, SIP establece una asociación entre la petición cliente PINT y el servidor PINT, responsabilizándose así de la las invocaciones dentro de la red telefónica. Estas dos entidades no son las mismas que la red telefónica utiliza para prestar los servicios de telefonía, los mensajes SIP llevan dentro de la carga útil el SDP, la descripción de la sesión de la red telefónica.

A continuación se darán algunos ejemplos de servicios que presta PINT, junto con un pequeño glosario que nos ayudará a entenderlos mejor.

- **Solicitante:** es un host en Internet el cual realiza la petición de un servicio.
- **Servicio PINT:** es un servicio invocado dentro de una red telefónica, en petición a la petición de un cliente PINT.
- **Cliente PINT:** es un host en Internet que envía las peticiones para la invocación de un servicio PINT.
- **Gateway PINT:** en Internet es un Host que acepta peticiones de un servicios PINT y las envía hacia una red telefónica.
- **Sistema Ejecutivo:** es una interfaz entre un servidor PINT y la red telefónica, encargada de ejecutar los servicios PINT, no necesita estar directamente asociado a Internet, pero representa el servidor de transacciones con la entidades de Internet cuando se relaciona con ellas.
- **Usuario solicitante:** es el iniciador de la petición de un servicio.
- **Llamada del servicio (Party):** es una persona que inicia una llamada telefónica y que resulta en la ejecución de un servicio PINT.

A.3.1.1 Servicios PINT

Petición para llamada: es una petición que se hace desde un host IP para originar una llamada conectando un abonado A con otro B.

Petición para Fax: es una petición que se hace desde un host IP, para enviar un fax a una maquina B. La petición puede contener un puntero con los datos del fax el cual puede residir en la red IP o en la red telefónica. el contenido del fax puede ser texto o puede contener imágenes, los detalles de la transmisión del fax no son accesibles por la red IP, los cuales permanecen en la red Telefónica.

Nótese que este servicio no se relaciona con el de fax sobre IP, la red IP solo se utiliza aquí para enviar la petición de envío de fax, por su puesto que el envío del fax sobre la red telefónica puede ser la solución de tiempo real para la red IP, pero esto es completamente transparente para el usuario.

A.3.1.2 Arquitectura Funcional de PINT

Los clientes y servidores PINT, son clientes y servidores SIP, el cual es utilizado para transportar las peticiones sobre la red IP al servidor PINT correcto de una manera segura y SDP es utilizado para describir la sesión sobre la red telefónica.

Un sistema PINT utiliza proxys SIP pero trabaja como otros servidores para propósitos comunes, en la mayoría de los casos deben ser servidores PINT para evitar introducir retardos dentro de la red telefónica, los servidores con éstas características se le llama gateway PINT, el cual aparece en el sistema SIP como un **agente usuario servidor.** Nótese que dentro de la estructura PINT este representa un usuario, mientras que dentro de la infraestructura de la red telefónica representa una pieza que puede proveer un conjunto de servicios.

El sistema servidor PINT es representado como una nube enfatizando que solo una sola petición PINT puede pasar a través de una serie de locaciones servidoras, servidores proxys

y servidores de redireccionamiento, para finalmente alcanzar el gateway correcto que puede realmente atender la solicitud del proceso pasándola a la red telefónica.

El gateway deberá tener una interfaz con la red telefónica o se podrá comunicar vía otro protocolo o alguna API con el sistema ejecutivo que es capaz de invocar servicios a la red telefónica.

Como un ejemplo, dentro de la Red Inteligente, el gateway PINT puede aparecer para realizar la función Gateway del servicio de control. Dentro de un ambiente de oficina este puede servir como compañero de un PBX, conectados ambos a la LAN local.

A.3.2 SPIRITS (Services in the PSTN Requesting Internet Services)

A continuación se dará una documentación especial sobre el trabajo que el grupo SPIRITS viene realizando, describiendo cuatro implementaciones para los servicios SPIRITS: Korea Telecom., Lucent Technoligies, NEC, y Telia en cooperación con Nortel Networks. Es de tener en cuenta que los servicios SPIRITS, son originalmente diseñados para la RTPC, y son extendidos en este momento a la RTPC, interaccionando con Internet.

Los detalles del servicio ICW, puede ser tratados de diferentes formas dependiendo de la implementación. Las características principales que deberán ser soportadas por cualquier implementación se consignan en el anexo 1.

Notificación de la llamada entrante:

El suscriptor es notificado de la llamada entrante sin tener ningún efecto sobre el MODEM que en ese momento esta manejando la conexión a Internet. Cuando una llamada esta entrando, se le presenta al suscriptor una cuadro de dialogo en el PC. En el cuadro de diálogo puede ser mostrado el número de la línea telefónica que origina la llamada, el nombre de la llamada y la hora de la llamada. Note que para poder mostrar el número de la llamada entrante, este debe estar disponible.

Disposición de la llamada entrante:

Una vez el suscriptor es notificado de la llamada entrante, él tiene varias opciones, las cuales son notificadas en el cuadro de dialogo, estas son:

- Aceptar la llamada sobre la RTPC y terminar la conexión vía MODEM.
- Aceptar la llamada sobre Internet, utilizando voz sobre IP, (VoIP).
- Rechazar la llamada.
- Presentar un mensaje previamente grabado y desconectando la llamada.
- Reenviar la llamada al correo de voz.
- Reenviar la llamada a otro número.
- Rechazar (o reenviar), o no responder: si el suscriptor decide no responder después de un determinado tiempo desde que el cuadro de voz fue mostrado, la llamada será tomada de acuerdo a una base predefinida por él mismo.

Manejo de múltiples llamadas, si mas de una llamada arriban durante la conexión a Internet o durante el proceso de una llamada en espera, varios mensajes serán mostrados al suscriptor indicándole las llamadas que en ese momento están entrando.

A.4 ARQUITECTURAS

A.4.1 TINA Telecommunications Networking Information Architecture

TINA es una arquitectura software abierta para la provisión de servicios de telecomunicaciónes e información que pretende ser aplicado en todas las partes de las telecomunicaciones y sistemas de información como terminales (computadores personales), servicios de transporte (switches, routers, etc.), servidores de servicios web y gestión de servicios (autenticación, facturación, etc.).

4.1.2 Arquitectura TINA

El propósito de éstos principios es garantizar la interoperabilidad, portabilidad y reusabilidad de los componentes software y la independencia de tecnologías específicas. Además de esto se pretende compartir la carga de creación y gestión de servicios, entre todos los diferentes participantes de este medio como son: consumidores, proveedor de servicios y proveedores de conectividad.
Principios TINA:

- **Análisis orientado a Objetos**, este captura la complejidad del sistema desde diferentes ángulos y divide el sistema en un conjunto de modelos.
- **Distribución**, esta parte distribuye los componente software sobre diferentes partes de la red, adecuando las características de tráfico, durabilidad de la red y especifica las diferentes demandas de los compradores. La distribución es soportada por el **Entorno de Distribución de Procesos (DPE).**

- **El desacople de los componentes Software**, se hace con el fin de cambiar un componente debido a cambios en tecnologías subyacentes.
- **Con respecto a la separación de inquietudes**, TINA provee dos separaciones importantes:

La primera separación es entre las aplicaciones y el ambiente en el cual se ejecuta o sea DPE. La segunda es la separación de aplicaciones de la parte de servicios específicos y la parte de gestión y control.

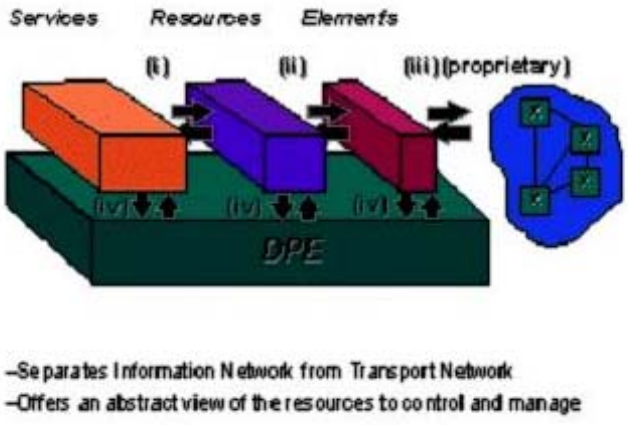

Figura 16: Separación de Redes

Conforme a los anteriores principios, TINA esta dividido en tres sub-arquitecturas:

Arquitectura computacional. Define los conceptos de modelado y el DPE. El DPE reside en diversas piezas de equipos y por ocultar su distribución, hace las funciones de un simple sistema de aplicaciones. El DPE de TINA es basado en CORBA con la adaptación para requerimientos para telecomunicaciones.

Arquitectura de Servicios. Define un conjunto de principios para proveer servicios. Utiliza el concepto sesiones para ofrecer un punto de vista coherente de varios eventos y relaciones que se originan durante la provisión de servicios.

Arquitectura de red. Describe una arquitectura genérica independiente del modelo tecnológico para poder configurar las conexiones y gestión de las redes de telecomunicaciones.

4.1.3 Modelos y Puntos de Referencia

TINA define un conjunto de puntos de referencia, derivados del Modelo de Negocios de TINA. Conforme a estos puntos de referencia, se asegura la interoperabilidad entre los productos de TINA, además TINA captura la complejidad de un sistema utilizando un numero de modelos similares a los ODP (Open Distribuited Processing).

Modelo de Negocios. Describe las diferentes partes que envuelven un servicio provisional y las relaciones entre cada uno de ellas, también son definidos un pequeño numero de roles las cuales reflejan las principales separaciones en el mercado los negocios de telecomunicaciones: proveedores, consumidores minoristas, comerciantes, otros proveedores de servicios y proveedores de conectividad.

Los puntos de referencia abarcan un conjunto de interfaces que describen la interacción que se forma en los comportamientos. Por ejemplo el punto de referencia “Consumidor”, describe la relación entre el proveedor y los consumidores.

Modelo de Información. Describe la información de conexión de la entidades, las relaciones con las otras y las reglas que manejan su comportamiento. Estas son descritas usando la técnica de modelamiento OMT (Object Modeling Object).

Modelo Computacional. Describe los objetos computacionales y las relaciones entre ellos. **El Lenguaje de Definición de Objetos (ODL)**, ha sido desarrollado para ayudar a definir objetos computacionales, esta es una extensión del lenguaje IDL de la OMG.

Se estudiará a continuación las diferentes arquitecturas.

Arquitectura computacional.

Define los conceptos de modelamiento y el DPE. El DPE es un ambiente que soporta la realización de la descripción de aplicaciones en el modelo computacional.

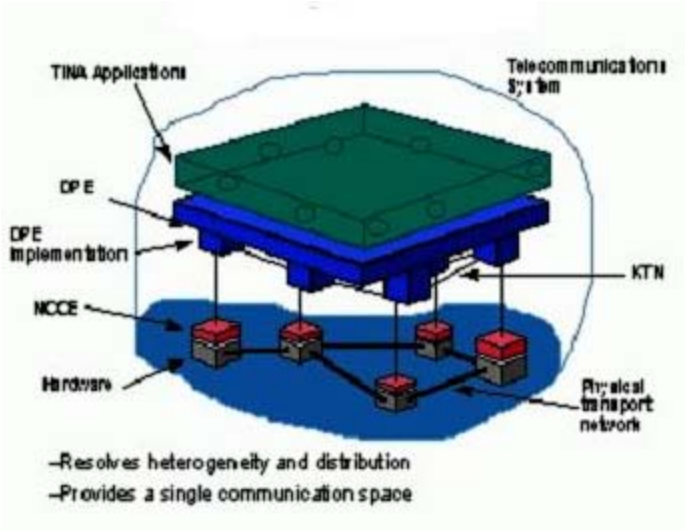

Figura 17: Utilización del DPE

El DPE asegura la transparencia de cierto numero de distribuciones (como localización y acceso), así los objetos computacionales para aplicaciones interactúan con cada uno de los otros vía el DPE, sin considerar detalles propios de la computación y del ambiente de desarrollo (NCCE).

Los servicios y facilidades del DPE, pueden ser distribuidos sobre múltiples nodos conectados por la **Red de Tranporte del Kernel (kTN)**. Tanto el nodo DPE como las kTN's son entidades lógicas y pueden ser implementadas de varias formas.

El DPE de TINA se basa en la tecnología CORBA, estas extensiones tecnológicas de OMG, proveen funciones específicas para las telecomunicaciones, como por ejemplo flujos continuos de información, comunicaciones, la composición de múltiples objetos de interfaz como las interfaces para los usuarios, para gestión y notificación de servicios. Algunas de estas ideas son introducidas por OMGs dominantes en el tema de las telecomunicaciones, las cuales están en proceso de adopción. Una de las metas más grandes de TINA es alinearse con las especificaciones DPE por medio de OMG.

La Arquitectura de Servicios.

Define un conjunto de principios para proporcionar servicios, estos principios se basan en las siguientes normas:

Utiliza el concepto de sesión. Una sesión representa la información usada por todos los procesos que envuelve la prestación de un servicio de cierta duración, por ejemplo en una conferencia multipunto, la información a cerca de las conexiones, cargar y utilización de los archivos de los usuarios asociados a los participantes de la conferencia. Las sesiones de ayuda mantienen información coherente hasta el final de la conferencia, una sesión puede ser muy simple como en el caso de una búsqueda web, el concepto de sesión permite redefinir la separación de accesos, utilización de servicios y comunicaciones.

Los objetos son separados entre **Objetos genéricos o comunes** a todos los servicios y **objetos de servicios específicos** representando a los servicios lógicos, de datos de gestión, etc.

Varias sesiones pueden ser definidas, correspondientes a diferentes tipos de actividad:

La sesión de acceso, corresponde al establecimiento de los términos y condiciones de la sesión durante la conexión del usuario al sistema, como autenticación, selección del perfil del servicio, etc. Esto permite a los usuarios combinar sesiones y participar de varios servicios.

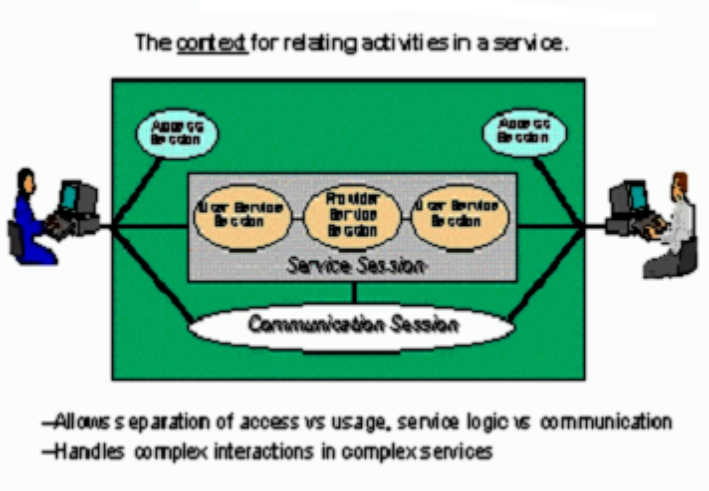

Figura 18: Sesiones TINA

La sesión de servicio, corresponde a la provisión de estos mismos servicios, (por ejemplo moderación de multi-conferencia con capacidades de retroinformación) y asegurar coherencia global del control de gestión, esta es dividido en:

- **Sesión de servicio de usuario**, que gestiona el estado y la actividad de cada usuario (vista local) y atribuye recursos, por ejemplo cargo de entorno, pagina corriente.
- **Sesión de provisión de servicios**, contiene los servicios lógicos y ofrece funciones que permiten al usuario unirse a una sesión, o ser invitado en otra, etc. (vista global).
- **Sesión de comunicación**, provee una vista abstracta de las actuales conexiones en la red de transporte. Esto es descrito en la arquitectura de Red.

La Arquitectura de Red

Define una tecnología genérica independiente, para configurar conexiones y gestión de redes de telecomunicaciones. Ésta hereda conceptos utilizados en la ITU-T y otras normas. Extiende estos conceptos a redes integrales y gestión de Software para diferentes tecnologías de redes.

La Arquitectura de Red esta compuesta por las siguientes capas:

- **Capa de sesión de comunicación**, provee interfaces de servicios independientes para el manejo de componentes del servicio de comunicación end-to-end, en un nivel abstracto. Por consiguiente a esta red le concierne ambas redes y terminales de comunicación.
- **Capa de sesión de Conectividad**, abstrae todas las tecnologías diferentes de redes y provee interfaces independientes de la tecnología para la capa de sesión de comunicación que se interconectar con los puntos terminales de nivel de las redes
- **Capa de Red**, generaliza las abstracciones de cada tecnología de red especifica.

A.5 CAMEL (Customized Application Mobile Enhanced Logic)

Para comenzar, debemos tener en cuenta que la presencia de CAMEL en una red es una ventaja y no un servicio suplementario, es una herramienta que ayuda al operador de red a proveer a los subscriptores servicios específicos cuando el roamming se realiza fuera de la red móvil local, en este documento, la Función de Control del Servicio de GSM (gsmSCF) se trata como parte de la HPLMN (Home Public Land Mobile Network) o Red Móvil Publica Local, considerando que en algunos casos las normas reguladores de algunos países requieren la posibilidad de que el gsmSCF y la HPLMN sean controlados por diferentes operadores y el gsmSCF y la HPLMN son por consiguiente entidades distintas.

A.5.1 Arquitectura Funcional de CAMEL

Se describe a continuación la arquitectura funcional de CAMEL, como también las exigencias funcionales de GSM, la Figura 19 muestra las entidades funcionales involucradas requeridas por CAMEL para dar soporte a una llamada. Esta arquitectura es aplicable a la segunda fase de CAMEL.

MSC y GMSC. La central de conmutación móvil (MSC) y la central de conmutación móvil frontera (GMSC) son los elementos que realizan el encaminamiento y conexionado de los usuarios para los servicios vocales, bajo la dirección del SCP.

SGSN. Serving GPRS Support Node **(SGSN).** Es el elemento que realiza el registro de los usuarios a la red GPRS y participa en la creación de los canales de datos con las redes a las

que quiere acceder el usuario. El registro y creación del portador de datos están controlados por las lógicas de servicio residentes en los SCPs.

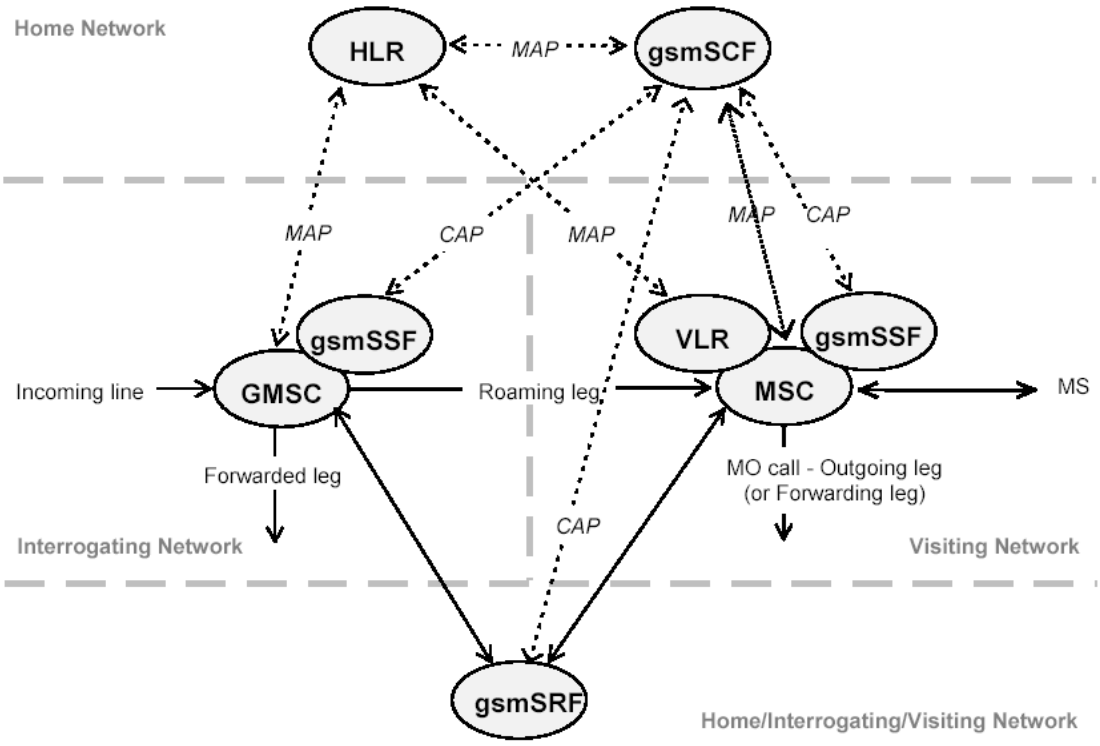

Figura 19: Arquitectura CAMEL

***HLR. Home Location Register* (HLR).** Es la base de datos que contiene los datos de los usuarios CAMEL.

***SMSC. El Short Message Service Center* (SMS).** El centro de mensajes cortos que permite la realización de SVA basados en el análisis y control de mensajes cortos, originados por los móviles mediante las lógicas de servicios residentes en los SCPs.

Puede observarse que en el modelo de CAMEL no está especificado el SDP. Esto no quiere decir que en la realidad no pueda utilizarse los SDPs para implementar lógicas de servicio

de CAMEL, sino que la interfaz que se defina entre el SCP y el SDP no está contemplada en el estándar. Debido al efecto de *roaming* la diferencia entre el escenario de una llamada saliente (*originating call*) y una llamada entrante (*terminating call*) es más importante en CAMEL que en la RI. En el caso de una llamada saliente, el MSC de la red visitada (*visited network*) puede pedir un tratamiento del servicio al SCP de la red de origen (*home network*). En cambio, una llamada entrante siempre pasa por el Gateway MSC de la red de origen, antes de ser desviado a la red visitada. En este caso, tanto el Gateway MSC de la *home network* como el MSC de la red de destino puede pedir un tratamiento de servicio.

El mecanismo de activar los servicios (*triggering*) también es diferente en CAMEL respecto a la RI. Un operador de red móvil, generalmente no puede manipular directamente las tablas de activación de servicios en los conmutadores de otra red. Por tanto, la activación de servicios se hace a través de la interfaz HLR-VLR que está definida en GSM.

Además del control de llamadas móviles, CAMEL ofrece una gran riqueza de posibilidades para asociar servicios de valor agregado a la gestión de movilidad. CAMEL permite al operador trasplantar los servicios clásicos de la RI al entorno de red móvil, pero además permite prestar nuevos servicios que son propios de los terminales móviles.

Un aspecto que aún no se ha tratado y que es crítico en la convergencia entre las comunicaciones móviles y los servicios de datos, es la posibilidad de ofrecer movilidad a servicios basados en IP de manera transparente para los protocolos de la capa superior. Es por esto que se ha decidido incluir a continuación una referencia de una de las soluciones que actualmente se tiene sobre este aspecto.

B.1 DESCRIPCIÓN DEL SISTEMA

A continuación se da una descripción del ambiente y las responsabilidades del sistema a implementar.

B.1.1 Entorno del sistema

El objetivo de este software es el de simular la prestación de servicios de telecomunicaciones usando la arquitectura JAIN, en particular el control de llamadas JAIN-JCC, la cual provee los mecanismos para la iniciar, observar, responder y manipular llamadas independientemente de la red en la cual se encuentran los abonados.

En el servicio de telefonía plana convencional, una llamada telefónica es un conexión entre dos usuarios, a los cuales se les llama abonados, cada uno de ellos se diferencia en cuanto a su función en el establecimiento de la llamada, de esta forma el abonado que solicita la llamada es conocido como abonado A y el que recibe la llamada se le conoce como abonado B, esta diferenciación puede tener implicaciones en cuanto al cobro de la llamada, de esta forma usualmente se le cobra el valor correspondiente al abonado A que fue el que solicito la llamada.

Adicionalmente, existen servicios que permiten dar un tratamiento especial a la llamada, por ejemplo, que la llamada sea recibida en un número telefónico diferente al discado o que la llamada no sea iniciada por el abonado A, si no por una aplicación diferente, a este tratamiento especial de la llamada se le conoce como servicio de valor agregado, y a la persona que contrata el servicio como suscriptor.

B.1.2 Requerimientos del sistema

Como se explicó en la monografía, el objetivo principal de este trabajo es el de desarrollar un software que permita observar la interacción de los diferentes elementos de red que intervienen en la arquitectura JAIN, para esto es necesario que el sistema pueda simular el envió y recepción de llamadas usando el API de control de llamadas JAIN.

Para simular el envió y recepción de llamadas es necesario contar con software que emule un terminal telefónico y que pueda interaccionar con el usuario, es decir, que el usuario pueda presionar teclas como lo hace en un terminal telefónico normal, recibir alertas del terminal, como es el caso de cuando entra una llamada telefónica, debe poder realizar llamadas y recibir llamadas usando dicho terminal.

Una vez logrado esto se busca montar usando este emulador los siguientes servicios:

- Click-to-Dial: Un usuario puede iniciar una llamada telefónica haciendo click en un botón durante una sesión Web, el abonado B y el abonado A pueden ser teléfonos IP o líneas telefónicas convencionales o móviles.
- Sígueme: El usuario puede registrar su teléfono para que las llamadas entrantes sean transferidas a otro número telefónico, está configuración de los datos del servicio se puede realizar a través de una sesión Web.

B.1.3 Modelo de Casos de Uso

A continuación se muestra el modelo completo de casos de uso de alto nivel que se implementaron.

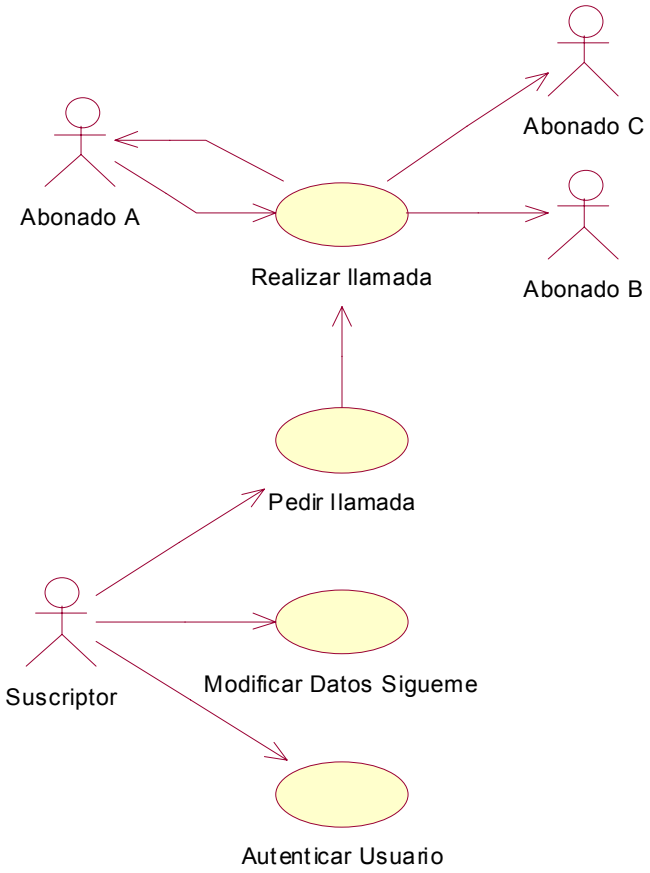

Figura 20: Diagrama de casos de uso

En el modelo de casos de uso se identificaron 4 actores:

- **Suscriptor**: es la persona que contrata el servicio y que tiene acceso para modificar los datos del servicio.
- **Abonado A**: es el usuario que inicia la llamada o al que se le cobra.
- **Abonado B**: es el usuario que recibe la llamada o al que estaba dirigida originalmente.
- **Abonado C**: es el usuario al cual se redirecciona la llamada en el caso del servicio sígueme.

1.3.1 Caso de Uso: Autenticar Usuario.

Iniciador: Suscriptor.

Propósito: Verificar el acceso al sistema.

Resumen: El sistema identifica al usuario según su número telefónico y la contraseña que introdujo, si el estos datos son validos se le permite el acceso.

Tipo: Primario.

Precondiciones:

- El sistema debe contar con la los datos del número telefónico y contraseña proporcionados por el suscriptor.

Flujo principal:

- El Suscriptor accede a la página que le pregunta su número telefónico y contraseña, Figura 21.
- El Suscriptor ingresa sus datos y presiona el botón aceptar para ingresar al sistema.
- El Suscriptor accede a la página que le despliega los servicio disponibles, estos servicios son: Realizar llamada o Transferir llamada, Figura 22.

Flujos de excepción:

- E1: Si los datos suministrados no son validos se despliega un mensaje de error, Figura 23

Interfaces de Usuario:

IU_Login

IU_Servicios

IU_Error

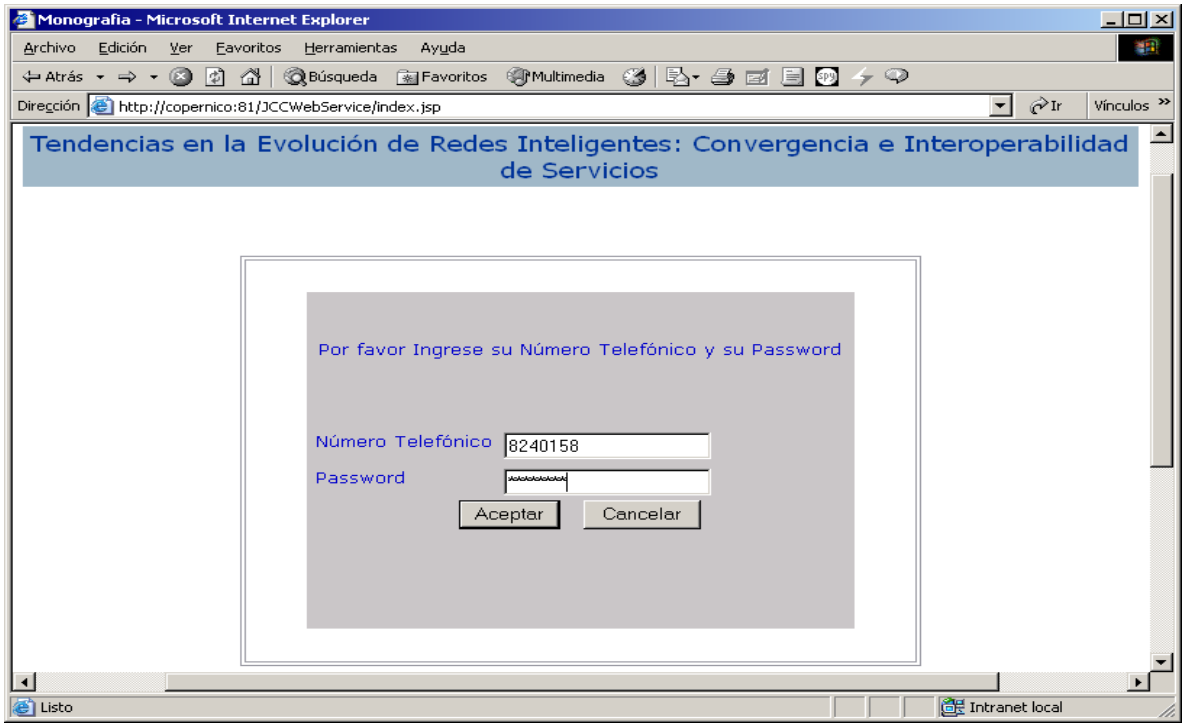

Figura 21: IU_Login

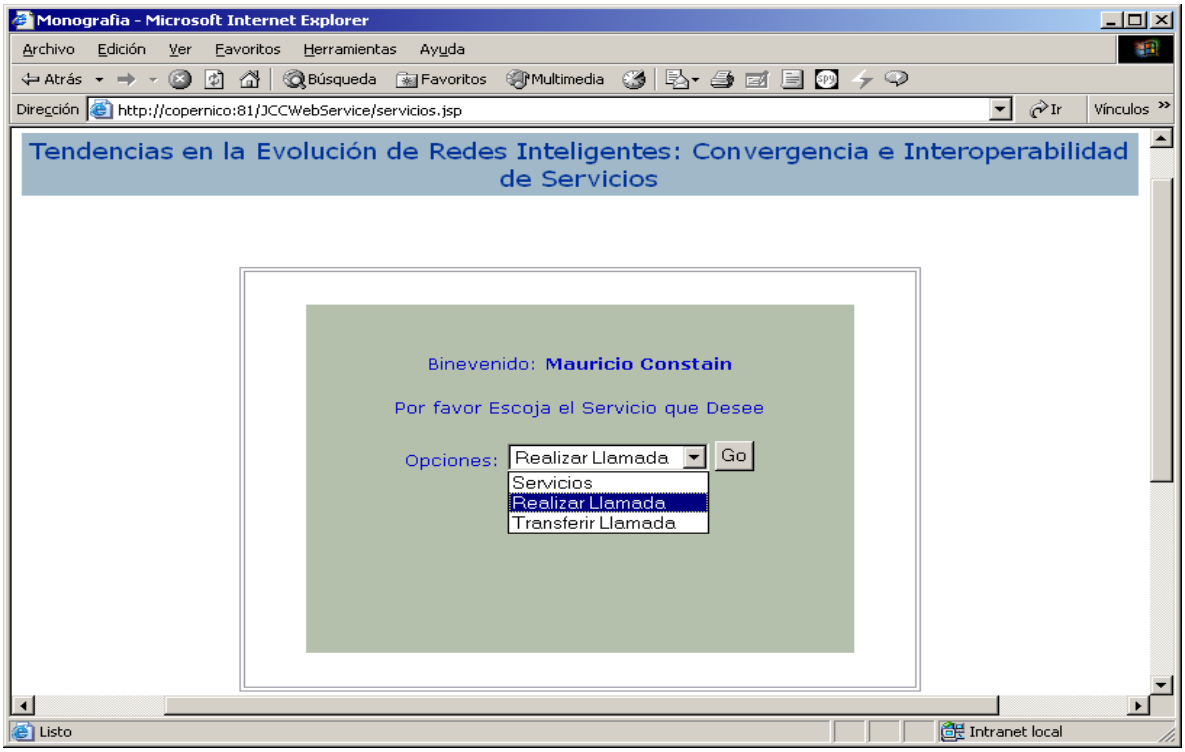

Figura 22: IU_Servicios

1.3.2 Caso de Uso: Modificar Datos Sígueme.

Iniciador: Suscriptor.

Propósito: Modifica los datos del servicio sígueme.

Resumen: El Suscriptor configura el número telefónico al cual desea redireccionar las llamadas.

Tipo: Primario.

Precondiciones:

- El Suscriptor debe haber sido identificado por el sistema usando el caso de uso Autenticar Usuario.
- El sistema debe contar con el número telefónico de destino de la transferencia.

Flujo principal:

- El Suscriptor accede a la página que le pregunta el número telefónico al cual desea transferir sus llamadas, Figura 24
- El Suscriptor ingresa el dato y presiona el botón aceptar para grabar los datos.

Flujos de excepción:

- E1: Si los datos suministrados no son validos se despliega un mensaje de error.

Interfaces de Usuario:

IU_Transferencia

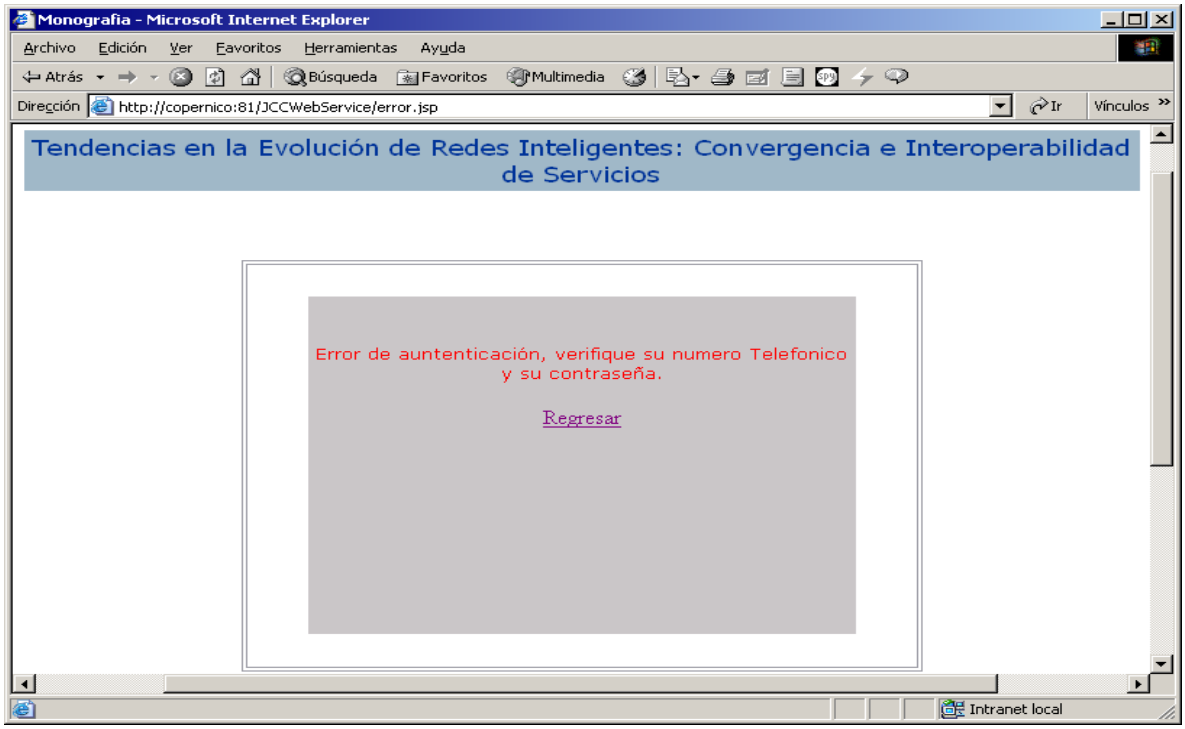

Figura 23: IU_Error

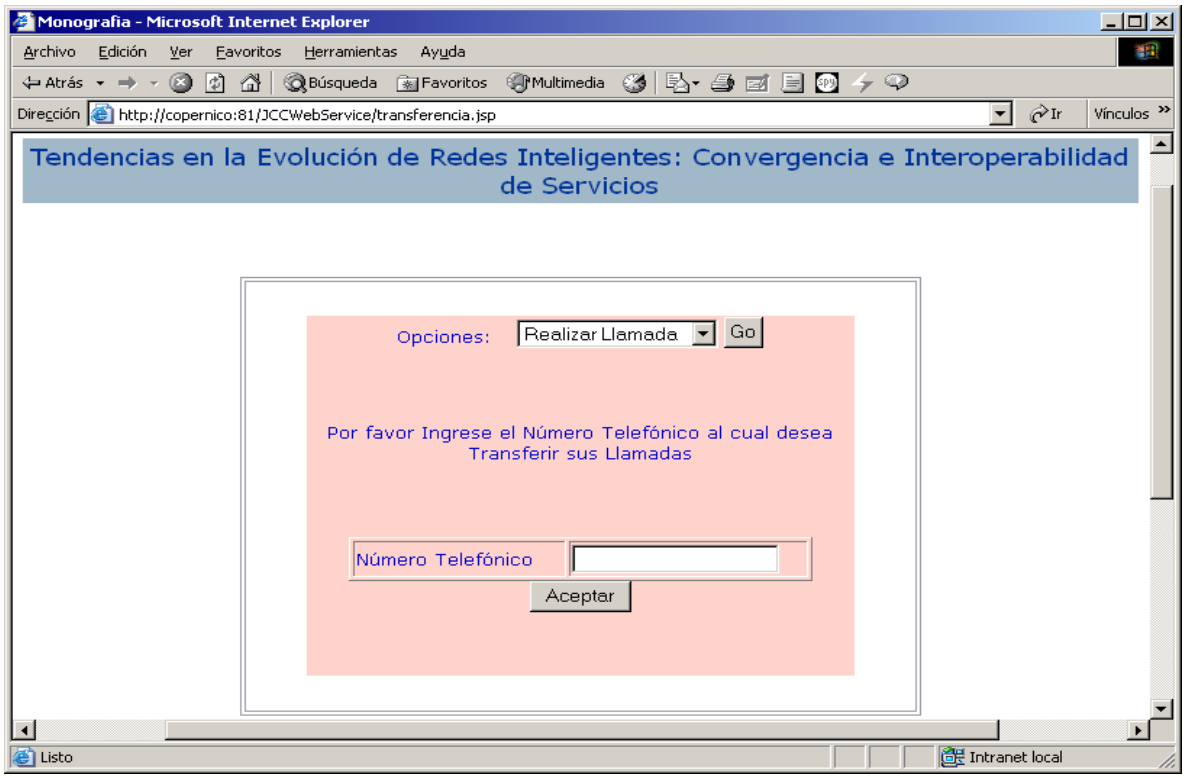

Figura 24: IU_Transferencia

1.3.3 Caso de Uso: Pedir Llamada.

Iniciador: Suscriptor.

Propósito: Iniciar una llamada telefónica entre el suscriptor y el número proporcionado.

Resumen: El Suscriptor pide se inicie una llamada telefónica entre su terminal y el de otro abonado.

Tipo: Primario.

Precondiciones:

- El Suscriptor debe haber sido identificado por el sistema usando el caso de uso Autenticar Usuario.
- El sistema debe contar con el número telefónico de destino de la llamada.

Flujo principal:

- El Suscriptor accede a la página que le pregunta el número telefónico al cual desea llamar, Figura 25.
- El Suscriptor ingresa el dato y presiona el botón llamar para pedir la llamada.
- El sistema le indica que la llamada se encuentra en progreso, Figura 26.

Flujos de excepción:

- E1: Si los datos suministrados no son validos se despliega un mensaje de error.

Interfaces de Usuario:

IU_Llamada

IU_Mensaje

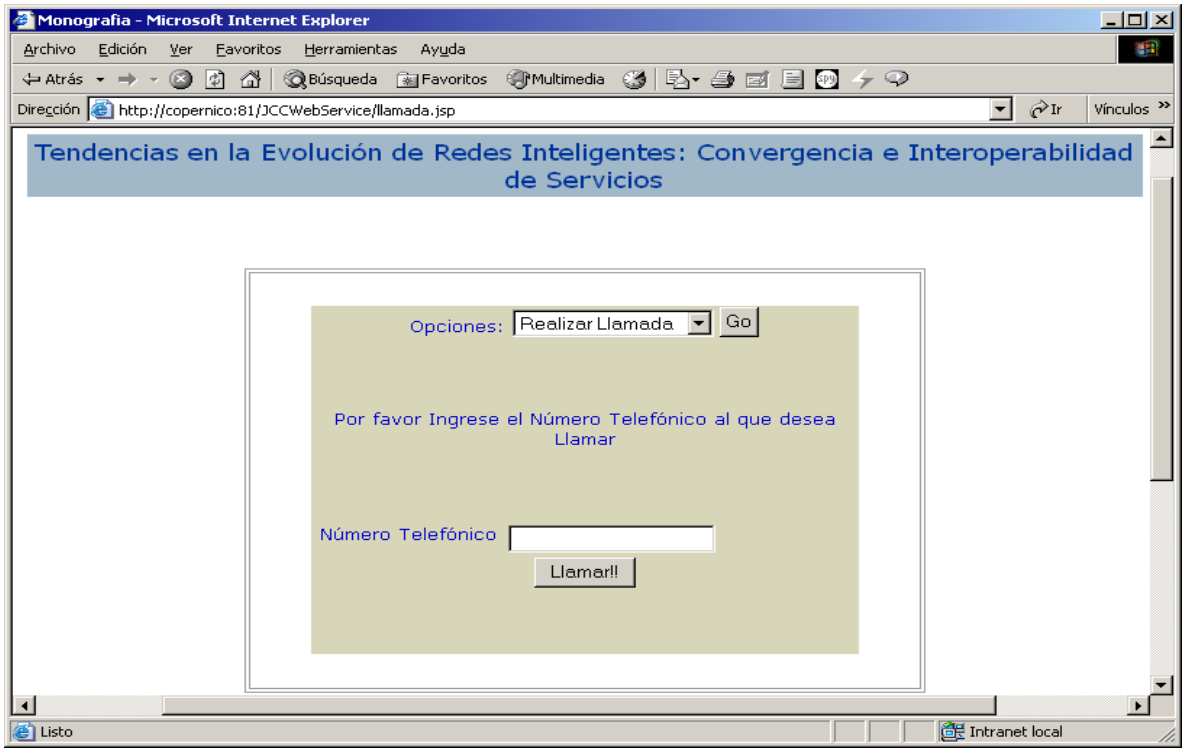

Figura 25: IU_Lamada

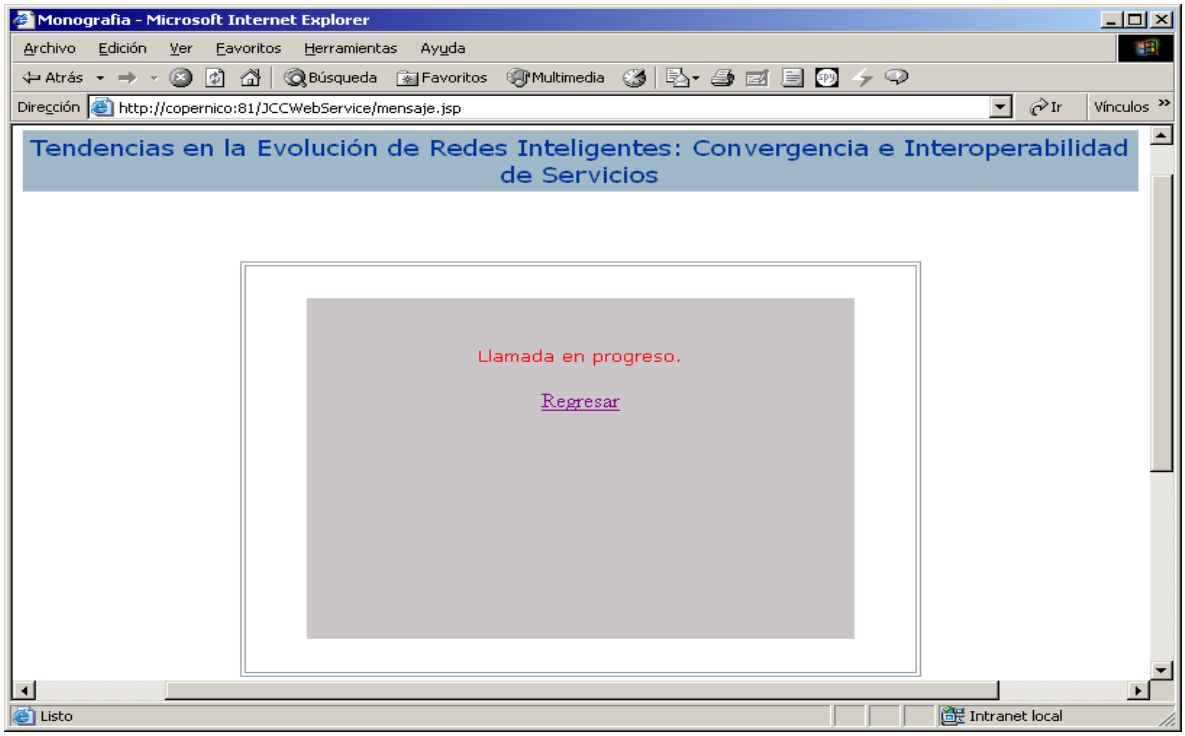

Figura 26: IU_Mensaje

1.3.4 Caso de Uso: Realizar llamada.

Iniciador: Abonado A, Caso de uso Pedir llamada.

Propósito: Iniciar una llamada telefónica entre dos abonados.

Resumen: Se identifica al los abonados A y B y se inicia la llamada.

Tipo: Primario.

Precondiciones:

- Se debe tener el número telefónico de origen (Abonado A) y del destino (Abonado B).
- Los abonados de origen y destino deben haber sido activados.

Flujo principal:

- Se deben activar los teléfonos necesarios para la simulación, abonados A, B y C.
- Se determina el origen de la llamada, el cual puede ser: uno de los terminales antes activados, en este caso se continúa la operación, o la aplicación web que envía una petición de llamada, en este caso se obtiene el número telefónico del origen de la información enviada por la aplicación web.
- Se determina si el destino de la llamada es el número telefónico suministrado, en este caso se continúa normalmente, o si se debe redireccionar la llamada a otro teléfono, en este caso se consulta la información del suscriptor para saber a que número se debe redireccionar la llamada.
- Se realiza la llamada entre los terminales correspondientes, el terminal del abonado que recibe la llamada debe timbrar y una vez presione la tecla de contestar la llamada queda establecida.
- Cuando alguno de los terminales cuelga, se liberan los recursos

B.1.4 Definición de la arquitectura de referencia

Debido a que el objetivo de la aplicación es mostrar el uso del API JCC de JAIN y su integración con aplicaciones externas, se define como arquitectura de la aplicación la que se

muestra en la Figura 27, en esta se pueden distinguir dos partes que son el objetivo de esta implementación:

- La aplicación JCC y servidor que se encargan de la simulación de la prestación de los servicios antes descritos.
- La aplicación web y el cliente de la aplicación, que son los encargados de permitir la personalización de los servicios.

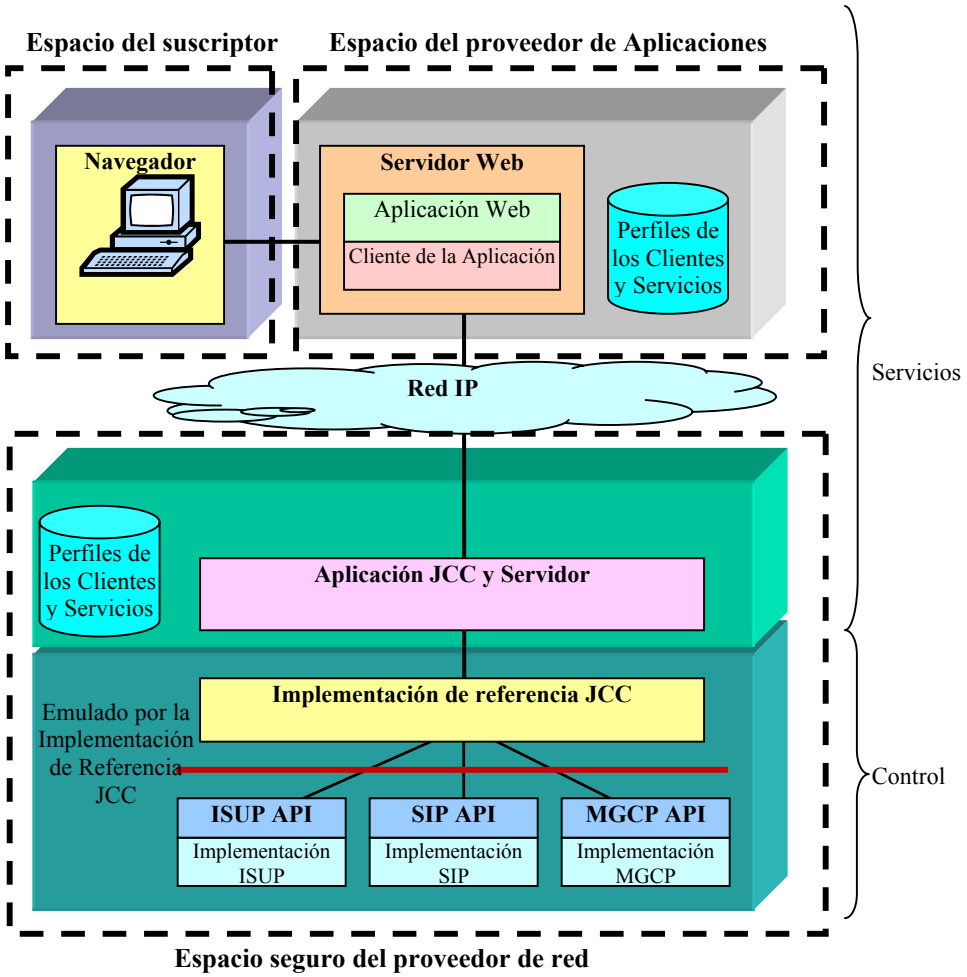

Figura 27: Arquitectura del sistema

Además también se requiere la implementación de las bases de datos de suscriptores y servicios para ambas aplicaciones.

B.2 ANALISIS DEL SISTEMA

En esta sección se pretende dar una descripción de los componente del sistema, se busca lograr una descomposición del sistema en clases de diseño, las cuales servirán de base para las siguientes fases del desarrollo.

B.2.1 Diagramas de Clases

Los diagramas de clases expresan en forma general la relación existente entre las diferentes clases que componen el sistema, en la Figura 28 se observa el diagrama de clases de la aplicación.

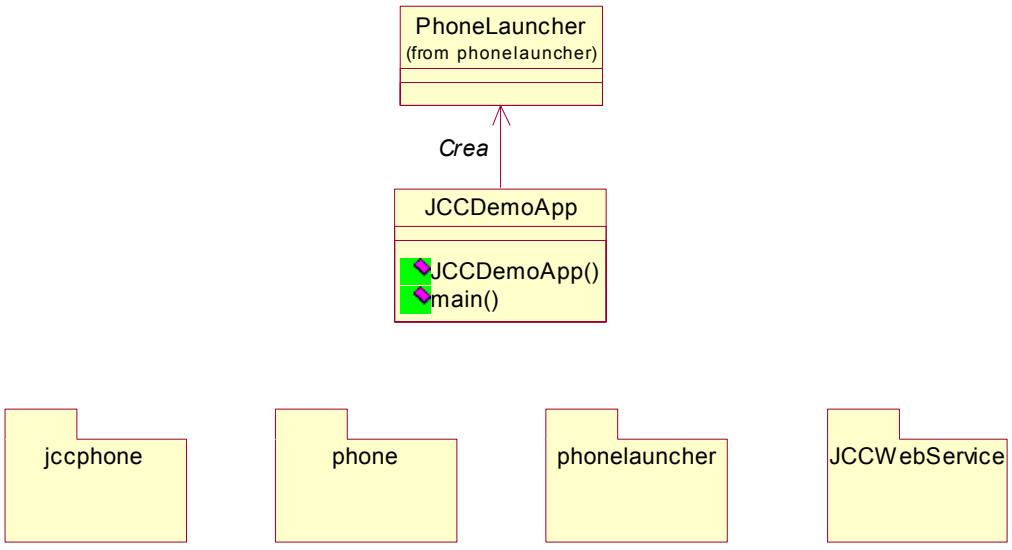

Figura 28: Diagrama de Clases de la aplicación

En la Figura 28 observamos la organización de las clases de la aplicación, para una mejor organización de las clases se han agrupado en 4 paquetes: jccphone, phone, phonelauncher

y JCCWebService, cuyos contenidos se describen en las secciones siguientes, además se observa una clase independiente llamada JCCDemoApp que se describirá a continuación.

JCCDemoApp	
Descripción:	Es la clase que contiene el método principal main() de la aplicación y por lo tanto es la primera que se ejecuta, se encarga de crear a la clase PhoneLauncher que es la que contiene la lógica y la interfaz gráfica de la aplicación.
Métodos:	• **JCCDemoApp:** constructor de la clase. • **Main:** método principal de la aplicación, se encarga de crear la clase PhoneLauncher.
Atributos:	Ninguno.

B.2.1.1 Clases del Paquete phone

En este paquete están contenidas las clases necesarias para la interfaz grafica de un terminal telefónico, se tienen además las imágenes y los sonidos necesarios para que el software se comporte como un terminal telefónico simulado. El diagrama de clases de este paquete se puede apreciar en la Figura 29, para una mejor comprensión se han omitido los métodos y atributos de las clases, los cuales se encuentran en las Figuras 11 a 13.

En este paquete se han identificado 9 clases, cuya relación general se muestra en la Figura 29 y cuya descripción sigue a continuación agrupándolas por su función, se iniciará con la descripción de las clases usadas para manejar las imágenes que componen la interfaz gráfica y que se encuentran desglosadas en la Figura 30.

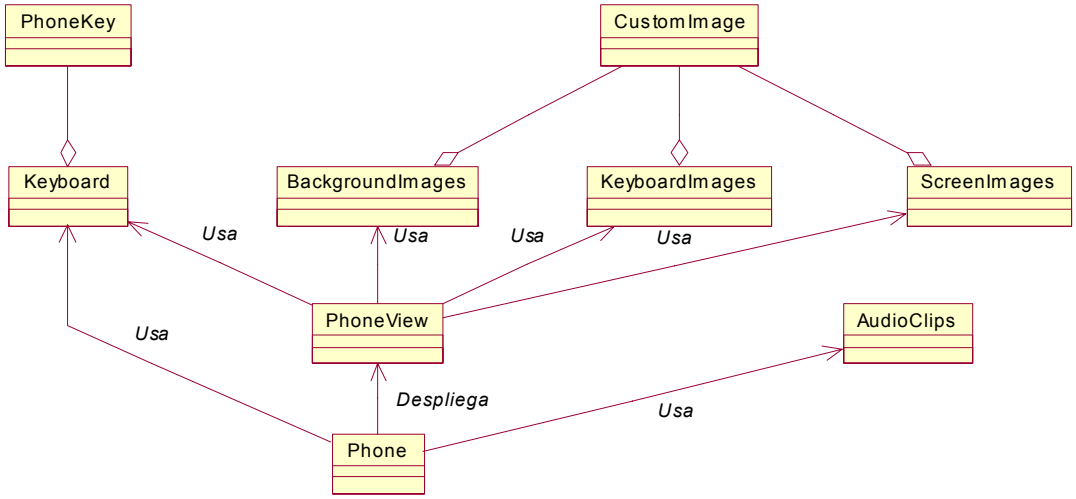

Figura 29: Diagrama de clases resumido del paquete phone

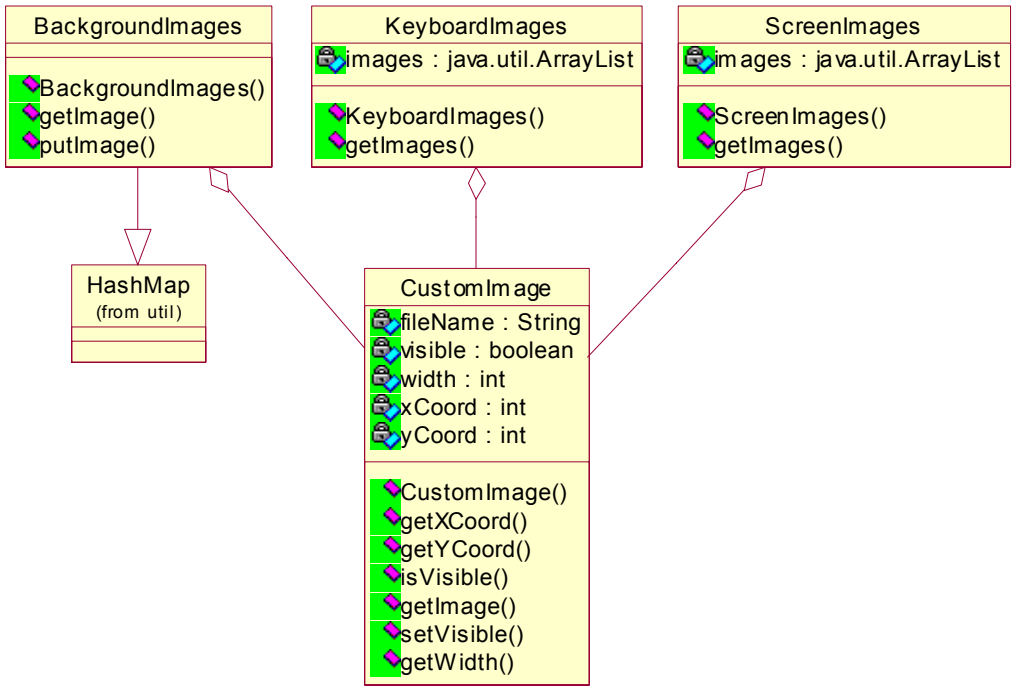

Figura 30: Diagrama de clases del paquete phone, parte 1

CustomImage	
Descripción:	Esta clase representa una imagen con algunos atributos adicionales que son de importancia para la aplicación, como son su posición en coordenadas, si se encuentra visible o no, etc.
Métodos:	• **CustomImage:** constructor de la clase. • **getXCoord**: retorna la coordenada X donde debe ser colocada la imagen. • **getYCoord**: retorna la coordenada Y donde debe ser colocada la imagen. • **isVisible**: retorna verdadero o falso dependiendo si la imagen es visible o no. • **getImage**: retorna un objeto de la clase Image, el cual es la representación binaria de la imagen a desplegar. • **setVisible**: cambia el estado de visibilidad de la imagen. • **getWidth**: retorna el ancho de la imagen.
Atributos:	• **fileName**: nombre del archivo que contiene la imagen. • **visible**: atributo que define si una imagen es visible o no. • **width**: ancho de la imagen. • **xCoord**: coordenada X donde debe colocarse la imagen. • **yCoord**: coordenada Y donde debe colocarse la imagen.

La imágenes usadas para construir la interfaz grafica se clasificaron en 3 tipos:

- BackgroundImages: Son las imágenes de fondo del teléfono por ejemplo la imagen que contiene la apariencia del teléfono.
- ScreenImages: Son las imágenes que se ubican dentro de la pantalla del teléfono y con las cuales se componen los mensajes que muestra dicha pantalla, por ejemplo, al marcar un número telefónico, la pantalla muestra el número discado.
- KeyboardImages: Son las imágenes que se usan para las animaciones cuando es presionada una tecla del teléfono.

BackgroundImages	
Descripción:	Esta clase representa las imágenes de fondo usadas para componer la interfaz grafica del terminal telefónico. Hereda de la clase java.util.HashMap.
Métodos:	• **BackgroundImages:** constructor de la clase. • **getImage**: retorna la imagen identificada por el nombre pasado como parámetro. • **putImage**: verifica si la imagen identificada por el nombre pasado como parámetro ya ha sido cargada, en caso contrario la carga.
Atributos:	Ninguno.

KeyboardImages	
Descripción:	Esta clase representa las imágenes usadas para las animaciones cuando se presiona alguno de los botones del teléfono.
Métodos:	• **KeyboardImages:** constructor de la clase. • **getImages**: retorna un arreglo conteniendo las imágenes.
Atributos:	• **images**: una lista de arreglos de objetos de la clase CustomImage, que almacena las imágenes.

ScreenImages	
Descripción:	Esta clase representa las imágenes de usadas para mostrar mensajes en la pantalla del terminal telefónico.
Métodos:	• **ScreenImages:** constructor de la clase. • **getImages**: retorna un arreglo conteniendo las imágenes.
Atributos:	• **images**: una lista de arreglos de objetos de la clase CustomImage, que almacena las imágenes.

Existen otras dos clases que abstraen el teclado del terminal telefónico y se usan para controlar las animaciones del teclado, así como para saber si se ha presionado una tecla valida, estas se pueden observar en la Figura 31.

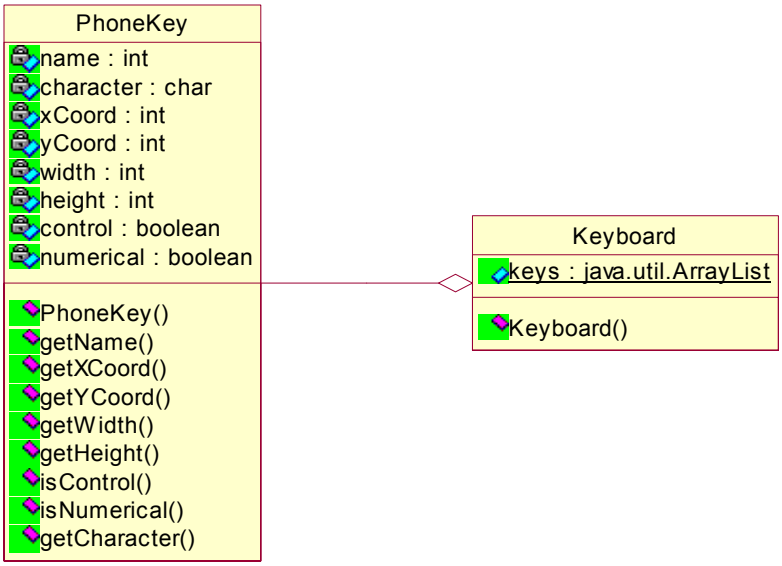

Figura 31: Diagrama de clases del paquete phone, parte 2

PhoneKey	
Descripción:	Esta clase representa una tecla del teclado del terminal telefónico, define algunos atributos necesarios para que la aplicación pueda identificar cual tecla fue presionada y así realizar las tareas correspondientes.
Métodos:	• **PhoneKey:** constructor de la clase. • **getName**: retorna el nombre de la tecla. • **getXCoord**: retorna la coordenada X de la tecla. • **getYCoord**: retorna la coordenada Y de la tecla. • **getWidth**: retorna el ancho de la tecla. • **getHeight**: retorna el alto de la tecla.

	• **isControl**: evalúa si la tecla es de control y retorna falso o verdadero. • **isNumerical**: evalúa si la tecla es numérica y retorna falso o verdadero. • **getCharacter**: retorna el carácter que representa la tecla.
Atributos:	• **name**: nombre de la tecla. • **character**: carácter que representa a la tecla, en el caso de que sea un dígito corresponde al carácter que representa a dicho dígito. • **xCoord**: coordenada X de la tecla. • **yCoord**: coordenada Y de la tecla. • **width**: ancho de la tecla. • **height**: alto de la tecla. • **control**: define si la tecla es una tecla de control, por ejemplo la tecla SEND. • **numerical**: define si la tecla es una tecla numérica.

Keyboard	
Descripción:	Esta clase representa el teclado del terminal telefónico, es una agregación de objetos de la clase PhoneKey.
Métodos:	• **Keyboard:** constructor de la clase.
Atributos:	• **keys**: una lista de arreglos de objetos de la clase PhoneKey, que almacena las teclas que conforman el teclado.

Otro componente de la aplicación es la clase AudioClips, que se puede apreciar en la Figura 32, y es el encargado de almacenar los sonidos correspondientes a los tonos DTMF, además de los sonidos usados cuando el terminal telefónico esta timbrando como se describe a continuación.

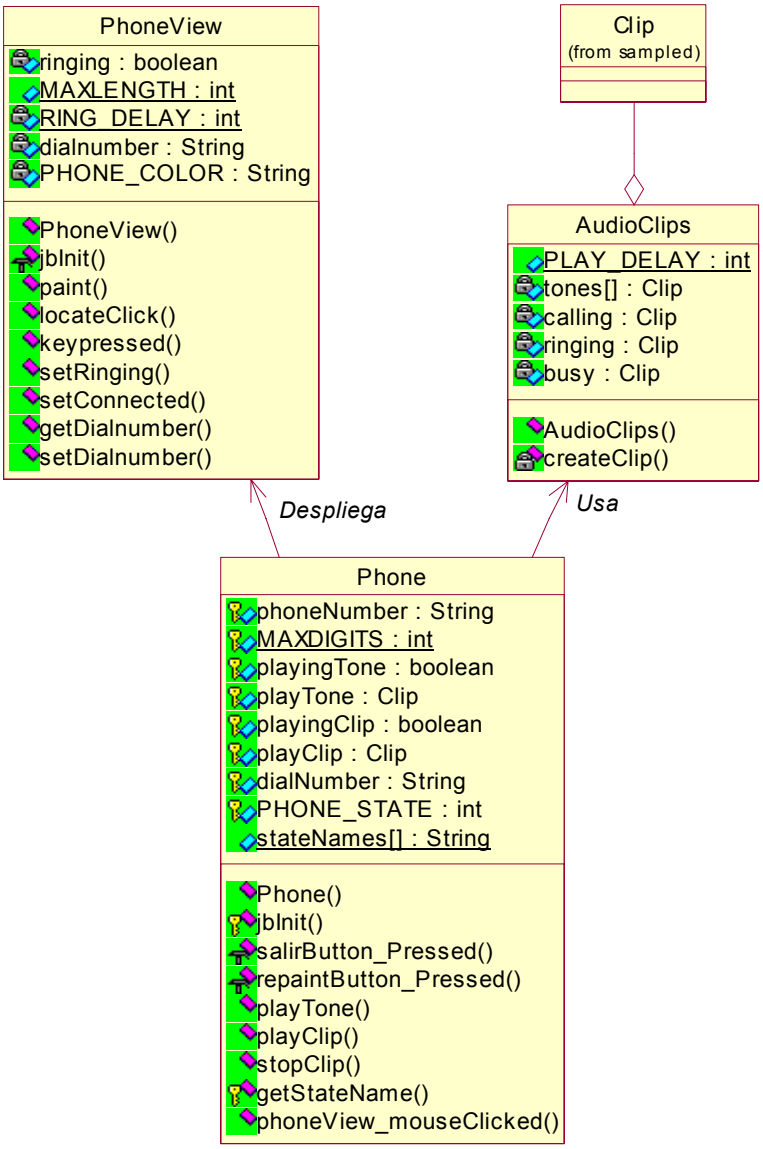

Figura 32: Diagrama de clases del paquete phone, parte 3

AudioClips	
Descripción:	Esta clase los sonidos usados al presionar las teclas y también para hacer timbrar el terminal telefónico, es una agregación de objetos de la clase javax.sound.sampled.Clip que es la clase usada para representar los datos de audio de un sonido.
Métodos:	• **AudioClips**: constructor de la clase. • **createClip**: carga el sonido del archivo especificado como parámetro y crea un objeto Clip que representa a dicho sonido.
Atributos:	• **PLAY_DELAY**: constante que define el tiempo que dura el sonido al presionar una tecla en milisegundos. • **tones**: arreglo de sonidos que corresponden a los tonos DTMF y que son usados al presionar una de las teclas numéricas. • **calling**: sonido que se usa cuando el teléfono está sonando para el abonado A. • **ringing**: sonido que se usa cuando el teléfono está sonando para el abonado B. • **busy**: tono de ocupado.

Las clases descritas hasta ahora se pueden considerar como clases básicas o componentes, las dos clases siguientes: PhoneView y Phone son las clases que tienen la lógica de la aplicación y usan a las clases componentes para realizar sus tareas, el diagrama de clases de estas se puede encontrar en la Figura 32.

PhoneView	
Descripción:	Esta clase controla la forma como se despliegan las imágenes en pantalla y recibe los eventos cuando el usuario presiona una tecla, hereda de la clase java.swing.JPanel que es la una clase básica usada

	para desplegar infamación en la pantalla.
Métodos:	• **PhoneView**: constructor de la clase. • **jbInit**: método que inicializa los componentes necesarios para el funcionamiento de la clase, es invocado por el constructor. • **paint**: método usado para dibujar el componente en la pantalla, dentro de el se despliegan las imágenes BackgroundImages, ScreenImages y KeyboardImages que estén visibles en el momento. • **locateClick**: este método es invocado con el objetivo de saber si un par de coordenadas X,Y se encuentran dentro de una tecla, en el caso afirmativo se retorna el código de la tecla, en caso contrario se retorna -1. • **keypressed**: este método se encarga de la animación cuando se presiona una tecla. • **setRinging**: se invoca cuando el terminal telefónico se encuentra timbrando, se encarga de la animación correspondiente. • **setConnected**: se encarga de activar las imágenes correspondientes para desplegar en pantalla el mensaje de llamada activa. • **getDialnumber**: retorna el número telefónico discado. • **setDialnumber**: modifica el número telefónico discado.
Atributos:	• **ringing**: atributo que define si el terminal telefónico está actualmente timbrando. • **MAXLENGTH**: constante que define la longitud máxima del número discado. • **RING_DELAY**: constante que define el tiempo que dura cada repetición del sonido de timbre cuanto entra una llamada. • **dialnumber**: número discado. • **PHONE_COLOR**: color del teléfono, usado para diferenciar mas fácilmente entre dos terminales telefónicos.

Phone	
Descripción:	Esta clase contiene la lógica que emula la respuesta de un terminal telefónico, se encarga de llevar el estado actual del terminal y de acuerdo a ello determina que sonidos, imágenes o animaciones se deben activar.
Métodos:	• **Phone**: constructor de la clase. • **jbInit**: método que inicializa los componentes necesarios para el funcionamiento de la clase, es invocado por el constructor. • **salirButton_Pressed**: evento invocado al presionar el botón salir, cierra la ventana y libera el terminal. • **repaintButton_Pressed**: evento invocado al presionar el botón repaint, dibuja en la pantalla el componente en caso necesario. • **playTone**: método encargado de tocar el tono DTMF correspondiente a una tecla cuando esta es presionada. • **playClip**: se encarga de tocar el tono de timbrado, la diferencia con el método playTone es que este sonido se repite a intervalos regulares hasta que el teléfono sea contestado o la llamada rechazada, mientras que en el evento playTone se coloca el sonido una sola vez. • **stopClip**: detiene un tono que se esté tocando actualmente. • **repaintButton_Pressed**: evento invocado al presionar el botón repaint, dibuja en la pantalla el componente en caso necesario. • **phoneView_mouseClicked**: este método es invocado cuando se hace click sobre el componente PhoneView, se encarga de averiguar si se presiono una tecla y activa las animaciones, imágenes o sonidos correspondientes.
Atributos:	• **phoneNumber**: atributo que almacena el número telefónico de este terminal.

- **MAXDIGITS**: constante que define el número máximo de dígitos que puede tener un número discado.
- **playingTone**: define si se está tocando un tono en un momento determinado.
- **playTone**: contiene el tono que se debe tocar en un momento determinado.
- **playingClip**: define si se está tocando un clip en un momento determinado.
- **playClip**: contiene el clip que se está tocando actualmente.
- **dialNumber**: contiene el número telefónico discado.
- **PHONE_STATE**: estado actual del teléfono.
- **stateNames**: nombre de los estados del teléfono, usado para desplegar mensajes más intuitivos.

B.2.1.2 Clases del Paquete jccphone.

El paquete jccphone contiene la clase JCCPhone que se encarga de implementar la interfaz JccConnectionListener y de esta manera, habilita al terminal telefónico para ser notificado de los eventos que produce el API JCC y por lo tanto para recibir llamadas y alertar a los abonados de las llamadas entrantes, el diagrama de clases se puede observar en la Figura 33.

La interfaz JccConnectionListener de API reporta los cambios en los objetos JccCall y JccConnection, y define varios métodos que deben ser implementados para responder a estos cambios, una descripción más detallada de estos objetos que forman parte del API se puede encontrar en la documentación del API JCC o en el capítulo 4 de esta monografía.

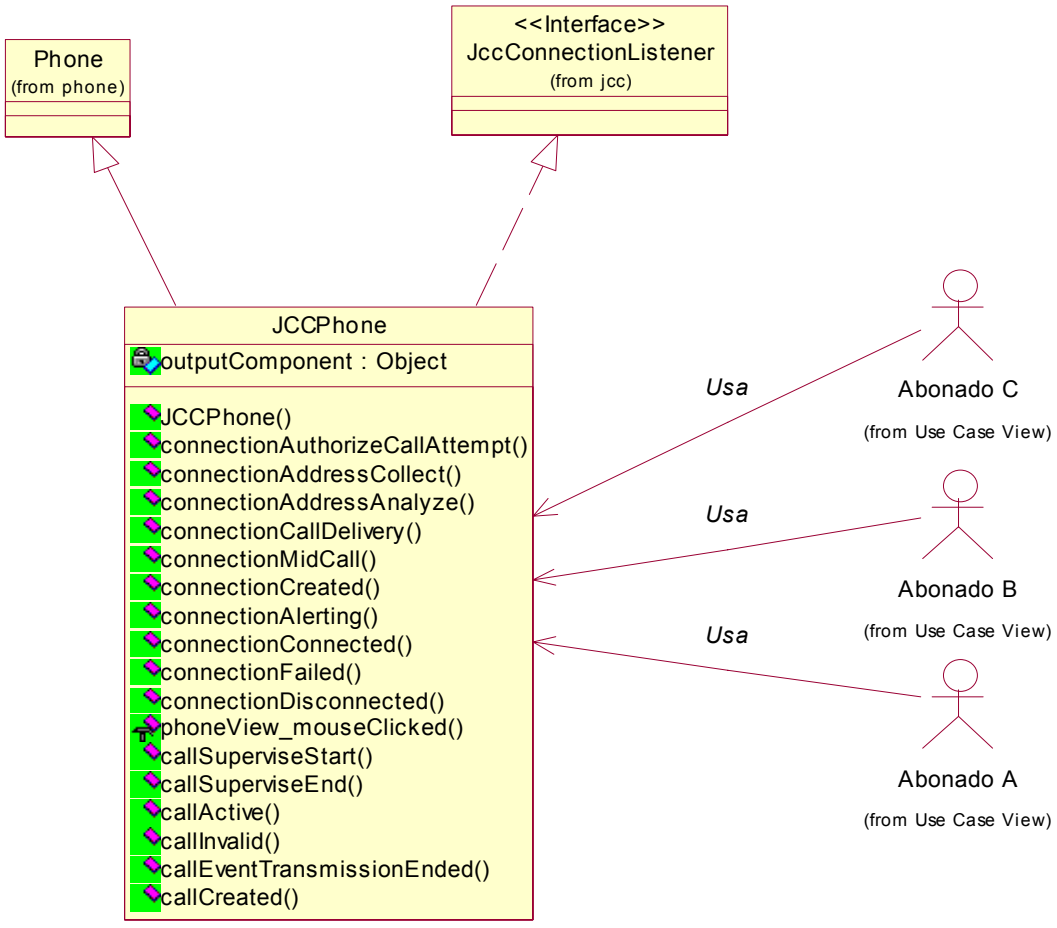

Figura 33: Diagrama de clases del paquete jccphone

JCCPhone	
Descripción:	Esta clase implementa la interfaz JccConnectionListener lo que le permite responder a los eventos de llamada del API JCC, además hereda de la clase Phone, pudiendo así simular un terminal telefónico que responde a las llamadas de API JCC y que interacciona con los abonados localmente.
Métodos:	• **JCCPhone**: constructor de la clase. • **connectionAuthorizeCallAttempt**: indica que la conexión ha

pasado al estado AUTHORIZE_CALL_ATTEMPT.

- **connectionAddressCollect**: indica que la conexión ha pasado al estado ADDRESS_COLLECT.
- **connectionAddressAnalyze**: indica que la conexión ha pasado al estado ADDRESS_ANALYZE.
- **connectionCallDelivery**: indica que la conexión ha pasado al estado CALL_DELIVERY.
- **connectionMidCall**: indica que la conexión ha sufrido un evento de mitad de llamada.
- **connectionCreated**: indica que la conexión ha sido creada.
- **connectionAlerting**: indica que la conexión ha pasado al estado ALERTING.
- **connectionConnected**: indica que la conexión ha pasado al estado CONNECTED.
- **connectionFailed**: indica que la conexión ha pasado al estado FAILED.
- **connectionDisconnected**: indica que la conexión ha pasado al estado DISCONNECTED.
- **phoneView_mouseClicked**: este método es invocado cuando se hace click sobre el componente PhoneView después de ser invocado el definido en la clase Phone, se encarga de averiguar si se presiono una tecla y activa las animaciones, imágenes o sonidos correspondientes.
- **callSuperviseStart**: indica que la supervisión de la llamada ha comenzado.
- **callSuperviseEnd**: indica que la supervisión de la llamada ha terminado.
- **callActive**: indica que la llamada ha pasado al estado ACTIVE.

	• **callInvalid**: indica que la llamada ha pasado al estado INVALID. • **callEventTransmissionEnded**: indica que la aplicación no recibirá más eventos JccCallEvent de la llamada. • **callCreated**: indica que la llamada ha pasado al estado IDLE.
Atributos:	• **outputComponent**: atributo que apunta al objeto al cual se envían los mensajes de error y de alerta, usualmente es el objeto que lo creo o sea el PhoneLauncher.

B.2.1.3 Clases del Paquete phonelauncher.

El diagrama de clases del paquete phonelauncher se puede observar en la Figura 34.

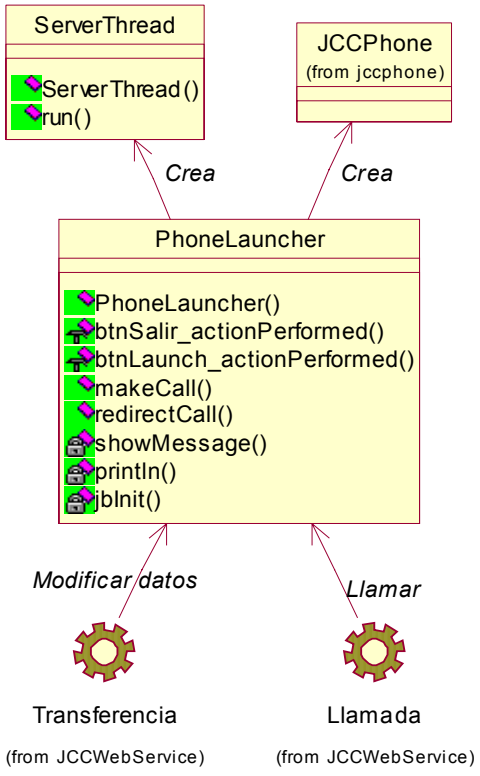

Figura 34: Diagrama de clases del phonelauncher

En este paquete se encuentran dos clases que son las que corresponden al servidor y a la aplicación JCC, la clase ServerThread es un hilo que escucha las peticiones de la aplicaión web, mientras que la clase PhoneLauncher es la clase que crea los diferentes terminales telefónicos necesarios para la simulación y controla las llamadas entre ellos. A continuación se describen estas dos clases.

ServerThread	
Descripción:	Esta clase sirve como hilo servidor, es la que recibe las peticiones desde el servidor web y ejecuta las acciones pertinentes, hereda de la clase java.lang.Thread.
Métodos:	• **ServerThread:** constructor de la clase. • **run:** método que implementa el hilo que escucha las peticiones.
Atributos:	Ninguno.

PhoneLauncher	
Descripción:	Esta clase funciona como fabrica de terminales telefónicos, los crea y los registra en el API para que puedan realizar llamadas entre ellos, también se encarga de iniciar a la clase ServerThead que es la que recibe la información del servidor web correspondiente a los servicios antes descritos, adicionalmente despliega mensajes de error o similares que los teléfonos generen.
Métodos:	• **PhoneLauncher:** constructor de la clase. • **btnSalir_actionPerformed:** método que se invoca al presionar el botón Salir, se encarga de terminar la aplicación. • **btnLaunch_actionPerformed:** método que se invoca al presionar el botón Launch, crea un nuevo terminal y le asigna un el número telefónico especificado por el usuario. • **makeCall:** inicia una llamada entre los abonados solicitados.

	• **redirectCall:** activa la redirección de llamadas para el terminal especificado. • **showMessage:** método que muestra una ventana adicional mostrando un mensaje, se usa para mensajes de error. • **println:** método que imprime un mensaje dentro del componente, para ser usado como salida de mensajes. • **jbInit:** método que inicializa los componentes gráficos que conforman esta clase, es invocado por el constructor.
Atributos:	Ninguno.

B.2.1.4 Clases del Paquete JCCWebService.

El paquete JCCWebService constituye una aplicación web independiente que se comunica con la clase ServerThread para realizar las acciones correspondientes, en este paquete se encuentran las clases que permiten hacer uso de los servicios antes descritos, el diagrama de clases se puede observar en la Figura 35.

IU_Error	
Descripción:	Interfaz que despliega suscriptor un mensaje de error.
Métodos:	• **regresar:** envía al usuario a la página anterior.
Atributos:	Ninguno.

Error	
Descripción:	Esta clase genera la pagina de error.
Métodos:	• **showError**: construye la página IU_Error.
Atributos:	• **mensaje**: mensaje a desplegar.

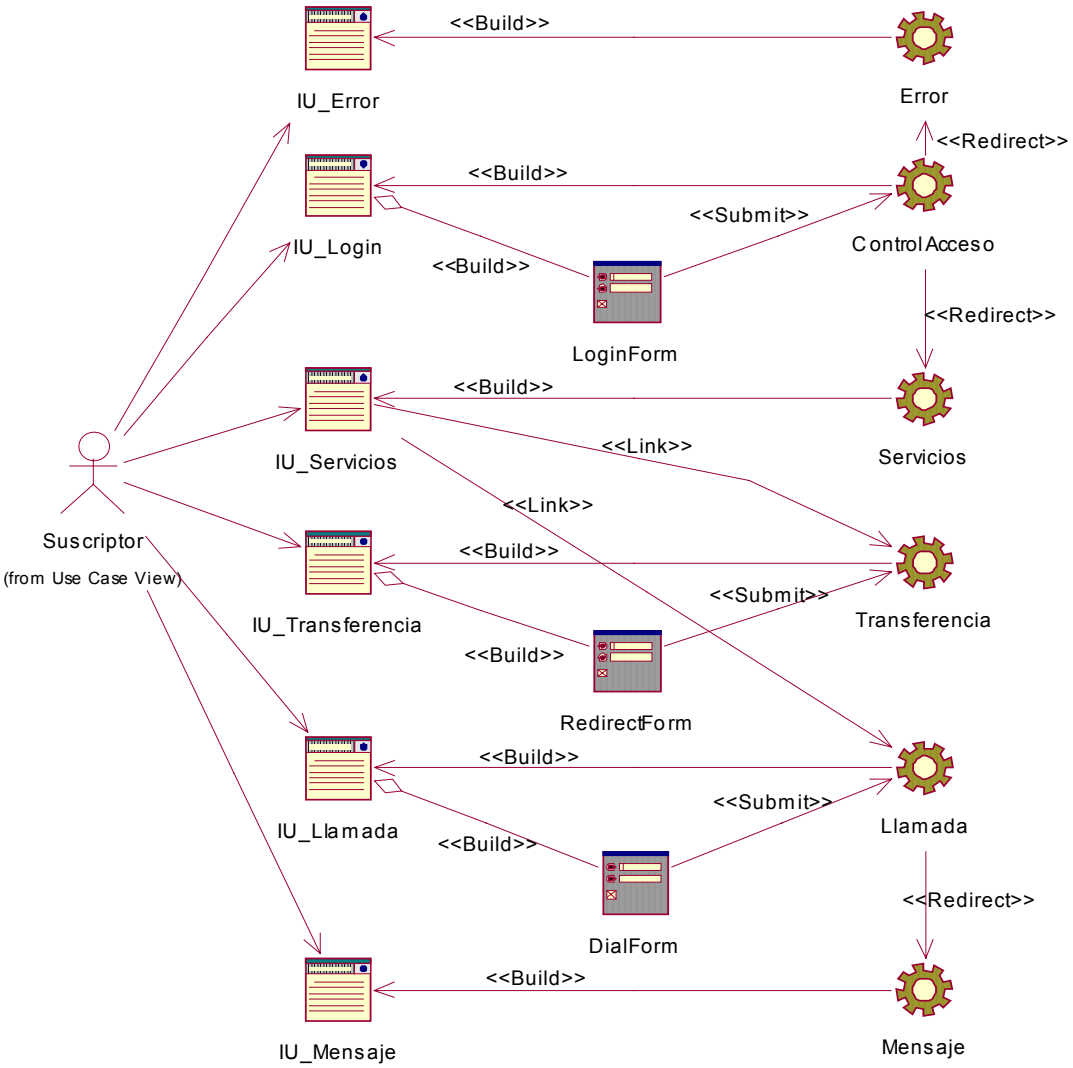

Figura 35: Diagrama de clases del paquete JCCWebService

IU_Login	
Descripción:	Interfaz que muestra al usuario el formulario que le pregunta sus datos para verificar el acceso, se despliega por defecto al entrar al sitio web.
Métodos:	Ninguno.

Atributos:	**LoginForm**: es el formulario que pregunta los datos necesarios para el control de acceso..

LoginForm	
Descripción:	Esta formulario almacena los datos necesarios para realizar el control de acceso al sistema como son número telefónico y contraseña.
Métodos:	• **clickAceptar:** envía los datos del usuario a la clase ControlAcceso. • **clickCancelar:** limpia los datos escritos.
Atributos:	• **telefono:** número telefónico del suscriptor. • **password:** contiene la contraseña del suscriptor.

ControlAcceso	
Descripción:	Esta clase autentica los datos del suscriptor, si los datos proporcionados son correctos, inicia una sesión y redirecciona a la clase Servicios, si son inválidos redirecciona a la clase Error con el mensajes de error adecuado.
Métodos:	• **AutenticarAcceso**: autentica el acceso del suscriptor, si los datos son correctos inicia una sesión y redirecciona a la clase Servicios, si son inválidos redirecciona a la clase Error con el mensajes de error adecuado. • **showLogin**: método muestra la interfaz de transferencia de llamadas IU_Login.
Atributos:	Ninguno.

IU_Servicios	
Descripción:	Interfaz que muestra al suscriptor los servicios que tiene activos.
Métodos:	• **clickTransferencia:** enlaza a la clase Transferencia. • **clicklLlamada:** enlaza a la clase Llamada.

Atributos:	Ninguno.

Servicios	
Descripción:	Esta clase muestra la lista de servicios que el suscriptor tiene activos.
Métodos:	• **showServicios:** consulta los servicios activos de un suscriptor y los despliega en la página IU_Servicios.
Atributos:	Ninguno.

IU_Transferencia	
Descripción:	Interfaz que muestra el formulario para activar el servicio de transferencia de llamadas sígueme..
Métodos:	• **clickTransferencia:** enlaza a la clase Transferencia. • **clicklLlamada:** enlaza a la clase Llamada.
Atributos:	• **RedirectForm**: formulario con los datos de la transferencia de llamadas.

RedirectForm	
Descripción:	Formulario que contiene los datos para la transferencia de llamadas.
Métodos:	• **cliclAceptar:** envía los datos del formulario a la clase Transferencia para su procesamiento.
Atributos:	• **redierct**: número telefónico al cual se deben redireccionar las llamadas del suscriptor.

Transferencia	
Descripción:	Clase que recibe los datos para la transferencia de llamadas, los verifica y los envía al servidor para configurar el servicio.
Métodos:	• **Transferencia:** recibe los datos del servicio, los verifica y los envía al servidor, en caso de haber algún error redirecciona a la clase

	Error y en caso de ser satisfactoria redirecciona ala clase Mensaje. • **showTransferencia:** método muestra la interfaz de transferencia de llamadas IU_Transferencia.
Atributos:	Ninguno.

IU_Llamada	
Descripción:	Interfaz que muestra el formulario para activar el servicio de click-to-dial.
Métodos:	• **clickTransferencia:** enlaza a la clase Transferencia. • **clicklLlamada:** enlaza a la clase Llamada.
Atributos:	• **DialForm**: formulario con los datos para la el servicio click-to-dial.

DialForm	
Descripción:	Formulario que contiene los datos para la el servicio click-to-dial.
Métodos:	• **clicklLlamar:** envía los datos del formulario a la clase Llamada para su procesamiento.
Atributos:	• **destino**: número telefónico al cual se desea realizar la llamada.

Llamada	
Descripción:	Clase que recibe los datos para el servicio click-to-dial, los verifica y los envía al servidor para iniciar la llamada.
Métodos:	• **Llamada:** recibe los datos del servicio, los verifica y los envía al servidor, en caso de haber algún error redirecciona a la clase Error y en caso de ser satisfactoria redirecciona ala clase Mensaje. • **showLlamada:** método muestra la interfaz para realizar llamadas IU_Llamada.
Atributos:	Ninguno.

IU_Mensaje	
Descripción:	Interfaz que muestra un mensaje al suscriptor.
Métodos:	• **regresar:** regresa a la página anterior.
Atributos:	Ninguno.

Mensaje	
Descripción:	Esta clase construye la pagina que muestra un mensaje al suscriptor.
Métodos:	• **showMensaje:** constructor la página IU_Mensaje.
Atributos:	• **mensaje:** mensaje a desplegar en la página IU_Mensaje.

B.2.2 Diagramas de secuencias de mensajes

A continuación se presentan los diagramas de secuencia de mensajes para los casos de uso anteriormente identificados.

B.2.2.1 Caso de Uso: Autenticar Usuario.

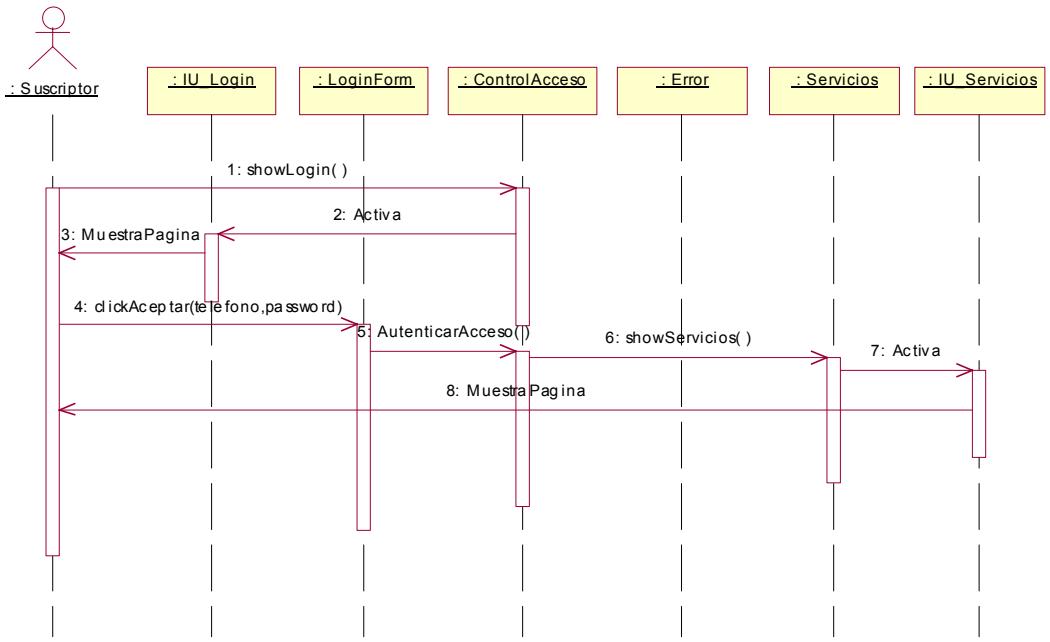

Figura 36: Diagrama de secuencia de mensajes del caso uso AutenticarUsuario, parte 1

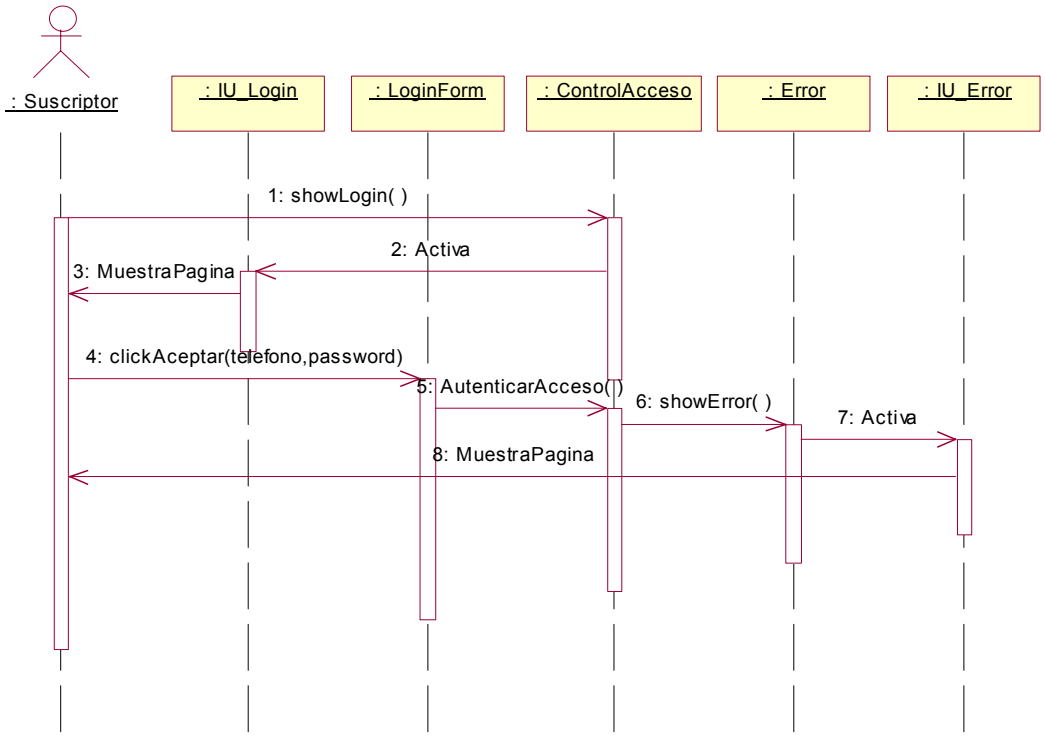

Figura 37: Diagrama de secuencia de mensajes del caso uso AutenticarUsuario, parte 2

B.2.2.2 Caso de Uso: Modificar Datos Sígueme.

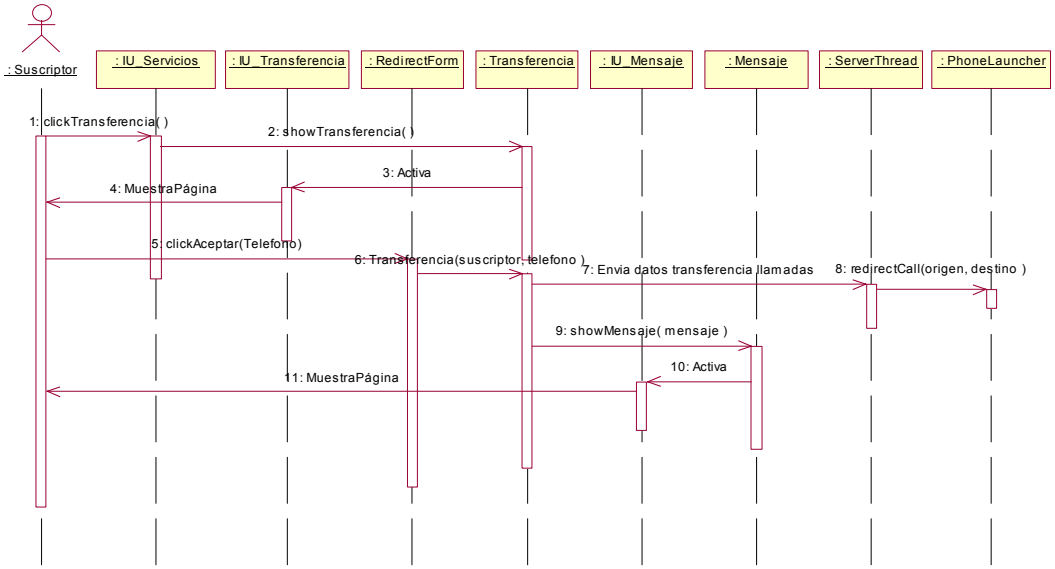

Figura 38: Diagrama de secuencia de mensajes del caso de uso Modificar Datos Sigueme, parte 1

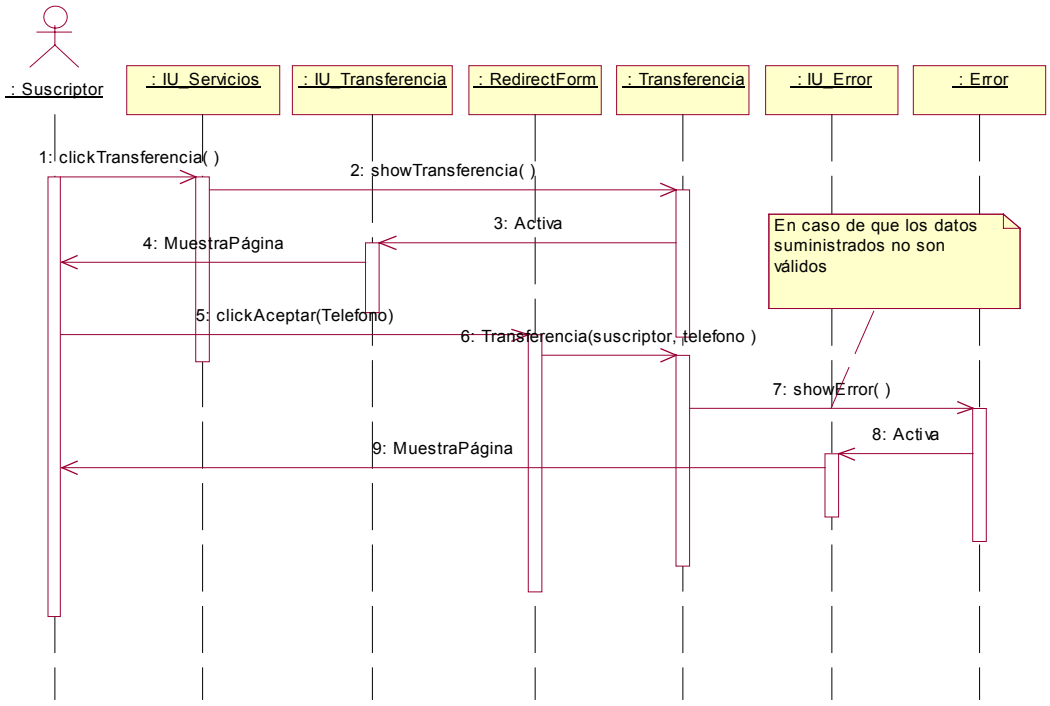

Figura 39: Diagrama de secuencia de mensajes del caso de uso Modificar Datos Sigueme, parte 2

B.2.2.3 Caso de Uso: Pedir Llamada.

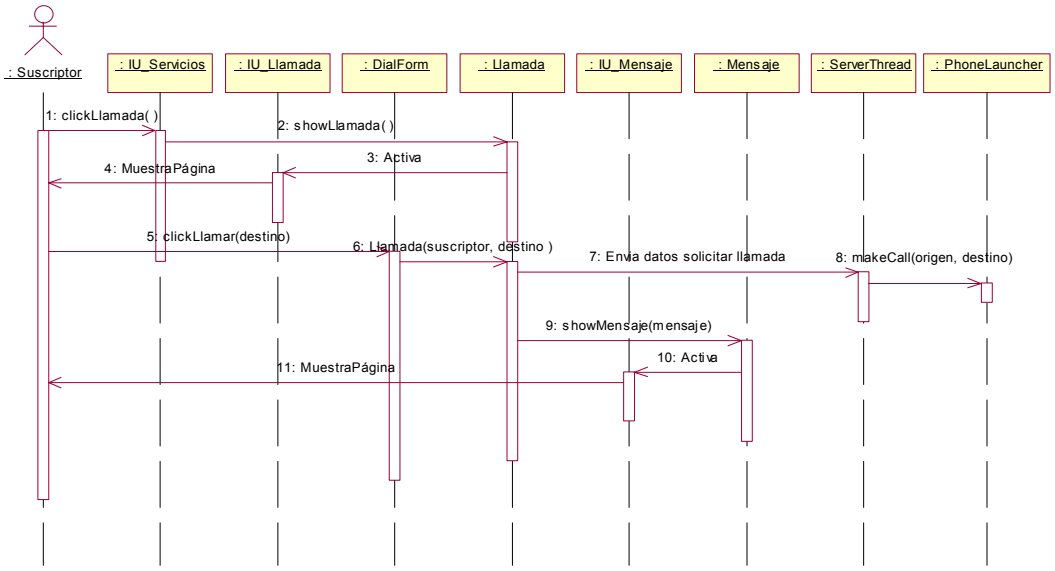

Figura 40: Diagrama de secuencia de mensajes del caso de uso Pedir Llamada, parte 1

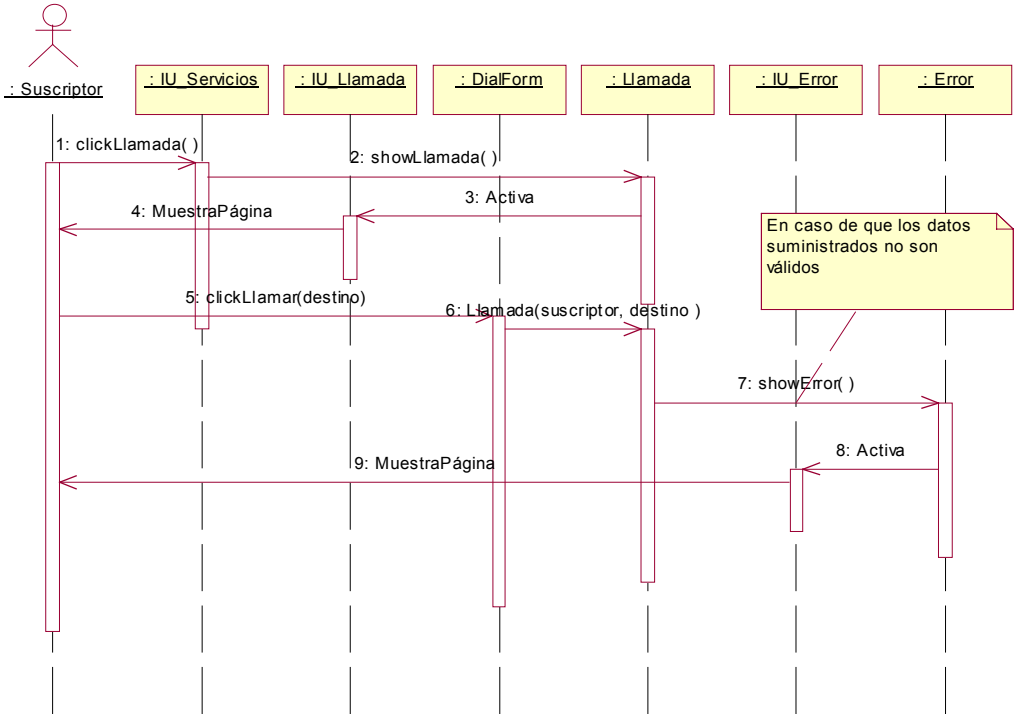

Figura 41: Diagrama de secuencia de mensajes del caso de uso Pedir Llamada, parte 2

B.2.2.4 Caso de Uso: Realizar llamada.

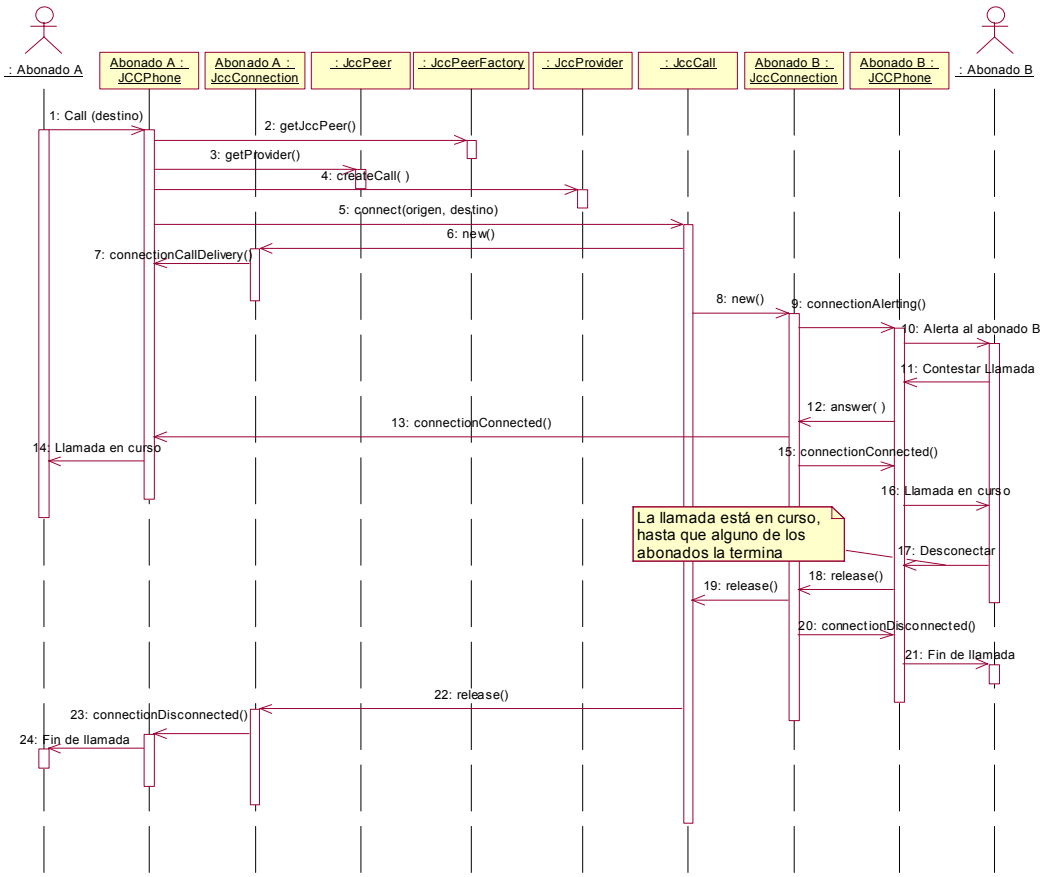

Figura 42: Diagrama de secuencia de mensajes del caso de uso Realizar Llamada

B.2.3 Diagramas de estados

Se identificaron los estados para la clase JCCPhone, los cuales están basados en los estados de la clase JccConnection del API JCC, la Figura 43 muestra el diagrama de estados de la clase JCCPhone.

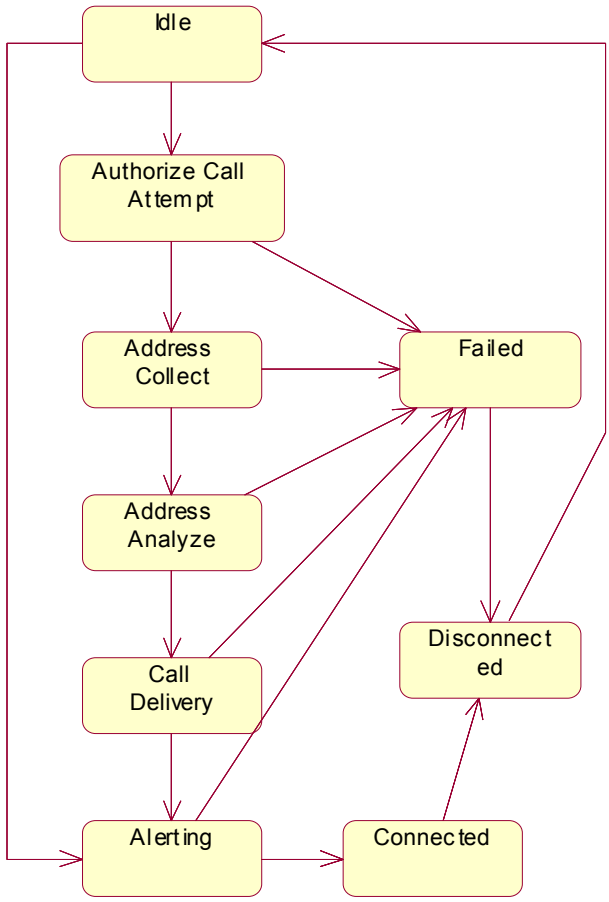

Figura 43: Diagrama de estados de la clase JCCPhone

B.3 DISEÑO DEL SISTEMA

El objetivo de esta sección es mostrar la estructura del sistema y su realización a partir de las clases que ya han sido identificadas y refinadas.

B.3.1 Diagrama de componentes

Para la organización del sistema se dividieron las clases en paquetes de acuerdo a la aplicación que los usa.

B.3.1.1 Componentes de la aplicación JCCDemoApp

La aplicación JCCDemoApp se divide en varios 7 componentes como se aprecia en la Figura 44:

- JCCDemoApp: Contiene las clases correspondientes al paquete de diseño JCCDemoApp.
- PhoneLauncher: Contiene las clases correspondientes al paquete de diseño PhoneLauncher.
- JCCPhone: Contiene las clases correspondientes al paquete de diseño JCCPhone.
- Phone: Contiene las clases correspondientes al paquete de diseño Phone.
- JCC-RI: Contiene la implementación de referencia del API JCC de JAIN.
- JCC-API: Contiene la especificación del API JCC de JAIN.
- DB_Servicios: La base de datos de los servicios.

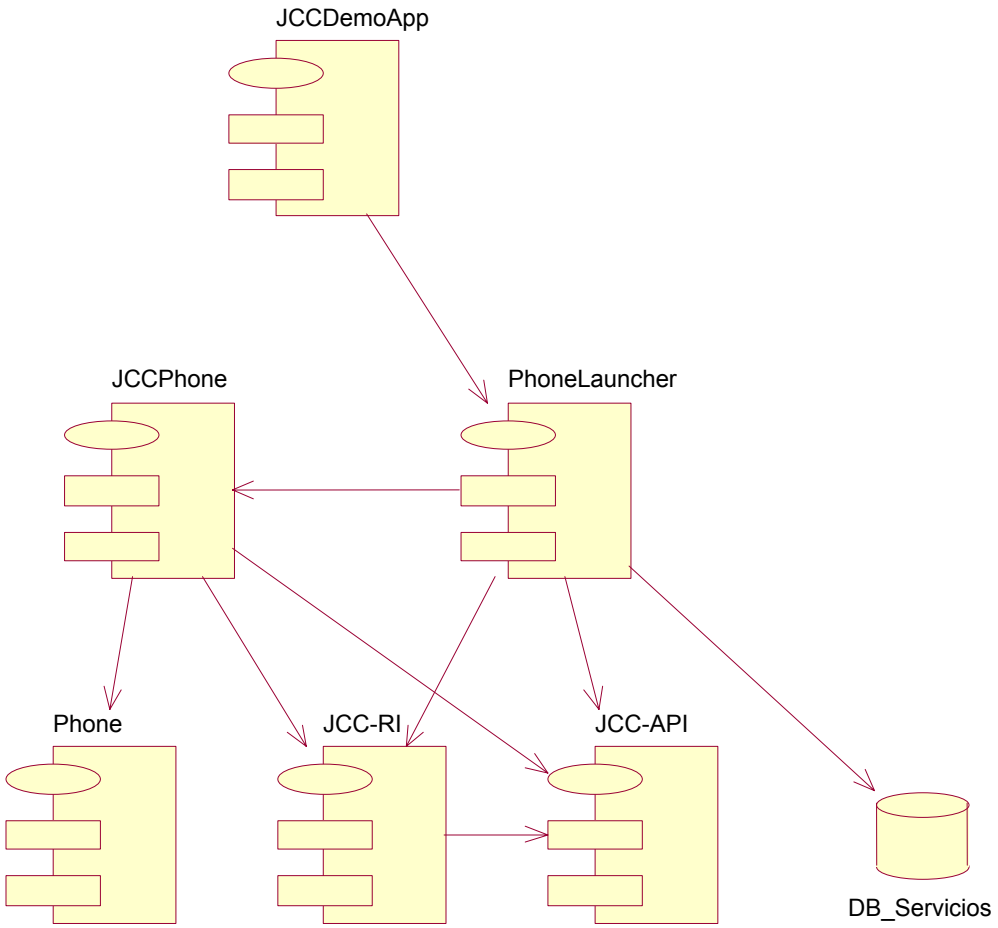

Figura 44: Diagrama de componentes de la aplicación JCCDemoApp

B.3.1.2 Componentes de la aplicación JCCWebApp

Los componentes de esta aplicación se encapsulan en un WebService que utiliza la base de datos de suscriptores del servicio para obtener los datos necesarios, como se muestra en la Figura 45.

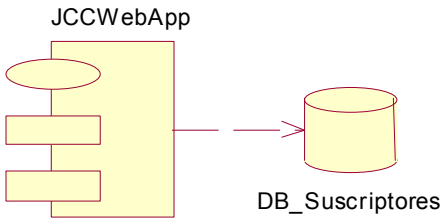

Figura 45: Diagrama de conponentes de la aplicación JCCWebApp

B.3.2 Diagrama de implantación

El diagrama de implantación muestra como encajan los diferentes elementos software en la arquitectura física.

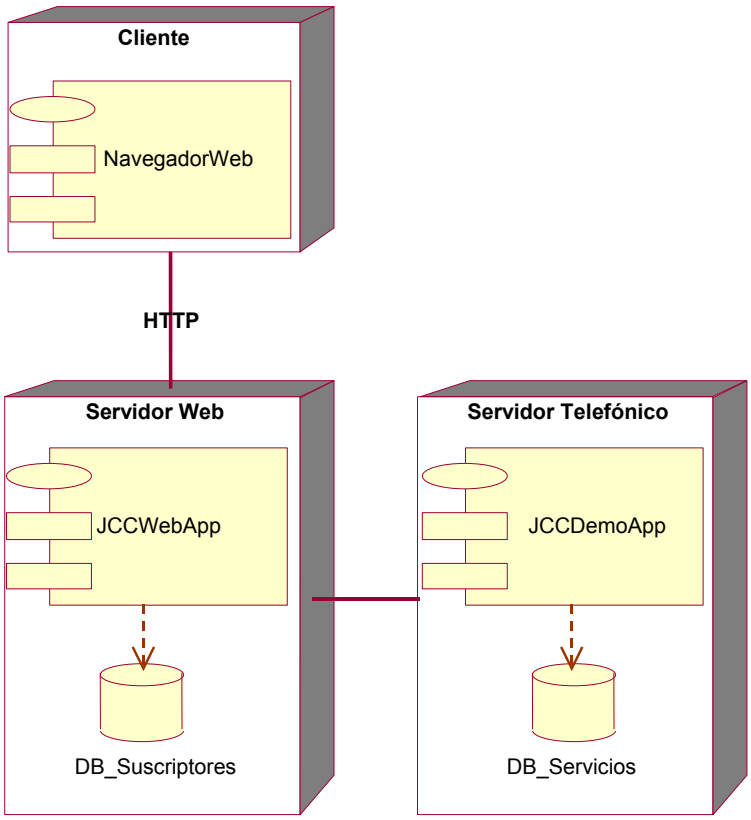

Figura 46: Diagrama de implantación

C.1 Manual de instalación del software

El hardware y software requerido depende del equipo que ese esté instalando, para más información ver la Tabla 2.

Tabla 2: Software necesario para cada equipo

	Servidor Telefónico	Servidor Web	Cliente Web
Memoria	• 256Mb	• 128Mb	• 64Mb
Sistema Operativo	• Recomendado: Windows 2000	• Recomendado: Windows 2000	• Cualquiera
Software Requerido	• Kit de Desarrollo de Java 1.4 o superior • Herramienta de compilación Ant • Motor de Base de Datos MySQL • Aplicación JCCDemoApp	• Kit de Desarrollo de Java 1.4 o superior • Servidor de JSPs y Servlets Tomcat • Aplicación JCCWebService	• Explorador Web

C.1.1 Instalación del Entorno de Desarrollo de Java (JDK)

La ultima versión del entorno de desarrollo de java se puede descargar de la siguiente dirección:

- http://java.sun.com/j2se/downloads.html

NOTA: La aplicación desarrollada utiliza clases disponibles a partir de la versión 1.4 de Java, por lo tanto se debe instalar una versión igual o superior a la 1.4, fue probada con la versión 1.4.1_02 que se encuentra en el CD de documentación de este trabajo.

1. Al hacer doble click sobre el archivo de instalación, se abre una la ventana de la Figura 47 que nos da la bienvenida al la instalación del JDK, aquí se debe hacer click en el botón **Next**.

2. Aparece la ventana de la Figura 48, en ella se muestra el contrato de licencia del software, para aceptarlo se debe hacer click en el botón **Yes**.

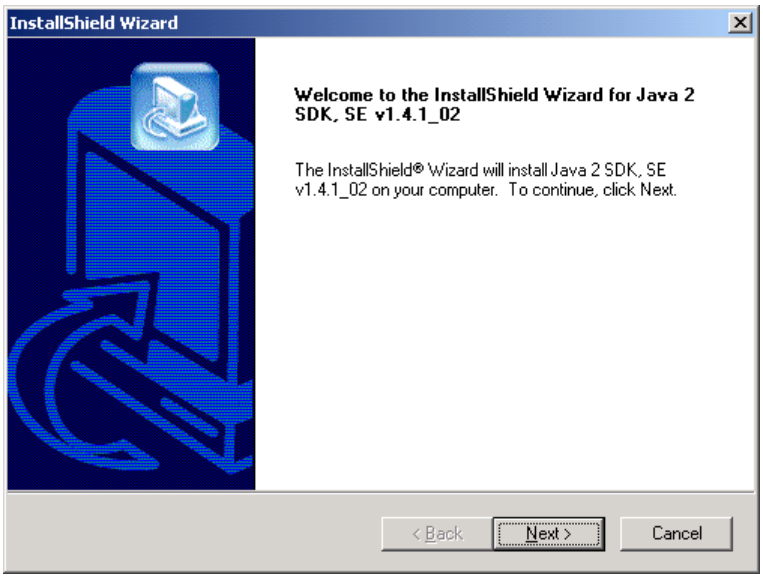

Figura 47: Inicio de la intalación del JDK

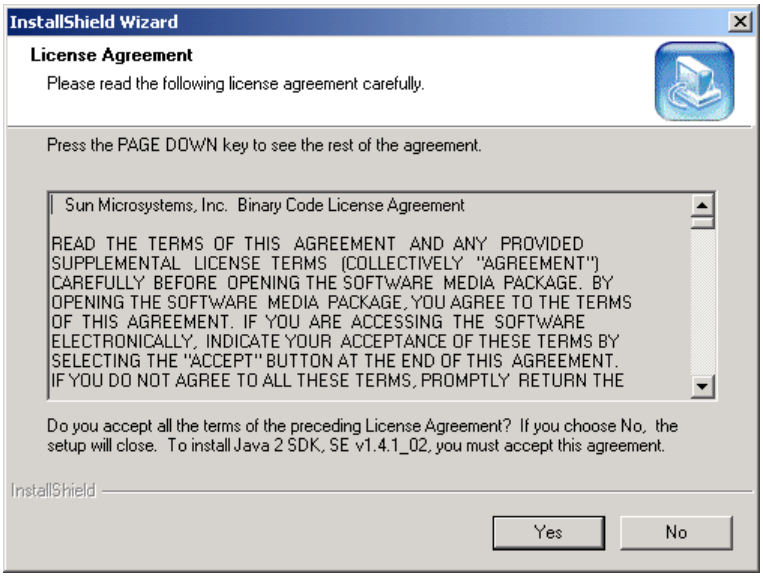

Figura 48: Contrato de Licencia

3. Al continuar, se despliega la ventana de la Figura 49, en la cual se debe escoger el directorio donde se desea instalar el JDK, en este caso se escogió el directorio D:\j2sdk1.4.1, tenga en cuenta este directorio ya que lo necesitará en el futuro. Una vez se ha elegido el directorio se hace click en el botón **Next**.

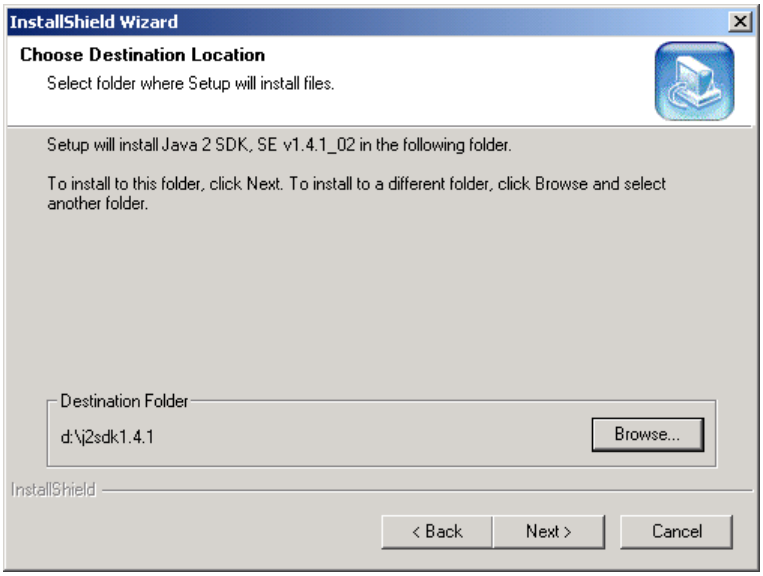

Figura 49: Directorio de instalación

4. En la siguiente ventana, Figura 50, se escogen las opciones de instalación, para este caso se pueden dejar las opciones por defecto y dar click en el botón **Next**.
5. Al continuar aparece una ventana para configurar esta instalación de Java como el entorno de ejecución por defecto para los navegadores instalados, Figura 51, se pueden dejar las opciones por defecto y hacer click en **Next**.
6. La siguiente ventana da por terminado el proceso de instalación, Figura 52, y se hace click en el botón **Finish** para salir del proceso de instalación.

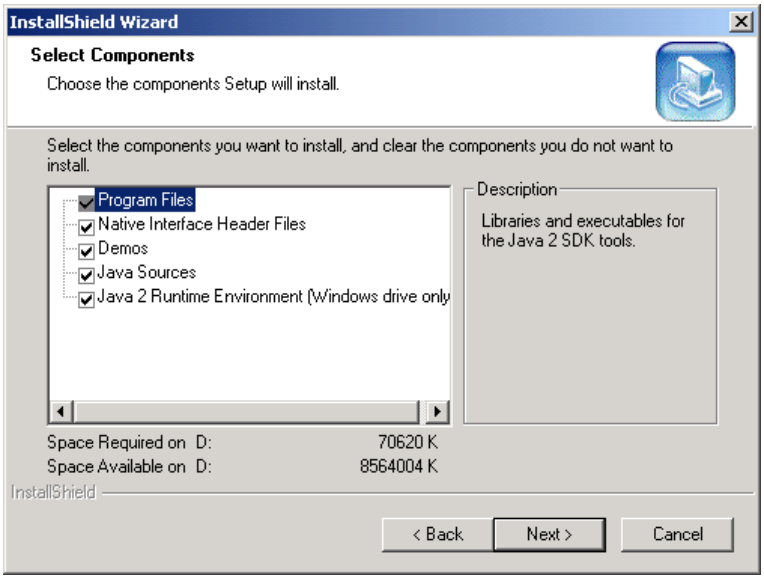

Figura 50: Opciones de instalación

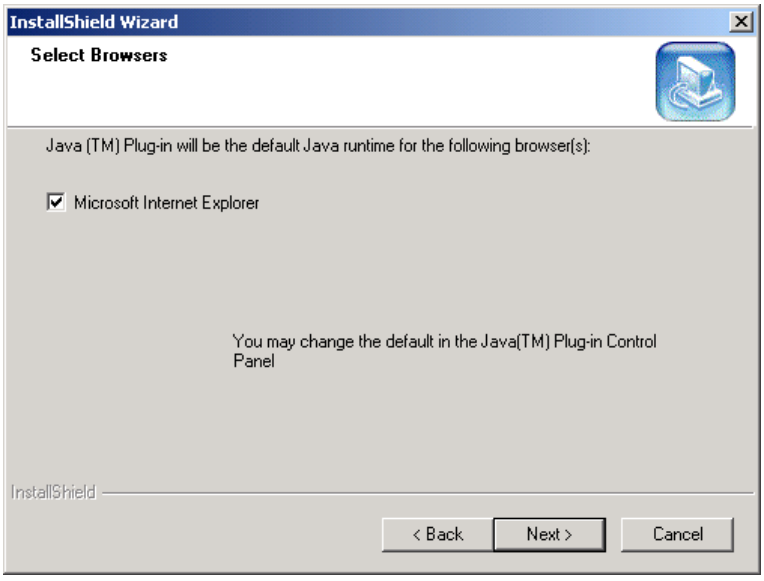

Figura 51: Configuración del navegador

7. Una vez terminada la instalación, es necesario configurar las variables de entorno del sistema para que el entorno de desarrollo de java funcione correctamente, para esto se deba hacer click-derecho sobre el icono de **Mi PC,** Figura 53, y hacer click sobre la

opción **Propiedades**, para abrir la ventana de **Propiedades del sistema** que se muestra la Figura 54.

8. En la ventana de **Propiedades del Sistema**, Figura 54, se debe escoger la pestaña llamada **Avanzado**, en esta ventana hace click en el botón **Variables de entorno...** para abrir la ventana de configuración de las variables del sistema.
9. En ventana abierta en el paso anterior, Figura 55, dentro de las variables del sistema se hace click en el botón **Nueva...**, lo cual abre la ventana de Nueva variable del sistema que se muestra en la Figura 56. En esta ventana se crea la variable del sistema **JAVA_HOME**, que debe tener como valor el directorio donde se instaló el JDK (el que se escogió en el paso 3), una vez llenada esta información se hace click enel botón **Aceptar**.

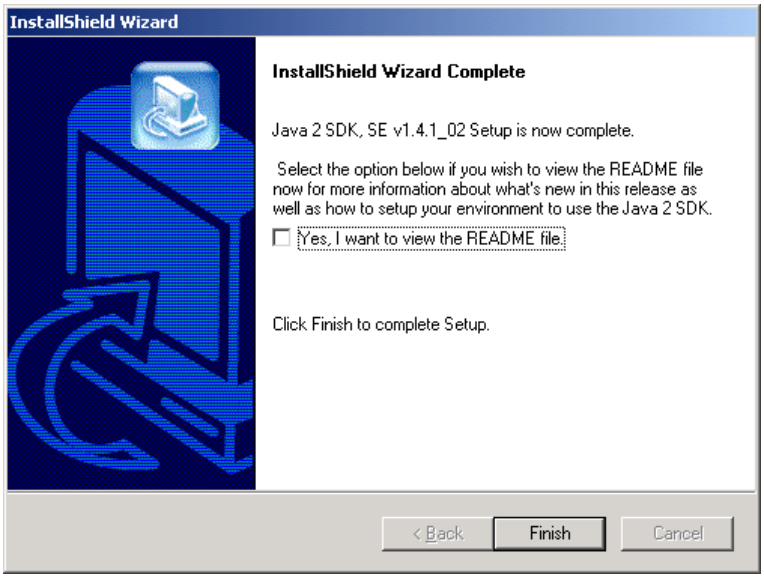

Figura 52: Finalización del la instalación

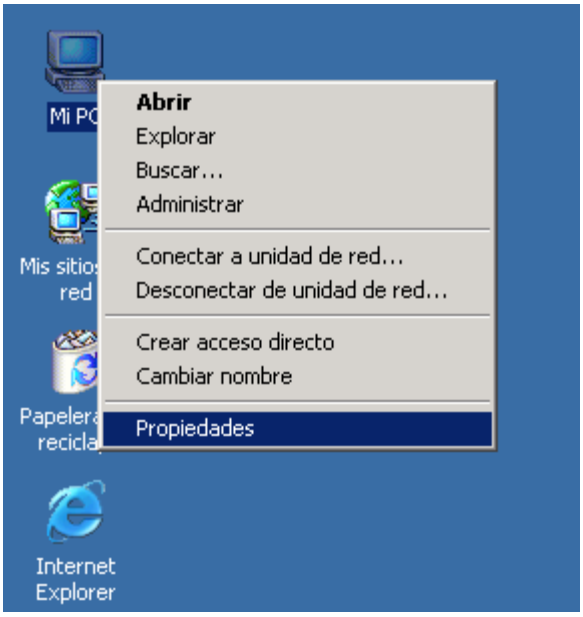

Figura 53: Edición de las variables de entorno

10. Ahora es necesario editar la variable del sistema **Path**, para lo cual se debe buscar entre la lista de variables del sistema y seleccionarla, luego se hace click sobre el botón **Editar..** de la ventana **Variables de entorno** mostrada en la Figura 55, esto abre la ventana **Modificar variable del sistema**, Figura 57. Aquí se debe agregar la siguiente cadena al final del valor de la variable: **;%JAVA_HOME%\bin**, para que los programas del JDK, como el compilador se puedan ejecutar desde cualquier ventana de terminal y desde cualquier directorio, una vez echo esto se hace click en el botón **Aceptar** y se cierran las demás ventanas haciendo click también en **Aceptar**.

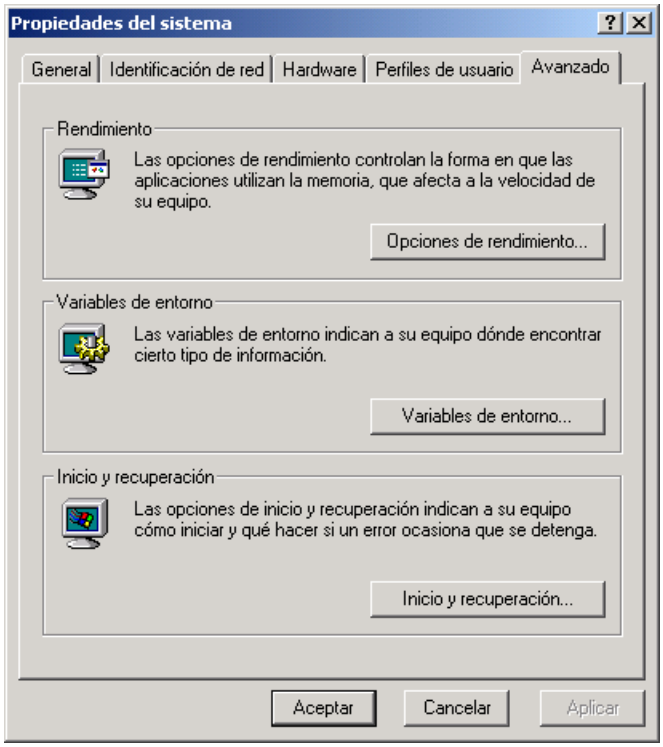

Figura 54: Propiedades del sistema

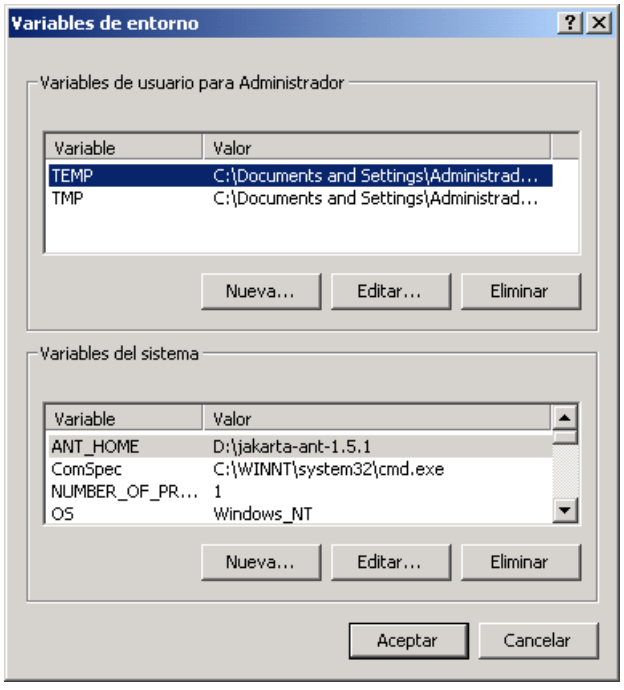

Figura 55: Variables de entorno

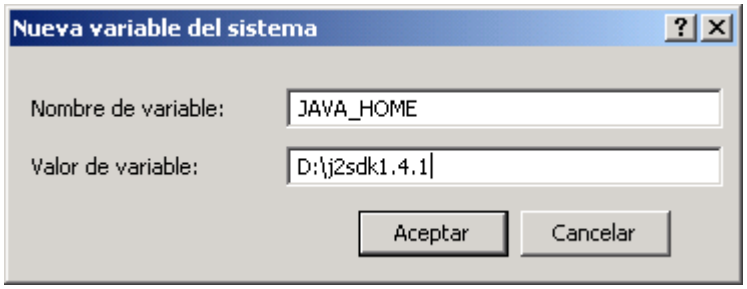

Figura 56: Nueva variable del sistema

11. Para probar la instalación del JDK, se abre una ventana de terminal, haciendo click en el menú **Inicio->Programas->Accesorios->Símbolo del sistema**, en esta ventana ejecutamos el comando: **java –versión**, y debemos obtener una salida como la que muestra la Figura 58, con esto se concluye la instalación y configuración del JDK.

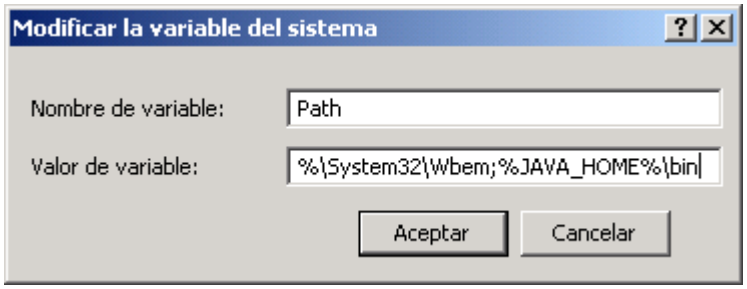

Figura 57: Modificar la variable del sistema

C.1.2 Instalación de la herramienta de compilación ant

Ant es un a herramienta usada para compilar, empaquetar e instalar aplicaciones, basada en Java y XML, la ultima versión se puede encontrar en:

- http://ant.apache.org/bindownload.cgi

NOTA: La aplicación desarrollada utiliza la herramienta para compilarse y empaquetarse, ha sido probada con la versión 1.5.2 que se encuentra en el CD de documentación de este trabajo.

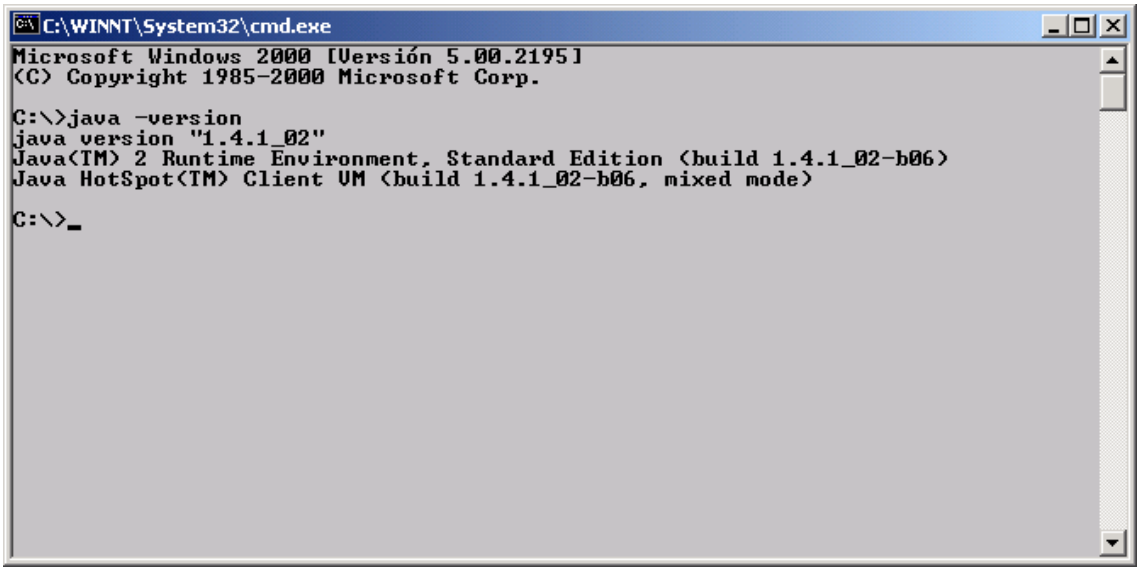

Figura 58: Prueba de la instalación del JDK

1. La herramienta ant viene empaquetada en un archivo comprimido en formato zip, para instalarlo es necesario descomprimirlo a un directorio usando una herramienta de descompresión como winzip, tenga en cuenta el directorio ya que lo necesitará en los pasos siguientes, una vez descomprimido se sigue con el próximo paso.
2. Es necesario modificar las variables de entorno, para esto se siguen los pasos 7 y 8 del proceso de instalación del entorno de desarrollo de java, que ya se describieron en la sección anterior.
3. En ventana abierta al seguir los pasos anteriores, Figura 55, dentro de las variables del sistema se hace click en el botón **Nueva...**, lo cual abre la ventana de Nueva variable del sistema que se muestra en la Figura 59. En esta ventana se crea la variable del sistema **ANT_HOME**, que debe tener como valor el directorio donde se descomprimió el software (paso 1), una vez llenada esta información se hace click en el botón **Aceptar**.

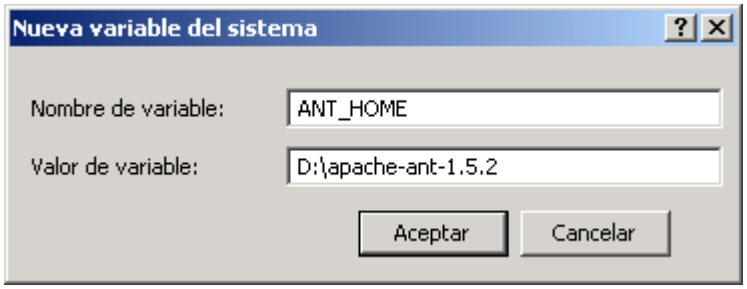

Figura 59: Nueva variable del sistema

4. Ahora es necesario editar la variable del sistema **Path**, para lo cual se debe buscar entre la lista de variables del sistema y seleccionarla, luego se hace click sobre el botón **Editar..** de la ventana **Variables de entorno** mostrada en la Figura 55, para abrir la ventana **Modificar variable del sistema**, Figura 60. Aquí se debe agregar la siguiente cadena al final del valor de la variable: **;%ANT_HOME%\bin**, para que la herramienta ant se pueda ejecutar desde cualquier ventana de terminal y desde cualquier directorio, una vez echo esto se hace click en el botón **Aceptar** y se cierran las demás ventanas haciendo click también en **Aceptar**.

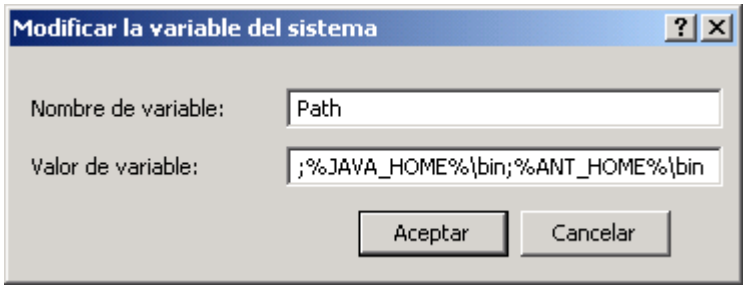

Figura 60: Modificar variable del sistema

5. Para probar la instalación de la herramienta ant, se abre una ventana de terminal, haciendo click en el menú **Inicio->Programas->Accesorios->Símbolo del sistema**, en esta ventana ejecutamos el comando: **ant –versión**, y debemos obtener una salida como la que muestra la Figura 61, con esto se concluye la instalación y configuración de ant.

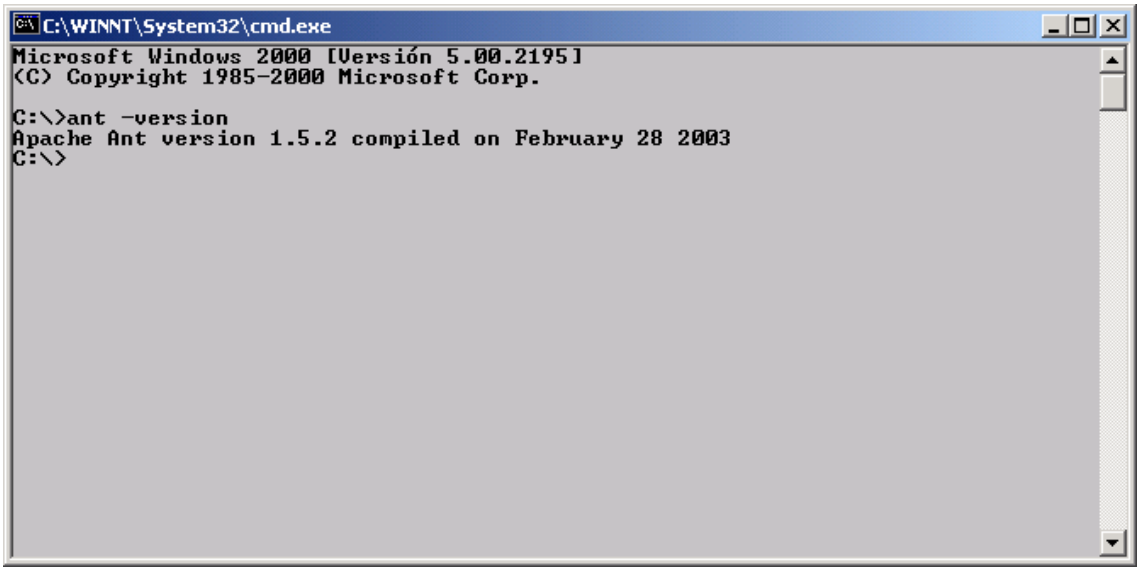

Figura 61: Prueba de instalación de la herramienta ant

C.1.3 Instalación del servidor de JSPs y Servlets

La ultima versión del servidor de JSPs y Servlets Tomcat se puede descargar de la siguiente dirección:

- http://jakarta.apache.org/site/binindex.cgi

NOTA: La aplicación desarrollada ha sido probada con la versión 4.1.24 que se encuentra en el CD de documentación de este trabajo.

1. El proceso de instalación del servidor Tomcat se inicia haciendo doble click en el archivo de instalación, el primer paso es la detección del entorno de desarrollo de java, el cual es necesario que esté instalado, si esto es correcto, aparece una ventana que nos indica que se ha encontrado el JDK, Figura 62, si esto no sucede se debe revisar que se halla instalado correctamente el JDK.
2. Al dar click en **Aceptar**, se procede a la ventana del contrato de licencia del software, Figura 63, en esta ventana se debe hacer click en **I Agree**.

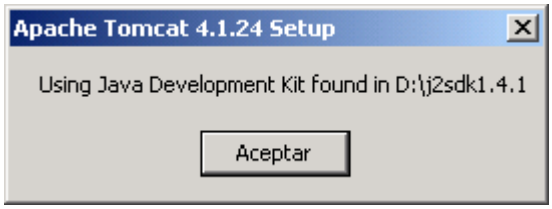

Figura 62: Detección del entorno de ejecución de java

3. Ahora se procede a configurar las opciones de la instalación, en la ventana de la Figura 64, en esta solo son necesarias las 4 primeras opciones: Tomcat, NT Service, JSP Development shell extensions y Tomcat Start menu group y se hace click en **Next**.
4. Se continua con la escogencia del directorio de instalación como se muestra en la Figura 65, una vez se ha llenado el dato correspondiente se puede dar click en el botón **Next**.

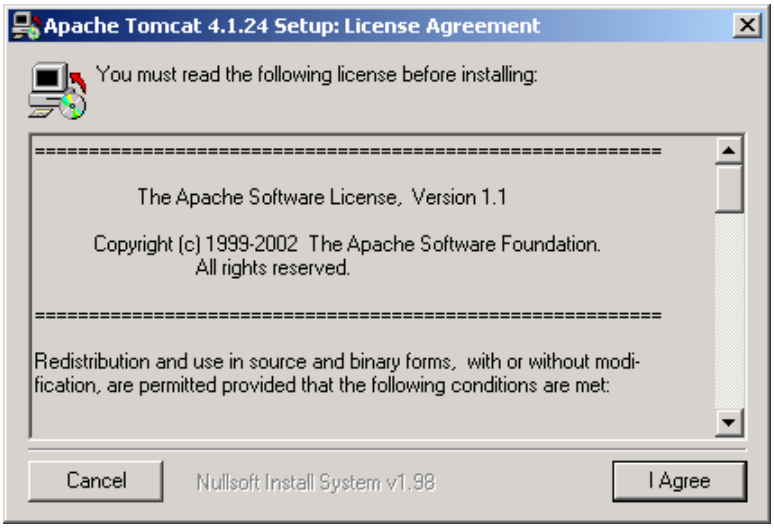

Figura 63: Contrato de licencia

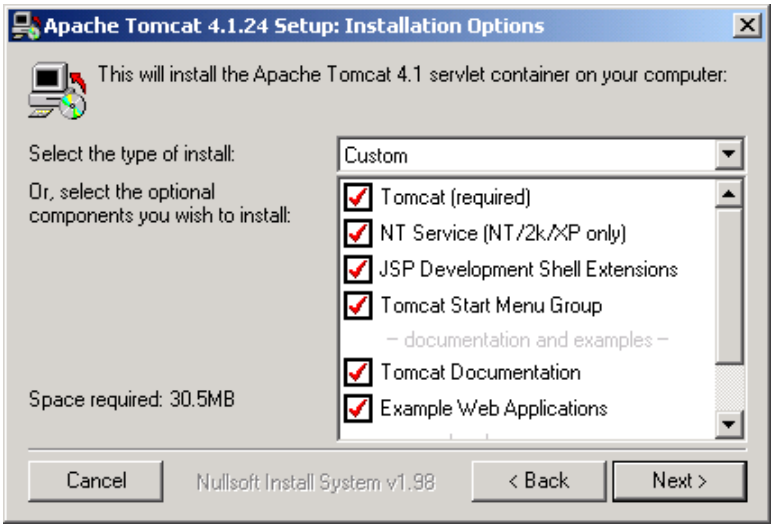

Figura 64: Opciones de instalación

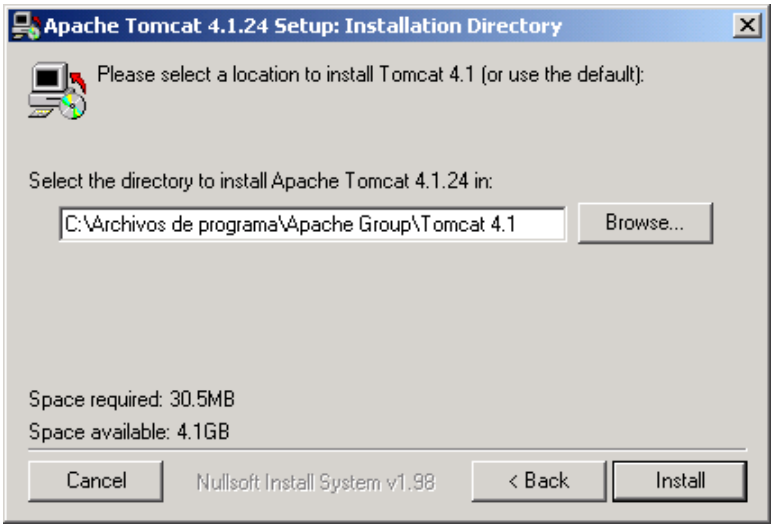

Figura 65: Directorio de instalación

5. Al continuar aparece la ventana de las propiedades del servidor, Figura 66, en la cual se escoge el puerto en el cual se escuchan las peticiones, el nombre del usuario administrador y su contraseña, llene los datos y téngalos en cuenta ya que se necesitarán en el futuro, una vez listo esto se hace click en **Next**.

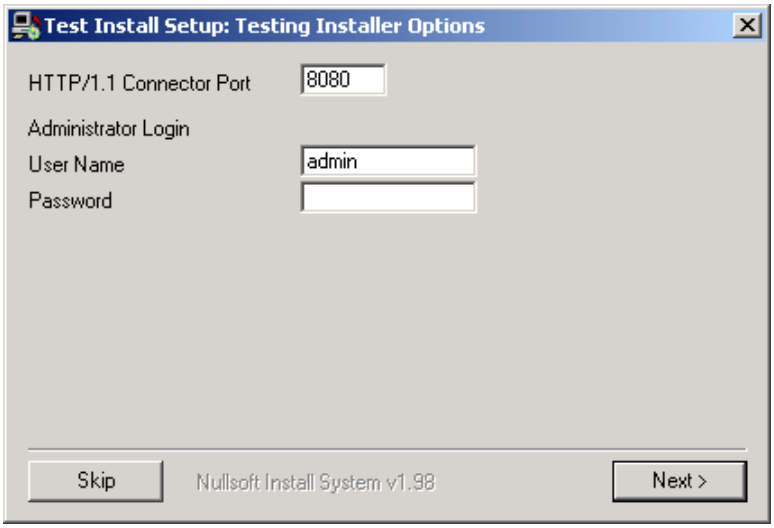

Figura 66: Opciones de intalación

6. El siguiente paso es leal progreso de la instalación, una vez terminada aparece la ventana de instalación completa, Figura 67, de la cual se sale haciendo click en el botón **Close**.

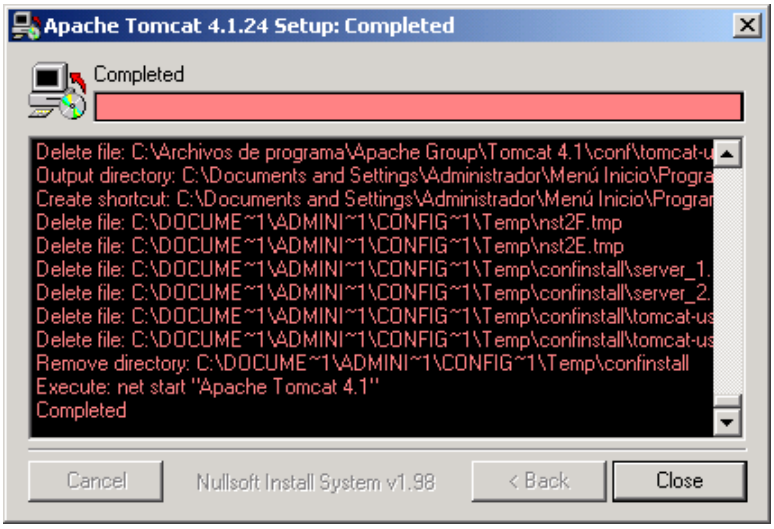

Figura 67: Instalación completa

7. Para probar la instalación de servidor Tomcat, se abre una ventana de navegación y se coloca la siguiente dirección web: http://localhost:8080/, y se debe cargar la página de prueba de tomcat que se muestra en la Figura 68.
8. Un paso adicional en la instalación de tomcat, es configurar la herramienta ant para que pueda instalar servicios web directamente en el servidor tomcat, esto se hace copiando el archivo **catalina-ant.jar** que se encuentra en el subdirectorio **server\lib** dentro del directorio en el que se instaló el tomcat, este archivo se debe copiar al servidor telefónico dentro del subdirectorio **lib** en la carpeta que se instalo la herramienta ant.

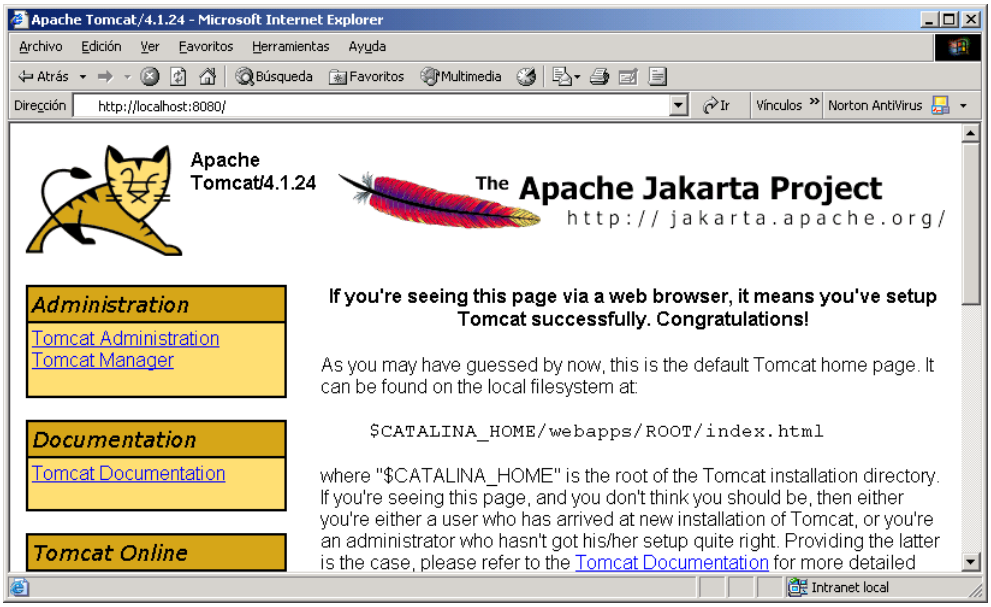

Figura 68: Página de prueba de Tomcat

C.1.4 Instalación del motor de bases de datos MySQL

La ultima versión del motor de bases de datos MySQL se puede descargar de la siguiente dirección:

- http://www.mysql.com/downloads/

NOTA: La aplicación desarrollada ha sido probada con la versión 3.23.52 que se encuentra en el CD de documentación de este trabajo.

1. Este software se distribuye empaquetado en un archivo de formato zip, es necesario descomprimirlo a un directorio temporal usando una herramienta como winzip.
2. Dentro del directorio temporal en el que se descomprimió el instalador, se ejecuta el archivo de nombre **setup.exe** para iniciar la instalación, hecho esto aparece la ventana de bienvenida de la Figura 69, se hace click en **Next**.
3. Ahora aparece la ventana de información, Figura 70, en la cual se hace click en **Next**.
4. El siguiente paso es la escogencia del directorio donde se debe instalar, Figura 71, en este se debe tener en cuenta que no es aconsejable que el nombre del directorio tenga espacios (por ejemplo C:\Archivos de Programa\mysql), es preferible instalarlo en el directorio c:\mysql, escoja el directorio y haga click en **Next**.
5. Ahora se debe escoger el tipo de instalación, Figura 72, escoja la opción **typical** y haga click en **Next**.
6. Una vez terminada la instalación, para probar el correcto funcionamiento es necesario ejecutar el programa **winmysqladmin.exe** que se encuentra en el subdirectorio **bin** dentro de la carpeta donde se instaló el servidor MySQL, al hacerlo aparece la venta que se muestra Figura 74 donde se debe escribir el nombre de usuario: **root** y la contraseña que se va a usar para la administración de la base de datos, estos datos se deben tener en cuenta para la configuración de la aplicación.

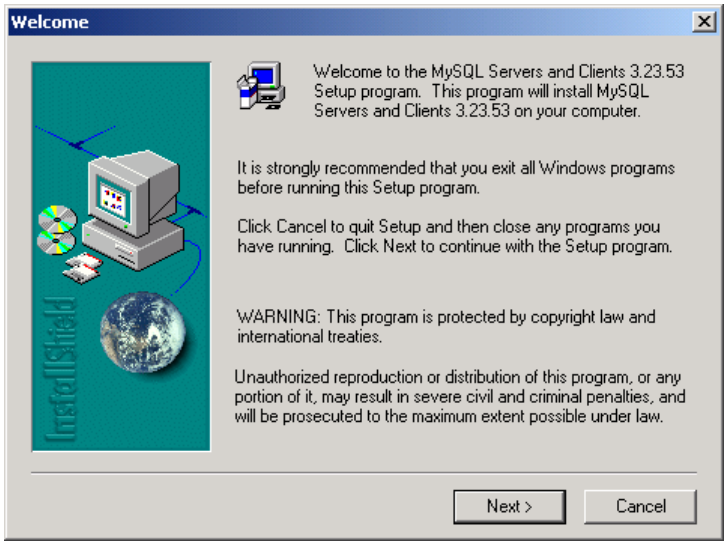

Figura 69: Bienvenido a la instalación

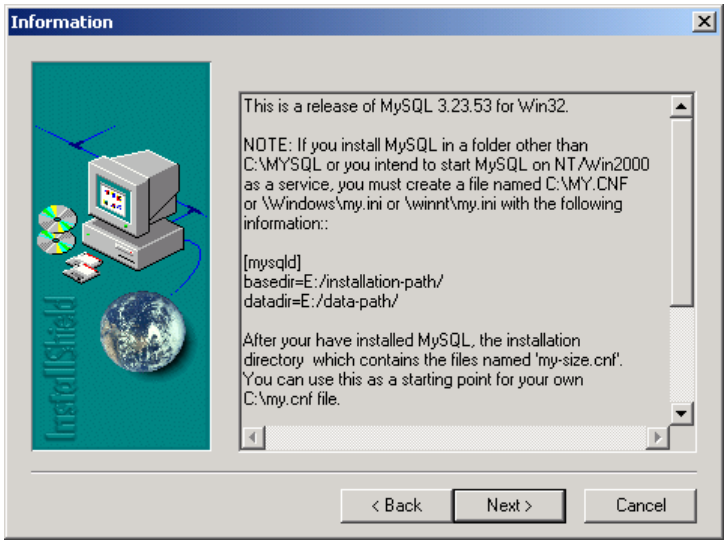

Figura 70: Información

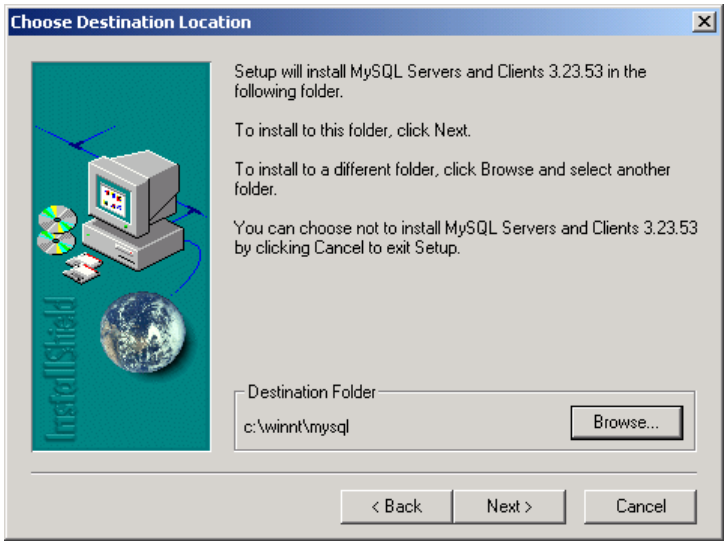

Figura 71: Directorio de instalación

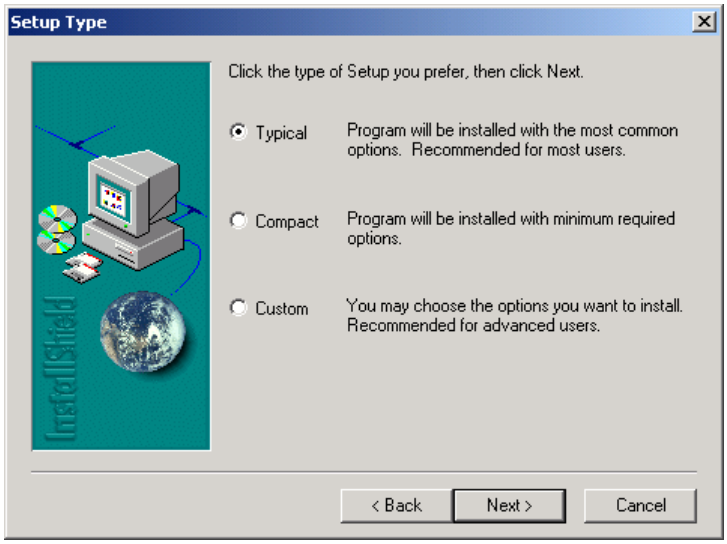

Figura 72: Tipo de instalación

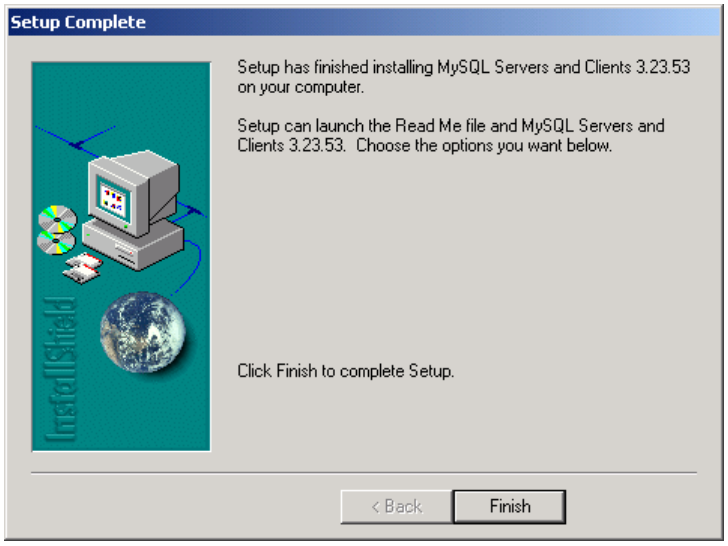

Figura 73: Instalación completa

Figura 74: Usuario y contraseña del administrador de MySQL

7. Al hacer click en **OK**, se abre la ventana de administración de la base de datos que se muestra en la Figura 75, al desplegarse esta ventana se verifica que la base de datos se está ejecutando correctamente si el semáforo que se encuentra al lado superior derecho se encuentra en verde. Con esto concluye la instalación del motor de base de datos MySQL.

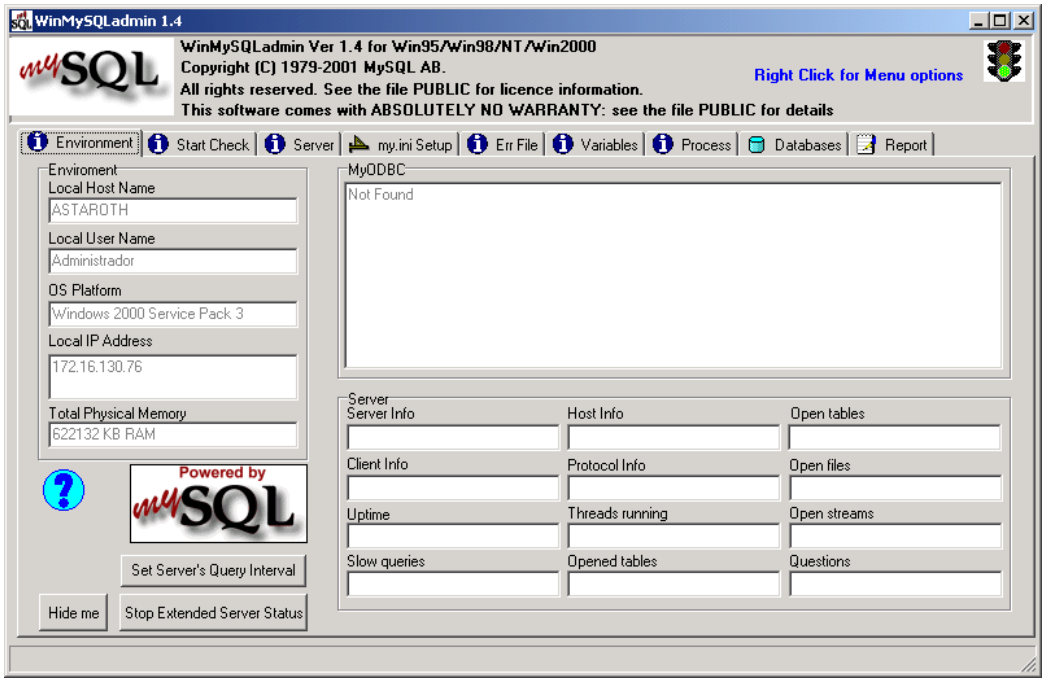

Figura 75: Herramienta de administración de MySQL

8. Ahora de debe crear la base de datos en la cual se alojarán las tablas, para esto ejecute el comando: **mysqladmin create grisat**, dentro de la carpeta **bin** del directorio de instalación de mysql, en este caso hemos llamado grisat a la base de datos, este nombre se debe tener en cuenta en los pasos siguientes para crear al usuario y para configurar la aplicación.
9. La creación del usuario se hace ejecutando el comando **mysql mysql**, al hacerlo entramos a la base de datos llamada mysql, dentro de esta se ejecutan los siguientes comandos:

   grant all privileges on grisat.* to mconsta@'%' identified by 'hola';

   grant all privileges on grisat.* to mconsta@localhost identified by 'hola';

   quit

   En estos comandos se uso **grisat** como nombre de la base de datos, **mconsta** como el nombre de usuario y **'hola'** como contraseña de la base de datos. Estos datos se deben tener en cuenta para la configuración de las aplicaciones. Al salir de nuevo al símbolo del sistema se ejecuta por último el comando **mysqladmin flush-privileges**.

10. En este paso se van crear las tablas e importar los datos necesarios, en el CD de instalación dentro de la carpeta MYSQL se encuentran dos archivos: **DB-ServidorTelefonico.txt** y **DB-ServidorWeb.txt**, escoja el que corresponda al servidor que se está instalando y cópielo a la carpeta **bin** del directorio de instalación de mysql y ejecute el siguiente comando:

    mysql grisat <nombre-archivo

C.1.5 Instalación de la aplicación JCCWebService

La aplicación web está empaquetada en un archivo war, el cual se encuentra en el directorio raíz de la aplicación, este archivo se llama JCCWebService.war. Para su instalación se siguen los siguientes pasos:

1. Copiar el archivo **JCCWebService.war** al directorio **webapps** dentro de la carpeta de instalación del Tomcat, al hacer esto el servidor Tomcat lo descomprime automáticamente en un subdirectorio con el nombre **JCCWebService**.
2. Se deben revisar los datos de configuración de la aplicación que se encuentran en el archivo **web.xml** dentro del directorio **WEB-INF**, estos datos deben acomodarse como se describe en la Tabla 3.

Tabla 3: Parámetros de configuración de la aplicación JCCWebService

Parámetro	ServidorTel	passDB	urlJDBC	userDB
Valor	Nombre o dirección IP del servidor telefónico.	Contraseña de la base de datos usada para los suscriptores.	URL de conexión a la base de datos.	Usuario de la base de datos.

3. Pruebe la aplicación abriendo un navegador y apuntando a la dirección **http://localhost:8080/JCCWebService/index.jsp** lo cual debe mostrar la pagina de inicio de la aplicación.

C.1.6 Instalación de la aplicación JCCDemoApp

La aplicación JCCDemoApp se encuentra en el directorio JCCDemo, para instalarla se deben seguir los siguientes pasos:

1. Copiar el directorio **JCCDemo** al disco duro del servidor telefónico.
2. Editar la configuración del servidor de base de datos, esta configuración se encuentra en el archivo **config.properties** dentro del directorio **src\unicauca\grisat\jccdemo**, es un archivo de texto plano que contiene las siguientes opciones:

Tabla 4: Parametros de configuración de la aplicación JCCDemoApp

Parámetro	phonelauncher.dbuser	phonelauncher.dbpass	phonelauncher.urljdbc
Valor	Usuario de la base de datos.	Contraseña de la base de datos usada para los suscriptores.	URL de conexión a la base de datos.

3. Una vez configurados estos datos se debe ejecutar el comando **ant** dentro del directorio de la aplicación.
4. Para iniciar la aplicación ejecute el comando **run** dentro del subdirectorio **build** de la carpeta de la aplicación.

C.2 Manual de usuario

Como se ha explicado anteriormente, el software se ejecuta en tres máquinas diferentes, las cuales se han llamado: cliente, servidor web y servidor telefónico, cada una de estas máquinas debe tener instalado el software requerido como se mostró en la Tabla 2.

Para iniciar el proceso de simulación de los servicios es necesario iniciar la aplicación JCCDemoApp y crear los terminales telefónicos que intervendrán en el proceso, esto se logra siguiendo estos pasos:

1. Para iniciar la aplicación JCCDemoApp en el servidor telefónico, es necesario ejecutar el archivo **run.bat**, que se encuentra en la carpeta **build** del directorio de instalación de dicha aplicación.
2. Una vez echo esto se abre la ventana que se muestra en la Figura 76, usando esta interfaz se crean los terminales telefónicos que intervendrán en la simulación del servicio.
3. Para crear un terminal telefónico, primero se escoge el color del terminal usando el selector de color, el siguiente paso es escribir el número telefónico que se le asignara al terminal en la casilla "número telefónico" y por ultimo, se crea el terminal haciendo click en el botón **launch**. Para simular los servicios es necesario que halla al menos 3 terminales telefónicos correspondientes a los abonados A, B y C, a manera de tutorial, se crearán los terminales cuyos datos se encuentran en la **¡Error! No se encuentra el origen de la referencia.**. Al hacer esto deben aparecer en la pantalla los terminales correspondientes como se observa en la Figura 77. Los terminales telefónicos se comportan de una manera similar a los telefónicos comunes, en este caso se han

habilitado 2 botones de control (**send** y **cancel** en la Figura 77) y los botones numéricos (1, 2 ... 0 en la Figura 77).

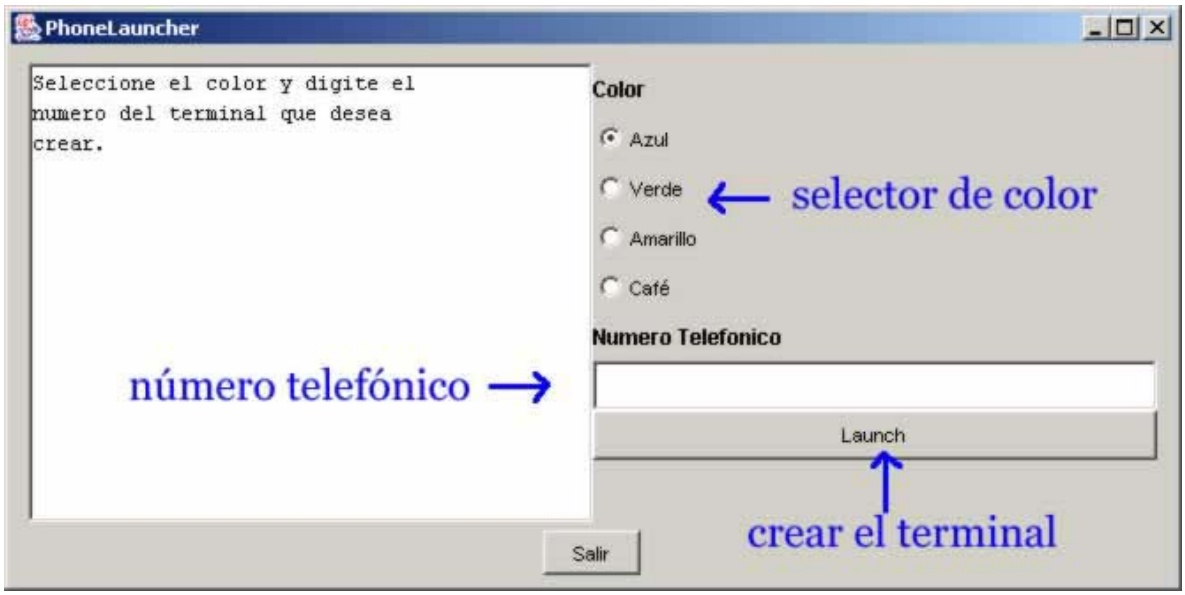

Figura 76: Ventana de la aplicación JCCDemoApp

Tabla 5: Terminales telefónicos

Número telefónico	Color del terminal
8240158	Azul
8203127	Verde
8209800	Amarillo

C.2.1 Uso de los terminales telefónicos para realizar llamadas

Para hacer uso de los terminales telefónicos, se hace click con el puntero del ratón en el botón deseado y el terminal responde de la manera adecuada, por ejemplo para hacer una llamada desde un terminal a otro se marca el número telefónico del terminal deseado desde cualquier terminal diferente al destino y una ves se ha terminado de marcar el número telefónico se hace click en el botón **send**, con esto se inicia el proceso de llamada, y el terminal de destino (abonado B) comienza a timbrar (Figura 78 izquierda), para aceptar la

llamada se hace click en el botón **send** del terminal de destino y para rechazar la llamada se hace click en el botón **cancel**. Cuando el terminal se encuentra en una llamada activa, aparece el mensaje **call in progress** (Figura 78 derecha), en este punto se puede terminar la llamada haciendo click en el botón **cancel**.

Figura 77: Terminales telefónicos usados en la simulación

C.2.2 Interfáz web

La interfaz web se usa para dos tareas: activar y configurar el servicio de transferencia de llamada (Sígueme) y para hacer uso del servicio click-to-dial como se describirá en la siguientes secciones.

El primer paso para ingresar a la interfaz web es la autenticación del usuario, para esto es necesario abrir un navegador web y apuntar a la siguiente dirección:

http://servidorweb:8080/JCCWebService/index.jsp, esto nos abre el formulario de ingreso al sistema que se muestra en la Figura 79. En esta ventana se debe escribir el número telefónico del suscriptor y su contraseña, estos datos deben estar registrados en la base de datos de suscriptores, para el propósito de este tutorial, se hará uso del usuario identificado con el número **8240158** cuya contraseña es "**hola2003**".

Figura 78: Terminal alertando y llamada en progreso

Al ingresar al sistema, el suscriptor tiene la opción de hacer uso del servicio click-to-dial (opción **Realizar llamada** del menú de servicios) o el servicio sígueme (opción **Transferir llamada** del menú de servicios) como se muestra en la Figura 80

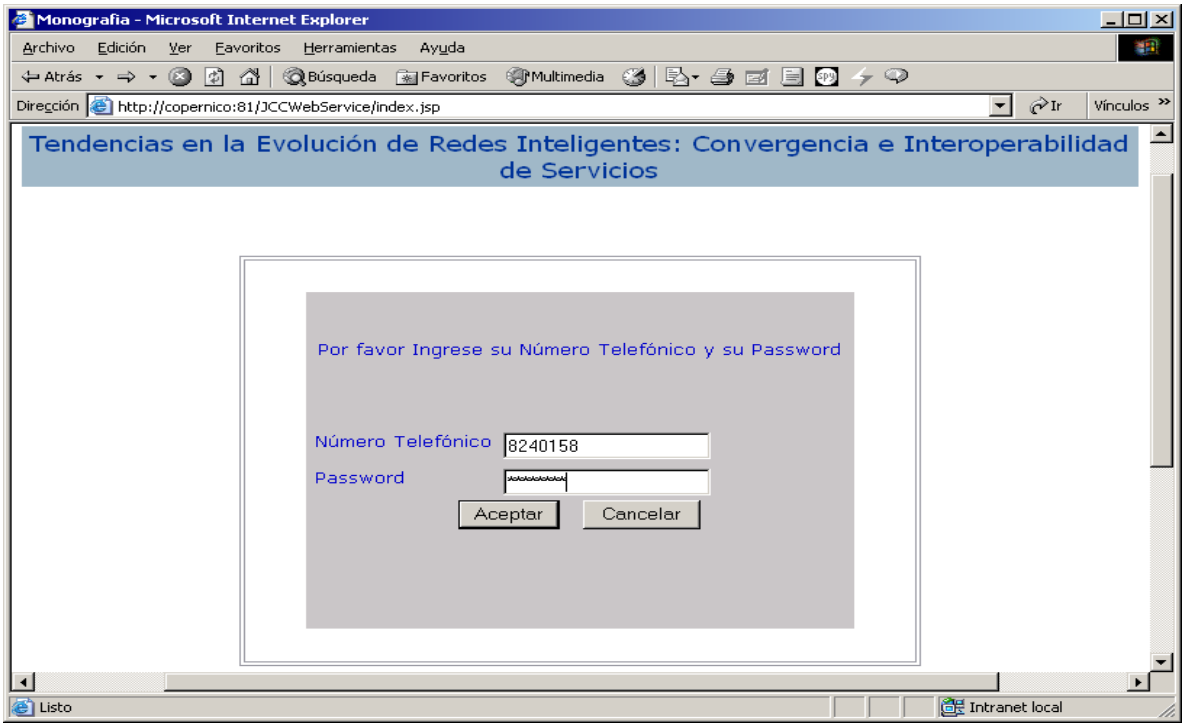

Figura 79: Formulario de ingreso al sistema

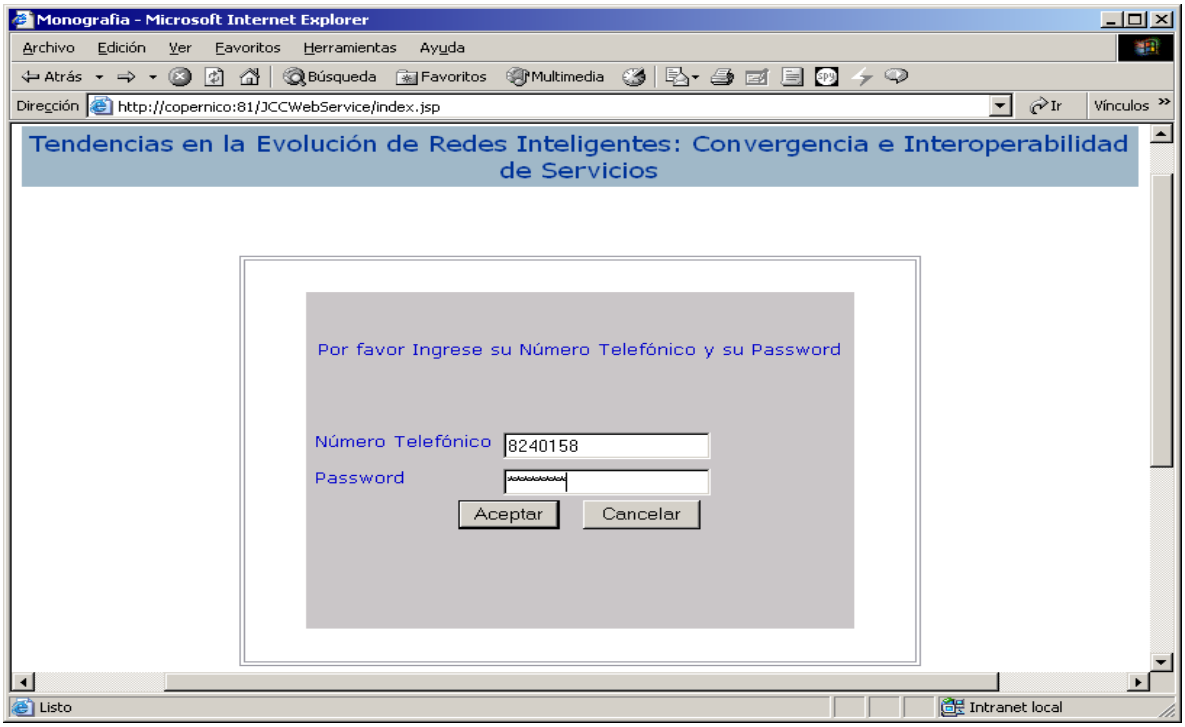

Figura 80: Formulario de servicios

C.2.2.1 Opción Realizar Llamada

Para realizar una llamada a través de la interfaz web, se escoge la opción "**Realizar llamada**" del menú de servicios (Figura 80), lo cual abre la ventana de la Figura 81, en este formulario se debe ingresar el número telefónico de destino y se presiona el botón **Aceptar**.

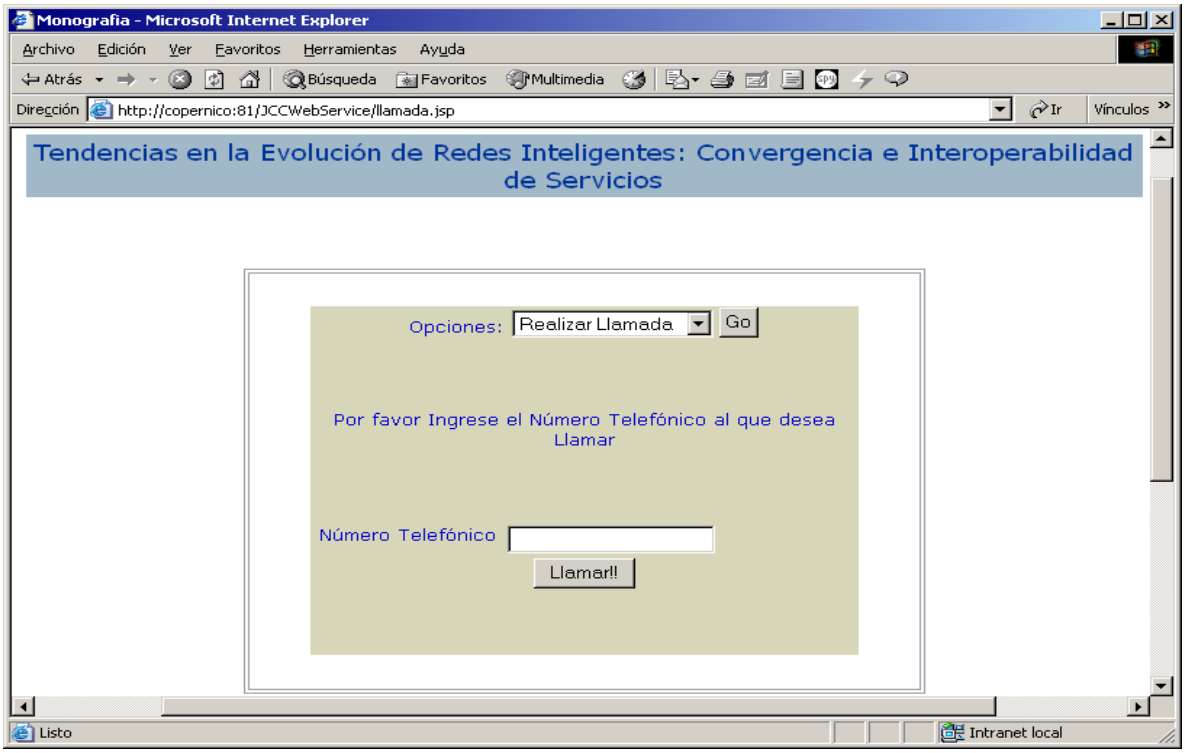

Figura 81: Formulario para realizar llamada

Al hacer esto se inicia el proceso de llamada y el abonado de destino timbra, una vez el abonado de destino ha aceptado la llamada presionando el botón **send** del terminal telefónico se alerta el abonado de origen, el cual a su vez debe aceptar la llamada presionando del botón **send** de su terminal telefónico, al hacer esto se establece la llamada como una llamada norma, para terminarla se hace click en el botón **cancel** del terminal telefónico.

C.2.2.2 Opción Transferir Llamada

Para usar el servicio de transferencia de llamada, se escoge la opción "**Transferir llamada**" del menú de servicios (Figura 80), lo cual abre la ventana de la Figura 82.

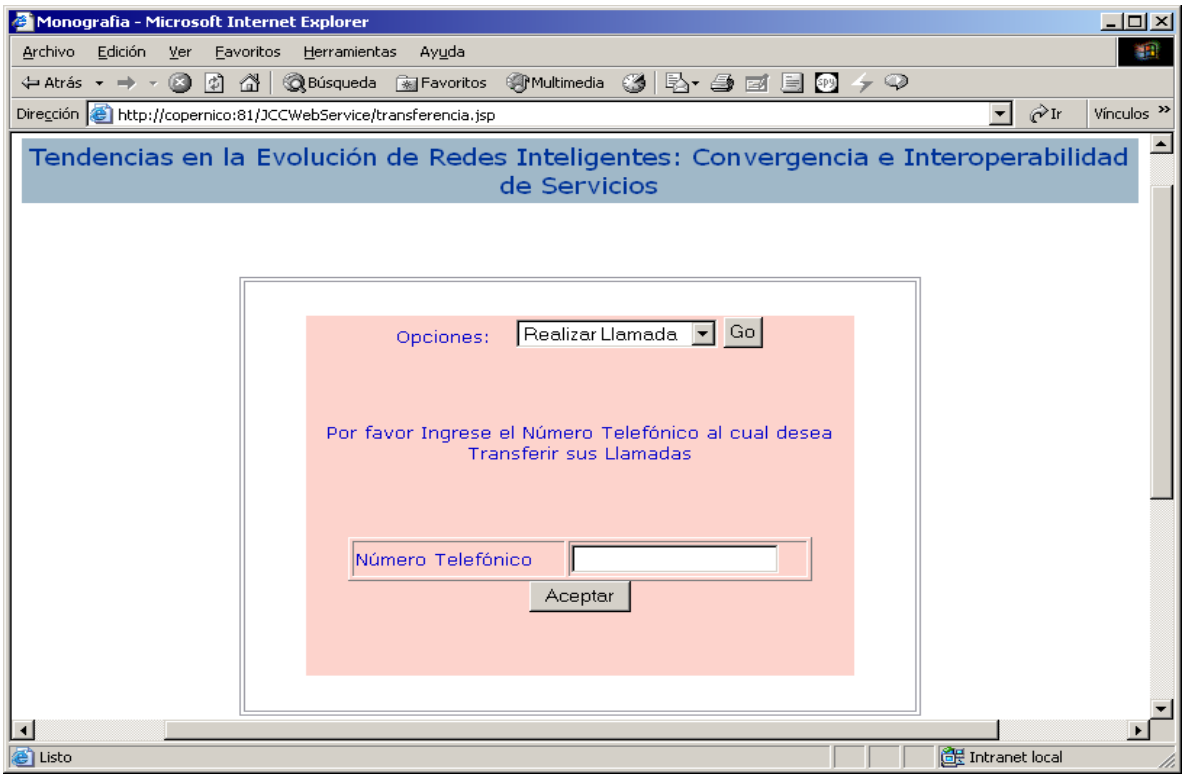

Figura 82: Formulario de transferencia de llamada

Para activar el servicio de transferencia de llamada se debe escribir el número telefónico de destino (por ejemplo 8209800) en la casilla "**Número Telefónico**" y se presiona el botón **Aceptar**, una ves se ha activado el servicio se puede proceder a realizar una llamada usando los terminales telefónicos creados anteriormente con destino al suscriptor que activó el servicio (8240158) y se verifica que la llamada se ha transferido al número deseado. Para desactivar la transferencia de llamadas se debe dejar en blanco la casilla "**Número Telefónico**" y se presiona el botón **Aceptar** con lo que los datos quedan grabados.