TRABAJO FIN DE CARRERA

TÍTULO: IP Centrex

AUTOR: Roger Massa Torrelles

DIRECTOR: Toni Oller Arcas

FECHA: 19 de enero de 2006

Título: IP Centrex

Autor: Roger Massa Torrelles

Director: Toni Oller Arcas

Fecha: 19 de enero de 2006

Resumen

Este documento recoge el trabajo realizado para diseñar e implementar una centralita o PBX para Telefonía IP basada en VoIP (Voz sobre IP) mediante SIP. Proporcionando una alternativa a las actuales centralitas de telefonía, basadas en hardware, que son caras y poco escalables.

Se detallan los conceptos VoIP, IP Centrex, se plantean diferentes esquemas para el diseño de IP Centrex y se presentan los detalles de la implementación de IP Centrex.

Para la implementación de IP Centrex se desarrolla una aplicación sobre un servidor de aplicaciones convergente HTTP/SIP basado en especificaciones SIP Servlet, propietaria de VozTelecom.

Title: IP Centrex

Author: Roger Massa Torrelles

Director: Toni Oller Arcas

Date: 19th January 2006

Overview

This document collects the work realized to design and implement a switchboard or PBX for IP Telephony based in VoIP (Voice over IP) using SIP. This provide an alternative to the current telephony switchboards, based in hardware, which are expensive and little extensible. Detailed some aspects

The concepts of VoIP and IP Centrex are detailed, different schemes for IP Centrex's design are planned and the details of IP's Centrex implementation are presented.

For IP Centrex's implementation, an application is developed over a HTTP/SIP convergent applications server based on SIP Servlet specifications, property of VozTelecom.

Para Laura por su
apoyo incondicional

ÍNDICE

ÍNDICE DE ILUSTRACIONES

ÍNDICE DE FIGURAS

INTRODUCCIÓN

IP Centrex es una tecnología que permite ofrecer los servicios de una centralita (o PBX) habitual para telefonía IP. Se basa en VoIP (Voz sobre IP), que a su vez, es una tecnología diseñada para las comunicaciones multimedia (voz, video, etc.) sobre banda ancha (Internet).

Los objetivos son diseñar e implementar una centralita para telefonía IP, como mínimo con un servicio. Para detallar el procedimiento empleado se ha separado el documento en el diseño de la arquitectura y en la implementación de IP Centrex. En la arquitectura se hallarán los diseños que se han considerado y en detalle el diseño final al que se ha optado. En la implementación se detalla cómo se ha puesto en práctica el diseño de IP Centrex, así cómo sus servicios.

Se hablará de los protocolos utilizados para la tecnología VoIP: SIP, protocolo de inicio de sesión para establecer las llamadas; RTP/RTCP, protocolo de transporte en tiempo real para el transporte de la voz y SDP, protocolo de descripción de sesión para la conjunción entre SIP y RTP.
También se hablará de la utilización de una interfaz diseñada por JAIN™, API JCC, para el control de llamadas y que ha facilitado y facilitará la implementación de servicios para IP Centrex. Y finalmente se hablará de los dispositivos utilizados por la tecnología IP Centrex: SER (SIP Express Router, para el registro de los terminales SIP); RTP Proxy (para la gestión de los flujos de voz) y Asterisk (proporcionará servicios de música y sonidos, además de poder mezclar la voz para una multiconferencia).

CAPÍTULO 1. CONCEPTOS

1.1. VozTelecom

VozTelecom es una empresa joven y con visión de futuro. El sector en el que trabaja es el tecnológico y se centra en ofrecer servicios de Telefonía IP.

VozTelecom Sistemas S.L. – B-6 3102149 – Avda. Parc Tecnològic 3 – CENT – Cerdanyola – 08290 – (Barcelona – SPAIN).

Actualmente VozTelecom ofrece a sus clientes el servicio **miCentralita.net.**

Según la empresa, "*miCentralita.net es un servicio tipo 'IP Centrex' que ofrece servicios de telefonía y funciones avanzadas de **PBX** a un conjunto de extensiones (teléfonos) a través de un acceso de banda ancha. Gracias a VozTelecom, grandes y pequeñas empresas pueden tener todos los servicios de telecomunicaciones que necesitan con una sola compra, sin tener que comprar o alquilar ninguna PBX basada en hardware*" **[22]**. Se detalla el concepto de PBX en el *apartado 1.3*.

VozTelecom lleva unos años ofreciendo este servicio IP Centrex a sus clientes, pero la tecnología que utilizan no es propietaria de VozTelecom y está limitada a un número de usuarios. Así que se empezó a diseñar un IP Centrex basado en la tecnología que utilizan, pero más robusto, extensible y propietario.

1.2. VoIP

VoIP es un estándar de la ITU (Internacional Telecommunications Union), creado en 1996 con el objeto de proporcionar una base desde la cual los desarrolladores puedan evolucionar en conjunto. El concepto de Telefonía IP es sinónimo de VoIP, es la implementación y utilización de VoIP.

La idea de transmitir voz a través de Internet, surgió en 1995 cuando Vocaltec, Inc. publicó su programa Internet Phone. Este programa estaba diseñado para ejecutarse en un 486 a 33Mhz con tarjeta de sonido, altavoces, micrófono y un módem. El software, comprimía la voz y la empaquetaba en paquetes IP para su transmisión a través del módem. Esto funcionaba perfectamente, el único problema era que los dos terminales tenían que tener instalado el software propietario de Vocaltec.

Poco después, empezaron a aparecer otros programas, aunque lo más importante, es que empezaron a crearse gateways (puertas de enlace) que permitían la intercomunicación entre la red IP (Internet) y la PSTN – Public Switched Telephone Network – (red telefónica pública conmutada, la red que se utiliza actualmente para la telefonía analógica convencional). Así se vieron posibilitadas las comunicaciones PC<->teléfono y teléfono<->teléfono a través de Internet.

La primera ventaja que observaron los usuarios es la de poder llamar a grandes distancias pagando la tasa de acceso a Internet, en vez de pagar la cantidad estipulada a través de la PSTN. Otra ventaja que existe es la de poder utilizar la infraestructura que se posee para la telefonía habitual. Finalmente, VoIP evita enviar datos cuando encuentra un silencio en la conversación, optimizando el ancho de banda utilizado.

VoIP no depende en gran medida de los proveedores de telefonía, debido a que la mayoría de conversaciones son *peer-to-peer* (P2P, se establece una comunicación entre dos únicos nodos). Pero si la comunicación que se desea establecer incluye como destino un teléfono de la red PSTN, entra en juego un gateway que trabaja entre las dos redes intercomunicándolas.

1.2.1. Telefonía IP

Se considera la telefonía IP como el servicio telefónico ofrecido sobre las redes de datos, tanto privadas como públicas. Este tipo de telefonía utiliza VoIP como tecnología para proporcionar sus servicios.

Para una mayor comprensión del proceso en una comunicación de telefonía IP se emplean los conceptos de plano de control y de plano de media.

Se diferencian dos planos debido a que el intercambio de información para el establecimiento de una llamada y la información enviada para la voz de dicha llamada, son distintos y siguen estándares distintos. Consecuentemente cada plano debe utilizar protocolos distintos.

Utilizar un mismo protocolo para establecer una comunicación mediante Telefonía IP permite poder usar cualquier terminal (teléfono, fax, etc.), sin necesidad de un ordenador con un software específico instalado. Los estádares utilizados para el plano de conrol son:

- H.323: H.323 **[1]** es un protocolo diseñado para la transmisión de datos en tiempo real entre usuarios. Se utiliza en Video Conferencias.
- SIP: SIP **[2]** es el protocolo por excelencia si se desea utilizar la telefonía IP. Más adelante se detallará el protocolo SIP (*ver apartado 1.2.2*).

Una vez se ha establecido la señalización mediante el plano de control, se realiza la transmisión de la información por el plano de media. El protocolo utilizado es RTP/RTCP.

RTP **[3]** (Real-time Transport Protocol) es un protocolo de transporte para comunicaciones en tiempo real. Va en conjunción con RTCP **[4]** (Real-time Transport Control Protocol) que controla la calidad de servicio del primero. RTP/RTCP se encuentra detallado mas adelante (*ver apartado 1.2.3.1*).

Usando SIP, el origen y el destino intercambiarán información para conocer los parámetros para la utilización de RTP y la manera de hacerlo se encuentra detallada en el SDP (*ver apartado 1.2.3.2*).

1.2.2. SIP

SIP (Session Initiation Protocol) se encuentra definido en el RFC 3261 **[2]** y es un protocolo que proporciona herramientas para trabajar con sesiones. En IP Centrex, las sesiones serán llamadas entre dos puntos y éstas se identifican por un call-ID. El Call-ID es un identificador de sesión que se crea mediante la dirección de origen, la de destino y otros parámetros de la sesión.

SIP proporciona el establecimiento de una sesión entre un terminal origen y un terminal destino. También permite poder localizar el destino, incluyendo mapeos de nombres, resolución de direcciones y redirección de destinatarios. Otra utilidad es la de determinar las capacidades del terminal de destino; para este fin se utiliza el protocolo SDP **[5]** (detallado en el *apartado 1.2.3.2*). Obtener la disponibilidad del destinatario también es una funcionalidad proporcionada; podría estar disponible, no disponible, ocupado, etc. Finalmente permite finalizar una sesión o que ésta sea transferida hacia otro destino.

Para IP Centrex se decide utilizar el protocolo SIP. Así lo que hay que tener en cuenta es que las dos partes implementen el estándar SIP e iniciar el proceso de establecimiento de la sesión.

1.2.2.1. Arquitectura

Para que un usuario A pueda llamar a un usuario B utilizará un elemento definido por SIP llamado UA (User Agent). Un UA puede comportarse como un UAC (User Agent Client) o como un UAS (User Agent Server). A un UAS le corresponde la tarea de enviar la petición de establecimiento de sesión SIP, al contrario que el UAC, el cual responde a la petición de establecimiento.
Los elementos existentes en las comunicaciones SIP se dividen en **clientes** y **servidores**. Un cliente SIP puede actuar como UAC o también puede actuar como UAS. Se considera un cliente cualquier terminal SIP (teléfonos IP, softphones, etc.) y a los gateways SIP. Un servidor puede incluir diferentes tipos de servidores:

- Servidor Proxy: igual que un Proxy habitual, recibe mensajes SIP y los reenvía hacia otro servidor SIP de la red. Puede realizar otras tareas como autenticación, autorización, control de acceso, encaminamiento, petición de retransmisión fiable y seguridad.
- Servidor de redirección: proporciona la información necesaria para saber el siguiente paso que debe hacer el mensaje. Una vez se obtiene esa información el cliente se pone en contacto con el destino pudiendo ser un servidor o el UAS (cliente destino).
- Servidor de registro: Se encarga de manejar las peticiones de registro de un UAC y habitualmente trabajan conjuntamente con alguno de los otros dos servidores. Dicha petición se utiliza para guardar la localización actual del UAC.

1.2.2.2. Establecimiento Normal

El procedimiento en una sesión sin incidentes se puede observar en *la Figura 1.1*.

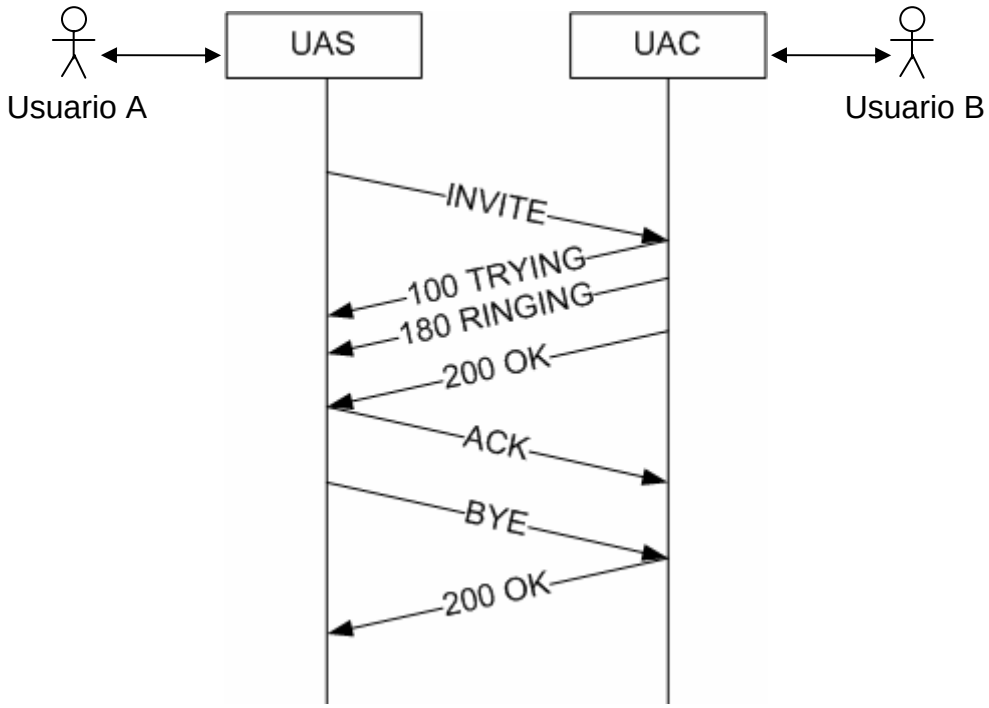


Figura 1.1. Flujo de mensajes en SIP

Se detallan el procedimiento y los mensajes transmitidos:

- **INVITE**: quien inicia la sesión (UAS, a partir de ahora LLAMADOR) envía un INVITE hacia el nodo con el que quiere iniciar la sesión (UAC, a partir de ahora LLAMADO).
- **TRYING (100)/RINGING (180)**: en cuanto el *llamado* recibe el INVITE realiza un proceso para notificar al usuario B del intento de establecer una sesión de parte del usuario A. Antes de empezar dicho proceso el *llamado* envía un TRYING al *llamador* indicando que se ha recibido correctamente el INVITE. En el momento que el proceso acaba satisfactoriamente (por ejemplo que el usuario B visualiza un teléfono sonando) el *llamado* se envía un RINGING.
- **200 OK**: Tanto el *llamado* como el *llamador* se encuentran esperando a que el usuario B indique si quiere establecer la sesión o no. En el momento que el usuario B se decide, el *llamado* se envía la confirmación (200 OK), indicando que desea establecer la sesión.
- **ACK**: finalmente cuando el *llamador* recibe la confirmación envía el reconocimiento (ACK), indicando que el *llamador* considera la sesión establecida. En el momento que el *llamado* recibe dicho reconocimiento también considera la sesión establecida.
- **BYE**: cualquiera de los dos UA puede enviar una petición de cierre de sesión. Si fuera el caso que el *llamador* envía el BYE, el *llamado* lo recibirá.
- **200 OK**: en cuanto el *llamado* recibe el BYE envía una confirmación a dicha petición y considera la sesión como cerrada. El *llamador* recibe la confirmación y también considera la sesión cerrada.

1.2.2.3. Establecimientos alternativos

Desde el punto en que el *llamador* recibe el 180, es decir, que el usuario B ha sido notificado, se pueden dar situaciones alternativas al envío del 200OK. Para la notificación de estas situaciones no normales se utilizan los mensajes con códigos 4xx y 5xx. Los mensajes 4xx son errores del cliente (UAC) y los 5xx son errores del servidor (UAS).

Se detallan dos ejemplos de situaciones alternativas:

1) El *llamado* no desea establecer la sesión
   - **486 BUSY**: el *llamado* envía una indicación de usuario ocupado (486 BUSY) para indicar que no desea establecer la sesión.
   - **ACK**: el *llamador* reconoce la recepción del 486 y considera la sesión terminada.

2) El *llamador* se retracta de querer iniciar la sesión
   - **CANCEL**: el *llamador* envía una cancelación del inicio de sesión. El mensaje CANCEL no es un mensaje de error en sí (no es ni 4xx, ni 5xx), es una petición para proceder a la cancelación del inicio de sesión.
   - **200 OK**: el *llamado* confirma la recepción de la cancelación del inicio de sesión.
   - **487 Request Terminated**: una vez el *llamado* también quiere cancelar el inicio de sesión, envía un mensaje de error indicando que se cancele la sesión.
   - **ACK**: el *llamador* recibe el mensaje de error y considera la sesión definitivamente cerrada enviando un reconocimiento (ACK) al *llamado*. El ACK indica al *llamado* que considere la sesión como terminada.
3) El *llamado* no está disponible
   - **480 Temporarily Unavailable**: el *llamado* envía un mensaje indicando su no disponibilidad y ambos consideran la sesión cerrada.
   - **ACK**: el *llamador* recibe el mensaje de error y considera la sesión definitivamente cerrada enviando un reconocimiento (ACK). El ACK indica al *llamado* que considere la sesión como terminada.

En los anexos se pueden observar distintos ejemplos de flujos de mensajes SIP (*ver anexos apartado D)*.

1.2.3. Flujos de Media

Un flujo de media es un flujo de información en cualquier formato multimedia (voz, video, etc.). Como IP Centrex utiliza el protocolo SIP y el estándar de VoIP, se deben utilizar protocolos que definan como enviar los flujos de media.

1.2.3.1. RTP/RTCP

RTP son las siglas de Real-time Transport Protocol (Protocolo de Transporte en tiempo Real) **[3]**. Es un protocolo de nivel de transporte utilizado para la transmisión de información en tiempo real como por ejemplo audio y video en una video-conferencia.

Se usa frecuentemente en sistemas de streaming, junto a RTSP **[6]**, en videoconferencias y en sistemas push-to-talk (en conjunción con H.323 o SIP). Representa también la base de la industria de VoIP.

Se ha diseñado otro protocolo que va muy unido al RTP, el RTCP (Real-time Transport Control Protocol – Protocolo de Control de Transporte en tiempo Real)y se encuentra definido en el RFC 3550 **[4]**. RTCP proporciona información de control para un flujo RTP. No transmite datos en sí, pero periódicamente envía paquetes de control a los participantes de una sesión de multimedia en streaming con información estadística. Su tarea es crear estadísticas de una conexión de media, como por ejemplo cuantos bytes se han enviado, paquetes perdidos, jitter, etc. Una aplicación puede utilizar esta información para incrementar la calidad de servicio quizás limitando el flujo, o usando una compresión mayor.

RTCP se usa para informar de la Calidad de Servicio (QoS – Quality of Service), pero no proporciona ninguna encriptación o autenticación. SRTP/SRTCP **[8]** (Secure RTP/Secure RTCP) se utilizaría para dichos fines.

1.2.3.2. SDP

El SDP, o Session Description Protocol, es un protocolo que define un formato para describir los parámetros de inicialización de un streaming de media. Ha sido caracterizado por la IETF en el RFC 2327 **[5]**.

SDP empezó como un componente de SAP **[9]** (Session Announcement Protocol), pero se han encontrado otras utilidades en conjunción con RTSP o SIP e incluso en un formato a parte, describiendo sesiones multicast.

En la descripción de un flujo se observa:

- El tipo de media (video, audio, etc.)
- El protocolo de transporte (RTP/UDP/IP, H.323, etc.)
- El formato de la media (H.261 video, MPEG video, etc.)

En una sesión IP multicast, se encuentra:

- Dirección multicast para la media
- Puerto de transporte para la media

Para una sesión IP unicast, se encuentra:

- Dirección remota para la media
- Puerto de transporte para la dirección de contacto

Un ejemplo de descripción SDP en unicast es:

```
v=0
o=roger1 1019836832 1019836945 IN IP4 192.168.1.3
s=eyeBeam
c=IN IP4 192.168.1.3
t=0 0
m=audio 8000 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
```

```
a=fmtp:101 0-15
a=sendrecv
```

Donde se haya información del origen (o=…) y los codificadores que es capaz de utilizar.

Esta descripción SDP se encuentra dentro de un INVITE enviado por un eyeBeam (softphone **[11]**, un cliente SIP). Por defecto, en un mensaje INVITE, se envía una descripción de los codificadores que puede utilizar el cliente y en la respuesta 200OK, se encuentra la descripción SDP indicando qué codificadores que puede utilizar el servidor.

Un ejemplo de descripción SDP en el 200OK es:

```
v=0
o=roger2 762320458 762322633 IN IP4 192.168.1.2
s=eyeBeam
c=IN IP4 192.168.1.2
t=0 0
m=audio 8000 RTP/AVP 0 8 3 98 97 101
a=rtpmap:0 pcmu/8000
a=rtpmap:8 pcma/8000
a=rtpmap:3 gsm/8000
a=rtpmap:98 iLBC/8000
a=rtpmap:97 speex/8000
a=rtpmap:101 telephone-event/8000
a=fmtp:101 0-15
a=sendrecv
```

1.3. PBX

Una PBX (acrónimo de Private Branche eXchange) es una centralita telefónica con fines privados. El término PBX proviene de las antiguas centralitas manuales (PMBX o PBX Manual) donde una persona interconectaba las distintas clavijas para intercomunicar otras dos personas. Posteriormente apareció un dispositivo electromecánico que conmutaba automáticamente las distintas terminaciones (PABX o PBX automático). Con el tiempo se adoptó el término PBX para referirse a las centralitas telefónicas privadas automáticas.

La principal utilidad de una PBX es la de evitar conectar todos los teléfonos privados de una empresa a la PSTN **[10]**. Si una llamada de una empresa tiene como destino un terminal de la misma empresa, con una PBX se evita que ésta salga y vuelva a entrar, evitando los costes de las llamadas internas. También se evita tener que contratar una línea de telefonía por cada terminal de la empresa.

Los distintos elementos conectados a la PBX (teléfonos, faxes, etc.) son denominados extensiones. Dos empresas pueden interconectar sus PBX y realizar llamadas internas entre ellas. Si se desea realizar una llamada externa, con sólo marcar un código (generalmente 0 o 9) automáticamente se selecciona la red troncal (PSTN **[10]**).

1.3.1. PBX y Telefonía IP

La tecnología IP Centrex está diseñada para implementar una PBX con acceso de banda ancha. Para transportar los flujos de voz sobre banda ancha se utiliza el estándar VoIP (*ver apartado 1.2*). Y el protocolo utilizado para establecer las llamadas entre las distintas extensiones es **SIP** (*ver apartado 1.2.2*).

1.4. IP Centrex

1.4.1. La tecnología

El concepto de IP Centrex se define como una tecnología que ofrece soluciones para **VoIP** (Voice over IP – Voz sobre IP) o **Telefonía IP** (*ver apartado 1.2*). Tiene como objetivo combinar los servicios de la Telefonía IP con los servicios proporcionados por una PBX tradicional (*ver apartado 1.3*).

IP Centrex permite unificar las comunicaciones de una empresa con diferentes ubicaciones geográficamente distantes (trabajadores, sucursales, central, etc.). Se destinará a clientes residenciales y pequeñas empresas que deseen prescindir de las centralitas analógicas privadas, e interactuará con ellos mediante terminales IP (softphones **[11]**, teléfonos IP o teléfonos analógicos con un ATA) o a través de un acceso de banda ancha (Internet).

En resumen, IP Centrex unirá los servicios de la Telefonía IP con los servicios de una PBX analógica, pero rebajando considerablemente las dificultades económicas y logísticas de su utilización.

Las partes de esta tecnología son el núcleo, una administración del servicio y una interfaz de usuario (se comentarán a fondo más adelante). Del conjunto, este TFC se centra en el núcleo.

1.4.2. Los Servicios

Se han diseñado un conjunto de servicios que ofrecerá IP Centrex para implementar las funcionalidades básicas.

En todos los escenarios se toman como hipótesis las siguientes condiciones:

- Los usuarios de los terminales tienen configurado su terminal para que se registre al Proxy SIP (SER) de IP Centrex (detallado en el *apartado 3.2.1*).
- El usuario tiene a su disposición una aplicación Web (Interfaz de Usuario, detallado en el *apartado 2.2.3*), para administración y uso del servicio de IP Centrex. Dicho usuario se habrá registrado informando de la extensión del terminal a su disposición. Así, por ejemplo, el usuario del terminal 1 con extensión 3423 se registrará con la extensión 3423.

1.4.2.1. Llamada por UI (Click2Dial)

Es el servicio más básico, el servicio de llamada asistida. El usuario puede usar la Interfaz de Usuario para establecer una comunicación. Si desea llamar al usuario que tiene visualizado en la página del navegador, lo único que tiene que hacer es presionar sobre un hipervínculo y se establecerá la comunicación gracias al servicio Click2Dial. La interacción del usuario con el explorador Web se verá con más detalle en el *apartado 2.2.3*.

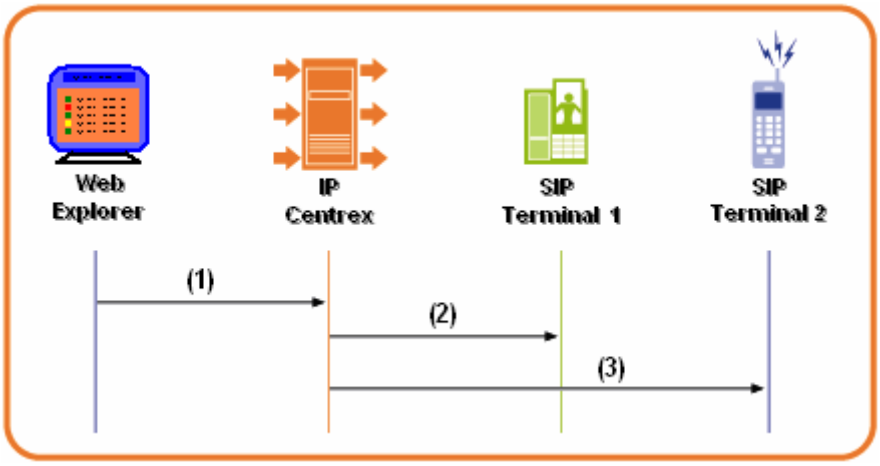


Ilustración 1.1. Representación del servicio Click2Dial

1. El usuario presiona sobre el hipervínculo y se envía la petición HTTP hacia IP Centrex
2. IP Centrex establece una llamada con el terminal 1 (es el terminal del usuario que se ha registrado en el Web Explorer)
3. Se establece la llamada con el terminal 2

1.4.2.2. Llamada por terminal (IncommingCall)

Otra posibilidad para establecer una comunicación es introduciendo el numero de extensión del terminal de destino en el propio terminal del usuario.

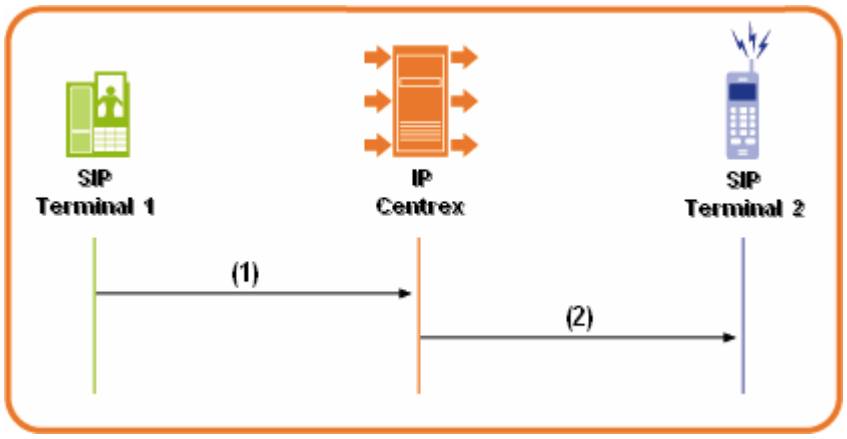


Ilustración 1.2. Representación del servicio Llamada por terminal

1. El usuario del terminal 1 introduce la extensión del terminal 2 y se establece la llamada con IP Centrex (diseño B2BUA, se detalla en el *apartado 2.1.4*)
2. IP Centrex establece una llamada con el terminal 2. Los usuarios del terminal 1 y 2 pueden hablar

1.4.2.3. Captura de llamada (PickUp)

Las líneas o una porción de las líneas, pueden ser configuradas para que sean miembros de un grupo de captura, por lo tanto, una llamada entrante a cualquiera de los terminales del grupo de captura puede ser respondida por cualquier terminal del grupo.

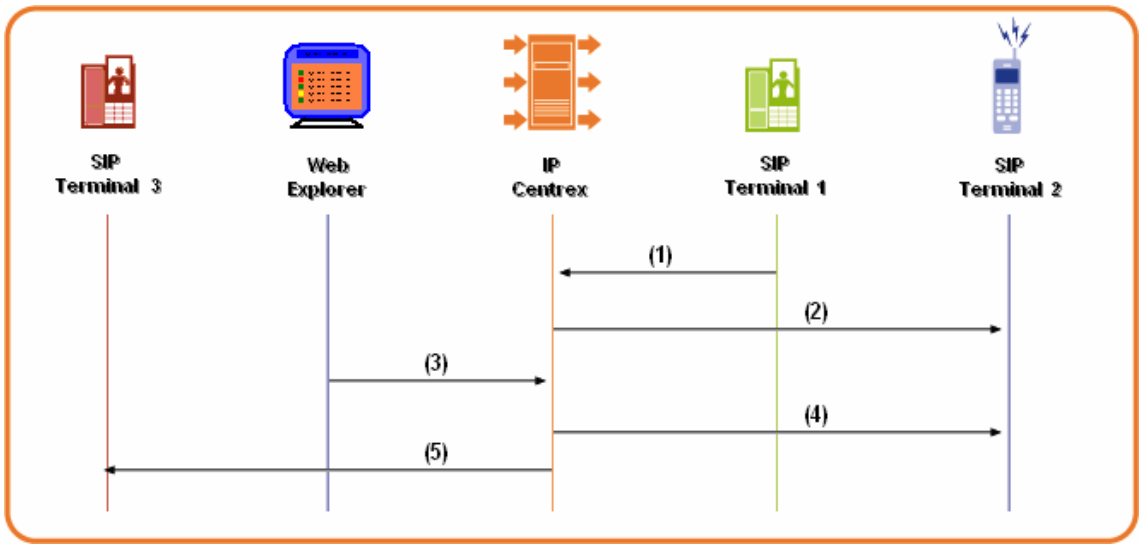


Ilustración 1.3. Representación del servicio Captura de llamada (PickUp)

1. El usuario del terminal 1 desea establecer una comunicación con el terminal 2, IP Centrex establece la llamada con el terminal 1
2. IP Centrex intenta establece contacto con el terminal 2
3. El usuario del terminal 3 presiona en el hipervínculo antes que el usuario del terminal 2 conteste
4. IP Centrex deja de intentar establecer la sesión con el terminal 2
5. Se establece la comunicación con el terminal 3, los usuarios del terminal 1 y el 3 pueden hablar

1.4.2.4. Transferencia de llamada (TransferCall)

Transfiere una llamada en curso a otro terminal. Esto implica colocar en espera la llamada entrante, mientras se hace efectiva la transferencia. Se puede establecer una comunicación entre el usuario que transfiere y el que recibe la transferencia para informar sobre esta transferencia, pudiendo este último aceptarla (conmutación de la llamada) o rechazarla.

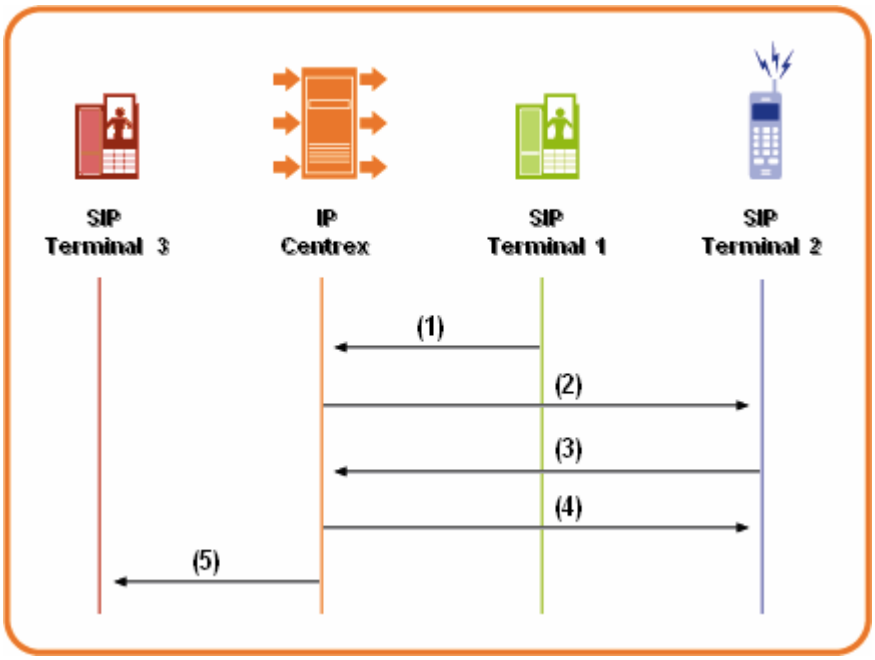


Ilustración 1.4. Representación del servicio Transferencia de llamada

1. El usuario del terminal 1 quiere llamar al terminal 2, IP Centrex establece la llamada con el terminal 1
2. IP Centrex establece la llamada con el terminal 2, los usuarios del terminal 1 y 2 hablan
3. El usuario del terminal 2 quiere transferir la llamada al terminal 3
4. IP Centrex pone en espera al terminal 2
5. IP Centrex establece una llamada con el terminal 3, el usuario del terminal 1 y 3 hablan

1.4.2.5. Desvío de llamadas (DeflectingCall)

Las llamadas entrantes a una extensión o número dado, se reenvían a otros números preseleccionados bajo ciertas circunstancias. Pueden desviarse las llamadas si la extensión llamada está ocupada, no se recibe respuesta después de un determinado tiempo o simplemente configurar el desvío hacia otra extensión en algún momento. Se pueden desviar las llamadas a un número en particular o a una serie de números predeterminados.

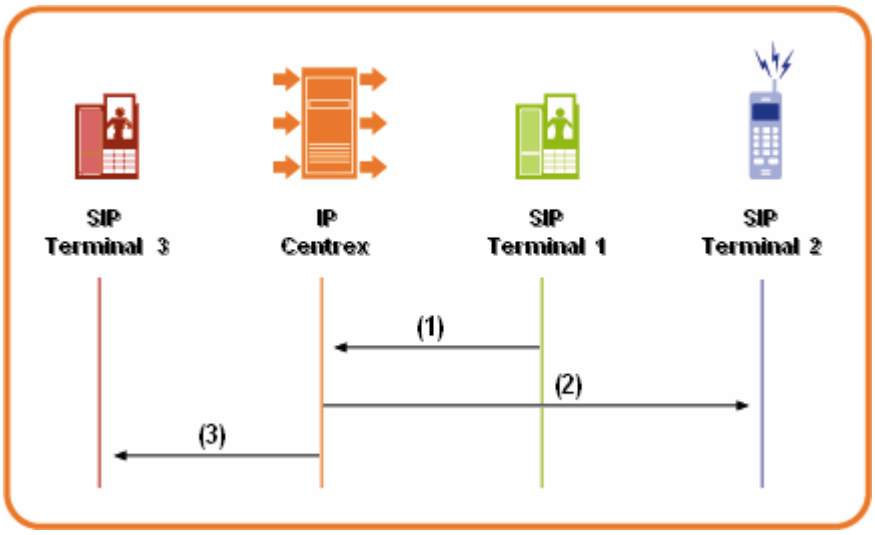


Ilustración 1.5. Representación del servicio Desvío de llamadas

1. El usuario del terminal 1 quiere llamar al terminal 2, IP Centrex establece la llamada con el terminal 1
2. IP Centrex intenta establecer una llamada con el terminal 2
    a. El terminal 2 se encuentra ocupado
    b. Se ha establecido previamente que si el terminal 2 no se encuentra disponible se desvíe la llamada al terminal 3
3. IP Centrex establece una llamada con el terminal 3. Los usuarios del terminal 1 y 3 pueden hablar

1.4.2.6. Filtrado de llamadas salientes (OutGoingFilteredCall)

Automáticamente se rechazan llamadas salientes destinadas a números previamente configurados. Si el número saliente coincide con la lista de restricciones el terminal no realizará la llamada.

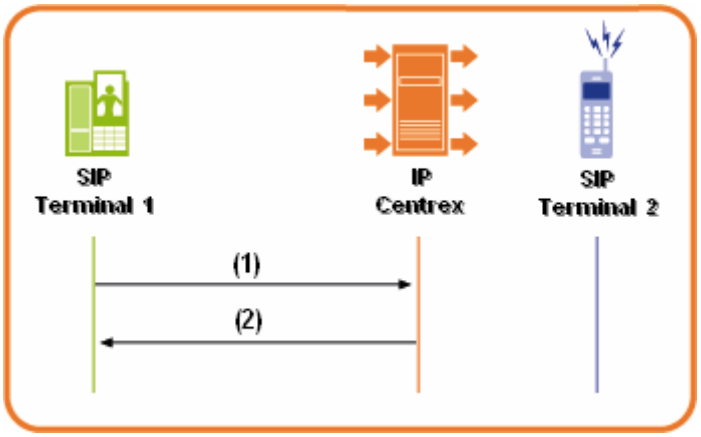


Ilustración 1.6. Representación del servicio Filtrado de llamadas salientes

1. El usuario del terminal 1 quiere llamar al terminal 2
2. IP Centrex establece que el terminal 1 no se le permite llamar al terminal 2, se le niega el establecimiento de la llamada

1.4.2.7. Música para llamada en espera (MusicOnHold)

Proporciona una pausa musical a los llamantes que han sido puestos en espera.

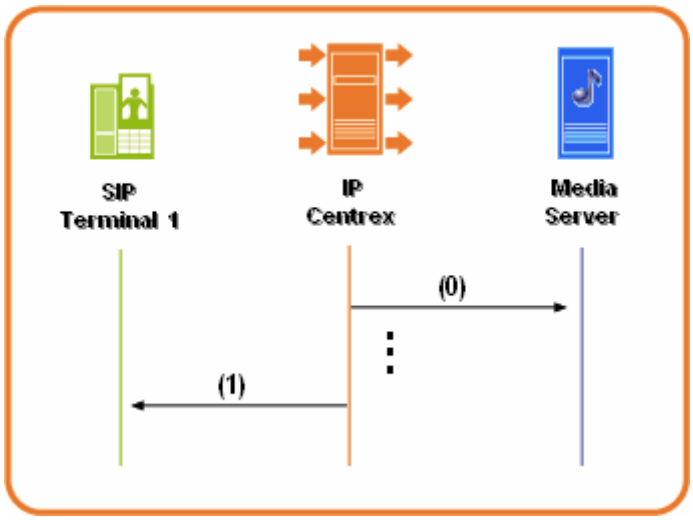


Ilustración 1.7. Representación del servicio Música en espera

0. Antes de cualquier comunicación con IP Centrex se establece una llamada con un servidor de media, lo que proporcionará un canal de música
1. En el momento que IP Centrex pone en espera al terminal 1, intercomunica el servidor de media con el terminal 1. El usuario del terminal 1 escucha música

1.4.2.8. Multiconferencia (Multiconference)

Permite a los usuarios añadir a una tercera persona a una conversación existente, formando una llamada con tres interlocutores simultáneamente. Cabe destacar que el mezclado de los flujos (*ver apartado 1.2.3*) se realiza en el plano de media mediante el servidor de media (Asterisk, detallado en el *apartado 3.2.4*).

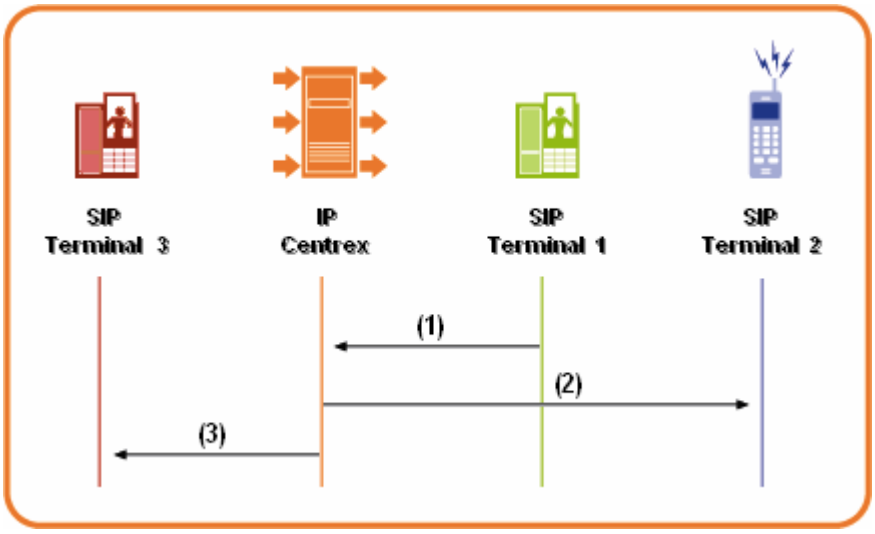


Ilustración 1.8. Representación del servicio Multiconferencia

1. El usuario del terminal 1 quiere llamar al terminal 2, IP Centrex establece la llamada con el terminal 1
2. IP Centrex establece una llamada con el terminal 2. Los usuarios del terminal 1 y 2 pueden hablar
3. IP Centrex establece una llamada con el terminal 3, para participar en la llamada. Los usuarios del terminal 1, 2 y 3 pueden hablar juntos

1.4.2.9. Identificación de llamadas entrantes (IdentifiedIncommingCall)

Permite al usuario identificar el nombre y el número de teléfono de la persona que está realizando la llamada antes de responderla, o bien solo el número de teléfono si es una llamada externa.

CAPÍTULO 2. ARQUITECTURA

2.1. Primeros Diseños

Se plantean cuatro diseños para la implementación de IP Centrex, separados en dos tipos:

- Distribuido o Centralizado
- SIP Proxy o B2BUA

Se detallarán cada uno de ellos y se señalarán las ventajas del diseño junto con una figura o ilustración explicativa.

2.1.1. Diseño Distribuido

El diseño distribuido, es un diseño donde la mayoría de procesos y comprobaciones se realizan en una "caja negra" (componentes, hardware y software, que se configuran antes de utilizarse). Esta caja negra, en este diseño, se entrega a la empresa, obteniendo tantas cajas negras como empresas (considérese empresas o particulares que contraten el servicio IP Centrex).

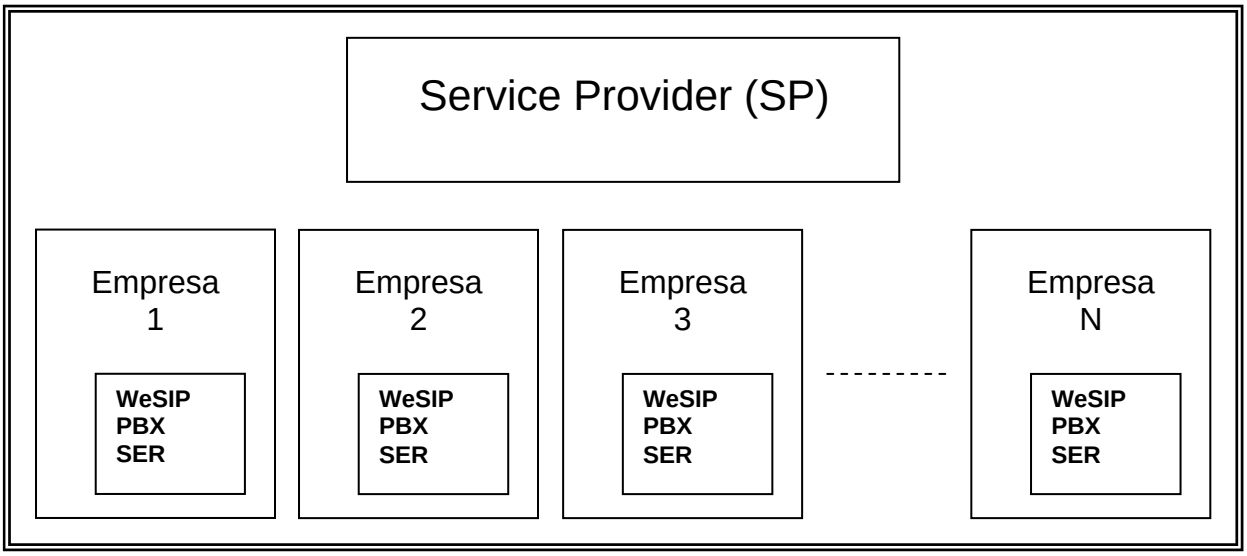


Figura 2.1. Diseño distribuido de IP Centrex

Ventajas:

- Con este diseño se distribuye la carga de un servidor central
- Posibilidad de personalizar el servicio para cada empresa
- Si el coste de fábrica de la caja negra se incrementa y se cobra a la empresa, se podrían obtener beneficios

En este diseño se observa como cada empresa se responsabiliza de todas las tareas (mediante la caja negra). Registra a los usuarios (mediante el SER, *ver apartado 3.2.1*), reenvía las diferentes llamadas dentro de la empresa o hacia fuera (función de PBX, *ver apartado 1.3*) y contiene todos los servicios

implementados (WeSIP, *ver apartado 3.2.2*). El proveedor es el encargado de administrar las diferentes “cajas negras”, proporcionando un servicio técnico.

En este caso, la PBX de la Empresa 1, solo tiene conocimiento de los usuarios pertenecientes a la Empresa 1. Si se detectara una llamada a una extensión externa a la empresa, se podría reenviar hacia el exterior (según la política de la empresa). En este punto el Proveedor de Servicios podría intervenir administrando todas las llamadas externas de las distintas empresas y si pertenece al grupo de empresas del SP reenviarla, o en caso contrario, mandarla hacia la red de telefonía convencional (PSTN **[10]**). Otra opción es que el SP no intervenga y cada empresa reenvíe sus llamadas externas hacia la PSTN.

Personalizar el servicio para cada empresa es una ventaja, pero también comporta un inconveniente. Si hay un cambio en el diseño interno de IP Centrex, se deberá ir empresa por empresa actualizando el servicio de manera personalizada.

2.1.2. Diseño Centralizado

Un segundo diseño, contrario al anterior, es un diseño completamente centralizado. Todos los procesos de autenticación, distribución de las llamadas y otros procesos lógicos se encuentran en un servidor (o conjunto de servidores) de la empresa que ofrece el servicio (SP).

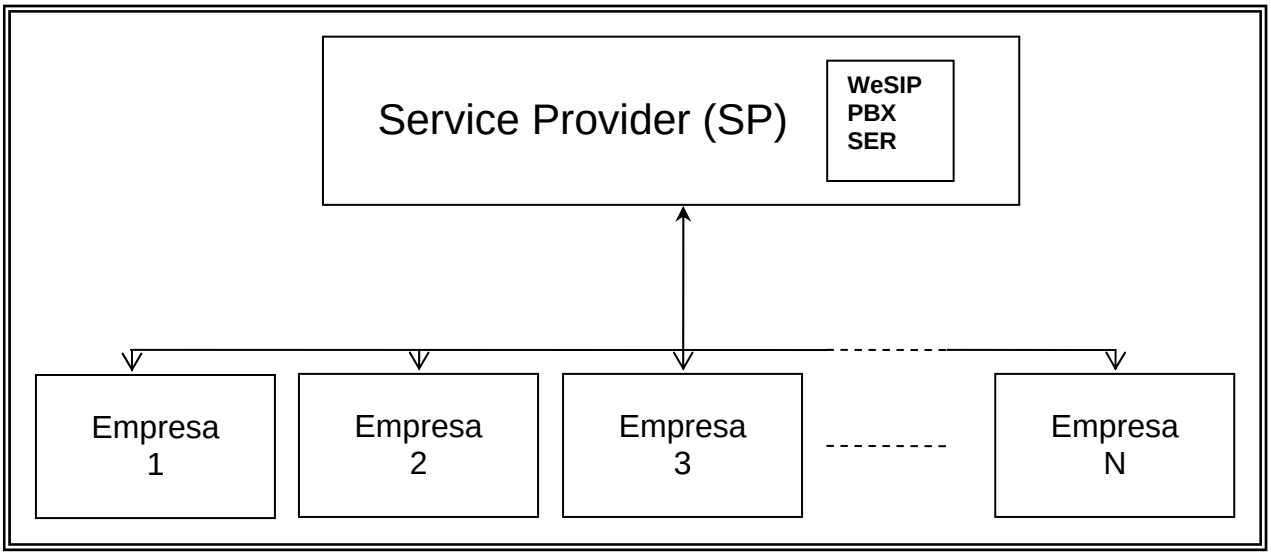


Figura 2.2. Diseño centralizado de IP Centrex

Ventajas:

- No se necesitan una gran cantidad de administradores
- El coste de base para el SP es menor, así el coste para la empresa que contrata el servicio también puede ser menor
- Control total sobre el uso y el estado del servicio ofrecido

La principal diferencia con el diseño anterior, es la duplicidad que existía, ya que en este caso solo hay UNA "caja negra". Al tener las comunicaciones centralizadas en un punto, se minimizan los escenarios de trabajo a uno solo (se detalla en el *apartado 2.2*). Con el diseño descentralizado hay tantos escenarios de trabajo como empresas, debido a que cada empresa cuenta con condiciones y variables de trabajo distintas (NAT, Proxy, router, ethernet, token ring, etc. *Ver anexos apartado A*).

Al existir diferentes "cajas negras", el coste de cada una podría ser para el Proveedor de Servicios o para la empresa, pero si se reduce el número de estas cajas a una sola más compleja, se evita el coste para la empresa. También se reducirá el número de administradores necesarios para el servicio técnico.

La función de PBX será bastante más compleja que las individuales (diseño descentralizado), debido a que recibirá las peticiones de todas las empresas y deberá comprobar si el usuario se encuentra dentro de esa empresa o de otra y si tienen permisos para ello. Si es una extensión de fuera del SP, podría reenviar la llamada a la PSTN **[10]**, etc.

Un punto bastante característico y atractivo, es que todas las comunicaciones y procesos se efectúan en el Proveedor de Servicios, obteniendo un gran control. Si cada empresa tiene su "caja negra" se podrían dar casos de actos maliciosos por parte de los usuarios de esa empresa, si pudieran tener acceso a esa "caja negra". Si se centraliza todo en un punto, el acceso malicioso se reduce considerablemente y se pueden tomar las medidas adecuadas.

2.1.3. Diseño SIP Proxy

Un primer diseño del segundo tipo para IP Centrex es un diseño SIP Proxy. Como todo Proxy, simplemente se encargará de enviar todos los mensajes SIP que recibe hacia su destino (*ver apartado 1.2.2*). Habrá una única sesión y Call ID por llamada. El SIP Proxy está diseñado como una máquina de estados, si se encuentra en el estado IDLE y recibe un INVITE redirecciona el INVITE hacia el TO del mensaje. Si por otro lado se encuentra en el estado RE_INVITE (el destinatario ha recibido el INVITE) pero si recibe un 480 BUSY en vez de un 200 OK, se pasa al estado BYE para cerrar las sesiones.

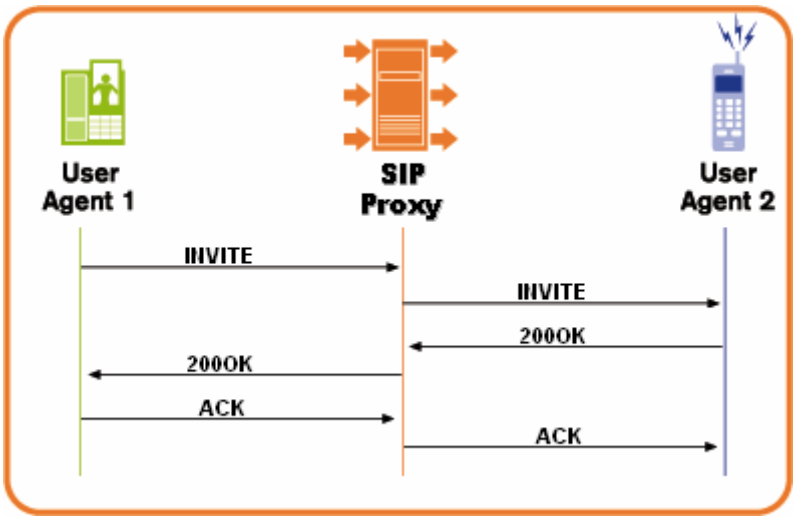


Ilustración 2.1. Ejemplo de funcionamiento de un SIP Proxy

La máquina de estados se puede encontrar en un estado, al recibir una INPUT. Dicha INPUT hace cambiar de estado mediante una TRANSITION. En el caso anterior, un estado podría ser el IDLE (sin nada que hacer), una INPUT sería un mensaje INVITE y la transición al siguiente estado podría ser RE_INVITE (redireccionando el INVITE).

Ventajas:

- Fiel al protocolo SIP
- Más sencillo de implementar (hasta la aparición de la multiconferencia)
- Se puede mantener un control de los estados de las llamadas

Con este diseño empiezan a aparecer problemas cuando se plantea una multiconferencia. ¿Cómo se puede establecer un origen y múltiples destino con SIP? Los estados del SIP Proxy se multiplican exponencialmente por cada usuario añadido a una llamada.

2.1.4. Diseño B2BUA

B2BUA es un acrónimo de Back to Back User Agent.

Los conceptos UAC y UAS se han detallado anteriormente en el *apartado 1.2.2.1*.

Un B2BUA puede actuar al mismo tiempo de UAS (UA Server) y UAC (UA Client). De este modo se obtienen dos llamadas distintas, con identificador de sesión y call-ID distintos (*ver apartado 1.2.2*). A partir de ahora ya no habrán tantos problemas para establecer llamadas con destinos distintos ya que éstas se tratan como llamadas independientes y es IP Centrex quien se encargará de conmutar o mezclar los flujos (*ver apartado 1.2.3*).

Ventajas:

- Los obtenidos con el diseño SIP Proxy
- Se pueden manejar multiconferencias

En el caso de una llamada por Click2Dial (*ver apartado 3.4.1*) se puede observar que el B2BUA se encarga de enviar los dos INVITE, actúa en ambos lados como UAC. Primero establece una sesión con el UA1 (User Agent 1) y acto seguido envía el INVITE hacia el UA2 (User Agent 2) para establecer otra sesión.

Ahora bien, en el caso de una llamada entrante del UA1, el B2BUA actúa más o menos como un SIP Proxy, pero en vez de redireccionar el INVITE, establece una sesión con el UA1 y envía un nuevo INVITE hacia el UA2. Se establecen dos sesiones completamente distintas (*ver Ilustración 2.2*).

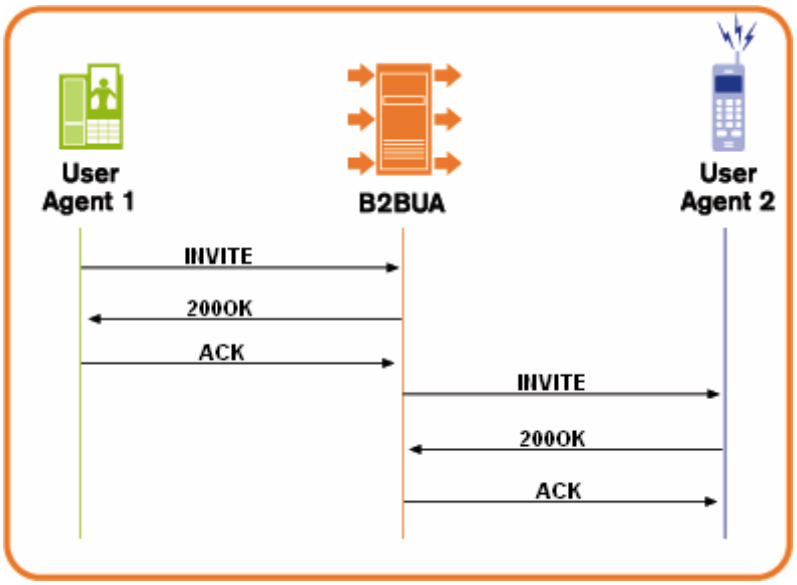


Ilustración 2.2. Ejemplo de una llamada entrante

Otra opción es que el mismo B2BUA llame a los dos participantes en la llamada (*ver Ilustración 2.3*). Esta opción se utiliza para el servicio Click2Dial (*ver apartado 1.4.2.1*).

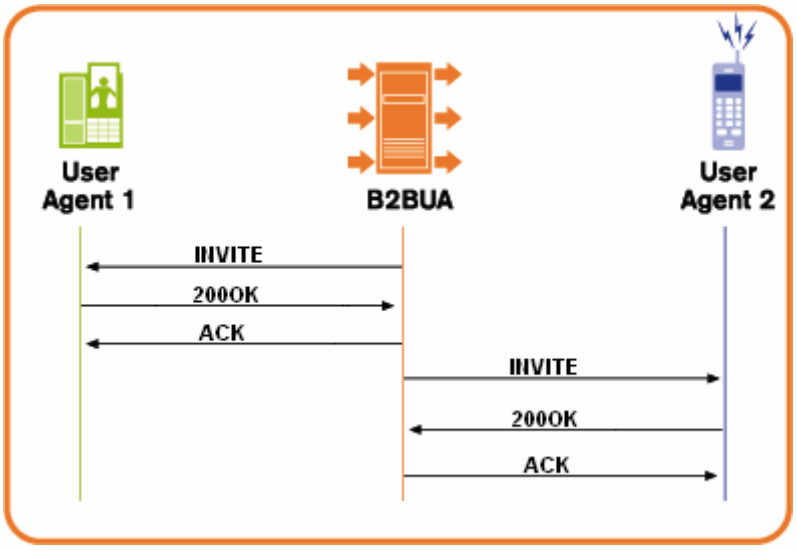


Ilustración 2.3. Ejemplo de una llamada desde el B2BUA

2.2. Diseño Final de IP Centrex

El diseño escogido para IP Centrex es de B2BUA junto con el diseño centralizado.

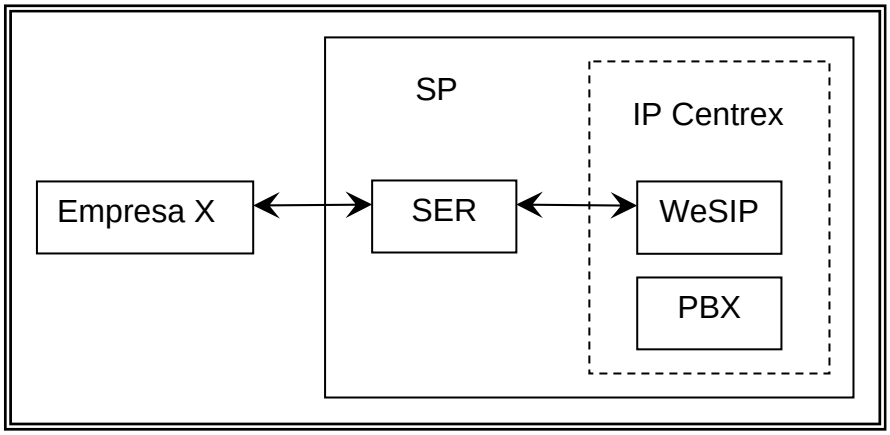


Figura 2.3. Relaciones en el diseño general de IP Centrex

Una vez elegido el diseño general se entra la tecnología en sí. IP Centrex está distribuido en tres grandes partes: el núcleo, la administración del servicio y por último, pero no menos importante, la interfaz de usuario.

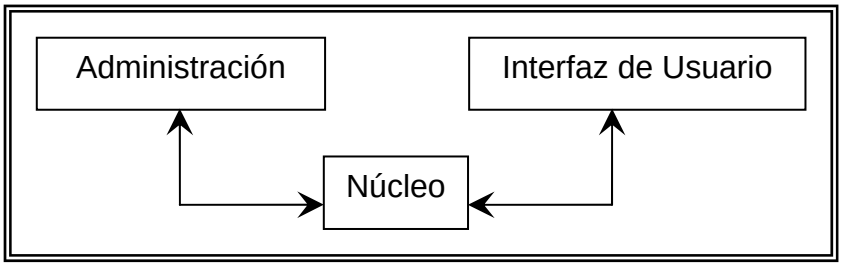


Figura 2.4. Relación entre las partes de IP Centrex

2.2.1. Núcleo

El núcleo, la parte más densa de IP Centrex, se encuentra dividida en dos bloques: **HTTP Servlet – SIP Servlet** y **JCC**.

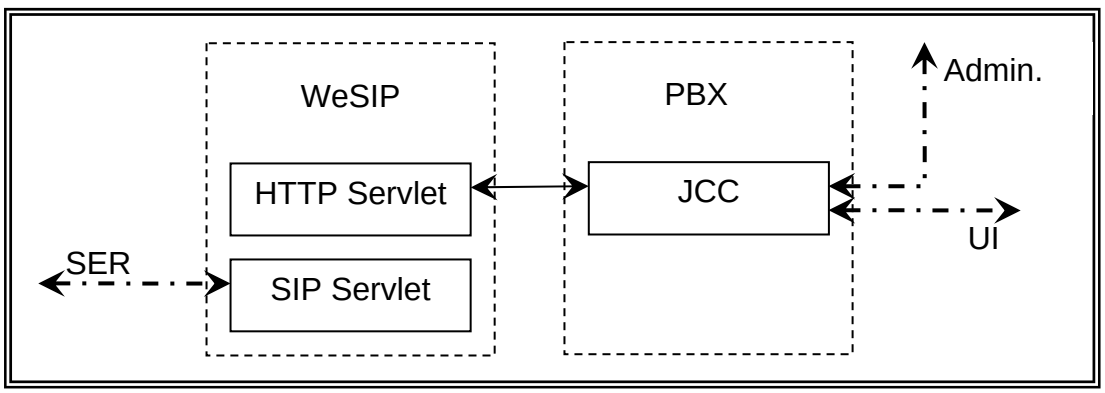


Figura 2.5. Relación entre los bloques del núcleo

2.2.1.1. HTTP Servlet – SIP Servlet

Este bloque se encuentra incrustado en un Tomcat. Su cometido es recibir y enviar mensajes SIP y recibir y responder mensajes HTTP. En el *apartado 2.3* se encuentra detallado este bloque.

2.2.1.2. JCC

Con el bloque JCC se obtiene una abstracción de todos los procesos del bloque de mensajes (HTTP Servlet – SIP Servlet), mediante una API (*ver apartado B.5)* del grupo JAIN™, API JCC 1.0 **[13]**. Dicha API es únicamente una plantilla con la que trabajar de manera estándar, así que se ha creado una implementación para ser utilizada por IP Centrex. En el *apartado 2.4* se encuentra detallado este bloque.

2.2.2. Administración del servicio

La administración del servicio contiene toda la lógica para autenticar usuarios y empresas, y para obtener toda la información necesaria de los clientes.

La administración podría perfectamente formar parte del núcleo de IP Centrex, pero se ha separado a causa de su gran complejidad y grandes contenidos.

El núcleo se comunica con la administración mediante una interfaz Java. Ésta interfaz permitirá obtener:

- la empresa, ya sea por el teléfono o por el nombre de la empresa
- los productos y servicios contratados por una empresa
- los usuarios registrados a una empresa
- si un usuario tiene permisos para un producto concreto

2.2.3. Interfaz de usuario (UI)

La parte de interacción con el usuario es la parte visual de IP Centrex. Gracias a la tecnología **Ajax [15]**, es posible mostrar una lista de usuarios con su estado y nombre únicamente proporcionando una lista XML (*ver apartado B.1*) desde el núcleo, sin tener que cargar una página entera. Y todo esto con tan solo un **Navegador Web**.

2.2.3.1. Ajax

AJAX, acrónimo de Asynchronous JavaScript And XML (JavaScript y XML asíncronos), es una técnica de desarrollo Web para crear aplicaciones interactivas. Utiliza una herramienta que proporcionan los exploradores, HTTP Push (*ver apartado B.3*) y la combinación de tres tecnologías ya existentes: HTML, JavaScript y XML.

El HTML **[16]** (HyperText Markup Language o Lenguaje de documentos con formato en hipertexto) es el lenguaje habitual utilizado para crear páginas web. Se utiliza para estructurar documentos mediante etiquetas y se muestra en modo hipertexto (documento digital que se puede leer de forma no secuencial). Mediante hipervínculos, el cliente puede navegar por los distintos documentos.

JavaScript **[17]** se define como un lenguaje interpretado para las páginas web. Para evitar incompatibilidades entre los distintos navegadores (Internet Explorer, Mozilla, etc.), el World Wide Web Consortium (w3c) **[12]** diseñó el estándar DOM (Document Object Model o Modelo de Objetos del Documento) **[19]**. Con este lenguaje se podrá ejecutar código en el cliente y tratar la información enviada por el servidor.

XML **[18]** es una tecnología de etiquetado de documentos, que facilita la organización de la información en un documento sin tener como objetivo la presentación. Esta tecnología se encuentra detallada en el *apartado B.1*.

Las aplicaciones Web habituales no sacan provecho de todo el ancho de banda utilizado. Para recibir información del servidor, el cliente tiene que enviar una petición de esa información (mediante un hipervínculo). La mayoría de veces, la información nueva que envía el servidor es mucho menor que la información total enviada.

Por ejemplo, para recibir la actualización de precios de una página, el cliente tiene que hacer una petición cada vez y posteriormente se envía la página completa, aunque el único cambio podría ser un número.

Con las aplicaciones diseñadas con AJAX se puede enviar información al cliente con una única petición al servidor. Mediante XML se envía la información que el cliente deberá procesar, de esta manera se distribuye la carga del servidor de procesar la información al cliente. Mediante JavaScript y DOM se procesa la información y se modifica el contenido de la página que se está visualizando. Como resultado, el cliente hará una sola petición y cada cierto tiempo recibirá una actualización de la página visualizada.

2.2.3.2. Navegador Web

Para poder usar toda la interfaz gráfica del servicio, tan sólo se necesita un navegador de Internet con conexión al servidor donde se encuentra el Ajax. En cuanto el usuario se registra introduciendo el usuario, la contraseña y el dominio (o empresa) al que pertenece, se le hace una redirección a una página donde irá visualizando el estado de todos los usuarios registrados en su mismo dominio. En este modo podrá realizar acciones sin necesidad de marcar con el teléfono.

Ilustración 2.4. Ejemplo de Interfaz de Usuario

En el *apartado 3.4.1* se detalla la implementación del servicio Llamada por UI.

2.3. Bloque HTTP Servlet – SIP Servlet

Utilizando la aplicación Tomcat (*ver apartado I.B.4*), en un mismo contexto se haya los dos servlets funcionando (*ver apartado I.B*). Así ambos comparten variables y se pueden comunicar.

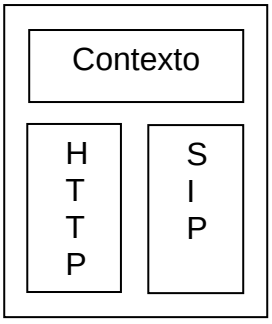


Figura 2.6. HTTP Servlet y SIP Servlet

Ni el SIP Servlet ni el HTTP Servlet tienen conocimiento de los estados de las llamadas o los usuarios, así como el bloque JCC no tiene conocimiento de cómo enviar o recibir mensajes SIP. Mediante una relación de todas las sesiones establecidas, ambos bloques pueden trabajar juntos.

2.3.1. HTTP Servlet

Toda comunicación entre el bloque HTTP Servlet – SIP Servlet y el bloque JCC se realiza mediante este HTTP Servlet y mediante peticiones y respuestas HTTP. Las peticiones enviadas por el bloque JCC se denominan **acciones** y las respuestas del HTTP Servlet se denominan **eventos**.

En *la Figura 2.7* se muestra cómo la interfaz Java Action, hereda de la interfaz Message, de esta manera tanto las acciones como los eventos poseen los métodos de Message:

```
public int getMessageType();
public String getAttribute(String attributeName);
….
```

2.3.1.1. Acción

Petición HTTP enviada por el bloque JCC para indicar la ejecución de un proceso.

- Un ejemplo podría ser la acción para crear una nueva llamada, dentro del cuerpo de la petición se enviará información necesaria para establecer la nueva llamada.
- Otro ejemplo seria el de responder a una llamada entrante

2.3.1.2. Evento

Respuesta HTTP enviada por el bloque del HTTP Servlet para indicar la recepción de un tipo de mensaje. No es condición que se haya recibido una acción para enviar un evento. Los eventos que no responden a una acción se enviarán por el **canal asíncrono**. Los que sí responden a una acción, se enviarán por el **canal síncrono**, creado al enviar la petición o acción.

- Un ejemplo de un evento que responde a la acción de crear una nueva llamada podría ser el de destino sonando (180 Ringing).
- Un ejemplo de evento que no responde a una acción podría ser el de recepción de un 200OK. Se envía cuando un usuario acepta establecer la llamada, momento que el bloque JCC desconoce.

2.3.1.3. Canal Asíncrono

El objetivo de este canal asíncrono es el de poder enviar desde el bloque HTTP Servlet – SIP Servlet al bloque JCC, información (eventos ocurridos) que no responden a ninguna acción concreta.

El canal asíncrono se crea haciendo que el thread (*ver apartado I.B.6*) del bloque JCC, que espera la respuesta del bloque de servlets, no termine nunca. De este modo cada vez que recibe una respuesta (evento) realiza los procesos necesarios y vuelve a esperar a recibir otra respuesta. El bloque de servlets por su parte se guarda la petición para los eventos producidos sin previa acción, como respuestas a esa primera petición.

2.3.1.4. Canal Síncrono

Es el canal HTTP habitual. Se envía una petición y se espera una respuesta a esa petición, una vez se ha recibido la respuesta se cierra el canal en ambos bloques y continúan los hilos de ejecución en ambos bloques.

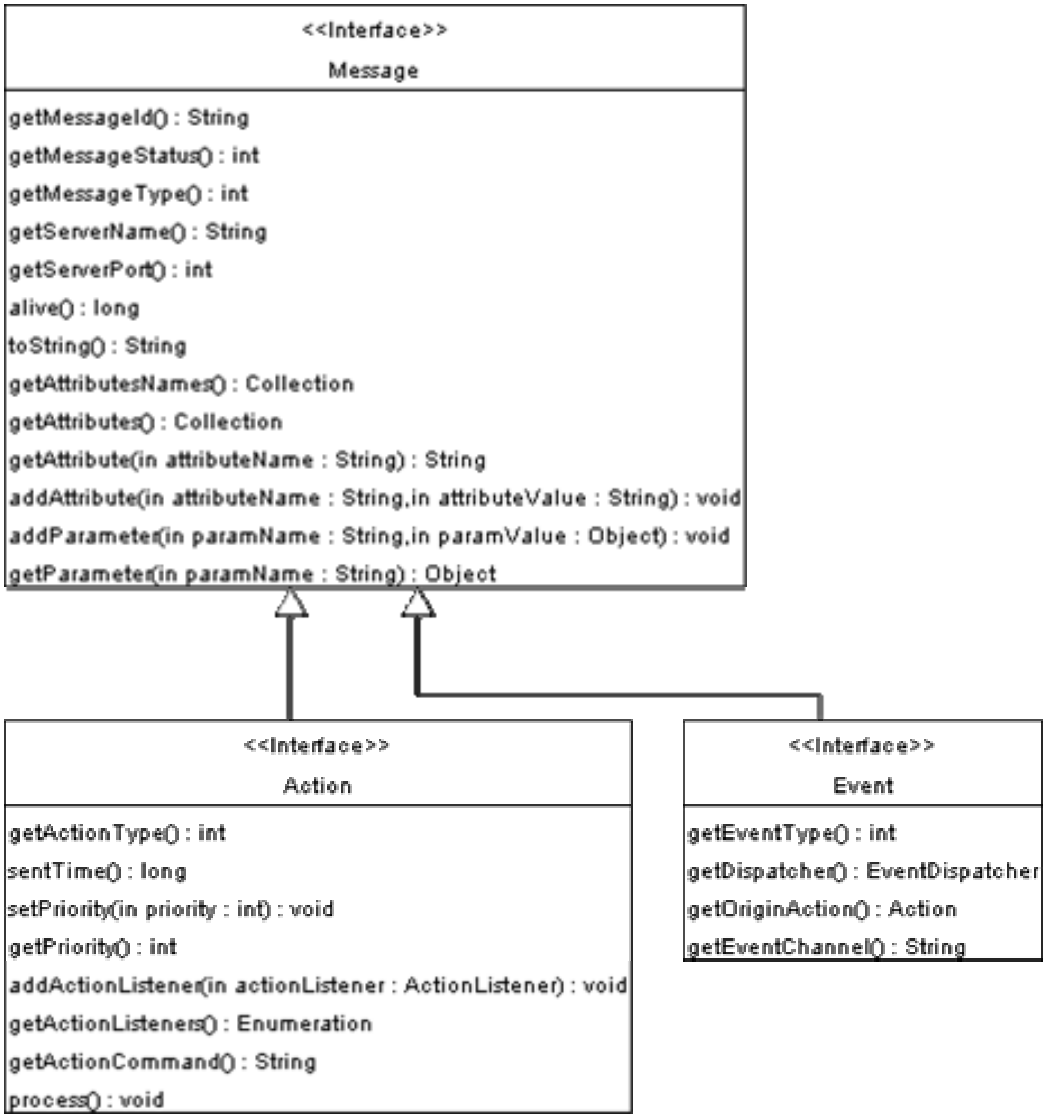


Figura 2.7. Diagrama de clases de las acciones y los eventos del núcleo

2.3.2. SIP Servlet

En IP Centrex, para poder establecer cualquier comunicación VoIP (*ver apartado 1.1*) se utiliza SIP *(ver apartado 1.2.2*). Pero para intercomunicar todos los procesos que pertenecen a SIP de los procesos del bloque JCC se utiliza el HTTP Servlet (*ver apartado B.3*) en el mismo contexto que el SIP Servlet. Consecuentemente si se recibe cualquier mensaje SIP por el SIP Servlet, mediante el HTTP Servlet se hará llegar un evento al bloque JCC. E inversamente si se recibe una acción del bloque JCC a través del HTTP Servlet, se traducirá en uno o más mensajes enviados por el SIP Servlet.

Cuando el servlet recibe un mensaje SIP, indica al HTTP Servlet que envía el evento junto con información necesaria para los procesos que se llevaran a cabo en el bloque JCC.

- Un ejemplo podría ser que al recibir un INVITE se envía un evento de nueva llamada y se le añade un identificador de sesión, el call-ID de la sesión SIP (*ver apartado 1.2.2*) y la dirección de origen y de destino.

2.4. Bloque Java Call Control – JaiCC

Este bloque se encarga de todo lo relacionado con la gestión de las llamadas, de los servicios y de la interacción con la interfaz de usuario (*ver apartado 2.2.3*). La lógica de este bloque será la que permitirá que un usuario pueda llamar a otro o hacer una transferencia de la llamada. A través del bloque HTTP Servlet – SIP Servlet podrá enviar y recibir mensajes SIP para el establecimiento de las diferentes llamadas. La principal ventaja de este bloque es su alto nivel. Las instrucciones para establecer una llamada se pueden minimizar a:

```
JccConnection con = call.createConnection (parámetros para crear la llamada con
        un usuario);
con.routeConnection (true);
```

La API (*ver anexos apartado B.5*) del grupo JAIN™, API JCC 1.0 **[13]** es la que proporciona el nivel de abstracción. La implementación de esta API es tarea y propiedad de la empresa que la utilice, de esta manera cualquier empresa puede utilizar esta API pero cada una tendrá sus procesos internos. Así el método routeConnection () de JccConnection en la implementación para IP Centrex (JaiCC, *ver apartado 2.4.2*) deberá, por lo menos, enviar una acción al bloque HTTP Servlet – SIP Servlet para establecer la llamada, indicando que desea enviar un INVITE (*ver apartado 1.2.2.2*) al destinatario de la llamada.

Del mismo modo si el bloque JCC recibe un evento de llamada entrante y desea contestar simplemente deberá hacer:

```
event.getConnection ().answer ();
```

El método answer () de JccConnection se habrá implementado para que, por lo menos, envíe una acción al bloque de Servlets indicando el envío de un 200OK (*ver apartado 1.2.2.2*).

2.4.1. Api JCC

Varias compañías pertenecientes al grupo JAIN™ crearon la API Java Call Control **[13]** para abstraer todo protocolo de red (SIP, H.323, ISUP, INAP, etc. *Ver anexos apartado A*) realizando la apropiada implementación.

La API JCC es una interfaz para controlar todos los posibles procesos de una sesión (crear, monitorizar, controlar, manipular o cerrar), facilitando así, la comunicación entre dos o más participantes y otros elementos de la red.

La abstracción de todo proceso permite utilizar la implementación de JCC en cualquier plataforma que soporte la API JCC. Así se puede reutilizar código, contratar a otra empresa para que realice la implementación de la API mientras se dedican recursos en utilizarla o incluso comprar la implementación para su utilización.

JCC define cuatro objetos básicos: Provider, Address, Call y Connection. Un Provider y una Address son estáticos (aunque quien implemente la API puede decidir que se creen dinámicamente), mientras que una Call y una Connection

son creados dinámicamente para cada llamada, por un Provider y una Call respectivamente.

- Un Provider proporciona la creación de llamadas y la obtención de la información necesaria para saber el estado de cada llamada
- Una Address es un terminal lógico (dirección SIP, dirección IP, etc.)
- Una Call representa una llamada en sí y puede mantener dos o más terminales (Address) unidos. En *la Figura 2.11 s*e representan los estados posibles para una Call
- Una Connection representa la relación que existe entre una Call y una Address. Se utiliza para saber en detalle el estado de cada participante en la llamada. En *la Figura 2.10* se observan los distintos estados por los que puede pasar una Connection

En *la Figura C.1* se observa la relación que existe entre una JaiCCConnection, una JaiCCCall y un JaiCCProvider. Como indica el diagrama, un JaiCCProvider guarda varias JaiCCCall y las crea mediante el método

```
public JccCall createCall()
```

implementado de la API JCC.
El mismo procedimiento ocurre con una JaiCCCall, ya que contiene varias JccConnection y las crea mediante el metodo

```
public JccConnection createConnection(String targetAddress, String
        originatingAddress, String originalCalledAddress, String
        redirectingAddress)
```

Aunque sea JaiCC y no JCC, las relaciones son las mismas debido a que JaiCC es la implementación de JCC para IP Centrex (*ver apartado 2.4.2*).

La API de JCC utiliza dos patrones de diseños definidos para Java, el patrón de la Factory y el de Java Listeners. El patrón Factory, se utiliza con una PeerFactory que proporciona un objeto definido por la API de JCC, un Peer. Un Peer se define según Java como “una implementación particular de una interfaz Java o API que depende de la plataforma”, por ejemplo, JaiCCPeer es un Peer de JCC para IP Centrex.

Otro de los patrones que utiliza la API de JCC son los Java Listeners. Estos Listeners permiten informar de eventos ocurridos a un objeto que lo haya solicitado. Por ejemplo, en JCC cuando llega una llamada entrante, el objeto Connection propaga un evento ConnectionEvent hacia los Listeners que tenga suscritos para informarles del cambio.

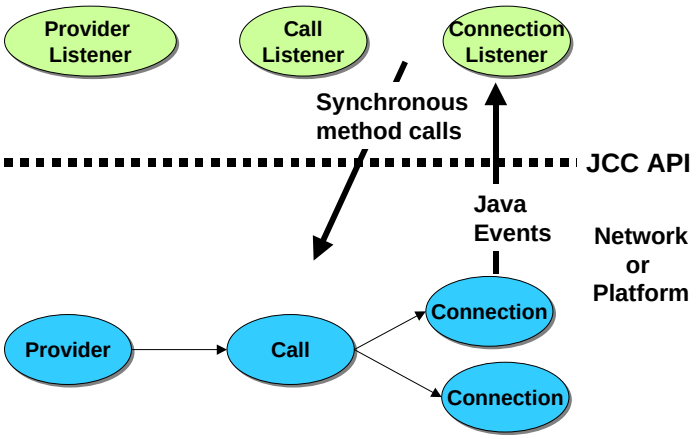


Figura 2.8. Patrón de diseño utilizando Java Listeners

En *la Figura 2.9* se muestra como un Connection Listener es al mismo tiempo un Call Lister (hereda). Un evento ocurrido en una Connection se propagará por todos los Connection Listeners. Entonces si un objeto quiere recibir los Connection Events de todas las conexiones de una Call, solo tendrá que suscribirse a la Call como Connection Listener. Si solo quiere recibir los eventos de una Connection concreta deberá suscribirse únicamente a esa Connection como Connection Listener.

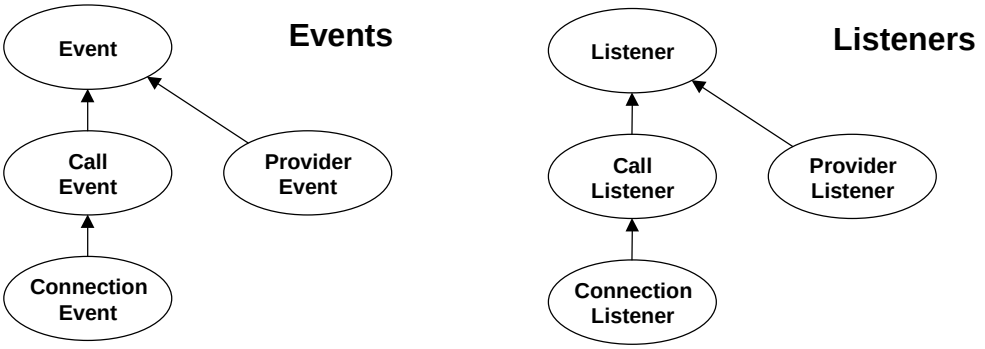


Figura 2.9. Diagrama de herencias de los JCC Event y Listeners

2.4.2. JaiCC

JaiCC es la implementación de la API JCC **[13]**, propietaria de VozTelecom (*ver apartado 1.1*). Esta implementación está diseñada para trabajar con SIP **[2]**, algunos métodos de la API JCC se traducen directamente en mensajes SIP (mediante el bloque HTTP Servlet – SIP Servlet, *ver apartado 2.3*), como por ejemplo el answer ().

Se ha implementado un Peer por defecto, JaiCCPeer, que se obtiene mediante JaiCCFactory. JaiCCPeer permite crear objetos JaiCCProvider y se crearán tantos JaiCCProviders como empresas (empresas o particulares) hayan contratado el servicio IP Centrex, diferenciados por un identificador.

Una vez obtenidos todos los JaiCCProviders necesarios se crea un servicio que extienda de JaiCCConnectionListener (implementación de JccConnectionListener) que contendrá toda la lógica de un servicio.

Por ejemplo para el servicio Click2Dial (*ver apartado 1.4.2.1*) se crea un JaiCCConnectionListener y cada vez que el usuario presiona en un hipervínculo se crea una nueva conexión se le hace un routeConnection () y en cuanto esta conexión se encuentra en estado Connected llega un evento al método connectionConnected. Es entonces que al hacer routeConnection (true) de la segunda conexión, el destinatario que recibe su INVITE.

Gracias al patrón de Listener se pueden implementar tantos servicios como se desee. Cada vez que una JccConnection cambia de estado se notifica a todos los objetos JccConnectionListener que tenga suscritos. Como cada Listener tiene una serie de métodos que se ejecutarán según el estado al que haya cambiado el JccConnection, se podrán ejecutar las instrucciones necesarias según el estado de las conexiones.

2.4.3. Diagramas de clase

En *la Figura C.2* se puede observar el diagrama de las clases JaiCCConnection y JaiCCCall. Estas dos clases son la implementación de la API JCC (JccConnection y JccCall respectivamente) que se ha diseñado para IP Centrex.

En *la Figura C.3* se plasma la implementación de dos servicios, el IpcClick2Call y el IncommingCall. La interfaz Service solo define un método abstracto, que obliga a que cualquier servicio implemente dicho método:

```
public abstract boolean hasEvent(JccConnectionEvent event);
```

DefaultService es una clase abstracta, que implementa esta interfaz. Tanto DefaultService como todas las clases que hereden de ella implementan la interfaz JaiCCConnectionListener. DefaultService implementa todos los métodos de JaiCCConnectionListener pero en cada uno sólo muestra un mensaje. El servicio diseñado (por ejemplo IpcClick2Call) implementará los métodos de JaiCCConnectionListener que le interesen. Los otros métodos se ejecutarán igual cuando se produzca el evento pero sólo se mostrará el mensaje. Por ejemplo, en el caso de IncommingCall implementa

```
public void connectionAlerting(JccConnectionEvent event)
```

debido a que tiene que tratar las llamadas entrantes, IpcClick2Call no, debido a que ningún UA realizará un intento de establecer una llamada (*ver apartado 1.4.2.1*).

La Figura C.4 muestra dos clases, ServiceManager y JccClientImpl. Service Manager será quien reciba todos los eventos producidos en IP Centrex del tipo connectionAlerting y únicamente mirará si hay algún servicio IncommingCall (*ver apartado 1.4.2.2*) tratando ese evento, sino creará uno para que trate ese evento. La clase ServiceManager nos será útil para administrar los eventos y para ejecutar alguna instrucción cada vez que llega un evento, ya que se ejecuta el método:

```
private void execute ()
```

Por ejemplo, para cada evento se actualiza la Interfaz de Usuario (*ver apartado 2.2.3*).
JccClientImpl, se podría considerar el motor de la parte JCC, se encarga de suscribir a los diferentes JaiCCProviders (cada empresa tiene su Provider y JccClientImpl los suscribe al bloque SIP/HTTP Servlets para que reciban eventos asíncronos, *ver apartado 2.3*). También es la interfaz donde ataca la parte UI (ver apartado 2.2.3), las acciones de los usuarios vía Web son recibidas en el método:

```
public void commandPerformed(Command cmd)
```

2.4.4. Diagrama de estados de JCC

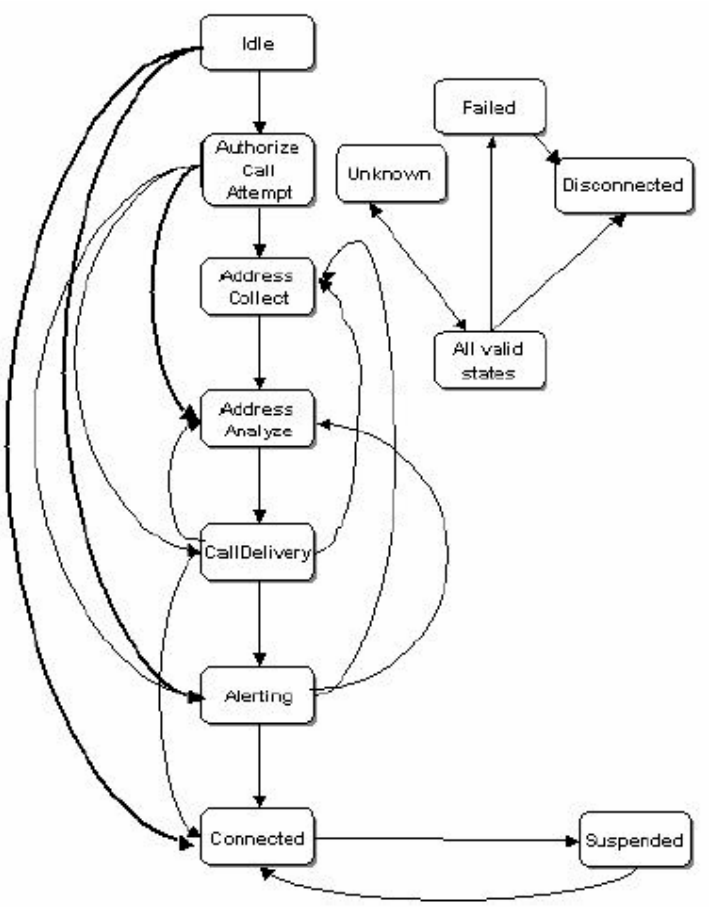


Figura 2.10. Flujo de los estados de JccConnection

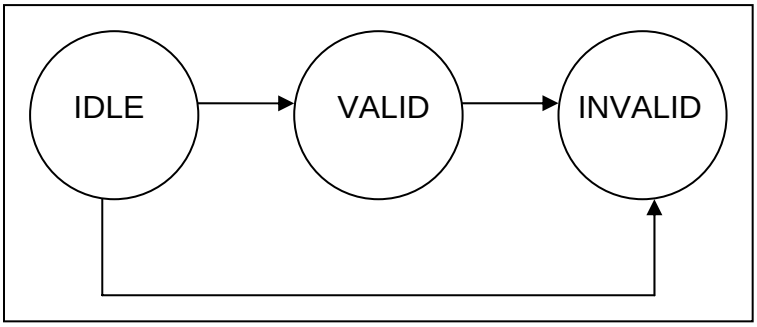


Figura 2.11. Flujo de los estados de JccCall

CAPÍTULO 3. Implementación

3.1. Escenario de trabajo

Para llevar a cabo la implementación de IP Centrex se va a utilizar un escenario de trabajo donde vamos a encontrar los siguientes elementos:

- IP Centrex
  - WeSIP (HTTP Servlet – SIP Servlet)
  - JCC / Interfaz de Usuario
- Proxy SIP (SER)
- RTP Proxy
- Servidores de media

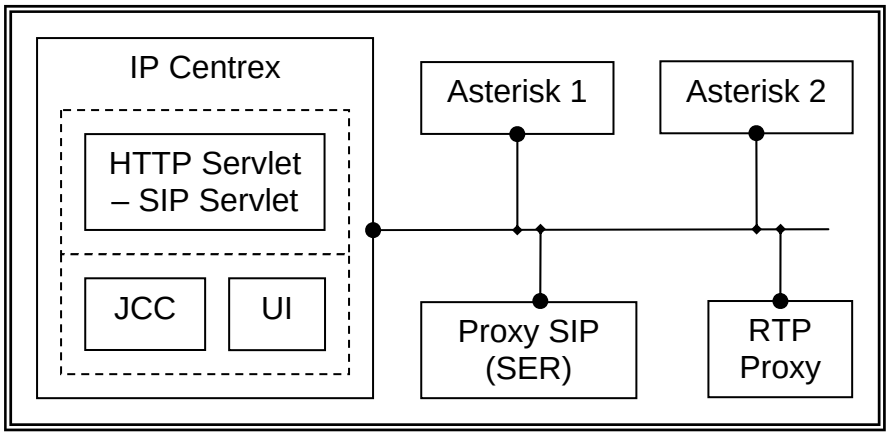


Figura 3.1. Escenario de IP Centrex

3.2. Elementos en el escenario

A continuación se detallan las características de los dispositivos utilizados en el escenario de trabajo.

3.2.1. SER

SER, acrónimo de SIP Express Router **[24]**, es un servidor SIP (*ver apartado 1.2.2.1*) gratuito, configurable y de alto rendimiento. Puede actuar como registrador, proxy o servidor de redirección de SIP. Como registrador, responde a los mensajes SIP REGISTER pudiendo pedir autenticación y registrando el usuario en una base de datos. Como proxy puede enrutar los mensajes SIP hacia otra red. Y como servidor de redirección puede simplemente redireccionar los mensajes hacia otro destino. Entre otros servicios se puede destacar su administración vía web y monitorización del estado del servidor.

Para IP Centrex se utiliza SER para registrar los usuarios con su extensión y así enviarles los pertinentes mensajes SIP.

En el *apartado 3.3.1* se detalla el uso de un SER para IP Centrex.

3.2.2. WeSIP

WeSIP es el nombre comercial del servidor de aplicaciones convergentes de VozTelecom. Según VozTelecom "*Hoy en día, los Proveedores de Servicio pueden crear aplicaciones convergentes y servicios de multimedia a través de servidores de aplicaciones basadas en SIP de una forma facil, elegante y poco costosa. La clave para hacerlo es el protocolo SIP y arquitecturas de telecomunicaciones IP, como IMS. El servidor de aplicaciones WeSIP es la propuesta de VozTelecom para el desarrollo de aplicaciones y su despliegue ya sea con las premisas y red del Proveedor de Servicio como dentro de la plataforma SIP de VozTelecom*" **[23]**.

WeSIP está diseñado para que pueda actuar como UAC, UAS, B2BUA o proxy dentro de cualquier entorno SIP, permitiendo así contener una amplia gamma de aplicaciones. Por ejemplo, VoIP, compartir archivos, etc. Utilizar al mismo tiempo un HTTP Servlet permite una convergencia entre aplicaciones SIP y HTTP.

IP Centrex está diseñado especialmente para ser desplegado en el servidor de aplicaciones WeSIP de VozTelecom. La convergencia de HTTP/SIP soportará todo el núcleo de IP Centrex. Aún así, el diseño de IP Centrex nos permite separar las distintas partes: el bloque de servlets sobre WeSip en un servidor y el bloque JCC y la UI en otro servidor distinto.

3.2.3. RTP Proxy

Un RTP Proxy permite enviar y recibir flujos de media (*ver apartado 1.2.3*) entre diferentes redes. De este modo se obtiene un plano de gestión en paralelo con un plano de media. Como ya se ha comentado, IP Centrex está diseñado para actuar como mediador entre dos usuarios, cuando detecta que los dos usuarios se encuentran listos para hablar utiliza el RTP Proxy para juntar las dos conversaciones (diseño B2BUA, *ver apartado 2.1.4*).

Un RTP Proxy tiene la capacidad de recibir flujos RTP por un puerto y retransmitirlo por otro puerto. También puede copiar el flujo de un puerto a otro puerto, obteniendo dos puertos que envían el mismo flujo RTP.

En el *apartado 3.3.2* se detalla el uso de un RTP Proxy para IP Centrex.

3.2.4. Asterisk

Digium™ ha creado una PBX (*ver apartado 1.3*) basada completamente en software, Asterisk **[25]**. Funciona sobre Linux, BDS y MacOSX. Asterisk puede operar con casi todos los elementos telefónicos basados en estándares utilizando una infraestructura mínima. Proporciona entre muchos otros servicios Voicemail, que puede servir como buzón de voz para almacenar mensajes, cuando los usuarios de IP Centrex no se encuentren activos. Y da la posibilidad de reproducir música a través de un flujo RTP, se usará para la música en espera.

No necesita ningún hardware adicional, con una tarjeta de red en un Linux, ya se puede levantar un Asterisk.

Asterisk fue creado originalmente por Mark Spencer de Digium, Inc. Todo el código ha sido desarrollado y testeado por programadores en open source (código abierto).

En el *apartado 3.3.3* se detalla el uso de un Asterisk para IP Centrex.

3.3. Dispositivos de IP Centrex

IP Centrex utiliza una serie de dispositivos que son óptimos para un proceso en concreto. Se utiliza un **SER**, un **RTP Proxy** y un **Asterisk**.

3.3.1. SER

IP Centrex utiliza un SER (*ver apartado 3.2.1*) para registrar los usuarios. Todas las peticiones SIP de establecimiento de la sesión (ver apartado 1.2.2) pasan por el SER y se reenvían a la extensión indicada. Cuando éste recibe una petición, en el TO del INVITE se encuentra la extensión del usuarios al que va destinada la llamada, así sabe a quien reenviar el mensaje.

IP Centrex envía las peticiones INVITE hacia el SER para que éste se las redireccione a los usuarios oportunos, de esta manera se optimiza el proceso de registrar y comunicarse con los usuarios. IP Centrex sólo se preocupa de indicar al SER que desea establecer una sesión con el usuario destino.

Se utiliza otro servicio del SER, la capacidad de redireccionar toda petición hacia IP Centrex. El resultado es que el usuario enviara una petición hacia el SER pero en realidad será el IP Centrex quien procese esa petición, así IP Centrex podrá estar en una red privada con comunicación exclusiva con el SER.

3.3.2. RTP Proxy

Uno de los elementos más utilizados, un Proxy para RTP. Necesario en cada una de las llamadas establecidas por IP Centrex.

Las llamadas se gestionarán con el diseño B2BUA, pero para los flujos de media (*ver apartado 1.2.3*), que se separan en el plano de media, se utiliza el RTP Proxy.

Con el servicio Click2Dial, el bloque de servlets enviará un INVITE al UA1 sin SDP (*ver apartado 1.2.3.2*). Cuando el mismo bloque recibe el 200OK se configura el RTP Proxy para que abra un puerto por donde enviará el flujo de media al UA1 (indicado en el 200OK). Al mismo tiempo se abre otro puerto por donde el UA1 tendrá que enviar su flujo, que junto con otros parámetros se guardan en una configuración SDP. Ésta configuración SDP se envía con un ACK al UA1.

En este momento el UA1 enviará su flujo a un puerto del RTP Proxy y éste, por otro puerto, enviará el flujo hacia el UA1. El flujo de un UA al RTP Proxy se ha denominado UpStream y el flujo del RTP Proxy al mismo UA DownStream.

Una vez la sesión con el UA1 está establecida, se establece la llamada con el UA2 enviando un INVITE sin SDP, siguiendo el mismo procedimiento para establecer el UpStream y el DownStream.

Finalmente, cuando las dos llamadas se encuentran establecidas y todos los canales de flujo (UpStream y DownStream) configurados, se juntan los flujos: el UpStream del UA1 con el DownStream del UA2 y el UpStream del UA2 con el DownStream del UA1. Este proceso se obtiene indicándole al RTP Proxy que copie el contenido de un puerto a otro puerto.

3.3.3. Asterisk

Un Asterisk es un software con muchas posibilidades y de hecho es una PBX por si mismo (*ver apartado 3.2.4*), pero de momento, para IP Centrex únicamente se utiliza una de sus posibilidades, música en espera.

Gracias al RTP Proxy se permite mantener un UpStream del Asterisk con música en espera (mp3 u otros formatos) y cuando se hace que una llamada esté en espera, se desacopla la llamada de la conversación y se la une al Asterisk. Este proceso se realiza haciendo que el RTP Proxy deje de copiar los puertos y uniendo el DownStream del usuario con el UpStream del Asterisk (*ver apartado 3.3.2*). Así se consigue que el usuario escuche el flujo que esta enviando el Asterisk, música en espera.

3.4. Servicios

Cada servicio tiene que extender de la clase DefaultService (*ver apartado 2.4.2*). Ésta clase implementa JccConnectionListener. Todo servicio implementará los métodos que necesite para cumplir su cometido. Si implementa

```
public void connectionAlerting (JccConnectionEvent event) {}
```

cada vez que un JccConnection, al que esté suscrito, cambie a CONNECTED, se ejecutará este método (*ver apartado 2.4*). En los anexos se puede observar el flujo de mensajes de cada servicio (*ver anexos apartado D*).

3.4.1. Llamada por UI (Click2Dial)

Con solo presionar en el hipervínculo *[Click2Dial]* de la interfaz de usuario se envía un evento a la JCC para que ejecute el servicio Click2Dial con dos parámetros, la extensión del origen (el usuario que ha hecho la petición de llamada) y el destino, ambos proporcionados por la UI. Una vez se ejecuta el servicio se envía un INVITE al origen para establecer una JaiCCConnection con el usuario que ha deseado establecer la llamada, si éste no contesta, se considerará el servicio terminado y el usuario destino no recibirá mensaje alguno.

En el momento que el usuario origen descuelga el teléfono, se envía otro INVITE (*ver apartado 1.2.2*) hacia la extensión destino para establecer una nueva JaiCCConnection. De nuevo, si el usuario final no desea establecer la comunicación se terminará el servicio y se notificará con un BYE al usuario originario de la llamada.

Al final, cuando los dos usuarios hayan aceptado las dos llamadas se establecerá la comunicación y podrán hablar hasta que uno de los dos cuelgue, cosa que provocará una notificación al otro usuario.

En *la Ilustración 3.1* se pueden observar los pasos del servicio Click2Dial cuando Roger se ha registrado y presiona sobre el [Click2Dial] de Óscar:

Ilustración 3.1. Ejemplo de servicio Click2Dial

1. Suena el teléfono de Roger y aparece cómo Ringing (Sonando)
2. En cuanto Roger contesta aparece como Busy (ocupado) y al instante suena el teléfono de Óscar, apareciendo como Ringing
3. Óscar responde al teléfono, aparecen los dos como Busy y ya se ha establecido la comunicación.

3.4.2. Llamada por terminal (IncommingCall)

El establecimiento de la llamada se hace con el diseño B2BUA (detallado en el *apartado 2.1.4*), primero establece la comunicación con el terminal que ha llamado y después la establece con el destino. El flujo de mensaje lo podemos observar en el ejemplo de *la Ilustración 2.2*.

3.4.3. Transferencia de llamada (TransferCall)

Gracias a la lógica B2BUA (detallado en el *apartado 2.1.4*), se establece una llamada con el origen otra con el destino inicial y una tercera llamada con el destino de la transferencia. De esta manera cuando el origen (la llamada entrante) está en espera, se cambia el flujo de voz que recibe (del destino inicial) por un flujo con un Asterisk. El Asterisk proporciona una herramienta para enviar una música de espera (*ver apartado 3.2.4*).

En estos momentos, el servicio de transferencia de llamada está en fase de implementación. El diseño se basa en crear dos JaiCCConnection con los dos primeros usuarios (por ejemplo, el empleado y la secretaria) y en cuanto la secretaria indique el destino de la transferencia (por ejemplo el jefe), se establecerá otra JaiCCConnection con el nuevo destino y se pondrá en espera a la secretaria (por ejemplo conectando el UpStream y el DownStream con un Asterisk, con música en espera), después se unirán el UpStream del empleado con el DownStream del jefe y el UpStream del jefe con el DownStream del empleado (*ver apartado 3.3.2*). En éste instante el empleado ha sido transferido y ya puede hablar con el jefe.

3.4.4. Captura de llamada (PickUP)

A través de la UI (detallado en el *apartado 2.2.3*) si se presiona sobre el hipervínculo [PickUP], se indicará al núcleo que se desea capturar esa llamada entrante. El bloque JCC deberá colgar el destino inicial para llamar al usuario que desea la captura. Al final, el origen estará conectado con el destino final (ha habido una captura de la llamada). En *la* Ilustración 3.1 podemos observar que cuando Óscar o Roger están sonando aparece el [PickUP] para capturar esa llamada entrante.

CAPÍTULO 4. PLANIFICACIÓN Y CONCLUSIONES

4.1. Pasado

4.1.1. Planificación del diseño de IP Centrex

ID	Task Name	Start	Finish	Duration
1	Recopilación de toda la información disponible sobre SIP	16/09/2005	05/10/2005	14d
2	Estudio y comprensión de las herramientas que se estaban utilizando en el proyecto IP Centrex, ya que ya estaba en marcha	**05/10/2005**	**07/11/2005**	**24d**
3	Estudio y comprensión del diseño de IP Centrex que se estaba desarrollando	05/10/2005	**07/11/2005**	**24d**
4	En primera instancia se utilizó un SIP Servlet con el diseño de SIP Proxy (ver apartado 2.1.3)	05/10/2005	21/10/2005	13d
5	Estudio de la API de JCC (ver apartado 2.4.1) para abstraer una parte del SIP Servlet	24/10/2005	07/11/2005	11d
6	**Posible implementación de JCC en el SIP Servlet disponible. No es viable a causa del tiempo de dedicación**	07/11/2005	23/12/2005	35d
7	Reunión con VozTelecom. Se cambian todos los esquemas y diseños, ahora se trabajará con un B2BUA (ver apartado 2.1.4)	07/11/2005	07/11/2005	1d
8	Estudio y compresión de la implementación de JCC en un SIP Servlet realizado por VozTelecom	07/11/2005	18/11/2005	10d

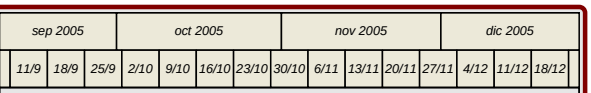


Ilustración 4.1. Diagrama Gantt de la planificación del diseño

4.1.2. Planificación de la implementación de IP Centrex

ID	Task Name	Start	Finish	Duration
1	Aprovechando la implementación de JCC que nos proporcionó VozTelecom, se empezó a implementar el primer servicio y la parte gráfica	**21/11/2005**	**09/12/2005**	**15d**
2	Implementación del servicio Click2Dial (ver apartado 3.3.1) con el nuevo diseño B2BUA y JCC	21/11/2005	09/12/2005	15d
3	Implementación de la UI (Interfaz de Usuario)	21/11/2005	09/12/2005	15d
4	Prueba piloto por separado	09/12/2005	09/12/2005	1d
5	Diseño del servicio Transferencia de Llamada	09/12/2005	20/12/2005	8d
6	Unir el núcleo con la UI	09/12/2005	20/12/2005	8d
7	Prueba piloto con el núcleo y la UI juntas	21/12/2005	21/12/2005	1d

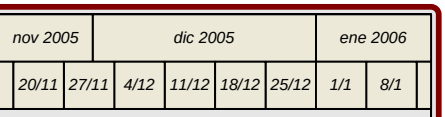


Ilustración 4.2. Diagrama Gantt de la planificación de la implementación

4.2. Conclusiones

Al terminar este TFC se dispone de un servicio completo (Click2Dial) y una comunicación completa entre el núcleo y la parte de UI.

Se está diseñando una modificación para la parte de la interfaz de usuario y terminando de pulir algunos detalles del núcleo. Al mismo tiempo también se va implementando el segundo servicio (Transferencia de llamada, *ver apartado 3.4.3*) y pensando como implementar un tercer servicio (PickUP, *ver apartado 3.4.4*).

El objetivo era diseñar e implementar una centralita de VoIP con al menos un primer servicio y se ha conseguido. Así que se pueden considerar logrados los objetivos.

Cabe destacar que este TFC es un proyecto real, se está llevando a cabo por un equipo de becarios y trabajadores de VozTelecom. Al colaborar una empresa privada en el proyecto, que en un futuro tiene previsto ponerlo a explotación, hay unos requisitos a cumplir. IP Centrex tiene que ser sobre todo robusto y escalable.

4.3. Estúdio de Ambientalización

Como todo proyecto, este Trabajo de Fin de Carrera ha tenido un seguimiento del impacto medioambiental al poner en práctica los diseños planteados. Aún siendo un proyecto de aplicación, no deja de tener consecuencias medioambientales en cuanto lo ponemos en práctica.

Primero, si tenemos en cuenta que la tecnología IP Centrex necesita más de un servidor (más de una máquina conectada día y noche), esto se convierte en energía consumida. Si la fuente de energía es renovable y no contaminante, el impacto ambiental en este punto no tiene mayores consecuencias. Pero si se da el caso de una obtención de energía por medio de energías no renovables, contribuimos a los efectos que estas energías provocan, contaminación atmosférica, lluvia ácida, efecto invernadero, etc.

En segundo lugar, si analizamos la cantidad de información que se transmite para lograr una comunicación clara mediante IP Centrex, se necesita una buena infraestructura de telecomunicaciones. Hoy en día la red de telefonía pública está más que extendida, pero teniendo en cuenta el ancho de banda que proporciona, está cada vez más desfasada. Lo que nos lleva a la necesidad de una infraestructura de más capacidad, pero el impacto medioambiental que esto supone es bastante grande. Para que las nuevas tecnologías de telecomunicaciones puedan llegar a cualquier punto, antes tiene que haber una industrialización y una contaminación visual, ya sea en torres de repetición, o cableado.

Finalmente, aun conociendo los impactos que pueden producir las nuevas tecnologías y en concreto IP Centrex, si se establecen una buenas políticas de ramificación de las comunicaciones, de reciclaje (para el hardware desechado) y de obtención de la energía, el proyecto IP Centrex es completamente viable.

BIBLIOGRAFÍA

[REF 1] ITU-T Recommendation H.323v.4 "Packet-based multimedia communications systems", November 2000.

[REF 2] Especificaciones para SIP, RFC 3261 [en línea]: *Session Initiation Protocol* [Última Consulta: 12 de enero de 2006] Disponible en: <http://www.faqs.org/rfcs/rfc3261.html>

[REF 3] Especificaciones para RTP, RFC 3550 [en línea]: *RTP: A Transport Protocol for Real-Time Applications*. [Última Consulta: 12 de enero de 2006] Disponible en: <http://www.faqs.org/rfcs/rfc3550.html>

[REF 4] Especificaciones para RTCP, RFC 3550 [en línea]: *RTP Profile for Audio and Video Conferences with Minimal Control*. [Última Consulta: 12 de enero de 2006] Disponible en: <http://www.faqs.org/rfcs/rfc3550.html>

[REF 5] Especificaciones para SDP, RFC 2327 [en línea]: *SDP: Session Description Protocol*. [Última Consulta: 12 de enero de 2006] Disponible en: <http://www.ietf.org/rfc/rfc2327.txt?number=2327>

[REF 6] Especificaciones para RTSP, RFC 2326 [en línea]: *Real Time Streaming Protocol* [Última Consulta: 12 de enero de 2006] Disponible en: <http://www.ietf.org/rfc/rfc2326.txt?number=2326>

[REF 7] Información sobre CGI (en línea). [Última Consulta: 12 enero de 2006]. Disponible en: <http://leo.worldonline.es/joferrer/tweb/tutoriales/cgi/cgi.html>

[REF 8] Especificaciones para SRTP, RFC 3711 [en línea]: *The Secure Real-time Transport Protocol* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.faqs.org/rfcs/rfc3711.html>

[REF 9] Información sobre SAP [en línea]: *Service Advertising Protocol (SAP)* [Última Consulta: 17 de enero de 2006] Disponible en: <http://msdn.microsoft.com/library/default.asp?url=/library/en-us/randz/protocol/sap.asp>

[REF 10] Información sobre PSTN [en línea]: *Public Switched Telephone Network* [Última Consulta: 17 de enero de 2006] Disponible en: <http://en.wikipedia.org/wiki/Public_Switched_Telephone_Network> <http://es.wikipedia.org/wiki/Red_Telef%C3%B3nica_Conmutada (en castellano)>

[REF 11] Información sobre softphones [en línea]: *EyeBeam y SIPPS* [Última Consulta: 17 de enero de 2006] Disponible en: EyeBeam:<http://www.xten.net/index.php?menu=eyeBeam> SIPPS:<http://www.nero.com/sippstar/esp/User.html>

[REF 12] Información sobre World Wide Web Consortium [en línea]: *World Wide Web Consortium* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.w3.org/>

[REF 13] API JAIN™ JCC [en línea]: *JSR 21: JAIN™ JCC Specification* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.jcp.org/en/jsr/detail?id=21>

[REF 14] Información sobre Apache [en línea]: *The Apache Software Fundation* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.apache.org/>

[REF 15] Información sobre AJAX [en línea]: *Ajax (programming)* [Última Consulta: 17 de enero de 2006] Disponible en: <http://en.wikipedia.org/wiki/Ajax_%28programming%29> <http://es.wikipedia.org/wiki/AJAX (en castellano)>

[REF 16] Información sobre HTML [en línea]: *HyperText Markup Language (HTML) Home Page* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.w3.org/MarkUp/>

[REF 17] Información sobre JavaScript [en línea]: *Client-Side JavaScript Reference* [Última Consulta: 17 de enero de 2006] Disponible en: <http://docs.sun.com/source/816-6408-10/contents.htm>

[REF 18] Información sobre XML [en línea]: *Extensible Markup Language (XML) 1.0 (Third Edition)* [Última Consulta: 17 de enero de 2006] Disponible en: < http://www.w3.org/TR/REC-xml/>

[REF 19] Información sobre DOM [en línea]: *Document Object Model (DOM) Level 1 Specification* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.w3.org/TR/1998/REC-DOM-Level-1-19981001/>

[REF 20] Información sobre SGML [en línea]: *Overview of SGML Resources* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.w3.org/MarkUp/SGML/>

[REF 21] Información sobre Servlets [en línea]: *Java Servlet Technology* [Última Consulta: 17 de enero de 2006] Disponible en: <http://java.sun.com/products/servlet/>

[REF 22] MiCentralita de VozTelecom [en línea]: *VozTelecom – MiCentralita.net* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.vozip.com/>

[REF 23] WeSip de VozTelecom [en línea]: *WeSIP* [Última Consulta: 17 de enero de 2006] Disponible en: <http://ssl.voztele.com:28080/wesip.htm>

[REF 24] Información sobre SER [en línea]: *iptel.org SIP Express Router* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.iptel.org/ser/>

[REF 25] Información sobre Asterisk [en línea]: *Asterisk™ - the Open Source PBX!* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.asterisk.org/about>

[REF 26] Información sobre Sun [en línea]: *Sun Microsystems* [Última Consulta: 17 de enero de 2006] Disponible en: <http://www.sun.com/>

[REF 27] Información sobre Java Community Process [en línea]: *The Java Community Process (SM)* [Última Consulta: 17 de enero de 2006] Disponible en: <http://jcp.org/en/home/index>

[REF 28] Información sobre PUSH y PULL con HTTP [en línea]: An exploration of dynamic documents [Última Consulta: 17 de enero de 2006]. Disponible en: <http://wp.netscape.com/assist/net_sites/pushpull.html>

Documentación referente a JCC y SIP [documento PDF en línea]: *Java™ Call Control (JCC) and Session Initiation Protocol (SIP)*. [Última Consulta: 12 de enero de 2006]. Disponible en: <http://www.argreenhouse.com/papers/jlbakker/JCC-SIP-final-e84-b_12_3096.pdf>

Información sobre VoIP y Telefonía IP [en línea]: *Internet Communications & Voice over IP Resources* [Última Consulta: 12 de enero de 2006]. Disponible en: <http://www.acertavoice.com/>

Información sobre un B2BUA [en línea]: *B2BUA: Enabling Class 5 Capabilities in SIP Designs* [Última Consulta: 17 de enero de 2006]. Disponible en: <http://www.commsdesign.com/story/OEG20030909S0008>

Tutorial para servlets y jsp [en línea]: *Servlets and JavaServer Pages (JSP) 1.0: A Tutorial* [Última Consulta: 17 de enero de 2006]. Disponible en: <http://www.apl.jhu.edu/~hall/java/Servlet-Tutorial/>

Información sobre Tomcat [en línea]: *Apache Tomcat* [Última Consulta: 17 de enero de 2006]. Disponible en: <*http://tomcat.apache.org/*>

Algunos de los conceptos definidos, se han obtenido de una enciclopedia en línea [en línea]: *Wikipedia – La enciclopedia libre* [Última Consulta: 17 de enero de 2006]. Disponible en: <*http://es.wikipedia.org*>

Información sobre Tomcat [en línea]: *Apache Tomcat* [Última Consulta: 17 de enero de 2006]. Disponible en: <*http://tomcat.apache.org/*>

Información sobre la empresa VozTelecom [en línea]: *VozTelecom* [Última Consulta: 17 de enero de 2006]. Disponible en: <http://www.vozip.com/>

AGRADECIMIENTOS

Este proyecto no se podría haber llevado a cabo sin la empresa colaboradora VozTelecom. A mis compañeros de trabajo y amigos Javier López, Oscar Santillana y Elias Baixas. Deseo extender un especial agradecimiento a quien me ha guiado y ha volcado una gran paciencia en este proyecto, Antonio Abajo. En especial agradecer el apoyo, la orientación, la dedicación y sobretodo la confianza depositada, a mi tutor y también compañero Toni Oller.

ANEXOS

A. Acrónimos

H.323 Recomendación del ITU-T (International Telecommunication Union), que define los protocolos para proveer sesiones de comunicación audiovisual en cualquier paquete de la red.

SIP Siglas de Session Initiation Protocol (Protocolo de Inicio de Sesión). Protocolo de señalización que se utiliza para iniciar sesiones multimedia interactivas entre usuarios de redes IP.

ISUP El ISDN User Part (Parte de usuario de la ISDN o RDSI) es parte de la señalización #7, la cual es usada para establecer la llamadas telefónicas en la PSTN.

INAP Acrónimo de Intelligent Network Application Part (Parte de Aplicación de la Red Inteligente) es un protocolo de señalización utilizado en la arquitectura de la red inteligente (pe. Servicios GSM).

ISDN Siglas de Integrated Services Digital Network (Red Digital de Servicios Integrados o también llamada RDSI). ISDN es un tipo de red de conmutación de circuitos, diseñada para permitir la transmisión de voz y datos sobre el cableado de cobre de la telefonía habitual, obteniendo una mejor calidad y altas velocidades que con los sistemas analógicos.

RTP Siglas de Real-time Transport Protocol (Protocolo de Transporte de tiempo Real). Es un protocolo de nivel de transporte utilizado para la transmisión de información en tiempo real como por ejemplo audio y video en una video-conferencia.

RTCP Siglas RTP Control Protocol (Protocolo de control para RTP). La función primaria es proporcionar información al origen de la calidad de servicio de la distribución de información.

SDP Siglas de Session Description Protocol (Protocolo de Descripción de Sesión). Está diseñado para describir las sesiones multimedia con el objetivo de anunciar sesiones, invitación de sesión y otras formas de inicio de una sesión de multimedia.

Streaming Término que describe una estrategia sobre demanda para la distribución de contenido multimedia a través del Internet.

RTSP Siglas de Real Time Streaming Protocol (Protocolo de streaming en tiempo real). Protocolo del nivel de aplicación para el control sobre el envío de información en tiempo real.

W3C El World Wide Web Consortium es una organización que crea estándares para la World Wide Web (o Telaraña Mundial, comúnmente llamada Internet).

CGI Un Common Gateway Interface (Pasarela de Interfaz Común) es una importante tecnología de la World Wide Web que permite a un cliente (explorador Web) solicitar datos de un programa ejecutado en un servidor Web.

SRTP Siglas de Secure Real-time Transport Protocol (Protocolo de Transporte Seguro en tiempo Real). Protocolo que proporciona confidencialidad, autenticación de mensajes y protección de la respuesta para el trafico RTP y el control de RTP (RTCP).

SAP Siglas de Service Advertising Protocol (Protocolo de Anuncio de Servicio). SAP se encuentra incluido en el protocolo IPX (Internetwork Packet Exchange, o Intercambio de Paquetes en una Red). SAP hace dinámico el proceso de añadir y quitar servicios en una red IPX.

PSTN Siglas de Public Switched Telephony Network (Red Telefónica Conmutada Publica) es una red de telefonía diseñada primordialmente para la transmisión de voz, aunque pueda también transportar datos, por ejemplo en el caso del fax o de la conexión a Internet a través de un módem acústico.

API Una API (del inglés Application Programming Interface - Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones de comunicación entre componentes software.

JAIN JAIN es una comunidad de al menos 80 empresas, especificando sobre 25 nuevas API que se centran en las redes IP y PSTN.

JCC Siglas de Java Call Control. JCC es una API de Java para crear, monitorizar, controlar, manipular y colgar sesione en un entorno convergente entre PSTN y conmutación de paquetes (Internet).

AJAX Acrónimo de Asynchronous JavaScript And XML (JavaScript y XML asíncronos). Técnica de desarrollo Web para crear aplicaciones interactivas mediante la combinación de tres tecnologías ya existentes: HTML, Document Object Model (DOM) junto con JavaScript y XML.

HTML El HTML, acrónimo de Hypertext Markup Language (lenguaje de etiquetaje de hipertexto), es un lenguaje de marcas diseñado para estructurar textos y presentarlos en forma de hipertexto, que es el formato estándar de las páginas Web.

SGML Siglas de Standard Generalized Markup Language (Lenguaje de etiquetaje Generalizado).

XML Acrónimo de eXtensible Markup Language (Lenguaje de etiquetaje extensible). Es un lenguaje informático de etiquetaje que deriva del lenguaje SGML y permite representar e intercambiar información entre ordenadores o programas, ya que organiza los datos de manera ordenada.

DOM Document Object Model (Modelo de Objetos de Documento) es una forma de representar documentos estructurados. Su finalidad es definir el conjunto de objetos que pueden componer documentos HTML (páginas Web) o XML, así como las estructuras que se definen dentro de él.

WeSip WeSIP es un contenedor convergente de HTTP y SIP Servlets, desarrollado por VozTelecom.

SER Acrónimo de SIP Express Router [24], es un servidor SIP gratuito, configurable y de alto rendimiento. Puede actuar como registrador, prosa o servidor de redirección de SIP.

NAT Acrónimo de Network Address Translation (Traducción de Direcciones de Red) es un estándar creado por la Internet Engineering Task Force (IETF), el cual utiliza una o más direcciones IP para conectar varios computadores a otra red (normalmente a Internet). Los computadores tienen normalmente una dirección IP no válida para Internet, privada, etc.

Proxy En inglés «apoderado» o «delegado», hace referencia a un programa o dispositivo que realiza una acción en representación de otro. La finalidad más habitual de esa representación es la de permitir el acceso a Internet a todos los equipos de una organización cuando sólo se puede disponer de un único equipo conectado, esto es, una única dirección IP.

Router El router (enrutador o encaminador) es un dispositivo hardware o software de interconexión de redes de ordenadores/computadoras que opera en la capa 3 (nivel de red) del modelo OSI. Este dispositivo interconecta segmentos de red o redes enteras.

Ethernet Norma o estándar (IEEE 802.3) que determina la forma en que los puestos de la red envían y reciben datos sobre un medio físico compartido que se comporta como un bus lógico, independientemente de su configuración física.

Token Ring Arquitectura de red desarrollada por IBM con topología lógica en anillo y técnica de acceso de paso de testigo. Cumple el estándar IEEE 802.5.

B. Otros Conceptos

B.1. XML

XML **[18]** son las siglas del inglés eXtensible Markup Language (lenguaje de marcado ampliable o extensible) desarrollado por el World Wide Web Consortium (W3C) **[12]**.

Es una versión simple del SGML **[20]**. Su objetivo principal es conseguir una página Web más semántica. Una de las principales funciones con las que nace XML sería suceder al HTML (*ver apartado 2.2.3.1*), separando la estructura del contenido y permitiendo el desarrollo de vocabularios modulares. Tiene otras aplicaciones entre las que destaca su uso como estándar para el intercambio de datos entre diversas aplicaciones o para archivos de configuración como Tomcat (*ver apartado B.4*).

Al igual que el HTML, se basa en documentos de texto plano en los que se utilizan etiquetas para delimitar los elementos de un documento. Sin embargo, XML define estas etiquetas en función del tipo de datos que está describiendo y no de la apariencia final que tendrán en pantalla o en la copia impresa, además de permitir definir nuevas etiquetas y ampliar las existentes.

Para IP Centrex se utilizará XML para enviar la lista de usuarios que están registrados con sus estados correspondientes a los usuarios que están utilizando un navegador. En el *apartado 2.2.3* se detalla la interacción con el usuario.

B.2. Servlets y CGI

Una Servlet **[21]** es una tecnología de Java que se ejecuta en un servidor Web y permite compilar páginas Web para que sean mostradas a los clientes. Cabe destacar que compila las páginas en tiempo real, cosa que permite recoger información del usuario y crear una página Web concreta para el cliente o cambiar la información de una página frecuentemente.

Otra posibilidad es utilizar CGI **[7]**. CGI es una interfaz que también permite interactuar con el cliente. Permite ejecutar scripts que reciben información de formularios Web rellenados por el cliente y utilizando esta información generar una respuesta específica y única para ése cliente en concreto.

La ventaja de los Servlets frente a los CGI es el modo de ejecución de los CGI. Un CGI se ejecuta una vez por cada petición que se recibe, un servlet se ejecuta una vez y se va accediendo a él por cada petición.
Así un servlet es persistente, se llama a la misma instancia por cada petición, pero un CGI no es persistente, se crea una instancia por cada petición.

Para el servicio IP Centrex se utilizan Servlets por las ventajas mencionadas con anterioridad.

B.3. Push con HTTP

En un principio los navegadores están diseñados para recibir entradas del usuario y efectuar alguna acción. Cuando apareció la necesidad de enviar información desde el servidor al navegador sin una previa acción (push), se encontró que los navegadores no estaban preparados.

Un claro ejemplo de push **[28]** es el usuario que desea ver los valores de la bolsa, actualizados cada 10 segundos. El cliente presiona en un link una sola vez y el servidor le ira mandando información cada 10 segundos.

Ventajas:

- No se abre una nueva conexión HTTP cada vez que se desea enviar nuevos datos (Si se tuviera que establecer una conexión cada segundo sería muy costoso).
- Se obtiene más control que si el cliente hace las peticiones cada cierto tiempo.

Inconvenientes:

- Se mantiene una conexión abierta indefinidamente (o hasta que el cliente presiona el botón de parada del navegador)

B.4. Tomcat

Tomcat de Apache **[14]** es el contenedor de servlets que se usa en la Implementación de Referencia oficial para las tecnologías de Java Servlets y JavaServer Pages. Estas tecnologías han sido diseñadas pos Sun bajo la Java Community Process **[27]**.

Tomcat se desarrolla en un entorno abierto y participativo bajo la Apache Software License **[14]**. Para los archivos de configuración del Tomcat se utiliza la tecnología XML (*ver apartado B.1*).

Para IP Centrex se ha escogido Tomcat de Apache por su código abierto y su eficacia y rapidez en lo que respecta a Servlets.

B.5. API

Una API (del inglés Application Programming Interface - Interfaz de Programación de Aplicaciones) es un conjunto de especificaciones de comunicación entre componentes software. Representa un método para conseguir abstracción en la programación, generalmente (aunque no necesariamente) entre los niveles o capas inferiores y los superiores del software. Uno de los principales propósitos de una API consiste en proporcionar un conjunto de funciones de uso general, por ejemplo, para dibujar ventanas o iconos en la pantalla. De esta forma, los programadores se benefician de las ventajas de la API haciendo uso de su funcionalidad, evitándose el trabajo de programar todo desde el principio. Las API asimismo son abstractas: el software que proporciona una cierta API generalmente es llamado la implementación de esa API.

B.6. Thread

Muchos lenguajes de programación, como Java, y otros entornos de desarrollo, soportan los llamados hilos o hebras (en inglés, threads). Los hilos son similares a los procesos en que ambos representan una secuencia simple de instrucciones ejecutada en paralelo con otras secuencias. Los hilos son una forma de dividir un programa en dos o más tareas que corren simultáneamente.

Un ejemplo de hilos es tener un hilo atento a la interfaz gráfica (iconos, botones, ventanas), mientras otro hilo hace una larga operación internamente. De esta manera el programa responde más ágilmente a la interacción con el usuario.

Los hilos se distinguen de los tradicionales procesos multitarea en que los procesos son típicamente independientes, llevan bastante información de estados, e interactúan sólo a través de mecanismos de comunicación dados por el sistema. Por otra parte, muchos hilos generalmente comparten otros recursos directamente. En sistemas operativos que proveen facilidades para los hilos, es más rápido cambiar de un hilo a otro dentro del mismo proceso, que cambiar de un proceso a otro. Sistemas Operativos como Windows NT, OS/2 y Linux (2.5 o superiores) han dicho tener hilos "baratos" y procesos "costosos" mientras que en otros sistemas no hay una gran diferencia.

Una ventaja de los programas multihilo es que pueden operar más rápido en sistemas de computadores con múltiples CPUs o a través de grupo de máquinas ya que los hilos del programa se prestan verdaderamente para ejecución concurrente. En tal caso el programador necesita ser cuidadoso para evitar condiciones de carrera (problema que sucede cuando diferentes hilos o procesos alteran datos que otros también están usando) y otros comportamientos no intuitivos. Los hilos generalmente requieren reunirse para procesar los datos en el orden correcto. Es posible que los hilos requieran de operaciones atómicas para impedir que los datos comunes sean cambiados o leídos mientras estén siendo modificados. El descuido de esto puede generar estancamiento.

El uso de hilos en programación frecuentemente causa una inconsistencia de estado. Un error común es crear una variable global e invocar subprogramas que dependen de ésta.

C. Diagramas de clase

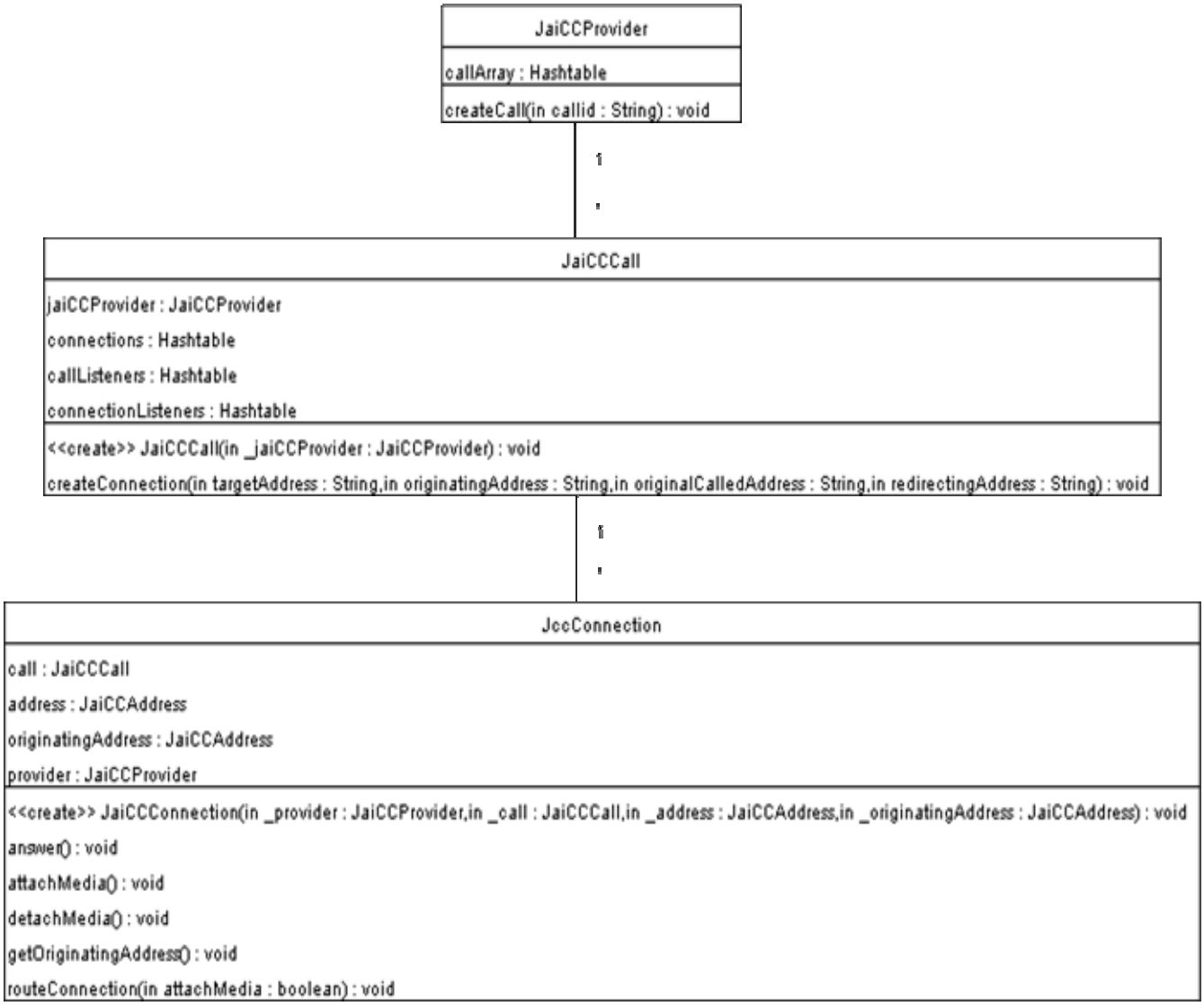


Figura C.1. Relación entre clases JaiCC

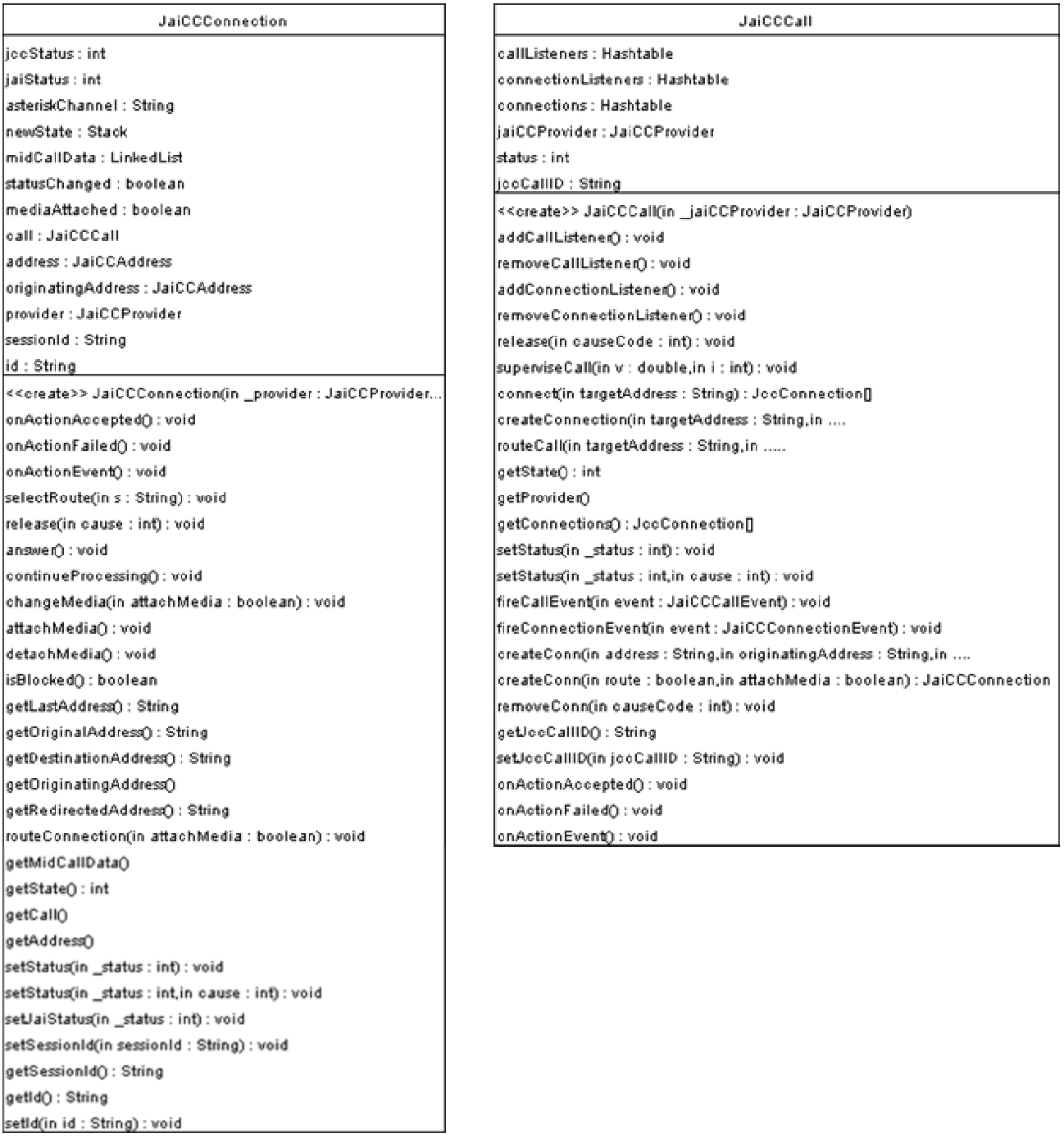


Figura C.2. Diagrama de clases de JaiCCConnection y JaiCCCall

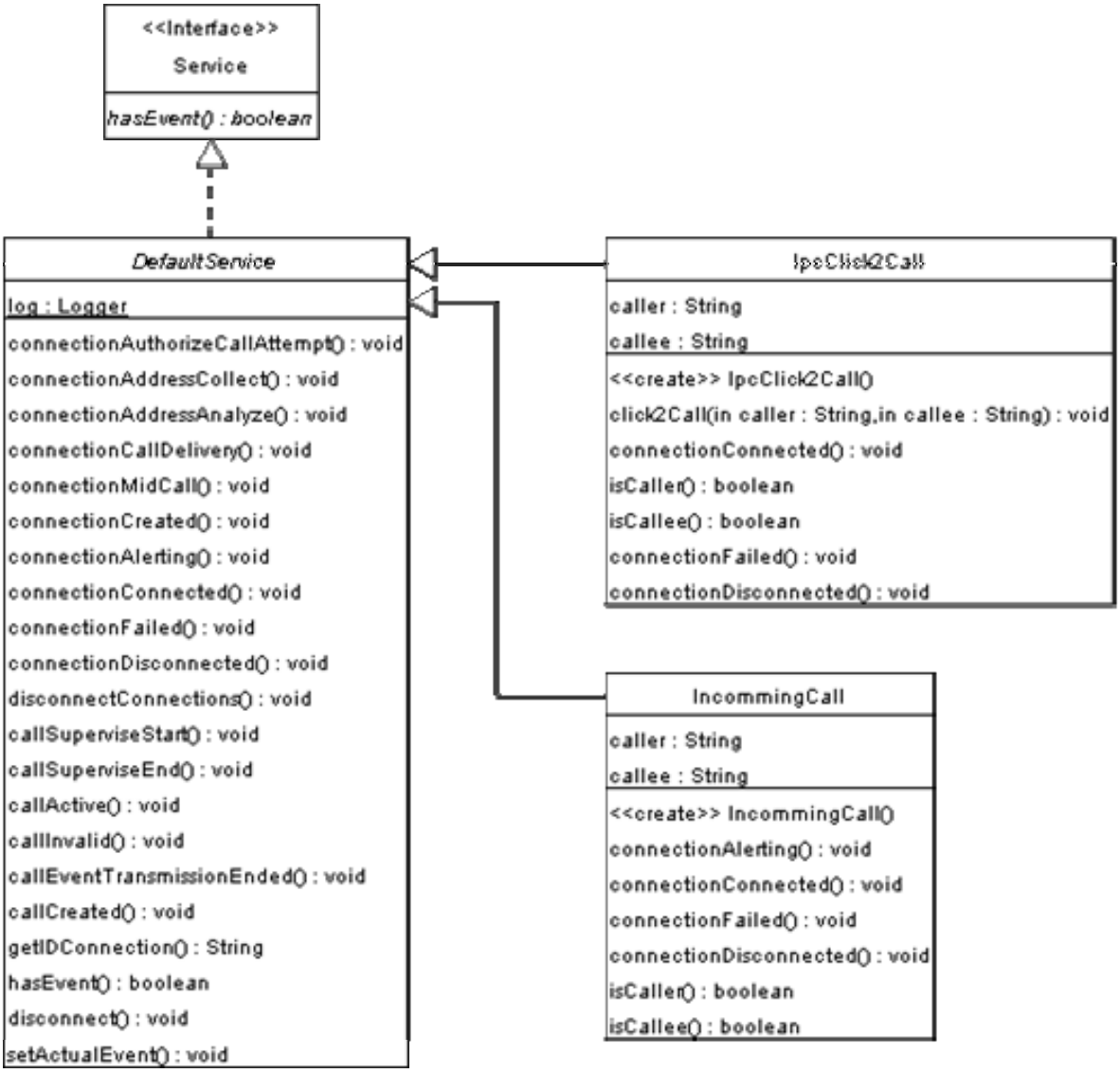


Figura C.3. Diagrama de clases de servicios

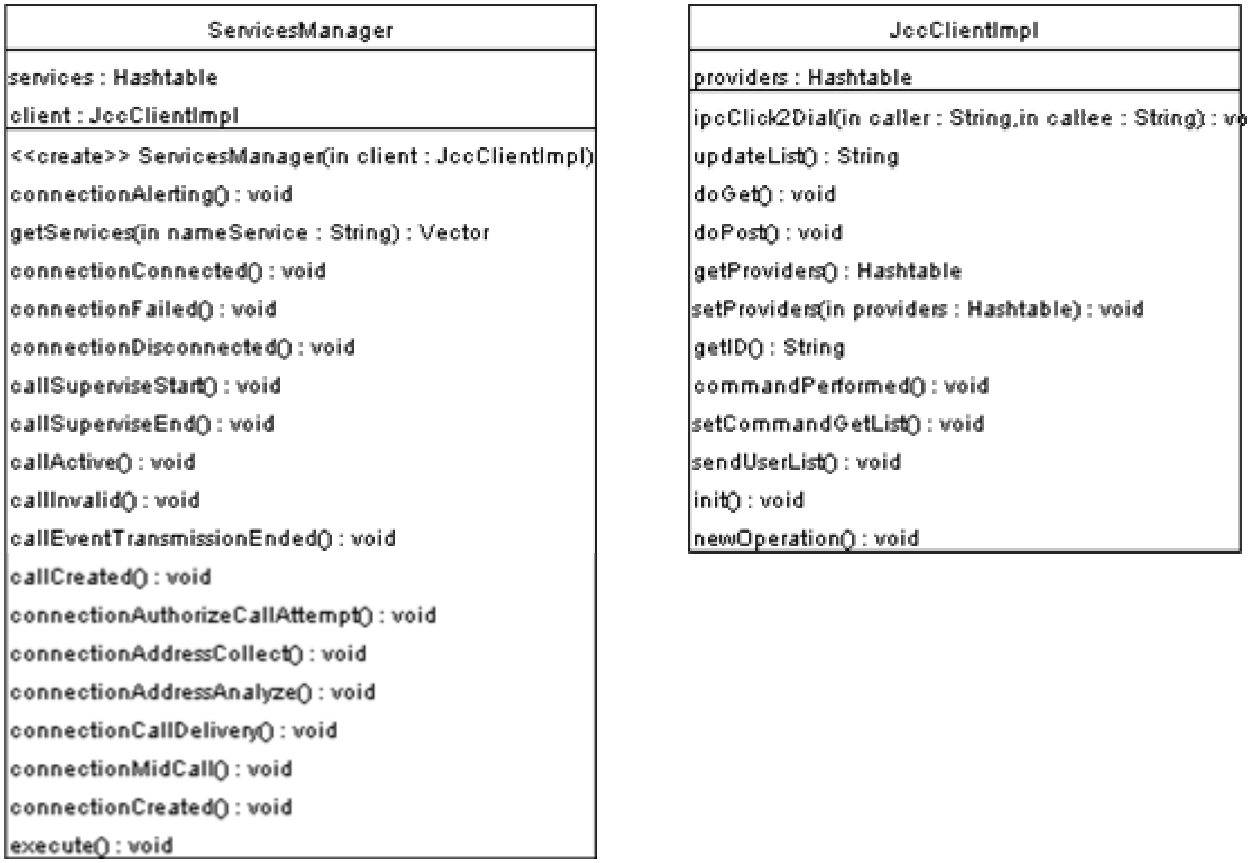


Figura C.4. Diagrama de clases del ServiceManager y el JccClientImpl

D. Diagramas de Flujo

D.1. Llamada entrante

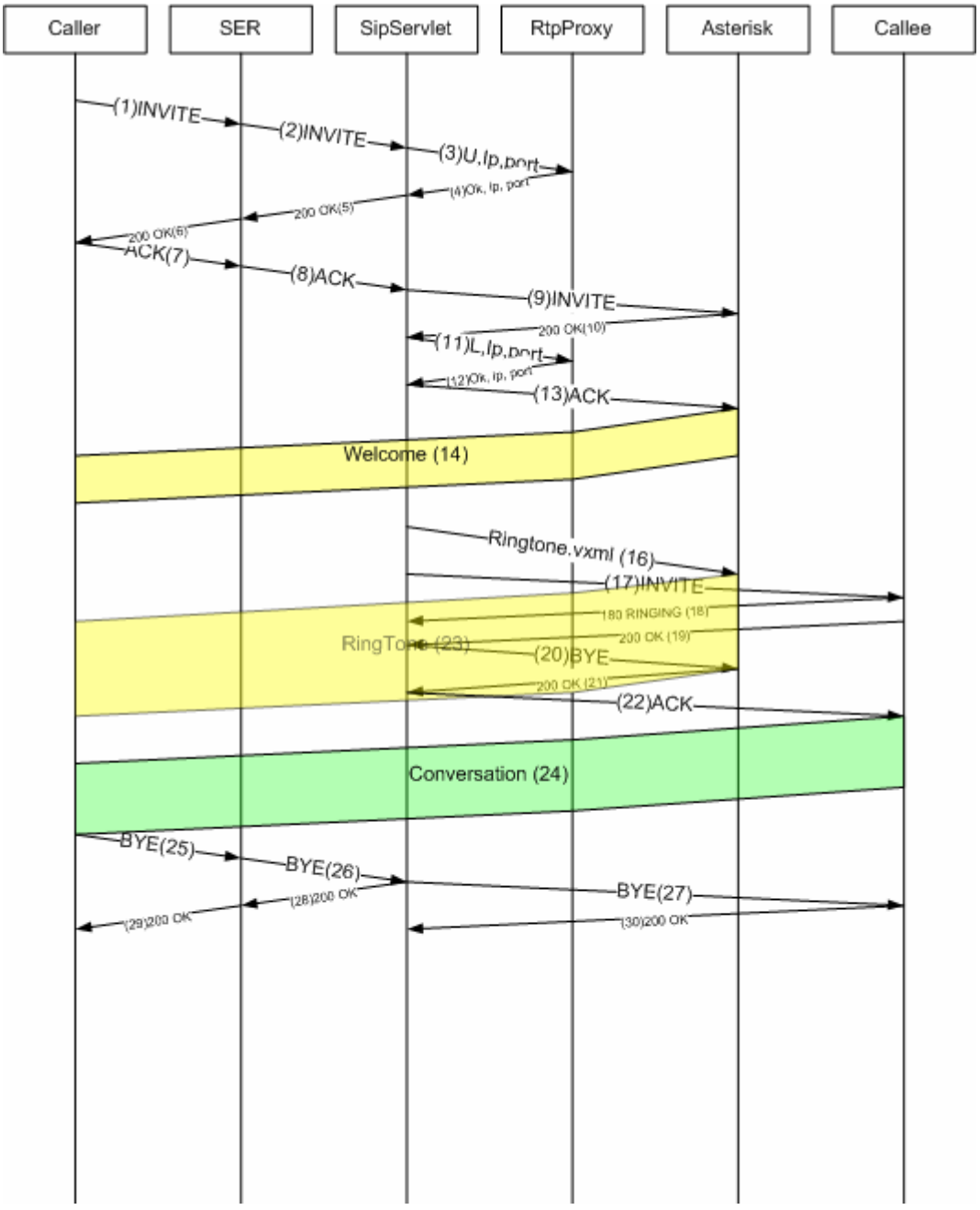
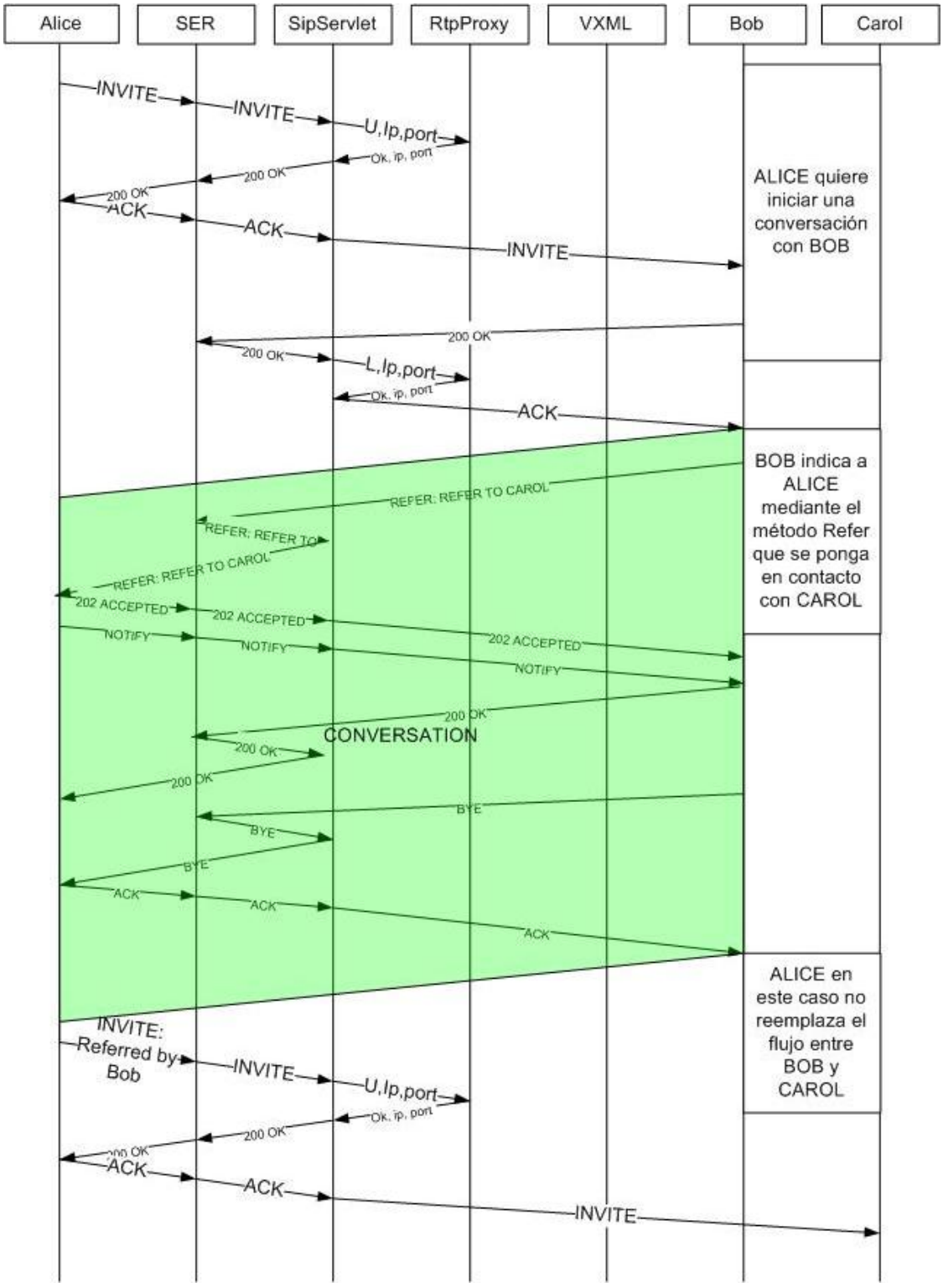


Ilustración D.1. Flujo de una Llamada entrante

D.2. Transferencia de llamada



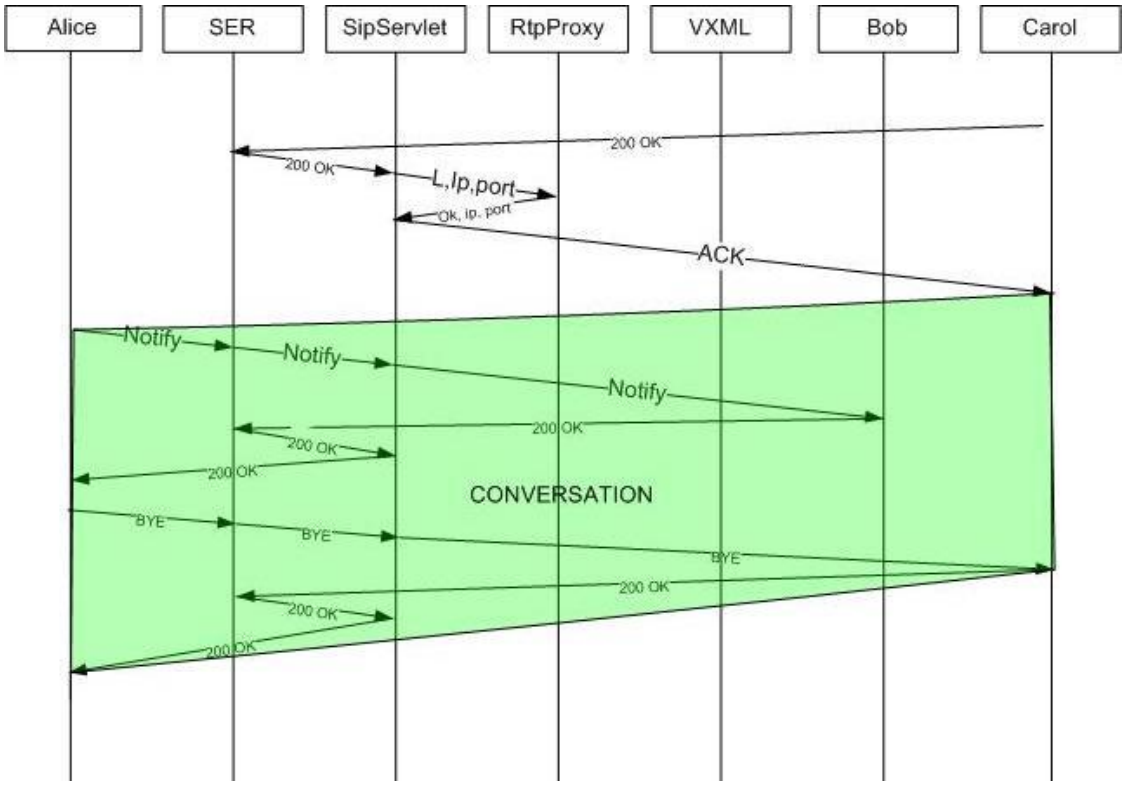
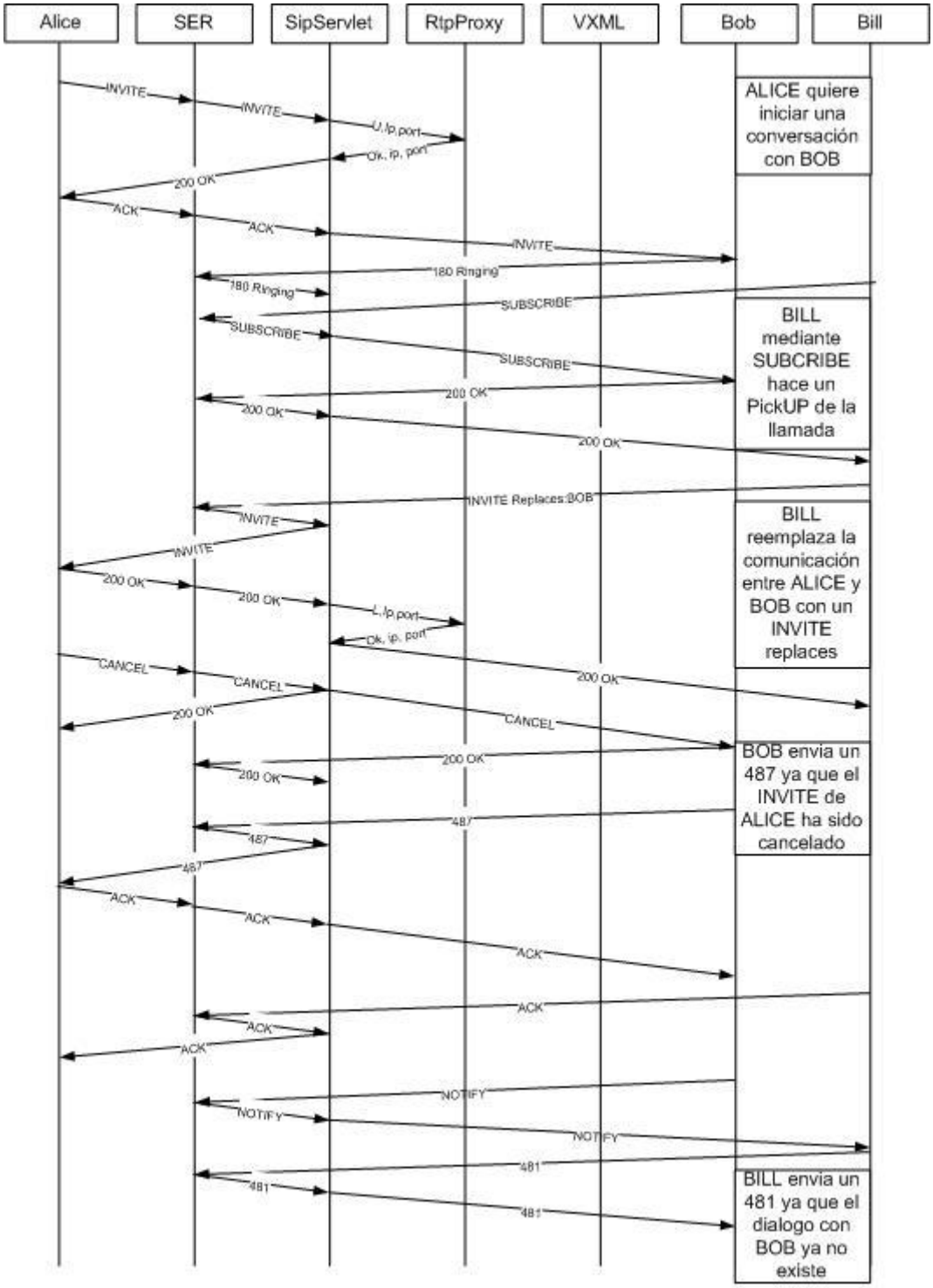


Ilustración D.2. Flujo de una Transferencia de llamada

D.3. Captura de llamada



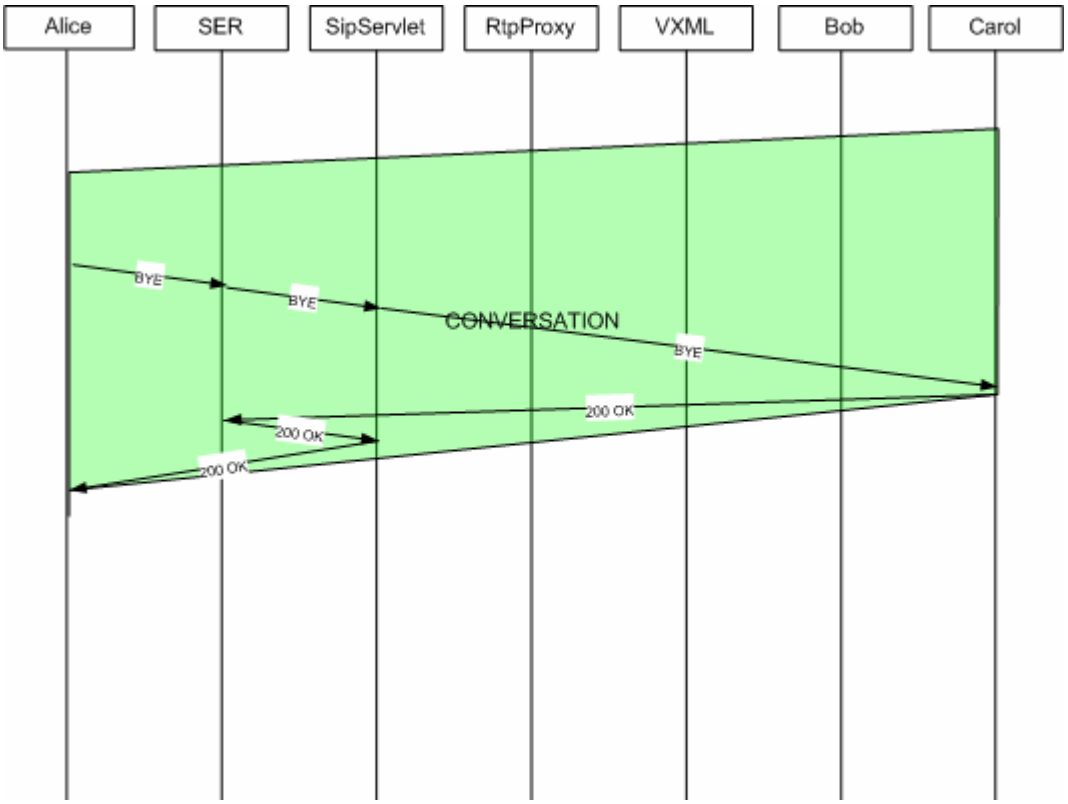


Ilustración D.3. Flujo de una Captura de llamada

D.4. Desvío de llamada

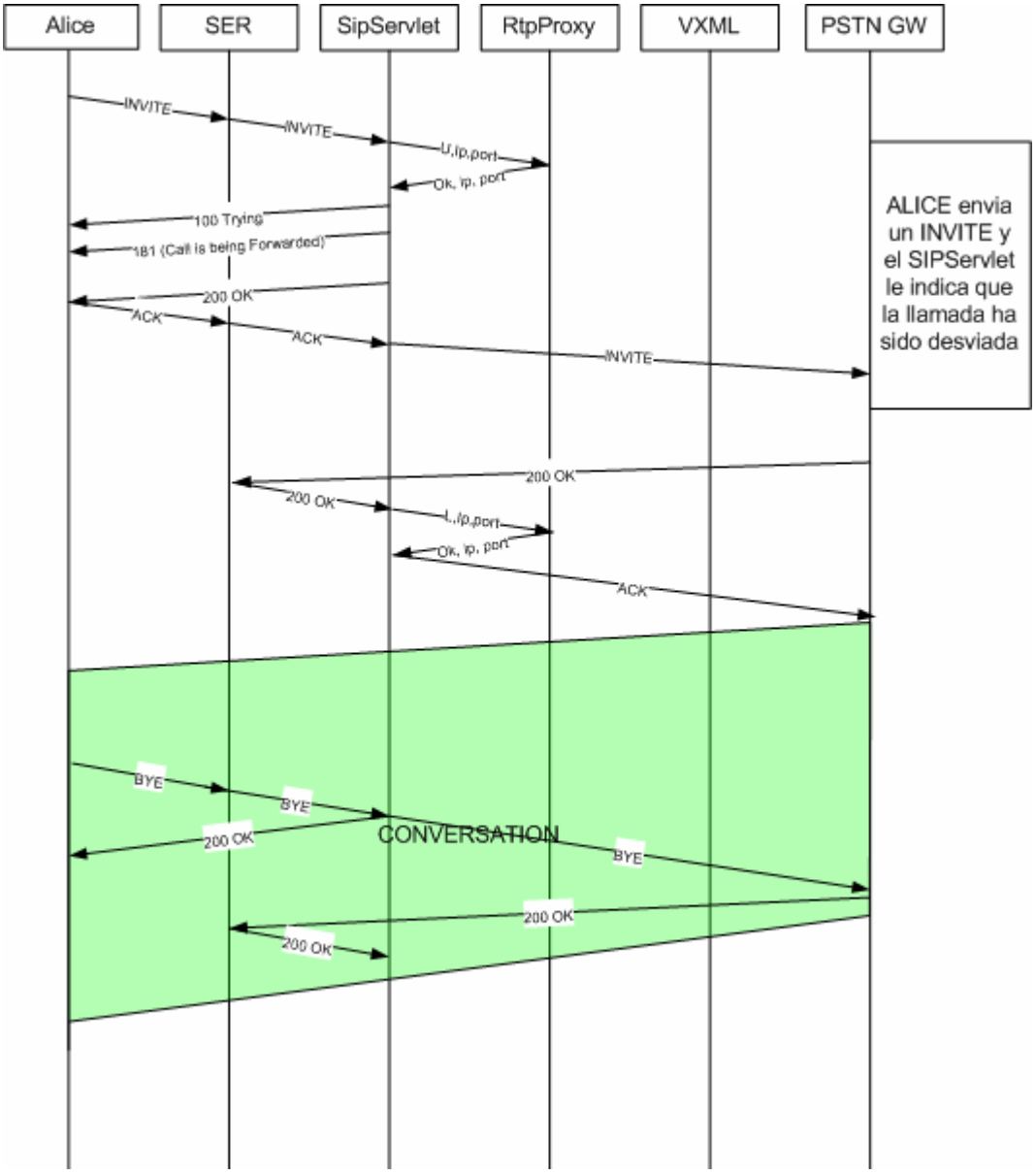


Ilustración D.4. Ejemplo de Desvío de llamada