UNIVERSITAT POLITÈCNICA DE VALÈNCIA

PROGRAMACION CLICK TO CALL

EN CENTRALITA SUTIL

Escuela Técnica Superior de Ingeniería Informática

Director: Miguel Sanchez

Autor: Alberto Serrano

Fecha: 25-09-2012

AUTOR

Este proyecto y su memoria han sido realizados por Alberto Serrano Molina para obtener el título de Ingeniero en Informática en la Universidad Politécnica de Valencia (UPV).

El presente proyecto ha sido supervisado por Miguel Sánchez, profesor del grupo DISCA en la Universidad Politécnica de Valencia, en calidad de tutor de proyecto. Además este proyecto ha sido dirigido y supervisado por Carlos García Pedro, Ingeniero en Informática en la empresa Premium Numbers S.L de Valencia, en calidad de director de las prácticas en empresa que dan lugar al siguiente proyecto.

1.- INTRODUCCIÓN

Dado que Internet sigue posicionándose como la principal fuente de información, son miles de usuarios los que diariamente acuden para investigar sobre determinados productos y/o servicios que requieren.

Click to call (Click2Call), se trata de un botón que se incorpora en la página web, el cual permite al cliente o visitante del sitio, comunicarse verbalmente, a través de dar click en el botón y proporcionar el número telefónico al cual quiere que le localicen. No requiere descargar ningún programa, ni configurar el sistema, dado que, la llamada se realiza directamente al número que el usuario introduce al dar click en el botón.

Es un sistema que mejora la atención al cliente, evita el cambio hacia la competencia, optimiza la captación de nuevos clientes, fideliza a sus clientes, brinda un servicio totalmente gratuito a sus clientes (no siempre) y prospectos, y optimiza tiempos de respuesta. En resumen, garantiza una comunicación eficaz y eficiente entre el comprador y el vendedor.

SUTIL (Sistema Unificado para el Tratamiento Inteligente de Llamadas) es un sistema de telefonía computarizada totalmente autónomo. Hace todas las funciones de una centralita telefónica de cualquier tipo, por lo que puede utilizarse en cualquier entorno donde podría funcionar una centralita.

Pero, además de ello, SUTIL puede realizar otras muchísimas funciones. Prácticamente cualquier labor que tenga que ver con automatizar procesos telefónicos.

SUTIL se puede aplicar principalmente en estos 7 entornos:

– Como centralita de empresa. Añade innumerables ventajas: grabar conversaciones, operadora virtual, etc.

– En entornos de telefonía de coste añadido 80x. Es la plataforma ideal en estos entornos.

– En contact-centers y call-centers. La mejor solución tecnológica para este tipo de entornos.

– En hoteles, residencias, hospitales, etc. Es la plataforma ideal para estos entornos, permite buzones para cada habitación, gestión inteligente del centro a través del teléfono, etc.

– Como sistema de enrutamiento masivo de llamadas: Gestión de tarjetas pre-pagos, pasarelas de llamadas, etc.

– En Centros de negocio. Permite una gestión inteligente de las empresas albergadas.

– Como plataforma de respuesta automatizada IVR (extensión utilizada para los script ejecutados con SUTIL). Funcionando de forma independiente o conectado a una centralita mayor puede proporcionar información de bases de datos a cualquier llamante.

SUTIL permite definir Extensiones Lógicas, de forma que pueden autentificarse en una extensión física o en un número de teléfono. Esto permite que un operador/a trabaje un día en su casa, otro día trabaje en una extensión física y otro día en otra extensión física, pero que en cualquier caso podemos obtener información de toda la actividad de esa extensión lógica, independientemente de que haya estado autentificada en distintos sitios.

SUTIL también nos permite asignar un scripts a una extensión lógica, de forma que al llamar a esa extensión lógica nos responderá ese script.

De esta forma cuando llamamos a una extensión lógica, Sutil, según proceda, llama a una extensión física, si es necesario llama a un número de teléfono o nos responde un script.

SUTIL define las llamadas como Entrantes, Salientes o Internas, tanto a bajo nivel como a alto nivel.

Como bajo nivel entendemos el nivel físico, es decir cuando sutil llama a un número de teléfono es una llamada Saliente, cuando Sutil recibe una llamada de teléfono es una llamada entrante y cuando desde una extensión física se llama a otra extensión física es una llamada interna.

Como alto nivel entendemos el nivel lógico, es decir cuando Sutil llama a una extensión que esta autentificada en un número de teléfono, esa llamada a alto nivel es interna, ya que se trata de una llamada a una extensión, sin embargo a bajo nivel es una llamada saliente.

Igualmente si SUTIL recibe una llamada desde un número de teléfono que esta asignado a una extensión lógica, a alto nivel se considera una llamada interna, ya que proviene de una extensión, y a bajo nivel es una llamada entrante.

SUTIL nos permite listar las llamadas filtrándolas por entrantes, salientes o internas tanto a bajo nivel como a alto nivel. De esta forma podemos, por ejemplo, listar todas las llamadas salientes a bajo nivel para comprobar la factura de nuestro operadora de telefonía, o listar todas las llamadas salientes a alto nivel a una extensión para comprobar todas las veces que hemos llamado a esa extensión lógica.

Una de las claves más importantes de la potencia de SUTIL está en su sistema de scripts. Cuando SUTIL recibe una llamada, dependiendo de a que número ha llamado el cliente se selecciona que se ejecute un determinado script.

Igualmente se puede hacer que se ejecute un script asociado a llamadas salientes. Mediante un script podemos programar un completo sistema IVR que interactúe con el cliente, una operadora virtual, sistemas de suministro de información a clientes, sistemas de encaminamiento masivo de llamadas, etc.

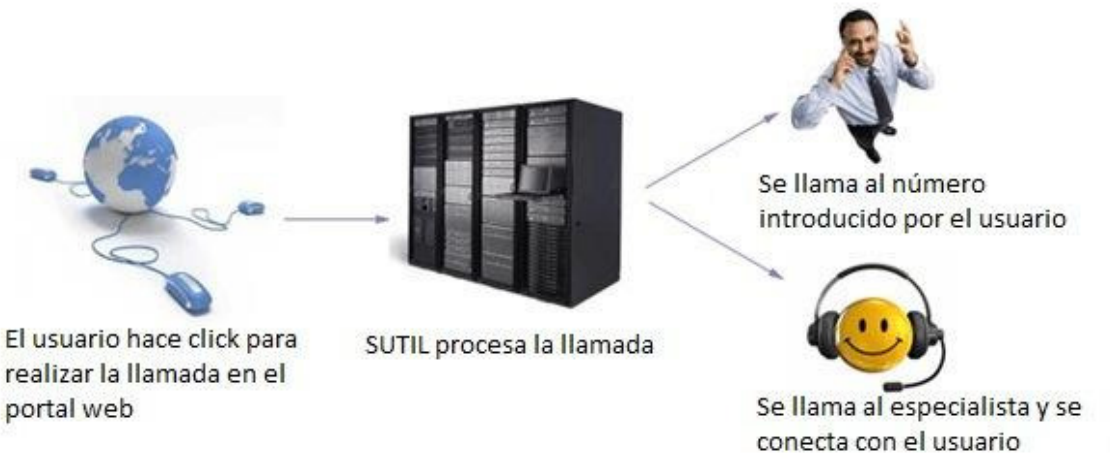
Esto hace de SUTIL un sistema muy versátil ya que en pocas horas se puede configurar para que realice todo tipo de tareas diferentes.

El presente proyecto se divide en 3 partes diferentes:

1. Portal web.
2. Script interno para efectuar las llamadas con SUTIL.
3. Sistema de cobro.

2.- MOTIVACIÓN

Este proyecto consiste en utilizar el sistema SUTIL para crear un servicio de "Click to Call" donde un usuario a través de la web solicite una llamada mediante un "Click" y el sistema automáticamente ponga en contacto al usuario ("Click") con alguien que pueda resolver su consulta ("Call").



Además se tratará de una "red social", que tan de moda están, formada por especialistas de cualquier ámbito, desde tarotistas y sus cartas hasta abogados y su asesoramiento judicial para dar una visión global de este servicio que se podrá usar independientemente del ámbito profesional.

Esto constituye un espacio de encuentro entre tarotistas y clientes o usuarios. Con el objetivo de que estén más cerca que nunca, el portal web permite interactuar mediante la publicación de reflexiones, configurar el perfil con toda la información que se considere oportuna y sobretodo hablar con los usuarios.

Los usuarios de esta web podrán seguir el perfil de un tarotista para recibir todas sus reflexiones, establecer contacto y comentar y/o puntuar la satisfacción por el servicio una vez los hayan usado.

3.- TECNOLOGÍAS EXISTENTES

Aquí se va a hablar de las tecnologías existentes que se pueden relacionar con *Click2Call* para dar una visión general de lo que actualmente hay en el mercado y sus diferencias.

5.1.- Formularios:

Un formulario web dentro de una página web permite al usuario introducir datos los cuales son enviados a un servidor para ser procesados. Los formularios web se parecen a los formularios de papel porque los internautas llenan dichos formularios usando casillas de selección, botones de opción, o campos de texto. Por ejemplo, los formularios web pueden ser usados para introducir datos de envío o datos de una tarjeta de crédito con el objetivo de solicitar un producto o bien ser utilizada para solicitar datos.

Desventajas:

El usuario accede a la web y quiere obtener más información de la que hay en la página web en cuestión y rellena este formulario y pone su correo electrónico para que le puedan proporcionar esa información, pero se tiene que ceñir a los campos que se le proporcionan en el formulario y puede que no tenga nada que ver con su duda para la cual quiere una aclaración directa y rápida.

5.2- Correos:

Correo electrónico (correo-e, conocido también como *e-mail*), es un servicio de red que permite a los usuarios enviar y recibir mensajes y archivos rápidamente (también denominados mensajes electrónicos o cartas electrónicas) mediante sistemas de comunicación electrónicos. Principalmente se usa este nombre para denominar al sistema que provee este servicio en Internet, mediante el protocolo SMTP, aunque por extensión también puede verse aplicado a sistemas análogos que usen otras tecnologías. Por medio de mensajes de correo electrónico se puede enviar, no solamente texto, sino todo tipo de documentos digitales.

Desventajas:

En este caso el usuario tiene más libertad para explicar lo qué necesita y no se ciñe a un estándar mediante unos campos rellenables. Pero ocurre lo mismo que en el caso de los formularios, el usuario no sabe cuándo va a ser resulta su duda o cuando tendrá alguna contestación.

5.3.- Teléfono Atención cliente

Cuando el usuario se encuentra en el sitio web existe un teléfono para consultas directas donde puede llamar y ser atendido.

Desventajas:

De las 3 propuestas está es la mejor, ya que, el contacto es directo entre el usuario y una operadora del portal web pero, ¿realmente es fiable el horario de atención al cliente del portal web?

5.4.-Desventajas generales:

El contacto no es directo.

El contacto no es inmediato.

Se desconoce finalmente si al cliente le van a resolver su duda.

El cliente no está satisfecho

No hay una comunicación entre el cliente y el vendedor.

4.- PLANTEAMIENTO DE LA SOLUCIÓN PROPUESTA

Como hemos mencionado anteriormente este proyecto se puede dividir en 3 partes:

1. Portal web.
2. Script interno para efectuar las llamadas con SUTIL.
3. Sistema de cobro.

La primera parte la compone la creación de un portal web donde agrupar a todos los especialistas. En este caso se trata de un portal de tarotistas y está alojado en http://www.todoslostarotistas.com .

IMAGEN 1: portal web

Como se puede observar existe un apartado arriba a la derecha para el registro de usuarios que asociarán o no su número de teléfono y su tarjeta de crédito para cuando realicen una consulta.

El usuario busca a través del buscador un tarotista con las características que necesita. Se listan una serie de tarotistas, como por ejemplo, los mejor valorados, los más consultados y los nuevos. Una vez el usuario ya ha elegido su tarotista se pone en contacto con él a través de la siguiente imagen.

Se indican una serie de pasos que el usuario tiene que cumplir para que la llamada se pueda efectuar. Para poder realizar una llamada el usuario tiene que estar registrado en el portal web, poner su teléfono de contacto e introducir su tarjeta de crédito.

IMAGEN 2: procedimiento de llamada

La segunda parte, que se desarrollará posteriormente, se centra en la conexión entre el portal web y el especialista elegido. Esta parte ocupa desde que el usuario ha pulsado el botón de llamar en el portal web hasta que está realizando su consulta con el especialista.
Aquí se hace uso de la centralita SUTIL para la ejecución de los script implicados en realizar este proceso.

La tercera, y última parte, engloba el sistema de cobro que tiene una llamada que ocupa desde que una llamada ha finalizado hasta que se conecta con el TPV para realizar el cargo en la tarjeta de crédito del cliente. En este caso es interesante cobrar la llamada porque un usuario está realizando una consulta a un especialista que está realizando su trabajo. No hay que decir que se puede o no cobrar dicha llamada dependiendo del ámbito en cuestión.
El sistema de cobro se ha llevado a cabo usando Microsoft Visual Studio 2005, concretamente en el lenguaje de programación de alto nivel C#.

Todas estas partes se relacionan entre sí mediante una base de datos usando
MySql. La estructura de la base de datos "tarotistas" es la siguiente:

> Tabla "administradores" (id, login, pass): acceso al panel de administración del portal web donde se pueden controlar los comentarios de los usuarios, las llamadas y datos personales entre otras cosas.
>
> Tabla "agenda" (id ,idTarotista , hora_desde, hora_hasta, dia, id_desde, id_hasta): disponibilidad del tarotista para recibir llamadas.

Tabla “denuncias” (idDenuncia, idItemDenunciado, tablaDenunciado, idDenunciante, idDenunciado, FechaHora, tipoDenunciante, revisado): Historico de los comentarios denunciados en el portal web guardando información sobre quien lo realiza, sobre quien y la hora como campos más importantes.

Tabla “especialidades” (idEspecialidad,nombre): relación numérica entre especialidades.

Tabla “historicoPrecios” (id, idTarotista,FechaHora,precio,nuevoPrecio): histórico sobre los precios que fija un tarotista y cuando son modificados. Esto es útil para evitar posibles fraudes.

Tabla “llamadas” (idLlamada, idTarotista, idusuario, ip, telefonoCliente, Telefonotarot1, TelefonoTarot2, TelefonoTarot3, teléfono_llamada, num_tarjeta, fecha_caducidad, cvv, Fecha, Hora, Duración, precioMinuto, coste, valorada, estado, pararscript): La tabla más importante de toda la base de datos. Se registra una llamada, los teléfonos de contacto, los datos de la tarjeta de crédito, el precio por minuto de la conversación…

Tabla “mensajetarotista” (idmensaje, idTarotista, Fecha, Mensaje, Denunciado): los mensajes por tarotista que aparecen en el portal web.

Tabla “seguidores” (idSeguidor, idTarotista, idUsuario): relación entre el tarotista y sus seguidores en el portal web.

Tabla “tarotistas” (idTarotista, Nombre, Apellidos, NombreAMostrar, Email, Especialidad, Precio, Foto, Presentación, DescripcionCorta, DescripcionDetallada, Experiencias, Publicaciones, presentacionAudio, PresentacionVideo, Login, Pass, telefonoContacto, Telefono1, Telefono2, Telefono3, Telf_premium, RazonSocial, Cif, Domicilio, NumCuenta, Status, IdTPV, DenunciaPerfil, NombreWeb, EspecialidadWeb, TerminosAceptados, FechaAceptacion, IpAceptacion): información detallada del tarotista.

Tabla “usuarios” (idUsuario, Nombre, Apellidos, Email, Login, pass, Telefono, Numtarjeta, FechaCaducidad, cvv, FechaNacimiento, Publicidad, tmp_key, Solicita_pass, Observaciones, status, NombreWeb): información detallada del usario.

Tabla “valoraciones” (idValoracion, idLlamada, idTarotista, idUsuario, valoración, texto, fecha, denunciado): relación entre el tarotista y el usuario para la valoración de una llamada.

Un posible diagrama Entidad-Relación sencillo podría ser:

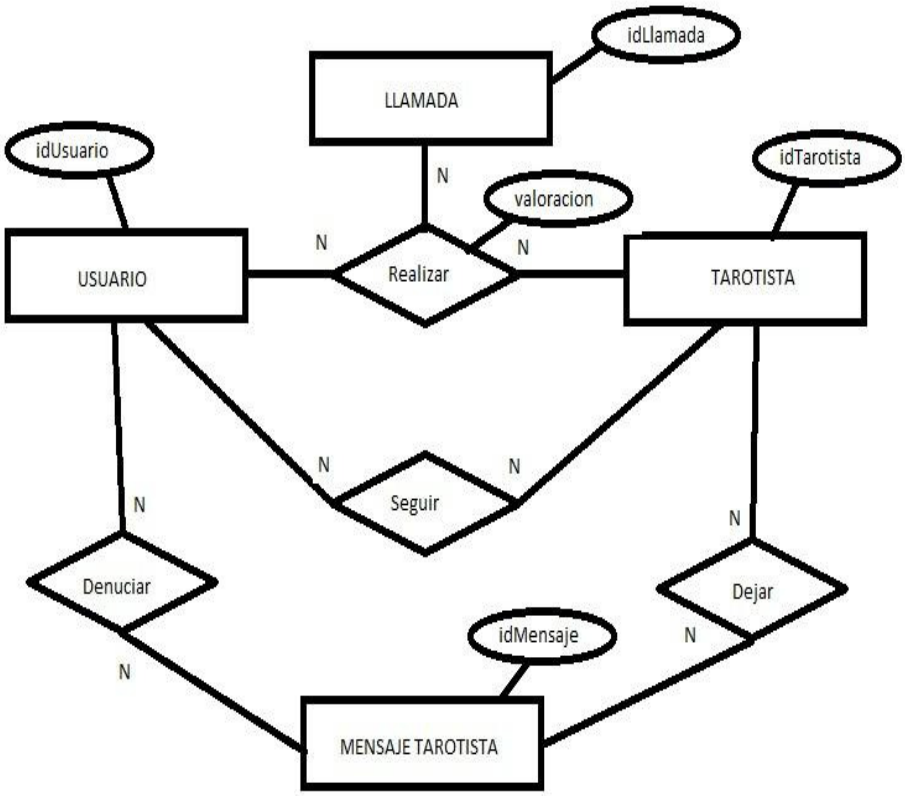


IMAGEN 3: Diagrama Entidad-Relación de la base de datos “tarotistas”. Solo se indican las claves primarias.

5.- DESARROLLO DE LA SOLUCIÓN

Como se ha comentado en el apartado anterior este proyecto se centra en las partes 2 y 3. Para conectar el portal web (parte 1) con la centralita SUTIL (parte 2) mediante scripts (.IVR)

Una de las claves más importantes de la potencia de SUTIL está en su sistema de scripts. Cuando SUTIL recibe una llamada, dependiendo de a qué número ha llamado el cliente se selecciona que se ejecute un determinado script.

Mediante un script podemos programar un completo sistema IVR que interactúe con el cliente, una operadora virtual, sistemas de suministro de información a clientes, sistemas de encaminamiento masivo de llamadas, etc.

Esto hace de SUTIL un sistema muy versátil ya que en pocas horas se puede configurar para que realice todo tipo de tareas diferentes. La programación de scripts es sencilla y está al alcance de cualquiera aunque jamás haya realizado un programa.

Los ficheros tienen extensión ***.ivr*** y tienen que estar grabados en el disco duro de SUTIL dentro del directorio ***Scripts*** dentro del directorio de SUTIL (normalmente ***C:\Sutil***). El nombre del fichero puede tener como máximo 20 caracteres más la extensión, en el fichero histórico solo se almacenan como máximo los 20 primeros caracteres del nombre del script incluyendo la extensión.

La sintaxis de los comandos no distingue entre mayúsculas y minúsculas. Los comandos pueden ir seguidos de un carácter ';' y un comentario. Este comentario será ignorado. Los espacios y tabuladores al comienzo de cada línea son ignorados.

Los scripts deben ser compilados para generar ficheros con el código binario que SUTIL utiliza, para generar estos ficheros compilados debemos ejecutar el comando Compilador.exe que se encuentra en el directorio de SUTIL.

Al ejecutar este comando compila todos los scripts que encuentre tanto en el directorio ***Scripts.*** Los ficheros compilados tienen de extensión ***.bin*** y se guardan en el directorio ***Bin***.

En el proceso de compilación por defecto todas las letras se cambian a mayúsculas, si queremos mantener algún texto tal cual, sin cambiarlo, debemos incluirlo dentro de comillas dobles, el compilador todo el texto que encuentra entre comillas dobles no lo modifica, las comillas dobles no aparecen en el fichero compilado.

Al compilar los scripts, si hay algún error, veremos un aviso del tipo de error y de la línea donde se encuentra el error, para facilitar la localización de esta línea donde está el error, en el directorio donde se encuentre dicho script (normalmente el directorio ***scripts***) podremos encontrar un fichero de nombre como el scripts al que además de la extensión .ivr se le ha añadido la extensión .obj y donde las líneas están numeradas, de forma que podemos localizar rápidamente y sin duda la línea donde se encuentra el error.

Una vez solucionado el error al volver a compilar, si no hay errores, el fichero .obj desaparece.

El fichero .ivr está formado por comandos que reconoce el compilador. Es texto plano y se ejecuta línea a línea. Para poder efectuar saltos se utilizan etiquetas. Un fichero debe de estar compilado para asegurarse de que los comandos están introducidos correctamente y no existe ningún error. De este modo se genera el archivo .bin que es el ejecutable cuando se hace una llamada a algún fichero .ivr. El compilador se ejecuta desde una consola MSDOS donde se indica el fichero .ivr que se quiere compilar. En esta misma consola devuelve si el fichero se ha compilado correctamente (se ha generado el .bin) o de lo contrario devuelve la línea del error.

"UserInit.ivr"

El script UserInit.ivr es el primer script en ejecutarse. En él se encuentran los scripts que se tienen que iniciar/ejecutar cuando SUTIL se reinicia por alguna razón como puede ser mantenimiento.

if %sutil==1

La centralita SUTIl puede estar instalada en diferentes motores. Con esta restricción indicamos que únicamente ejecutaremos en el motor 1 los script que se encuentren dentro de la sentencia IF-ELSE.

INVOKE script1

INVOKE c2cTarot_Bucle

INVOKE scriptN

Mediante el comando INVOKE ejecutamos el script y pasamos al siguiente comando. Tambíén se puede usar el comando CALL pero en este caso no sigue ejecutando el script sino que se espera a que acabe para devolver el control.

endif

EXIT

Con el comando EXIT salimos del script, se usa para parar un script.

"c2c_tarotIniciar.ivr"

El script c2c_TarotIniciar.ivr se encarga de iniciar el sistema para no depender del script "userInit.ivr" explicado anteriormente. De esta forma podemos realizar las pruebas oportunas para verificar que el sistema funciona como se desea.

OPENFILE 1 1 Trazas\c2cTarotAlbertoTLT.txt

Este comando se utiliza para crear un archivo de log, en este caso, extension .txt que se guarda en la carpeta Trazas con nombre c2cTarotAlbertoTLT.

El segundo parámetro indica el identificador del archivo.

WRITE2FILE 1 ===== %session,INICIO TAROT-INCIAR=====

Este comando escribe una línea en el fichero de log.

A través del primer parámetro se sabe en qué fichero se va a escribir.

La variable "session" es un identificador único para cada script. De esta forma se puede conocer el proceso integro de un script cuando en el mismo fichero de log escriben 2 o más scripts.

INVOKE c2cTarot_Bucle.ivr

Se invoca el script "c2cTarot_Bucle.ivr" pero continúa la ejecución de este script.

EXIT

Se finaliza el script.

"c2c_tarot_Bucle.ivr"

El script c2c_Tarot_Bucle.ivr se encarga de estar en continuo funcionamiento. Realiza peticiones continuas a la tabla "llamadas" de la base de datos "tarotistas" para ver si se tiene que producir una llamada. En este caso se diferencia por el campo estado de la tabla llamadas. Cuando una llamada entra, tiene estado 0 y cuando se ha procesado estado 1.

Este script no finaliza, al llegar al final salta al principio para volver a comprobar si hay alguna llamada nueva. Se comprueba cada 20 segundos.

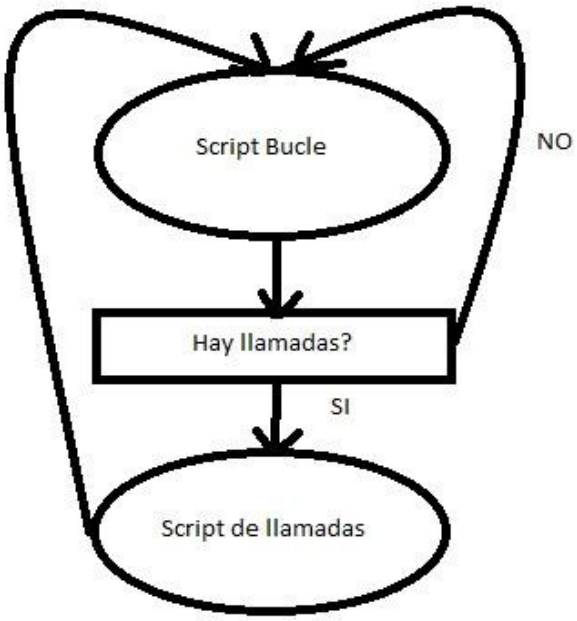


IMAGEN 4: Diagrama de flujo

OPENSOCKET 1 DireccionIP Puerto

Este comando se utiliza para abrir una conexión con la dirección y puerto indicado. En este caso donde está alojada la base de datos.

OPENFILE 1 1 Trazas\c2cTarotAlbertoTLT.txt

WRITE2FILE 1 ===== %session,INICIO TAROT-BUCLE=====

:INICIO_BUCLE

Etiqueta que se utlizara con el comando GOTO para volver a esta posición del script. Hay que decir que un script se ejecuta comando a comando como si se tratara de texto plano. Los bucles de los lenguajes de alto nivel se consiguen mediante etiqueta y el comando GOTO.

SEND2SOCKET 1 36 10

12A,"org.gjt.mm.mysql.Driver","jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=yes&characterEncoding=UTF-8","tarotistas","password";

El commando SEND2SOCKET con el parámetro 12A abre una conexión a la base de datos tarotistas y la almacena en la variable 36.

SEND2SOCKET 1 37 10 12B,%w36,"SELECT COUNT(*) FROM llamadas where estado=0";

El commando SEND2SOCKET con el parámetro 12B almacena la consulta que se quiere realizar en la variable 37. En este caso se comprueba cuantas llamadas hay con estado 0 para procesarlas.

SEND2SOCKET 1 38 10 12C,%w36,%w37;

El commando SEND2SOCKET con el parámetro 12C almacena en la variable 38 el resultado de la consulta realizada con los parámetros 36 y 37.

SEND2SOCKET 1 39 10 12D,%w36,%w37;

El commando SEND2SOCKET con el parámetro 12D indica la solicitud de fin de sesion con base de datos.

SEND2SOCKET 1 39 10 12E,%w36;

El commando SEND2SOCKET con el parámetro 12E indica la solicitud de fin de conexion con base de datos.

Si se realizan otras consultas sobre la misma base de datos no es necesario cerrar la conexión para cada consulta.

IF %w36 == ERR

Si la variable 36 origina un error se ejecuta esta parte de la sentencia IF.

PAUSE 10

Este commando indica que se realiza una pausa de 10 segundos en la ejecución de este script.

WRITE2FILE 1 ===== %session,%w36 =====

GOTO INICIO_BUCLE

Como se ha mencionado con antelación. Con este comando se salta a la etiqueta INICIO_BUCLE.

ENDIF

IF %w37 == ERR

PAUSE 10

WRITE2FILE 1 ===== %session,%w37 =====

GOTO INICIO_BUCLE

ENDIF

IF %w38 == ERR

PAUSE 10

WRITE2FILE 1 =====%session, %w38 =====

GOTO INICIO_BUCLE

ENDIF

Código identico para las variables 37 y 38. En el caso de que estas variables den error, se salta al inicio del script. El error puede venir dado de una caída de la base de datos temporal o que ha tardado mas del tiempo estimado en devolver un resultado.

IF %W38 == 0

PAUSE 10

WRITE2FILE 1 ===== %session,No hay llamadas =====

ELSE

WRITE2FILE 1 ===== %session,%w38 llamada(s) por procesar =====

ENDIF

Este fragmento de código se utiliza para conocer si hay o no llamadas y en el caso de haber cuantas se tienen que procesar.

12A,"org.gjt.mm.mysql.Driver","jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=yes&characterEncoding=UTF-8","tarotistas","password";

SEND2SOCKET 1 37 10 12B,%w36,"SELECT idUsuario,telefonoTarot1, telefonoTarot2,telefonoTarot3,idTarotista,idLlamada,telefonoCliente,precioMinuto, pararscript FROM llamadas where estado=0";

Por cada llamada a procesar obtenemos los datos necesarios para poder realizar la llamada y posteriormente poder hacer el cobro de la misma.

:BUCLE_LLAMADAS

SEND2SOCKET 1 38 10 12C,%w36,%w37;

Como se puede observar se ha agregado una nueva etiqueta. Esta etiqueta nos permite procesar todas las llamadas antes de volver a comprobar si tenemos nuevas.

También es interesante fijarse en que se ha dejado la conexión abierta hasta que se procesen todas las llamadas.

```
IF %w36 == ERR
   PAUSE 10
   WRITE2FILE 1 ===== %session,%w36 =====
   GOTO INICIO_BUCLE
ENDIF
IF %w37 == ERR
   PAUSE 10
   WRITE2FILE 1 ===== %session,%w37 =====
   GOTO INICIO_BUCLE
ENDIF
```

```
IF %w38 == ERR
  PAUSE 10
  WRITE2FILE 1 ===== %session,%w38 =====
  GOTO
INICIO_BUCLE ENDIF
IF %W38 == X
  SEND2SOCKET 1 39 10 12D,%w36,%w37;
  SEND2SOCKET 1 39 10 12E,%w36;
  WRITE2FILE 1 ===== %session,No hay más llamadas pendientes =====
  PAUSE 5
  GOTO
INICIO_BUCLE ENDIF
```

Sino quedan llamadas pendientes por procesar se salta a INICIO_BUCLE una vez pasados 5 segundos.

```
STRTOKEN 1 1 | %W38 ;idUsuario -> v1
STRTOKEN 2 2 | %W38 ;telefonoTarot1 -> v2
STRTOKEN 3 3 | %W38 ;telefonoTarot2 -> v3
STRTOKEN 4 4 | %W38 ;telefonoTarot3 -> v4
STRTOKEN 5 5 | %W38 ;idTarotista -> v5
STRTOKEN 6 6 | %W38 ;idLlamada -> v6
STRTOKEN 7 7 | %W38 ;telefonoCliente -> v7
STRTOKEN 8 8 | %W38 ;precioMinuto-> v8
```

STRTOKEN 9 9 | %W38 ;pararscript -> v9

WRITE2FILE 1 ===== %session,%W38 =====

Cuando una consulta se ejecuta de manera satisfactoria y se trata de un SELECT, la variable 38 contiene el resultado devuelto separador por el carácter " | ".

El comando STRTOKEN sirve para dividir una cadena en subcadenas mediante el separador indicado y almacenarlo en una variable.

IF %v9 == 0000-00-00

WRITE2FILE 1 ===== %session,CAMPO PARARSCRIPT = 0000-00-00 =====

GOTO FIN

ENDIF

En la variable 9 está almacenado el campo "pararscript". Este campo se utiliza cuando se quiere parar forzosamente el script. De no estar este campo la única manera de parar la ejecución del script sería reiniciando el motor de SUTIL pero perderíamos las acciones de los demás script que se estuvieran ejecutando. Hay que pensar que la centralita SUTIL puede estar ejecutando o procesando llamadas por lo que reiniciar el motor podría ser devastador. De esta forma se para la ejecución de este script y luego lo podemos volver a ejecutar con el script "c2c_TarotIniciar.ivr" como se ha comentado anteriormente por lo que este fragmento de código tiene mucha importancia.

IF %V6 ==

WRITE2FILE 1 %session, IDLLAMADA VACIO

GOTO BUCLE_LLAMADAS

Si el id de la llamada está vacío volvemos a la etiqueta BUCLE_LLAMADAS. Esto indica que la llamada ha sido mal insertada en la base de datos por lo que no se puede procesar.

```
ENDIF
IF %V6 == X
  WRITE2FILE 1 %session, IDLLAMADA X
  GOTO BUCLE_LLAMADAS
ENDIF
IF %V6 == ERR
  WRITE2FILE 1 %session, IDLLAMADA ERR
  GOTO BUCLE_LLAMADAS
ENDIF
INVOKE c2cTarotLlamada.ivr %v1 %v2 %v3 %v4 %v5 %v6 %v7
%v8 GOTO BUCLE_LLAMADAS
```

Se llama al script c2cTarotLlamada y se le pasa como parámetros el id de la llamada, el teléfono del usuario, los diferentes destinos del especialista y el precio por minuto entre otros.

Por último, como el script ya está procesando la llamada mediante INVOKE se vuelve a BUCLE_LLAMADAS para procesar la siguiente llamada.

“c2cTarotLlamada”

El script “c2cTarotLlamada” se encarga de procesar la llamada que recibe como parámetro desde el script “c2cTarot_Bucle.ivr”. En este script se llevan a cabo todas las operaciones para poner en contacto al usuario con el especialista y posteriormente la generación de un fichero para poder cobrar la llamada.

Lo primero que realiza el script es llamar a un extremo, en este caso, al especialista. Al especialista le suena su teléfono de contacto y escucha una locución llamada whisper(un whisper es una locución que se escucha antes de conectar los dos canales para que exista una comunicación). En este caso el whisper le sirve al especialista para elegir si quiere o no contestar la llamada. En el caso de aceptar la llamada, se llama automáticamente al teléfono de contacto que ha introducido el usuario en el portal web. Si el especialista rechaza la llamada, se llama el usuario y se le ofrece intentar la llamada pasado un tiempo, seleccionar otro especialista aleatoriamente o finalizar la llamada. En el caso de que sea el usuario el que no ha podido atender la llamada, se le avisa al especialista si quiere volver a intentarlo o no nuevamente.

Una vez la llamada ha finalizado, si ha tenido éxito, se genera un fichero con los datos de la llamada y el precio total de ésta. Este fichero es procesado posteriormente por una tarea que se explicará mas adelante.

Ahora se mostrará un diagrama de flujo con el procedimiento a seguir para poder entender, mejor si cabe, su funcionamiento:

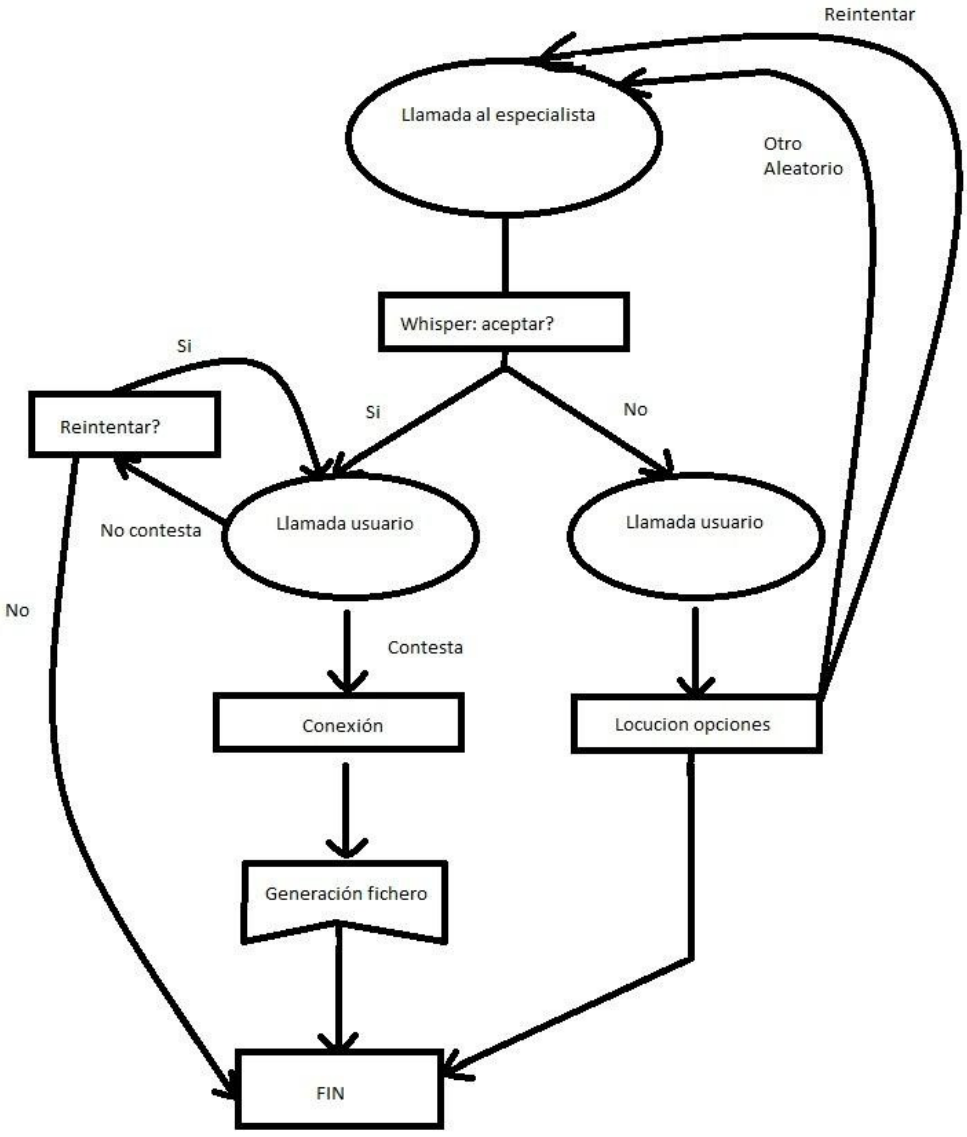


IMAGEN 5: Diagrama de flujo.

OPENSOCKET 1 192.168.1.2 3012 ;Para acceso a BD

OPENFILE 1 1 Trazas\c2cTarotAlbertoTLT2.txt

WRITE2FILE 1 %session, ===== INICIO c2cTarot.ivr =====

SEND2SOCKET 1 36 10
12A,''org.gjt.mm.mysql.Driver'',''jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=yes&characterEncoding=UTF-8'',''user'',''pass'';

SEND2SOCKET 1 37 10 12B,%w36,''UPDATE llamadas SET estado = 1 WHERE idLlamada = %V5'';

SEND2SOCKET 1 39 10 12E,%w36;

WRITE2FILE 1 %session,UPDATE llamadas SET estado = 1 WHERE idLlamada = %V5;

Se actualiza la llamada con estado igual a 1 para que no se vuelva a procesar en el script c2ctarotBucle.ivr.

SETLOCAL 27 0

Esta variable se utiliza para saber si el nuevo tarotista es aleatorio o no.

GOTO INICIO

:TAROTISTA_ALEATORIO

SEND2SOCKET 1 36 10
12A,''org.gjt.mm.mysql.Driver'',''jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=yes&characterEncoding=UTF-8'',''user'',''pass'';

SEND2SOCKET 1 37 10 12B,%w36,''UPDATE llamadas SET telefonoTarot1='%V1' ,telefonoTarot2='%V2',telefonoTarot3='%V3',precioMinuto='%V7', idTarotista='%V4' WHERE idLlamada = %V5'';

SEND2SOCKET 1 39 10 12E,%w36;

WRITE2FILE 1 %session,UPDATE llamadas SET telefonoTarot1='%V1' ,telefonoTarot2='%V2',telefonoTarot3='%V3',precio='%V7', idTarotista='%V4' WHERE idLlamada = %V5;

Se actualizan los campos con los datos del nuevo tarotista.

:INICIO

SETSELFCALLERID 902070777

Numero que aparecerá en las pantallas del teléfono cuando entra la llamada. Es un número en el cual el especialista o el usuario se pueden poner en contacto si surge algún problema.

SETLOCAL 19 1
SETLOCAL 17 1
SETLOCAL 18 1

Variables para controlar los intentos de llamada en el origen y destino.

```
:LLAMA_ORIGEN
 ONHOOK 0 FIN
 IF %W19==1
  SETLOCAL 12 %v1
 ELSE
  IF %W19==2
    SETLOCAL 12 %V2
  ELSE
   IF %W19==3
     SETLOCAL 12 %V3
   ENDIF
  ENDIF
 ENDIF
```

En las variables V1, V2 y V3 están guardados los teléfonos destino del especialista(origen) y en caso de no contactar con uno se intenta con el resto.

```
IF %W19 > 3
  GOTO ORIGEN_NO_DISPONIBLE
ENDIF
```

Si se supera el límite de intentos (1 por destino) se salta a la etiqueta ORIGEN_NO_DISPONIBLE.

```
IF %w12 ==
  INCLOCAL 19
  WRITE2FILE 1 %session, Destino vacio
  GOTO LLAMA_ORIGEN
ENDIF
;WRITE2FILE 1 %session, Destino= %w12
```

Si no hay un destino definido se salta al siguiente.

```
CALL2EXTERIOR 0 %W12 40
IFERR 1
  WRITE2FILE 1 %session, Timeout en
  LLAMA_ORIGEN INCLOCAL 19
  WRITE2FILE 1 %session, Contador= %W19
  GOTO LLAMA_ORIGEN
IFERR 2
  WRITE2FILE 1 %session, Comunicando en
  LLAMA_ORIGEN INCLOCAL 19
  WRITE2FILE 1 %session, Contador=%W19
  GOTO LLAMA_ORIGEN
IFERR 0
  WRITE2FILE 1 %session, Whisper en LLAMA_ORIGEN
  WRITE2FILE 1 %session, Contador= %W19
```

```
  GOTO WHISPER
 ENDIFERR
```

Se llama al destino del especilista con una duración máxima de 40 segundos. Si devuelve un 1 es que ha dado time out el destino y se intenta con el siguiente. Si devuelve un 2 es que el destino estaba comunicando y se intenta con otro. Si devuelve 0 se le lanza la locución whisper al especialista para que acepte o rechace la llamada.

```
 WRITE2FILE 1 %session, Otro error en LLAMA_ORIGEN
 INCLOCAL 19
 GOTO LLAMA_ORIGEN
```

En caso de que devuelva otro valor que no sea 0, 1 o 2 se incrementa la variable 19 para buscar otro destino.

```
:WHISPER

 PLAYFILE 0 c2ctarot\whispertarotista
 GOTO INPUT_WHISPER
```

Se pone la locucion por el canal 0 (canal del origen de la llamada)

```
:INPUT_WHISPER
 WRITE2FILE 1 %session, Input_Whisper
 INPUT 0 24 1 15
 IFERR 0
  IF %W24==1
   WRITE2FILE 1 %session, Tecla pulsada: %w24. Llamo al cliente
   %V6. PLAYTONE 0 0
   IF %W27 == 0
    GOTO LLAMA_DESTINO
   ELSE
    CONNECT 0 1
    GOTO CONEXION
```

```
    ENDIF
   ELSE
    WRITE2FILE 1 %session, Tecla pulsada: %w24. Llamada
    finalizada HOOK 0
    ;GOTO FIN
    GOTO ORIGEN_NO_DISPONIBLE
   ENDIF
  IFERR 1
   WRITE2FILE 1 %session, Timeout
   input_WHISPER GOTO WHISPER
  ENDIFERR
```

Se espera durante 15 segundo un pulso DTMF (pulsación de una tecla del teléfono) por el canal 0 que se almacenará en la variable 24. Si es un 1 se llama a continuación al cliente. Si es otra tecla se llama al cliente con la locución de que el origen está ocupado. Dentro de la aceptación de la llamada se puede diferenciar gracias a la variable 27 si se conectan los canales directamente porque el usuario viene de esperar un tarotista aleatorio y si se llama al cliente porque aun no se han puesto en contacto con él.

```
:FALLO_DESTINO
  IF %W17>3
   WRITE2FILE 1 %session, ===== FIN c2cTarot.ivr- INTENTOS fallados en el
cliente =====
   GOTO FIN
  ENDIF
```

Si no se ha podido lograr ponerse en contacto con el cliente pasados 3 intentos se finaliza el script.

```
  PLAYFILE 0 c2ctarot\intentoscliente
  INPUT 0 25 1 15
  IFERR 0
   IF %W25==1
```

```
    WRITE2FILE 1 %session, Tecla pulsada: %w25. Llamo al cliente intento %W17.
    PLAYTONE 0 0
    GOTO LLAMA_DESTINO
   ELSE
    WRITE2FILE 1 %session, Tecla pulsada: %w25. Llamada finalizada intento
%W17.
    HOOK 0
    GOTO FIN
   ENDIF
  IFERR 1
   GOTO FALLO_DESTINO
  ENDIFERR
```

Cuando el cliente no contesta, el especialista elige si quiere intentarlo otra vez o no mediante una locución. Si pulsa 1 se salta a la etiqueta LLAMA_DESTINO, si por lo contrario selecciona otra tecla cuelga el canal y se finaliza el script.

```
:LLAMA_DESTINO

  CALL2EXTERIOR 1 %v6 40
  IFERR 1
   WRITE2FILE 1 %session, Timeout cliente
   INCLOCAL 17
   GOTO FALLO_DESTINO
  IFERR 2
   WRITE2FILE 1 %session, Comunicando cliente
   INCLOCAL 17
   GOTO FALLO_DESTINO
  IFERR 0
   CONNECT 0 1
   GOTO CONEXION
  ENDIFERR
```

WRITE2FILE 1 %session, Otro error en LLAMA_DESTINO
INCLOCAL 17
GOTO FALLO_DESTINO

Caso idéntico que la sección LLAMA_ORIGEN pero en este caso con el cliente (destino)

:LLAMA_DESTINO2

IF %W18>3
WRITE2FILE 1 %session, ===== FIN c2cTarot.ivr- INTENTOS fallados en el cliente sin conexion tarotista=====
HOOK 1
GOTO
REINTENTO_LLAMADA ENDIF

SI no se puede poner en contacto con el cliente en 3 ocasiones se salta a la etiqueta REINTENTO_LLAMADA

CALL2EXTERIOR 1 %v6 30
IFERR 1
WRITE2FILE 1 %session, Timeout cliente
INCLOCAL 18
GOTO LLAMA_DESTINO2
IFERR 2
WRITE2FILE 1 %session, Comunicando cliente
INCLOCAL 18
GOTO LLAMA_DESTINO2
IFERR 0
GOTO CONEXION2
ENDIFERR

:CONEXION

```
STRSUB 20 0 2 %time
STRSUB 21 3 2 %time
STRSUB 22 6 2 %time
OPENFILE 2 0 c2cTarotistas\%V5%W20%W21%W22.txt
CONNECT 0 1
ONHOOK 0 CUELGUE_0
ONHOOK 1 CUELGUE_1
SETLOCAL 11 %timespent      ;tiempo inicio llamada
SETLOCAL 16 %V7/60          ;precio en segundos
WRITE2FILE 1 %session, PrecioMin=%V7, PrecioSeg=%W16.
WRITE2FILE 1 %session, Conversación en curso.
PAUSE 2700                  ;El maximo de la conversación son 45 min
WRITE2FILE 1 %session, Fin conversacion.
GOTO CALCULAR_TIEMPO
```

En caso de conexión entre ambos canales, se almacena la hora, se crea el fichero donde se guardarán los datos de la llamada y se inicializa el tiempo transcurrido de la conversación. Por defecto para que ninguna conversación se quede colgada se estima una duración máxima de 2700 segundos. También hay que indicar que con el comando ONHOOK se salta a la etiqueta correspondiente cuando uno de los canales cuelque y acabe la conversación.

:CONEXION2

```
PLAYFILE 1 c2ctarot\locuciontarotistaocupado
GOTO REINTENTO
```

Locución al cliente indicando que el tarotista está ocupado.

:CUELGUE_0

WRITE2FILE 1 %session, Ha colgado el tarotista.
HOOK 0
HOOK 1
GOTO CALCULAR_TIEMPO

Cuelga el origen pero se cierren los dos canales y se salta a la etiqueta CALCULAR_TIEMPO.

:CUELGUE_1

WRITE2FILE 1 %session, Ha colgado el cliente.
HOOK 0
HOOK 1
GOTO CALCULAR_TIEMPO

Cuelga el destino pero se cierren los dos canales y se salta a la etiqueta CALCULAR_TIEMPO.

:CALCULAR_TIEMPO

EVAL 15 %timespent - %W11 ;tiempo transcurrido de conversacion

WRITE2FILE 1 %session,Duracion=%W15,Coste=%W16*%W15;
SEND2SOCKET 1 36 10
12A,''org.gjt.mm.mysql.Driver'',''jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=yes&characterEncoding=UTF-8'',''user'',''pass'';
SEND2SOCKET 1 37 10 12B,%w36,''UPDATE llamadas SET duracion=%W15,coste=%V7/60*%W15,telefono_llamada='%W12' WHERE idLlamada = %V5'';

```
WRITE2FILE 1 %session,UPDATE llamadas SET
duracion=%W15,coste=%V7/60*%W15,telefono_llamada='%W12' WHERE
idLlamada = %V5;

SEND2SOCKET 1 37 10 12B,%w36,"SELECT idTPV FROM tarotistas
WHERE idTarotista=%V4";
SEND2SOCKET 1 38 10 12C,%w36,%w37;

SETLOCAL 12 %W38   ;idTPV

SEND2SOCKET 1 39 10 12D,%w36,%w37;
SEND2SOCKET 1 37 10 12B,%w36,"SELECT coste FROM llamadas WHERE
idLlamada = %V5";
SEND2SOCKET 1 38 10 12C,%w36,%w37;

SETLOCAL 13 %W38   ;coste

SEND2SOCKET 1 39 10 12D,%w36,%w37;
SEND2SOCKET 1 39 10 12E,%w36;

WRITE2FILE 2 %W12 %W13 %V6 %V5 ;Fichero para cobrar
WRITE2FILE 1 %session,SELECT idTPV WHERE idTarotista=%V4;
WRITE2FILE 1 %session, idTPV: %W12, Tiempo Conv: %W15, Coste:
%W13 GOTO FIN
```

Se calcula el tiempo transcurrido se multiplica por el precio del minuto del especialista y se actualiza el coste de la llamada en la tabla “llamadas”. También se escribe en el fichero generado de la llamada el idTPV, el coste total de la llamada, el teléfono del especialista que atiende la llamada y el id de llamada.

```
:REINTENTO
 ONHOOK 1 REINTENTO_LLAMADA ; Si cuelga se inicia todo el proceso

 INPUT 1 26 1
 20 IFERR 0
  IF %W26==1
   SETLOCAL 27 0
   WRITE2FILE 1 %session, Tecla pulsada: %w26. Reintentamos en 5
   min GOTO REINTENTO_LLAMADA
  ELSE
   IF %W26==2
    SETLOCAL 27 1
      PLAYFILE 1 c2ctarot\coors 4
    WRITE2FILE 1 %session, Tecla pulsada: %w26. Otro tarotista aleatorio
que este libre.
```

Se consulta la tabla aleatoriamente de quien esté libre en ese momento con estado=1 y con la misma especialidad o con la especialidad general (6). Se llama directamente.

```
 SEND2SOCKET 1 36 10
12A,"org.gjt.mm.mysql.Driver","jdbc:mysql://DirIP:Puerto/tarotistas?useUnicode=y
es&characterEncoding=UTF-8","user","pass";
    SEND2SOCKET 1 37 10 12B,%w36,"SELECT especialidad FROM tarotistas e
where e.idTarotista=%V4";
    SEND2SOCKET 1 38 10
    12C,%w36,%w37; SETLOCAL 28 %W38
    IF %w38 == X ;Si no hay resultados SETLOCAL 28 6 ;
     si falla por defecto tarot general(6)
    ENDIF
      IF %W38 == ERR
     SETLOCAL 28 6 ; si falla por defecto tarot general(6)
      ENDIF
```

```
        WRITE2FILE 1 %session, Especialidad es %W28
        SEND2SOCKET 1 37 10 12B,%w36,"SELECT
telefono1,telefono2,telefono3,t.idTarotista,Precio from tarotistas t,agenda a WHERE
status=1 AND especialidad=%W28 AND t.idTarotista!=%V4 AND dia=%day AND
hora_desde<='%time' AND hora_hasta>='%time' ORDER BY RAND() LIMIT 0,1";
        WRITE2FILE 1 %session, CONSULTA ALEATORIO: "SELECT
telefono1,telefono2,telefono3,idTarotista,Precio from tarotistas,agenda WHERE
status=1 AND especialidad=%W28
        WRITE2FILE 1 %session, AND idTarotista!=%V4 AND dia=%day AND
hora_desde<='%time' AND hora_hasta>='%time' ORDER BY RAND() LIMIT 0,1"
        SEND2SOCKET 1 38 10 12C,%w36,%w37;
      IF %w38 == X                          ;Si no hay resultados
       WRITE2FILE 1 %session, No se ha encontrado ningun tarotista.
       GOTO INICIO
      ENDIF
        IF %W38 == ERR
       WRITE2FILE 1 %session, No se ha encontrado ningun tarotista.
       GOTO INICIO
        ENDIF
        STRTOKEN 1 1 | %W38  ;telefono1 -> v1
        STRTOKEN 2 2 | %W38  ;telefono2 -> v2
        STRTOKEN 3 3 | %W38  ;telefono3 -> v3
        STRTOKEN 4 4 | %W38  ;idTarotista -> v4
        STRTOKEN 7 5 | %W38  ;Precio -> v7
      SEND2SOCKET 1 39 10 12D,%w36,%w37;
      SEND2SOCKET 1 39 10 12E,%w36;
      WRITE2FILE 1 %session, Nuevo Tarotista -- idTarotista=%V4, telefono1=%V1
        PAUSE 20
        GOTO TAROTISTA_ALEATORIO
    ELSE
     IF %W26==3
```

```
      WRITE2FILE 1 %session, Tecla pulsada: %w26. FIN.
        HOOK 1
        GOTO FIN
     ENDIF
    ENDIF
   ENDIF
IFERR 1
 GOTO CONEXION2
ENDIFERR
```

Si el usuario selecciona la tecla 1 se corta la llamada, se hace una pausa de 5 minutos y se vuelve a iniciar todo el proceso del script. Si el usuario pulsa la tecla 2 se selecciona un tarotista aleatorio y se repite el proceso. Si por el contrario selecciona la tecla 3 la llamada se corta y no se vuelve a intentar.

```
:ORIGEN_NO_DISPONIBLE

 WRITE2FILE 1 %session, ===== TALEAT=%W27 Llamamos al cliente con
locucion de que esta ocupado el tarotista. =====
 ;GOTO FIN2
 SETLOCAL 18 1
 IF %W27 == 0
  GOTO LLAMA_DESTINO2
ELSE
  GOTO
CONEXION2 ENDIF
```

Como se ha indicado anteriormente la variable 27 guarda el valor de 0 si aun no se ha seleccionado un tarotista aleatorio y 1 en caso contrario. Si se ha seleccionado un tarotista aleatorio el usuario ya está conectado y únicamente se le pasa la locución. Por el contrario se llama al usuario.

:REINTENTO_LLAMADA

HOOK 1
PAUSE 300
GOTO INICIO

Se cuelga el canal del destino, se hace una pausa de 5 minutos y se inicia el proceso

:FIN

WRITE2FILE 1 %session, ===== FIN c2cTarot.ivr- IDLLAMADA=%V5 =====
EXIT

:FIN2

WRITE2FILE 1 %session, ===== FIN c2cTarot.ivr-
Forzado===== EXIT

“Fichero de logs”

Ahora se va a presentar una captura del fichero de log donde se puede observar los mensajes que deja el script c2cTarotBucle y cuando se produce una llamada.

```
2012/07/11 09:34:28.167 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:34:46.292 ===== 4,No hay llamadas  =====
2012/07/11 09:34:47.230 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:34:54.105 ===== 4,1 llamada(s) por procesar  =====
2012/07/11 09:34:54.839 ===== 4,42|601275777|||115|76|659264335|1.1|0 =====
2012/07/11 09:34:54.855 32, ===== INICIO c2cTarot.ivr =====
2012/07/11 09:34:55.371 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:34:55.511 32,UPDATE llamadas SET estado = 1 WHERE idLlamada = 76
2012/07/11 09:35:09.730 32, Whisper en LLAMA_ORIGEN
2012/07/11 09:35:09.730 32, Contador= 1
2012/07/11 09:35:11.933 32, Input_whisper
2012/07/11 09:35:12.074 32, Tecla pulsada: 2. Llamada finalizada
2012/07/11 09:35:12.246 32, ===== TALEAT=0 Llamamos al cliente con locucion de que esta ocupado el tarotista. =====
2012/07/11 09:35:13.168 ===== 4,No hay llamadas  =====
2012/07/11 09:35:14.105 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:35:32.090 32, Tecla pulsada: 2. Otro tarotista aleatorio que este libre.
2012/07/11 09:35:32.230 ===== 4,No hay llamadas  =====
2012/07/11 09:35:32.699 32,     Especialidad es 6
2012/07/11 09:35:32.856 32, CONSULTA ALEATORIO: "SELECT telefono1,telefono2,telefono3,idTarotista,Precio from tarotistas,agenda WHERE status=1 AND especialidad=6
2012/07/11 09:35:32.856 32,     AND idTarotista!=115 AND dia=3 AND hora_desde<='09:35:32' AND hora_hasta>='09:35:32' ORDER BY RAND() LIMIT 0,1"
2012/07/11 09:35:33.168 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:35:33.324 32, Nuevo Tarotista -- idTarotista=54, telefono1=856921462
2012/07/11 09:35:51.293 ===== 4,No hay llamadas  =====
2012/07/11 09:35:52.246 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:35:54.903 32,UPDATE llamadas SET telefonoTarot1='856921462' ,telefonoTarot2='663482081',telefonoTarot3='',precio='1.45', idTarotista='54' WHERE idLlamada
2012/07/11 09:36:10.200 ===== 4,No hay llamadas  =====
2012/07/11 09:36:11.293 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:36:29.262 ===== 4,No hay llamadas  =====
2012/07/11 09:36:30.216 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:36:35.200 32, Timeout en LLAMA_ORIGEN
2012/07/11 09:36:35.200 32, Contador= 2
2012/07/11 09:36:37.856 32, Whisper en LLAMA_ORIGEN
2012/07/11 09:36:37.856 32, Contador= 2
2012/07/11 09:36:46.153 32, Input_whisper
2012/07/11 09:36:48.169 ===== 4,No hay llamadas  =====
2012/07/11 09:36:49.106 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:37:02.232 32, Timeout input_WHISPER
2012/07/11 09:37:07.232 ===== 4,No hay llamadas  =====
2012/07/11 09:37:08.169 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:37:10.513 32, Input_whisper
2012/07/11 09:37:26.310 ===== 4,No hay llamadas  =====
2012/07/11 09:37:26.310 32, Timeout input_WHISPER
2012/07/11 09:37:27.232 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:37:34.576 32, Input_whisper
2012/07/11 09:37:45.201 ===== 4,No hay llamadas  =====
2012/07/11 09:37:46.138 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:37:50.201 32, Timeout input_WHISPER
2012/07/11 09:37:58.482 32, Input_whisper
2012/07/11 09:38:04.264 ===== 4,No hay llamadas  =====
2012/07/11 09:38:05.201 ===== 4,No hay más llamadas pendientes, vuelvo al bucle inicial =====
2012/07/11 09:38:08.029 32, ===== FIN c2cTarot.ivr- IDLLAMADA=76 =====
```

“Sistema de cobro”

El sistema de cobro es una tarea creada en C# que se ejecuta cada 3 minutos comprobando si hay ficheros nuevos por cobrar en un directorio de la centralita SUTIL. Se analiza el fichero donde se encuentra el id de la llamada, el teléfono de contacto, el id del TPV del especialista y el importe. Una vez procesado se añade al final del fichero la cadena “COBRADO” para tener un histórico de todas las llamadas por si hubiera algún problema.

```
foreach (string file in Directory.GetFiles(directorioLocal))
        {

            contenido = File.ReadAllText(file, Encoding.Default);
            if(!contenido.EndsWith(" COBRADO"))
            {
               arrayContenido =contenido.Split(' ');
               idTPV = arrayContenido[0]; importe
               = arrayContenido[1]; telefonoCliente
               = arrayContenido[2]; idLlamada =
               arrayContenido[3];
               try
               {
                  string query = "SELECT HEX(AES_SCRIPT('" + idLlamada + "','" +
clave + "'))";
                  MySqlDataAdapter da = new MySqlDataAdapter(query, con);
                  DataSet ds = new DataSet();
                  da.Fill(ds, "tabla");
                  con.Close(); //Datos

                  int tam = 1;
                  for (int i = 0; i < tam; i++)
                  {
                     idLlamadaEncriptado = ds.Tables["usuarios"].Rows[i][0].ToString();
                  }
```

Se recorren todos los ficheros contenidos en el directorio de la maquina SUTIL y se comprueba cuales están cobrados o no. Si un fichero no está cobrado se encripta el id de la llamada junto con una clave para pasarla al TPV.

```
                //TPV
                try
                {
                    WebRequest req = HttpWebRequest.Create
("http://www.premiumnumbers.eu/tpv/tpvtlt.aspx?idll=" + idLlamadaEncriptado +
"&imp=" + importe);
                    req.Method = "GET";
                    req.ContentType = "application/text/html; charset=utf-8";
                    WebResponse response = req.GetResponse();
                    StreamReader rea = new
StreamReader(response.GetResponseStream(), System.Text.Encoding.UTF8);
                    responseStr = rea.ReadToEnd();
                    arrayRespuesta = responseStr.Split(',');
                    if (arrayRespuesta[0] == "OK" && arrayRespuesta[1] == idTPV)
                    {
                        File.AppendAllText(file, " COBRADO");
                        string query2 = "SELECT HEX(AES_SCRIPT('" + idLlamada +
"','" + clave + "'))";
                        MySqlDataAdapter da2 = new MySqlDataAdapter(query2, con);
                        DataSet ds2 = new DataSet();
                        da2.Fill(ds, "tabla");
                        con.Close();
                    }
                }
                catch { con.Close(); }
```

Se hace una petición al TPV con el id encriptado que se ha explicado anteoriormente. Se encripta porque la base de datos está ubicada en Paris y el TPV en Madrid y se desaconseja que los datos viajen desencriptados.

Una vez se ha hecho la petición, esta devuelve OK o KO. Si devuelve OK se marca el fichero como “COBRADO” y se pasa al siguiente. Si por un problema ajena devuelve KO no se hace nada con el fichero y en la siguiente ejecución se volvería a procesar.

6.- AMPLIACIONES

En este apartado se van a comentar los aspectos del proyecto que no formaban parte del proyecto original pero que viendo su viabilidad se llevaron a cabo y otras nuevas pendientes para hacer en un futuro próximo.

Se puede destacar la insistencia en que la llamada entre el usuario y el especialista se produzca hasta que una de las partes indique a través de su teléfono que no quiere continuar. Esto se traduce, por ejemplo, cuando un usuario no recibe la llamada porque está ocupado, se vuelva a intentar más adelante o cuando el usuario no indica que quiere finalizar la llamada se intenta llamar cada 5 minutos.

También podemos agregar una mejora en el apartado de seleccionar un especialista aleatorio. En este caso en vez de usar una consulta a la base de datos podríamos directamente poner como destino un gabinete con una centralita. De esta manera no se perdería tiempo buscando un especialista sino que se llamaría directamente a un número mediante el comando CALL2EXTERIOR.

Una mejora de futuro sería dotar al proyecto de un nuevo motor SUTIL que repartiera la carga entre ambos motores. Uno trataría las llamadas pares y otro las impares. Esta mejora solo sería útil si se desbordara un motor debido a numerosas llamadas.

7.- PRESENTACION DE RESULTADOS OBTENIDOS

Este apartado trata sobre cambios realizados posteriores a la concepción del proyecto en sí.

Se puede destacar el paso de la base de datos del servidor de Madrid a Paris. No se quiere saturar el servidor de Madrid y se quiere ofrecer un servicio rápido sin penalización. Se han realizado las pruebas oportunas y el tiempo entre que se hace una petición de consulta a la base de datos y ésta devuelve el resultado es de 3 segundos, un tiempo asumible.

Al estar ubicada la base de datos en París y el motor SUTIL en Madrid se ha optado por encriptar toda la información vulnerable que circula por la red. Se gana en seguridad y casi no se pierde en prestaciones.

Debido a que el desarrollo de la parte web ha ido en paralelo con la parte del script. Las pruebas generales sobre en el script consistían en ir cambiando el estado de una llamada en la base de datos directamente. El resultado siempre ha sido satisfactorio. La generación del fichero con la llamada y el importe coincide con los segundos hablados por el coste del minuto.

Han existido más problemas a la hora de integrar la parte web con el script, ya que antes de dar paso a la llamada se realiza una preautorización de la tarjeta para ver si tiene saldo. Para resolver este problema se ha optado por agregar un estado más a la llamada que indica si está preautorizada o no y puede realizarse la llamada antes de insertarse en la base de datos que está continuamente consultando el script. Debido a qué esta parte no dependía de este proyecto se han retrasado un poco los tiempos.

Finalmente todo está perfectamente integrado.

8.- CONCLUSIONES

El proyecto realizado ha cumplido con las expectativas de la empresa. Es un sistema robusto que no se ha conseguido que se quede "colgado". Todo funciona debidamente aunque siempre se intentan optimizar zonas de código y realizar llamadas de pruebas.

Este proyecto se puede extender a otros ámbitos donde se quiera tener un servicio *Click2Call* con tan solo cambiar la base de datos porque el sistema de llamada usuario-especialista es idéntico para todos.

La programación del script no ha sido costosa conceptualmente ya que he mirado numerosos ejemplos y existe una guía precisa sobre que hace cada comando acompañado de un ejemplo.

La parte del cobro de las llamadas tampoco ha sido costosa porque he trabajo bastante con C# Visual Studio

La base de datos ha ido creciendo a medida que necesitábamos almacenar más información, puede que haya sido un error pero no se podía prever con antelación los numerosos campos que se iban a utilizar. De todas formas, esto no ha sido un impedimento.

A título personal me he enfrentado a una parte de la Informática que desconocía como son las centralitas. He conseguido hacer viable el proyecto y he participado en un grupo de trabajo donde ha reinado un buen ambiente de trabajo.

Esto me da motivación para enfrentarme a nuevos retos y poder desarrollarme profesionalmente en este ámbito de la Informática que es muy amplio.

9.- ANEXO I

Listado de los comandos para los scripts de la centralita SUTIL utilizados en el proyecto y algunos más para ver las posibilidades que ofrece.

EXIT

Descripción: Finaliza la ejecución del fichero script.
Cierra todos los canales que queden abiertos y no hayan sido asociados entre sí.
Si el script tenía bloqueada alguna variable global la desbloquea.
Un EXIT en un script hijo provoca también un EXIT en el script Padre.

PAUSE *[Tiempo]*

Descripción: Espera el número de segundos indicado antes de continuar. *Tiempo* debe estar comprendido entre 0 y 2147483647 .
Si no indicamos Tiempo, se hace una pausa hasta que ocurra alguna interrupción (ONHOOK, RESERVEEXTENSION u ONSOCKET)

GOTO *etiqueta*

Descripción: Desvía la ejecución a la etiqueta indicada.

GOSUB *etiqueta*

Descripción: Desvía la ejecución a la etiqueta indicada, dejando la posibilidad de volver a la siguiente instrucción usando el comando RETSUB.
El tamaño máximo de la cola de direcciones de retorno es de 64.
Este comando no puede ser ejecutado dentro de un bloque SELECTCASE.

RETSUB

Descripción: Devuelve la ejecución a la siguiente instrucción del último comando GOSUB.

CALL *FicheroIVR Arg1 Arg.*

Descripción: Ejecuta un fichero IVR ajeno. El fichero debe tener la extensión ".IVR", en el comando se puede indicar el nombre del fichero con la extensión o sin ella. El actual fichero se queda parado hasta que el llamado retorne. Los argumentos que se pasan son copiados en las variables locales %v0, %v1, ... del fichero llamado. Dichas variables locales del fichero llamado son propias de este y cualquier modificación en ellas no varía nada en el fichero original.

Los canales, sin embargo, sí son comunes a ambos ficheros, por lo que el script hijo puede utilizar los canales abiertos en el script padre y cualquier modificación sobre estos actuará de forma definitiva para los dos, y de igual forma en el script padre puede utilizar los canales abiertos por un script hijo.

De igual forma el valor que se haya podido definir para nuestro CallerID también afectara a las llamadas externas que se realicen desde el fichero llamado y cualquier modificación que se haga del CallerID en el fichero llamado, afectara, al volver, a las llamadas que se hagan desde el fichero superior.

Las variables del sistema %nums, %numd y %op mantiene en un script hijo el contenido que tenían en el scritp padre.

Los comandos OPENSOCKET y OPENFILE hechos en un script se mantienen en un script hijo o en un script padre.

Una variable global bloqueada por un script puede ser desbloqueada por un script hijo,... o por un script padre.

Un comando EXIT ejecutado en un script hijo, hace que se ejecute también un comando EXIT en el script padre.

Asimismo un cuelgue estando en el script hijo, hace que se ejecute lo que este configurado por un comando ONHOOK en dicho script (o los valores por defecto) y posteriormente si se ejecuta un RETURN (NO un EXIT), al volver al script padre ejecuta lo que este configurado por un comando ONHOOK en dicho script (o los valores por defecto).

El retorno de una llamada CALL puede leerse en la variable %ret. Pueden anidarse llamadas CALL entre distintos ficheros.

Salidas anómalas: 4.- Fichero inexistente.

RETURN ***Valor***

Descripción: Retorna la ejecución a la siguiente instrucción al comando CALL previamente ejecutado para llamar al script en curso. Hace que la variable %ret tome el valor especificado por *Valor.*

Ejecutar un comando RETURN en un script que no ha sido ejecutado por un comando CALL, equivale a ejecutar un comando EXIT.

INVOKE ***FicheroIVR Arg1 Arg2***

Descripción: Ejecuta un fichero IVR ajeno. El fichero debe tener la extensión ".IVR", en el comando se puede indicar el nombre del fichero con la extensión o sin ella. El actual fichero continua su ejecución normalmente. Los argumentos que se pasan son copiados en las variables locales %v0, %v1, ... del fichero llamado.

Dichas variables locales del fichero llamado son propias de este y cualquier modificación en ellas no varía nada en el fichero original. Los canales y el valor que se le haya podido definir a nuestro CallerID, también son independientes, por lo que en el momento en el que comienza la ejecución del fichero invocado son totalmente independientes uno del otro.

Si uno de los parámetros que se pasa es una variable, todo el contenido de la variable pasa a la variable correspondiente en el fichero script invocado, incluso si contiene espacios en blanco.

Salidas anómalas: 4.- Fichero inexistente.

SETLOCAL ***NumVar [Cadena]***

Descripción: Asigna a la variable local indicada por NumVar el valor de *Cadena.* Esta *Cadena* puede contener espacios en blanco, en cualquier caso se le asigna a la variable todo el contenido después de NumVar hasta el final de la línea.

Si no se incluye *Cadena*, se le asigna a la variable una cadena vacía.

Salidas anómalas: 4.- Variable no existente (mayor que 39).

DECLOCAL ***NumVar***

Descripción: Decrementa el valor contenido en la variable local indicada por NumVar. Si la variable contiene un valor no numérico, se interpreta como 0, pasando a tomar el valor -1.

Salidas anómalas: 4.- Variable no existente (mayor que 39).

INCLOCAL ***NumVar***

Descripción: Incrementa el valor contenido en la variable local indicada por NumVar. Si la variable contiene un valor no numérico, se interpreta como 0, pasando a tomar el valor 1.

Salidas anómalas: 4.- Variable no existente (mayor que 39).

EVAL ***NumVarLoc ExpAlgebraica***

Descripción: Evalúa una expresión algebraica que puede utilizar +, -, * ó / de números enteros y/o variables, se pueden utilizar paréntesis, se pueden insertar espacios en blanco dentro de la expresión algebraica para hacer más clara su interpretación. Tanto el resultado final como los intermedios se evalúan como enteros. La cantidad resultante se asigna a la variable local indicada.

Salidas anómalas: 4.- Variable no existente (mayor que 39).

STRSUB ***NumVarDest Origen Longitud Cadena***

Descripción: Copia en la variable destino un sub-segmento de la longitud indicada de la cadena suministrada empezando en el carácter indicado por *Origen*. Cuando *Origen* vale cero se empieza en el primer carácter y asi sucesivamente.

Si *Longitud* es superior a los caracteres existentes desde *Origen* hasta el final de la *Cadena*, devuelve estos caracteres existentes desde *Origen* hasta el final de la *Cadena*, sin añadirle espacios en blanco ni ningún otro carácter de relleno.

Salidas anómalas: 4.- Variable no existente (mayor que 39).

STRTOKEN ***NumVarDest NumToken Separador Cadena***

Descripción: Descompone la cadena suministrada en subcadenas utilizando el separador indicado y almacena en la variable indicada el número de subcadena indicada por NumToken.

Si se indica un *NumToken* superior al número de Tokens contenidos en *Cadena*, se devuelve un error 5 y la variable indicada por *NumVarDest* vacía.

Los separadores válidos son:

Cualquier letra en mayúscula o minúscula.

Cualquier número.

! # $ % * , - . / : < = > @ ^ |

Salidas anómalas: 4.- Variable no existente (mayor que 39).

5.- Token inexistente.

INCOMINGCALL ***Accion [Causa]***

Descripción: Cuando entra una llamada al sistema y se pone en marcha un script, esta llamada se asocia con el canal 0. Sin embargo, la llamada no se acepta, rechaza o se le indica que avanza la llamada, hasta que se ejecuta este comando.

Accion puede valer:

0.- Se acepta la llamada.

1.- Se rechaza la llamada

2.- Se envía un Alert indicando que la llamada se esta cursando.

Si la llamada se recibe por una línea analógica, solo es válida la primera acción.

En el caso de de que rechacemos la llamada podemos enviar la causa por la que se rechaza, según la tabla de causas por las que no tiene éxito una llamada según el protocolo RDSI. La ejecución de este comando con la opción 1 no provoca el salto a la etiqueta que pudiera estar configurado con un comando ONHOOK.

Solo se puede utilizar la acción de enviar un Alert en el caso de que tengamos configurado *AutoAlert = 0* en el fichero Sutil.ini.

Si tenemos configurado *AutoAlert = 0* en el fichero Sutil.ini, podemos ejecutar primero este comando con *Accion = 2* y posteriormente volverlo a ejecutar con *Accion = 0* o directamente ejecutarlo con *Accion = 0* sin haberlo ejecutado previamente con *Accion = 2*.
Salidas anómalas: 6.- Código incorrecto.

CALL2EXTENSION *Canal NExt Timeout [MensajeLCD]*

Descripción: Realiza una llamada a una extensión o grupo y la asocia al canal especificado por *Canal*. Si la extensión no responde o comunica, el sistema no pasa automaticamente a la opción de dejar un mensaje. Si se quiere que esto ocurra deberá ser implementado con los correspondientes comandos dentro del fichero SCRIPT.
Canal es el canal que se abrirá, al que se asociará la extensión, debe tener un valor entre 1 y 9.
NExt es el número de extensión o grupo al que se intenta llamar.
Timeout indica el máximo tiempo en segundos a esperar, este valor debe estar comprendido normalmente entre 20 y 40. Aproximadamente cada 3 seg. equivalen a un Ring.
En el caso de que la extensión este autentificada en un teléfono externo, un valor por debajo de 20 puede ser demasiado pequeño de forma que puede que la llamada no llegue a sonar en el destino, sobre todo en el caso de llamadas a móviles o al extranjero. Un valor superior a 40 seg. puede no tener sentido, ya que normalmente las Operadoras tienen un tiempo máximo, para que tenga éxito la llamada, de alrededor de 40 sg., de forma que la Operadora daría la llamada por fallida antes del *Timeout* indicado por nosotros.
MensajeLCD es una cadena que se mostrará en el display lcd de la extensión, el tamaño máximo es 32 caracteres (incluyendo el carácter 0 de fin de cadena), si es mayor se trunca.
Cuando la extensión este autentificada en un teléfono externo, se utiliza la *Política de LCR* 0 (cero) para seleccionar la linea a utilizar para realizar la llamada saliente, si no hay definida una política de LCR 0 para llamar al número de teléfono en cuestión, Sutil realiza la llamada utilizando de forma cíclica (Round Robin) todas las líneas existentes (tanto analógicas como digitales).
Si se ha indicado un grupo en vez de una extensión, Sutil decide a que extensión del grupo debe llamar según como se haya configurado la distribución de las llamadas al grupo. Si la

extensión a la que corresponde llamar es una extensión externa y al llamarla comunica, Sutil intenta llamar a otra extensión, mientras quede alguna disponible a la que no se haya llamado.

Si el grupo tiene el parámetro RetryCall configurado a 1 (en el fichero Sutil.ini), si la extensión a la que se llama no contesta, Sutil llama a otra, mientras quede alguna disponible a la que no se haya llamado.

Salidas anómalas: 1.- Timeout.

2.- Si se está llamando a una extensión: la extensión esta comunicando o no esta activa.

Si se está llamando a un grupo: todas las ext. Del grupo están comunicando o no están activas.

4.- Si se está llamando a una extensión: la extensión no existe o no esta autentificada.

Si se está llamando a un grupo: el grupo no existe o no hay en ese momento ninguna extensión del grupo autentificada (incluyendo el caso de que el grupo no tenga definida ninguna extensión).

5.- No hay líneas disponibles para llamar (solo posible en el caso de extensiones externas).

7.- Canal en uso.

CALL2EXTERIOR *Canal NTeléfono Timeout [NLinea1] [NLinea2]*

Descripción: Hace una llamada a un número de teléfono externo y la asocia al canal especificado.

Canal es el canal que se abrirá, al que se asociará la llamada, debe tener un valor entre 1 y 9.

Nteléfono es el número de teléfono al que se intenta llamar.

Timeout indica el máximo tiempo en segundos a esperar, este valor debe estar comprendido normalmente entre 20 y 40, un valor por debajo de 20 puede ser demasiado pequeño de forma que puede que la llamada no llegue a sonar en el destino, sobre todo en el caso de

llamadas a móviles o al extranjero. Un valor superior a 40 seg. puede no tener sentido, ya que normalmente las Operadoras tienen un tiempo máximo, para que tenga éxito la llamada, de alrededor de 40 sg., de forma que la Operadora daría la llamada por fallida antes del *Timeout* indicado por nosotros. Aproximadamente cada 4,5 seg. equivalen a un Ring.

NLinea1 Nllinea2 ... indica por qué enlace(s) se debe intentar efectuar
la llamada.
Estos valores pueden ser:

Un número que indica una *Línea* , incluyendo analógicas y digitales. Si alguna de los números de línea pertenece a un enlace RDSI se intentará hacer por cualquiera de los canales del mismo.

– ***L****x*, de esta forma hacemos referencia directamente a la línea analógica número x.
– ***B****x*, de esta forma hacemos referencia directamente a todos los canales del BRI número x.
– ***P****x*, de esta forma hacemos referencia directamente a todos los canales del PRI número x.
– ***S****x*, de esta forma hacemos referencia al Servidor de enrutamiento x.

En el caso de que ninguna de las líneas indicadas estuviera disponible para realizar la llamada (por estar ocupadas o averiadas), no lo intenta por otras que pudieran existir.
En el caso de que no se especifiquen estos parámetros, se utiliza la *Política de LCR* 0 para seleccionar la línea a utilizar para realizar la llamada, y en el caso de no haber definida una *Política de LCR* 0 que incluya al teléfono a llamar, se van utilizando sucesivamente todas las líneas disponibles (tanto analógicas como digitales).

Salidas anómalas: 1.- Timeout.
2.- Comunicando.
5.- No hay líneas disponibles para llamar.
7.- Canal en uso.

CALL2EXTERIORBG *Canal NvarEstado NvarCausa NTeléfono CanalSecEsc SecEsc Timeout [NLinea1] [NLinea2]*

Descripción: Hace una llamada a un número de teléfono externo en BackGround y la asocia al canal especificado, una vez se ejecuta esta instrucción la ejecución del script

continua, mientras se esta realizando la llamada el script debe ser el encargado de vigilar el avance de la llamada, consultando la variable NvarEstado y tomar las medias oportunas (poner tono de ocupado o llamando, descolgar, rechazar la llamada, etc...).

La llamada deja de realizarse cuando la variable *NvarEstado* toma uno de los valores 2, 3, 4 ó 5, o cuando se realiza un HOOK de *Canal*.

Canal es el canal que se abrirá, al que se asociará la llamada, debe tener un valor entre 1 y 9.

NVarEstado es el número de variable donde nos va devolviendo el estado en el que se encuentra la llamada:

0.- Progresando.

1.- Alerta (solo puede aparecer en el caso de líneas digitales, PRIs o
BRIs). 2.- La llamada no ha tenido éxito.

3.- Conectado.

4.- Se ha pulsado la secuencia de Escape.

5.- Timeout

NvarCausa es el número de variable donde nos devuelve la causa RDSI en el caso de que la llamada no haya tenido éxito.

Nteléfono es el número de teléfono al que se intenta llamar. *CanalSecEsc* indica por qué canal se espera recibir la secuencia de escape.

SecEsc indica cual es la secuencia de escape, esta secuencia de escape puede ser uno o más digitos, si se pulsan estos digítos esta instrucción se detiene y en la variable *NvarEstado* se indica dicha circunstancia. Si no queremos poner una secuencia de escape debemos poner X.

Timeout indica el máximo tiempo en segundos a esperar, este valor debe estar comprendido normalmente entre 20 y 40, un valor por debajo de 20 puede ser demasiado pequeño de forma que puede que la llamada no llegue a sonar en el destino, sobre todo en el caso de llamadas a móviles o al extranjero. Un valor superior a 40 seg. puede no tener sentido, ya que normalmente las Operadoras tienen un tiempo máximo, para que tenga éxito la llamada, de alrededor de 40 seg.,

de forma que la Operadora daría la llamada por fallida antes del *Timeout* indicado por nosotros. Aproximadamente cada 4,5 seg. equivalen a un Ring.

NLinea1 Nllinea2 ... indica por qué enlace(s) se debe intentar efectuar la llamada. Estos valores pueden ser:

– Un número que indica una *Línea* , incluyendo analógicas y digitales. Si alguna de los números de línea pertenece a un enlace RDSI se intentará hacer por cualquiera de los canales del mismo.

– ***L****x*, de esta forma hacemos referencia directamente a la línea analógica número x.

– ***B****x*, de esta forma hacemos referencia directamente a todos los canales del BRI número x.

– ***P****x*, de esta forma hacemos referencia directamente a todos los canales del PRI número x.

– ***S****x*, de esta forma hacemos referencia al Servidor de enrutamiento x.

En el caso de que ninguna de las líneas indicadas estuviera disponible para realizar la llamada (por estar ocupadas o averiadas), no lo intenta por otras que pudieran existir.

En el caso de que no se especifiquen estos parámetros, se utiliza la política de LCR 0 para seleccionar la línea a utilizar para realizar la llamada, y en el caso de no haber definida una *Política de LCR* 0 que incluya al teléfono a llamar, se van utilizando sucesivamente todas las líneas disponibles (tanto analógicas como digitales). *Salidas anómalas:* 4.- Canal o nº de variables no válidos.

5.- No hay líneas disponibles para llamar.

7.- Canal en uso.

SETSELFCALLERID *[NúmeroPropio [HIDE]]*

Descripción: Define el valor de nuestro CallerID que se envía en todas las llamadas externas que hagamos en un script, a partir de esta instrucción, por una línea Digital (Básico o Primario), con los comandos CALL2EXTERIOR, CALL2EXTERIORBG o CALL2EXTENSION (en el caso de extensiones externas).

Si la llamada se realiza por una linea analógica esta instrucción no tiene ningún efecto.

Cuando realizamos una llamada por una línea digital, la operadora puede tener configurado por defecto, que se mande CallId Oculto, o que se mande uno de los números asignados a esa línea digital.

NúmeroPropio es el número de teléfono que queremos que le llegue al destino como número origen de la llamada, este NúmeroPropio, en principio, tiene que pertenecer al grupo del que forma parte el canal por el que hagamos la llamada, si no pertenece, nuestra operadora puede permitirnos que lo enviemos o puede no tenga ningún efecto y la llamada se realice acorde a lo que tenga configurado la operadora por defecto.

HIDE indica que se envíe CallId oculto, en este caso también es necesario que el NúmeroPropio indicado pertenezca al grupo del que forma parte el canal por el que hagamos la llamada, si no pertenece no tiene ningún efecto y la llamada se realiza acorde a lo que tenga configurado la operadora por defecto.

Si no se indica ningún parámetro la llamada se realiza acorde a lo que tenga configurado la operadora por defecto.

Cuando se llama a otro fichero script con el comando CALL se transmite el valor definido con esta intrucción, e igual ocurre al volver al fichero desde el que ha sido llamado un script con el comando CALL.

Cuando se lanza un fichero script con el comando INVOKE no se transmite el valor definido con esta instrucción

El valor de CallerID definido aquí, tiene prioridad sobre el valor que pudiera haberse definido en el fichero Sutil.ini para la línea que se utilice para realizar la llamada.

CONNECT *Canal1 Canal2 [Modo]*

Descripción: Conecta *Canal1* con *Canal2*.

Modo puede valer:

0 .- Es el valor por defecto. Realiza una conexión bidireccional desconectandolos antes totalmente y parando ejecuciones previas en ambos canales de comandos PLAYFILE, PLAYTONE.

1 .- Realiza una conexión unidireccional del *Canal1* al *Canal2*. Es decir el *Canal2* puede escuchar al *Canal1* pero no a la inversa.

2 .- Realiza una conexión unidireccional del *Canal2* al *Canal1*. Es decir el *Canal1* puede escuchar al *Canal2* pero no a la inversa.

Si sobre *Canal1* o *Canal2* se está ejecutando una instrucción CALL2EXTERIORBG o CALLSIMULTANEOUSBG, no se comprueba el *Modo* y se realiza automáticamente una

conexión en modo 1 si la instrucción de llamada saliente se ejecuta sobre el *Canal1* y en modo 2 si se ejecuta la instrucción de llamada saliente sobre el *Canal2*. Es decir la conexión se hace de forma que el otro
canal escuche al canal por el que se está ejecutando la instrucción CALL2EXTERIORBG o CALLSIMULTANEOUSBG.
Salidas anómalas: 4.- Canal no abierto o modo incorrecto.

HOOK C*anal [Causa]*

Descripción: Cuelga la línea indicada por *canal*, antes de colgar dicha línea, si el canal esta conectado con otro a través del comando CONNECT o con un salón con CONNECT2ROOM, lo desconecta, por lo que no es necesario realizar antes un UNCONNECT Canal.
Opcionalmente podemos enviar la causa por la que se cuelga la llamada, según la tabla de causas RDSI.
Este comando no provoca el salto a la etiqueta que pudiera estar configurado con un comando ONHOOK.
Salidas anómalas: 4.- Canal no abierto.

ONHOOK *Canal [Etiqueta]*

Descripción: En el caso de que se corte la comunicación en el canal indicado por causas exteriores al script (es decir, en cualquier caso salvo que se ejecute un comando HOOK o INCOMINGCALL 1), salta a la etiqueta indicada. Si no se indica etiqueta, no se hace nada en caso de que se corte. Por defecto, las opciones son que se acaba la ejecución del script en caso de que se produzca un corte en el canal 0 y no se hace nada en el caso de que se produzca un corte en cualquiera de los otros canales.

PLAYFILE *Canal Fichero [Opcion [NivelTraza]]*

Descripción: Reproduce en *Canal* el *Fichero* wav indicado. La reproducción en background (*Opcion* 3, 4, 5 y 6) se para por otro comando PLAYFILE, PLAYNUMBER, PLAYTIME, PLAYTONE, INPUT (Solo en los casos de *Opcion* 3 y 4, no en los casos 5 y

6), CONNECT, CONNECT2ROOM, CONNECT2SPEAKER o RECORDFILE (No se para por un RECORDFILEBG).

Fichero puede llevar incluida o no la extensión .wav .

Fichero se busca por defecto en *Suti\WavFiles* , y puede incluir un camino hasta el fichero, en cualquier caso se parte del directorio *Suti\WavFiles* , salvo que *Fichero* comience por:

"\" en este caso se parte del directorio raíz de la unidad donde se este ejecutando Sutil.exe.

"letra de unidad:" en este caso, se indique explicitamente o no, siempre se parte del directorio raíz de la unidad indicada.

"\\Nombre_ordenador\Recurso_compartido" o

"\\Dir_IP\Recurso_compartido" en estos dos casos, se indique explicitamente o no, siempre se parte del directorio raíz del Recurso_compartido.

En todos estos casos podemos utilizar "\" o "/" indistintamente.

Opcion, puede valer de 0 a 4. 0 es el valor por defecto.

0 - El mensaje parará al recibir un tono DTMF o al acabar.

1 - El mensaje parará al acabar, pero no si se reciben tonos DTMF.

2 - El mensaje se reproduce continuamente hasta que se reciba un DTMF. 3 - El mensaje se reproduce en background.

4 - El mensaje se reproduce en background continuamente.

5 - El mensaje se reproduce en background, parará al recibir un tono DTMF o al acabar.

6 - El mensaje se reproduce en background continuamente, parará al recibir un tono DTMF.

Si la reproducción se para por recibir un tono DTMF, se abre un buffer (si no estaba ya abierto) y se almacena en él, el DTMF recibido para que pueda ser recogido después por un comando INPUT.

NivelTraza, puede valer de 0 a 5, nos permite modificar el nivel con que se generan las 2 siguientes trazas:

- "No existe el fichero que se quiere reproducir" (108.09.42015.4).
- "El canal no esta abierto o no hay canales disponibles" (108.09.42016.3).

Si no ponemos nada no se modifica el nivel con que se generan estas 2 trazas.

Mientras se esta ejecutando este comando se comprueba si se produce alguna interrupción, si se produce un cuelgue del canal por el que esta reproduciendo este comando y está

deshabilitado el ONHOOK este comando da error 4 en ese momento (ademas de generar una traza 108.09.42016.3). Por lo que se recomienda tener activo el ONHOOK adecuado para el caso de que se produzca el cuelgue del canal durante la ejecución de este comando.

Salidas anómalas: 4.- Fichero inexistente o canal no abierto. 5.- Recurso de voz no disponible.

INPUT *Canal NVar Ncaract Timeout [CaracteresFin [CaractAdmitidos]]*

Descripción: Recibe una cadena de tonos DTMF en *Canal* y la introduce en la variable *Nvar*. También detecta la pulsación de la tecla R, en cuyo caso devuelve una R. A partir de la ejecución del primer comando INPUT en un canal o del corte de un comando PLAYFILE, RECORDFILE o RECORDFILEBG por la recepción de un tono DTMF, permanecerá abierto un buffer donde se almacenaran todos los tonos DTMF que se tecleen entre INPUTs. En caso de que se quieran borrar, habrá que utilizar el comando CLEARINPUT.

Ncaract es la máxima longitud. Cuando se han entrado estos caracteres, acaba.

Timeout es el máximo tiempo en segundos que espera sin pulsar ninguna tecla en el teléfono. La pulsación de la entrada completa puede tardar más tiempo.

CaracteresFin es una cadena opcional que contiene unos caracteres DTMF que provocan la finalización de la entrada. El carácter que haya provocado la salida no se incorpora a la cadena leída.

CaractAdmitidos es una cadena opcional que contiene los caracteres que serán admitidos. En el caso de existir esta cadena, cualquier carácter que no pertenezca a ella será descartado.

Salidas anómalas: 1.- Timeout.
4.- Canal no abierto o variable inexistente.
5.- Recurso de voz no disponible.

OPENSOCKET *Handle DireccionIP Puerto*

Descripción: Abre un socket para poder comunicarse con un programa externo.
DireccionIP y Puerto son los del destino donde esta escuchando el programa externo, y donde los comandos SEND2SOCKET enviarán los paquetes UDP, no se deben confundir con la dirección IP y el puerto desde donde el comando SEND2SOCKET envía los paquetes UDP, evidentemente la dirección IP desde donde el comando SEND2SOCKET envía los paquetes es la dirección IP de Sutil y el puerto lo determina el S.O. la primera vez que se ejecuta el comando SEND2SOCKET en cada script que se ejecuta, el S.O. va dando puertos consecutivos desde el 1025 que estén libres.
Salidas anómalas: 5.- No se puede abrir el socket.
7.- Handle en uso.

SEND2SOCKET *Handle NVarRespuesta Timeout CadenaEnvio*

Descripción: Envía un paquete UDP (según el formato indicado anteriormente) conteniendo *CadenaEnvio* a la dirección Ip y al puerto establecido por el comando OPENSOCKET correspondiente al *Handle* indicado. Después se queda a la escucha (el tiempo indicado por Timeout) en el puerto por el que se ha enviado el paquete UDP y esperando una respuesta que almacenará en la variable indicada.
Una vez transcurrido el tiempo indicado en timeout sin recibir una respuesta, el comando deja de escuchar en dicho puerto, y termina devolviendo el error 1. En este caso se genera una traza en el fichero Trazas.log, si no queremos que se genere dicha traza indicaremos el timeout terminando en N, es decir 10 significa timeout de 10 seg. Y 10N significa también un timeout de 10 seg. pero que no dejara traza en caso de haber timeout.
Es importante recalcar que el puerto donde se queda a la escucha el comando SEND2SOCKET es el puerto desde donde se ha enviado el paquete UDP, que no tiene nada que ver con el puerto destino a donde se ha enviado el paquete UDP. El puerto desde donde se envía el paquete UDP lo determina el S.O. la primera vez que se ejecuta el comando SEND2SOCKET en cada script que se ejecuta, el S.O. va dando puertos consecutivos desde el 1025 que estén libres.
Handle es el asignado en el comando OPENSOCKET.
NVarRespuesta vale entre 0 y 39 e indica donde almacenar el resultado.

Timeout indica el número de segundos para esperar la respuesta.

CadenaEnvio es la cadena a enviar y se toma hasta el final de la línea o de un carácter ";" (comienzo de comentario) si lo contiene dicha línea, incluyendo los espacios o tabuladores que pueda contener.

Salidas anómalas: 1.- Timeout.
4.- Variable no existente (mayor que 39).
No se ha hecho un OPENSOCKET de este Handle.
5.- Error en el envío/recepción (error de red).

OPENFILE *Handle Modo Fichero*

Descripción: Abre un fichero para escribir trazas en él.

Handle lo utiliza el comando WRITE2FILE para indicar en que fichero escribe la traza. Los valores válidos para *Handle* son entre 1 y 100.

Modo puede valer 0 ó 1 :

0 Al comienzo de cada traza no se añade la fecha y hora actual.

1 Al comienzo de cada traza se añade la fecha y hora actual.

Fichero se genera por defecto en *Suti*\ , y puede incluir un camino hasta el fichero, en cualquier caso se parte del directorio *Suti*\ , salvo que *Fichero* comience por:

"\" en este caso se parte del directorio raíz de la unidad donde se este ejecutando Sutil.exe.

"letra de unidad:" en este caso, se indique explicitamente o no, siempre se parte del directorio raíz de la unidad indicada.

"\\Nombre_ordenador\Recurso_compartido" o

"\\Dir_IP\Recurso_compartido" en este caso se parte del directorio raíz del Recurso_compartido.

En todos estos casos podemos utilizar "\" o "/" indistintamente.

Salidas anómalas: 5.- Handle no válido (fuera del rango 1-100)
7.- Handle ya en uso.

WRITE2FILE ***Handle Cadena***

Descripción: Escribe una traza en un fichero.

Handle indica en que fichero escribe la traza, este fichero se debe abrir previamente con el comando OPENFILE.

Cadena es la traza a escribir en el fichero.

Salidas anómalas: 4.- No se ha hecho un OPENFILE de este Handle.

5.- Error al escribir (Disco duro lleno, error al acceder a unidad de red).

SUTIL posee variables propias que dan información sobre el estado del sistema, como por ejemplo:

Las variables siempre se tratan como cadenas de caracteres internamente.

Hay cuatro tipos de variables: de usuario locales numéricas, de usuario locales definidas, de usuario globales y del sistema.

Las variables de usuario locales numéricas son: %v0, %v1, ... %v9, %w00, %w02, ... %w39. Cuando usamos %v solo podemos poner una cifra detrás (por eso solo podemos poner del 0 al 9) y si utilizamos %w tenemos que poner 2 cifras, pero la variable %v0 es la misma que la variable %w00 y así sucesivamente hasta la variable %v9, que es la misma que la variable %w09.

Las variables de usuario locales definidas, son variables que el usuario a declarado previamente con un nombre cualquiera. Como máximo podremos definir 60 de estas variables en un mismo script.

Las variables de usuario globales son %g0, %g1, ... %g9.

Y las variables del sistema son:

%err Resultado del último comando.

%suberr Sub-resultado del último comando. Si un resultado principal de un comando, (%err), puede ser originado por varios motivos, en esta variable distinguiremos entre las distintas causas que han provocado el resultado principal, (%err).

%numd Destino donde ha llegado la llamada, esta variable solo tiene sentido en script ejecutados para atender una llamada entrante.

En una Línea RDSI Destino es: Número de teléfono para el que viene la llamada.

En una Línea Analógica Destino es: Número de teléfono asignado a la línea por la que entra la llamada.

En un Recurso VoIP mediante protocolo SIP Destino es: Contenido a la izquierda de la @ del usuario SIP indicado en el campo *To* de los paquetes SIP.

%nums Origen desde el que nos ha llegado la llamada, esta variable solo tiene sentido en script ejecutados para atender una llamada entrante.

En una Línea RDSI Origen es: Número de teléfono desde el que nos ha llegado la llamada, en las llamadas ocultas este valor está vacío.

En una Línea Analógicas Origen es: Número de teléfono desde el que nos ha llegado la llamada, en las llamadas ocultas este valor está vacío.

En un Recurso VoIP mediante protocolo SIP Origen es:

%phandle Handle único para cada script ejecutándose que lo identifica.

%session Handle del script principal. En el caso de que el script actual haya sido llamado con el comando CALL, esta variable contiene el handle del script padre original que originó las llamadas.

%source Si en la memoria del sistema existe el Número de teléfono origen de la llamada, contiene el Nombre asociado a dicho número y si no, contiene el Numero de telefono origen de la llamada.

%op Operador por defecto, esta variable contiene el valor indicado en la columna *Operador* en la configuración de los scripts que atienden las llamadas.

%Sutil Número, dentro del Dominio al que pertenezca, del Sutil donde se esta ejecutando el script, este número se define en el fichero Sutil.ini, si no se ha definido se devuelve el valor 1, porque se entiende que solo hay un Sutil.

%ret Valor retornado en una llamada con CALL.

%timespent Número de segundos que han pasado desde el inicio del script.

%date Fecha actual en el formato: aaaa/mm/dd

%time Hora actual en el formato hh:mm:ss

%day Dia de la semana, 1 = lunes, ... 7 = domingo

10.- BIBLIOGRAFÍA:

SUTIL

- http://www.grupoei.com/

Click2Call

- http://www.innovae.com/es/servicios/software-de-seguridad-de-fabricantes/26-tecnologia-click-to-call
- http://ilifebelt.com/clic-to-call-mejorando-las-telecomunicaciones-a-traves-de-internet/2011/12/

Polibuscador (UPV)

- http://www.upv.es/entidades/ABDC/infoweb/bg/info/707225normalc.html
- **El lenguaje de programación C#**
  Francisco Javier Ceballos Sierra Madrid : Ra-ma cop. 2002
- **MySQL**
  Paul DuBois Madrid etc. : Prentice Hall D.L. 2001

Wikipedia

- http://www.wikipedia.es