Desarrollo de servicios en NGN: caso Emcali – Telecomunicaciones

NGN services development: Emcali-Telecomunicaciones case

Oscar Mondragon[1]. M.Sc. y Zeida Solarte[2]
1. Especialización en Telemática, Universidad Autónoma de Occidente, Colombia
2. Programa de Ingeniería Electrónica, Universidad Autónoma de Occidente, Colombia
ohmondragon@uao.edu.co; zsolarte@uao.edu.co

Recibido para revisión 3 de abril de 2009, aceptado 29 de enero de 2010, versión final 10 de marzo de 2010

Resumen—**Las Empresas Municipales de Cali – EMCALI adquirieron recientemente una plataforma de red multiservicios ZTE (Zhong Xing Telecommunication Equipment Company Limited) que permite la oferta de servicios integrados de voz, datos y video, con altos estándares de calidad y que soporta la arquitectura OSA/Parlay para la construcción de servicios, acordes con las tendencias mundiales actuales de convergencia propiciadas por la aparición de modelos de Redes de Nueva Generación o NGN. Para hacer eficiente la creación de servicios sobre esta plataforma es necesario definir un ambiente de desarrollo propicio que utilice las herramientas y tecnologías adecuadas, y que permita el desarrollo rápido de aplicaciones telemáticas, ya sea por EMCALI o por terceros.**

Palabras Clave—**Red de Nueva Generación, OSA/Parlay, Servicios Telemáticos, EMCALI.**

Abstract—**The Empresas Municipales de Cali - EMCALI recently acquired a ZTE (Zhong Xing Telecommunication Equipment Company Limited) multi-services network platform which permits offering integrated voice, data and video, with high quality standards and based on OSA / Parlay architecture for building services, in line with current global trends of convergence fostered by the emergence of models of Next Generation Networks or NGN. To make efficient creation of services on this platform is necessary to define a favorable development environment using suitable tools and technologies that enables the fast development of telematics applications, either by EMCALI or by third parties.**

Keywords—**Next Generation Network, OSA/Parlay, Telematic Services, Development Model, EMCALI**

I. INTRODUCCION

Recientemente la empresa de servicios públicos de la ciudad de Cali - Emcali, adquirió, para el área de telecomunicaciones, una plataforma NGN multiservicios, con el ánimo de mejorar la oferta de servicios a sus usuarios.

Esta plataforma permite ofertar accesos de banda ancha, servicios LAN to LAN, servicios de conexión a Internet, servicios de voz a través de líneas POTS, voz sobre IP, telefonía sobre IP, servicios de red inteligente, correo de voz, mensajería unificada, servicios prepago, servicios de IVR y televisión sobre IP, entre otros.

Además, la plataforma cuenta con soporte para la arquitectura Osa/Parlay que permite de una manera abierta la creación de nuevos servicios por funcionarios de la empresa ó por terceras personas.

Con el fin de aprovechar de la mejor manera dicha infraestructura, se tomó la decisión de unir esfuerzos entre la Universidad Autónoma de Occidente y las Empresas Municipales de Cali - Emcali, lo cual se concretó en un proyecto de investigación enfocado a facilitar el despliegue de servicios telemáticos.

En este documento se plasman los resultados de dicho proyecto. En primer lugar se hace una descripción de la arquitectura de las NGN, luego se describen las herramientas existentes para la creación de servicios en estos ambientes y se hace una comparación de las mismas. A continuación se describe

el proceso de desarrollo de un servicio telemático sobre la plataforma de Emcali utilizando una de las tecnologías analizadas. Por último se presentan las recomendaciones y conclusiones del proyecto.

II. MARCO TEORICO

2.1. Redes de nueva generación - NGN

Actualmente se presenta una fuerte tendencia a integrar todo tipo de servicios de telecomunicaciones en una única infraestructura de red IP. Dicha convergencia implica una evolución las soluciones IP clásicas en características como la capacidad, la calidad de servicio, la seguridad y la fiabilidad [3]. Las redes de nueva generación (NGN – Next Generation Network) proponen una arquitectura para facilitar el despliegue de una amplia gama de servicios sobre una misma infraestructura implicando la mejora de la experiencia del usuario y el aumento de los ingresos del operador.

La red de nueva generación (Next Generation Network) es una red basada en paquetes, habilitada para proveer servicios de telecomunicaciones y que usa múltiples tecnologías de transporte de banda ancha con calidad de servicio y en la cual los servicios son independientes de las tecnologías de transporte subyacentes.

Esto permite a los usuarios acceso sin restricciones a las redes, a una amplia variedad de proveedores de servicios y a servicios de su preferencia, soportando además movilidad, permitiendo el aprovisionamiento ubicuo de servicios a los usuarios.

La arquitectura NGN esta conformada en general por cuatro capas:

- Acceso: Es la capa inferior y permite a todos los usuarios conectarse a la red y acceder a los servicios que son ofrecidos a través de esta. Soporta variedad de dispositivos y tipos de redes de acceso.
- Transporte: Proporciona conectividad entre puntos finales acorde con los requerimientos de servicio, las capacidades del terminal y la disponibilidad de los recursos de red.
- Control: Suministra los elementos necesarios de señalización para facilitar el establecimiento de sesiones multimedia. Realiza el control de todas las operaciones que se llevan a cabo dentro de la red y de todos los dispositivos que hacen parte de esta. Asegura el funcionamiento entre las demás capas obteniendo así los mecanismos necesarios para la provisión de los servicios.
- Servicios: Responsable de facilitar la creación y entrega de servicios por parte de desarrolladores internos ó terceras partes a través de plataformas de servicios con interfaces abiertas.

2.2. Análisis de tecnologías para la creación de servicios sobre las redes NGN

A continuación se realizará un análisis de las tecnologías más importantes utilizadas para el desarrollo de servicios en NGN teniendo en cuenta los siguientes factores [4]: capacidades de soporte de la red, arquitectura, abstracción de interfaces, tipo de interfaces (lenguajes de descripción), capacidad para el desarrollo de servicios por terceras partes, facilidad de uso, soporte y madurez.

Para definir el criterio referente a la arquitectura se tomó como base la categorización propuesta en el proyecto P1109, cuya arquitectura se muestra en la figura 1.

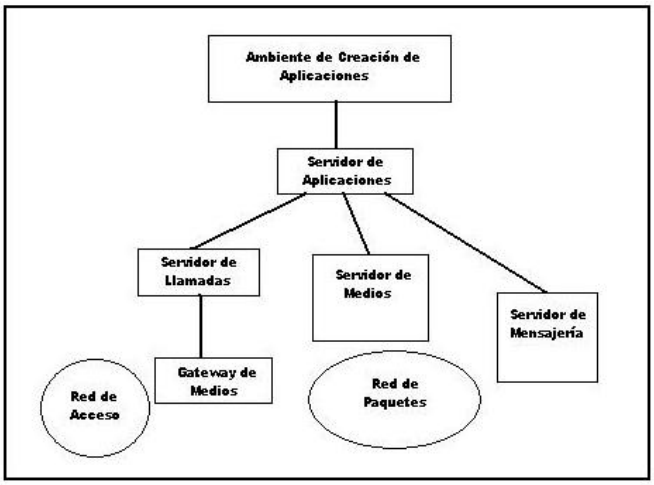


Figura 1. Arquitectura de Referencia P1109
Fuente: Analysis of NGN service creation technologies. Falcarin, 2003.

2.2.1. SIP Servlets

SIP (Session Initiation Protocol), es un protocolo que permite establecer, modificar y terminar sesiones IP multimedia. Soporta diferentes servicios como telefonía IP, mensajería instantánea, transferencia de llamada entre otros. Las aplicaciones SIP son programas que utilizan un SIP servlet el cual es un componente de aplicación basado en java que contiene un conjunto de librerías utilizadas para la creación de servicios sobre redes IP [5].

Un SIP servlet es gestionado durante su tiempo de vida por un contenedor, el cual hace parte de un servidor de aplicaciones que proporciona los servicios de red a través de los cuales se reciben y envían las peticiones y las respuestas. Aquí son alojadas las aplicaciones y el contenedor decide cual invocar y en qué orden, ya que puede recibir varias peticiones al mismo tiempo.

Las especificaciones del SIP servlet también tienen el objetivo de estandarizar los siguientes aspectos del contenedor: arquitectura, modelo de seguridad, el formato de datos (XML, DTD), formato de archivos.

SIP servlet es adecuado para servicios desarrollados por terceras partes. Se podría decir que el desarrollo de servicios por terceros es bastante simple ya que son vistos como librerías de java.

Teniendo en cuenta la referencia P1109 el contenedor SIP servlet se encuentra en el nivel de servidor de aplicación, y como es una API estándar es ideal para el desarrollo de servicios por terceras partes [4]. También tiene funciones SIP como agente de usuario y como Proxy.

2.2.2. JCC (Java Call Control)

JCC (Java Call Control) es un API, que provee una interfaz para un modelo de control de llamada genérico (GCC), esta interfaz es utilizada para implementar todos los procesos propios de un protocolo de señalización como crear sesiones, modificar, cancelar, cerrar, etc. Las características (SCF's) pueden ser invocadas o descargadas durante la configuración de sesiones. Por medio de JCC los desarrolladores pueden hacer aplicaciones para ser ejecutadas desde cualquier plataforma que soporte la API, permitiendo a los proveedores de servicios ofrecer mayor cantidad de servicios de forma rápida y eficiente, desarrollándolos ellos mismos, por outsourcing o por terceras partes [1].

JCC incluye las facilidades para observación, inicialización, respuesta, proceso, manejo de llamadas, permite también invocar aplicaciones y retornar resultados durante el desarrollo de las mismas. Una llamada puede incluir sesiones multimedia, tripartita, etc, sobre la red. El API es usada para implementar una gran variedad de aplicaciones de voz y de datos como: originar y terminar llamadas, VPN (Voice Virtual Private Network), traducción de número de teléfono, activación de marcación de voz, clic para marcar, conferencias, entre otros.

JCC está encaminado hacia redes convergentes, y está previsto para ser implementado con softswitches, call agents, proporcionando una interfaz abstracta para las aplicaciones. Puede ser utilizado para manejar sesiones en redes PSTN, redes de paquetes IP o ATM, redes inalámbricas o una combinación de ambas, sin afectar el desarrollo de los servicios usando la API.

Con respecto a la arquitectura P1109, JCC puede ser implementado de dos formas: puede ser incluida como una librería en el servidor de aplicaciones a fin de dar conectividad a la red con el call Server o puede ser efectuado directamente con el nivel de media gateway.

2.2.3. Osa/Parlay

Osa/Parlay define una arquitectura que permite la interoperabilidad entre las aplicaciones de las tecnologías de la información y las características de las telecomunicaciones en las redes, a través de una interfaz abierta estandarizada [7]. Osa/Parlay es una API que permite la creación rápida de servicios de telecomunicaciones.

Osa (Open Service Architecture), es una arquitectura flexible que hace parte de la tercera generación de redes de telecomunicaciones para servicios móviles desarrollada por el 3GPP (3rd Generation Partnership Program). Parlay es el grupo que busca crear los estándares necesarios que permita la unión de las tecnologías de la información con las telecomunicaciones.

Las API's de Osa/Parlay están definidas por el Parlay Group, el cual es una organización sin ánimo de lucro conformada por 65 empresas del área de las telecomunicaciones e industrias de las tecnologías de información. Algunas compañías miembros del Parlay Group son Alcatel, British Telecom, Ericsson, Fujitsu, HP, IBM, Lucent, NTT, Siemens, Telecom Italia entre otras.

Osa/Parlay es una tecnología independiente y fue diseñada para ser usada por redes móviles, redes fijas y redes de nueva generación (NGN), basadas en el protocolo IP. Esta tecnología se caracteriza principalmente por permitir a los operadores y aplicaciones de terceros acceder a las funcionalidades de la red por medio del conjunto de interfaces abiertas estandarizadas. Para la creación de aplicaciones los desarrolladores pueden utilizar diferentes lenguajes de programación como java y C++.

Osa/Parlay se basa en estándares abiertos como CORBA, IDL, Java, UML y servicios Web (SOAP, XML y WSDL).

La arquitectura Osa/Parlay está definida por los siguientes componentes:

- Servidor de aplicaciones: Sobre este servidor son ejecutadas todas las aplicaciones, puede estar conformado por uno o más hosts. En algunos documentos se refieren al servidor de aplicaciones como Parlay client o Parlay client Proxy. Sobre estos servidores también corren los servicios desarrollados por terceras partes. Esto es una gran ventaja ya que el proveedor del servicio utiliza las mismas interfaces para sus propios servicios y los servicios desarrollados por terceras partes.

- Parlay gateway: Este elemento del modelo Osa/Parlay permite conectar las aplicaciones utilizando las API's de Parlay con los elementos de la red. El parlay gateway se encuentra bajo el control del operador o del proveedor del servicio y es el punto por el cual pasan todos los procesos que se vayan a llevar a cabo con Osa/Parlay. Gracias al Parlay gateway las aplicaciones son independientes de los protocolos utilizados por el proveedor de la red, y las redes pueden ser modificadas sin afectar el funcionamiento de las aplicaciones y los servicios. Dentro del Parlay gateway se encuentran los servicios y el Framework.

- Servicios: El parlay gateway esta conformado por varios SCS (Service Capability Server), los cuales son entidades funcionales que proveen interfaces Osa/Parlay a través de las aplicaciones. Cada SCS es visto por las aplicaciones como uno o varios SCF's (Service Capability Features, capacidades de servicio), que son abstracciones de las

funcionalidades ofrecidas por la red e interactúan con los elementos de la red como SSP, HLR, CSE, servidores de localización entre otros. Estos SCF's reciben también el nombre de servicios y se puede acceder a ellos por medio de las API's de Parlay.

Cada SCF o varios pueden ser utilizados por una aplicación de acuerdo a la función del servicio que se vaya a implementar.

Con respecto a la arquitectura P1109 Osa/Parlay ha hecho un gran esfuerzo por estandarizar las interfaces entre el nivel del servidor de aplicaciones y todos los niveles subyacentes (Call Server, media Server, messaging Server). Actualmente se están desarrollando rápidamente servidores especializados para aplicaciones Osa/Parlay y su difusión ha sido exitosa.

Las API's de Osa/Parlay proveen un nivel medio de abstracción. Muchos de los productos de Osa/Parlay están disponibles, como Parlay gateway, Framework, servidores de aplicaciones, muchas de estas ofertadas a redes móviles. Por las facilidades que ofrece Osa/Parlay actualmente muchas empresas están desarrollando nuevos servicios utilizando esta tecnología.

2.2.3. Selección de la tecnología a usar

Se puede concluir que Osa/Parlay se destaca en el conjunto de tecnologías analizadas, dado que ofrece interfaces hacia un amplio conjunto de capacidades de la red, lo que facilita la implementación de una gran variedad de servicios avanzados de telecomunicaciones. Además, por ser una tecnología ya establecida garantiza un alto nivel de estandarización y adopción por parte de la comunidad desarrolladora de servicios y por los proveedores de los mismos.

III. DESARROLLO DE UN SERVICIO DE PRUEBA EN LA RED DE EMCALI-TELECOMUNICACIONES

La capa de servicios de la red multiservicios de EMCALI está conformada por la plataforma de servicios ZXUP10 [9], el cual esta conformado por un servidor de aplicaciones convergente, modular, orientado hacia NGN y adopta la arquitectura Osa/ Parlay. El sistema ZXUP10 tiene las siguientes características:

- Está diseñada para trabajar tanto con la red existente como con la red NGN
- Provee interfaces estándar y abiertas lo cual permite el desarrollo de servicios por terceras partes
- Soporta múltiples protocolos como INAP, CAMEL.WIP, SIP, MGCP y SMPP
- Las aplicaciones pueden ser ejecutadas desde diferentes servidores de aplicaciones reduciendo posibles fallas en el sistema
- Proporciona encapsulamiento y des-encapsulamiento de los recursos de la red

La plataforma ZXUP10 está constituida por el servidor de medios, el servidor de aplicaciones, el administrador de operaciones, la consola de mantenimiento (OAM), el gateway de señalización y el parlay gateway,

El parlay gateway es el núcleo del sistema ZXUP10, el cual encapsula varios niveles de la red de recursos, provee las interfaces API's abiertas de acuerdo al protocolo Parlay y ofrece una plataforma para el desarrollo de servicios por terceras partes. Es el enlace entre las aplicaciones y los elementos de la red.

En el parlay gateway se aloja una implementación de parlay llamada PCH (Parlay Client Hub) [8], que recibe las invocaciones de aplicaciones de terceras partes permitiéndoles comunicarse con las otras capas de la red.

Para facilitar el desarrollo de servicios por terceras partes usando PCH, se desarrolló un **M**odelo para la **C**reación de **S**ervicios en la red NGN de **E**mcali MCSE, basado en el metamodelo SCMM (Service Creation MetaModel) [6] que es un marco genérico de referencia del cual se pueden extender modelos específicos para el desarrollo de servicios en redes de nueva generación utilizando el concepto de herencia.

Basados en este metamodelo, se definen 3 capas de referencia para el modelo MCSE: Servicios, Descripción de Servicios, Creación de Servicios. Estas se muestran en la figura 2.

Figura 2. Estructura del modelo MCSE

La capa de servicios especifica los procedimientos para acceder a las APIs que definen las capacidades de servicio que provee PCH. MCSE define tres niveles jerárquicos de servicios, estos niveles son:

- Nivel 3. Este nivel ofrece capacidades de propósito general aplicables a todos los servicios y están en el nivel más alto de la jerarquía. Se basa en la definición de las capacidades de enviar eventos e invocar comandos.
- Nivel 2. En este nivel se agrupan capacidades comunes a grupos de servicios. Se clasifican los servicios en familias y se definen interfaces genéricas para cada tipo de ellos. Las

capacidades de nivel 2 heredan de las de nivel 3.

- Nivel 1. En este nivel se definen interfaces específicas para las capacidades PCH.

En la capa descripción de servicios se posibilita el anuncio y aprendizaje de las capacidades soportadas por cada uno de los servicios disponibles. Para realizar el descubrimiento de servicios se necesita el soporte de un lenguaje para describir los servicios que se exportan a través de interfaces abiertas. SCMM define un lenguaje basado en XML [2] para la descripción de servicios denominado GSDL (Generic Service Description Language), el cual se adaptó en el modelo MCSE.

La capa de creación de servicios ofrece una interfaz amigable y sencilla que encapsula los detalles específicos de la plataforma. Para ello se desarrolló un lenguaje XML denominado GSCL (Generic Service Creation Language).

Sobre el modelo MCSE se implementó la aplicación NP (Number Portability, número portable), la cual permite que cuando un usuario cambie de número telefónico siga recibiendo las llamadas que realicen a su antiguo número. Cuando se realiza una llamada a un usuario suscriptor, la aplicación recibe una notificación informando sobre este evento, seguidamente se consulta en una base de datos el número actual y se realiza la transferencia.

A continuación se presenta el diseño de la aplicación, teniendo en cuenta las tres capas del modelo.

3.1 Capa de Servicios

La aplicación define una clase principal *Portability* que extiende de la clase *DummyApplication*. *Portability* hace uso de las interfaces de PCH para su inicio, terminación, enrutamiento de llamadas, etc. En la figura 3 se muestra el diagrama de clases de la aplicación.

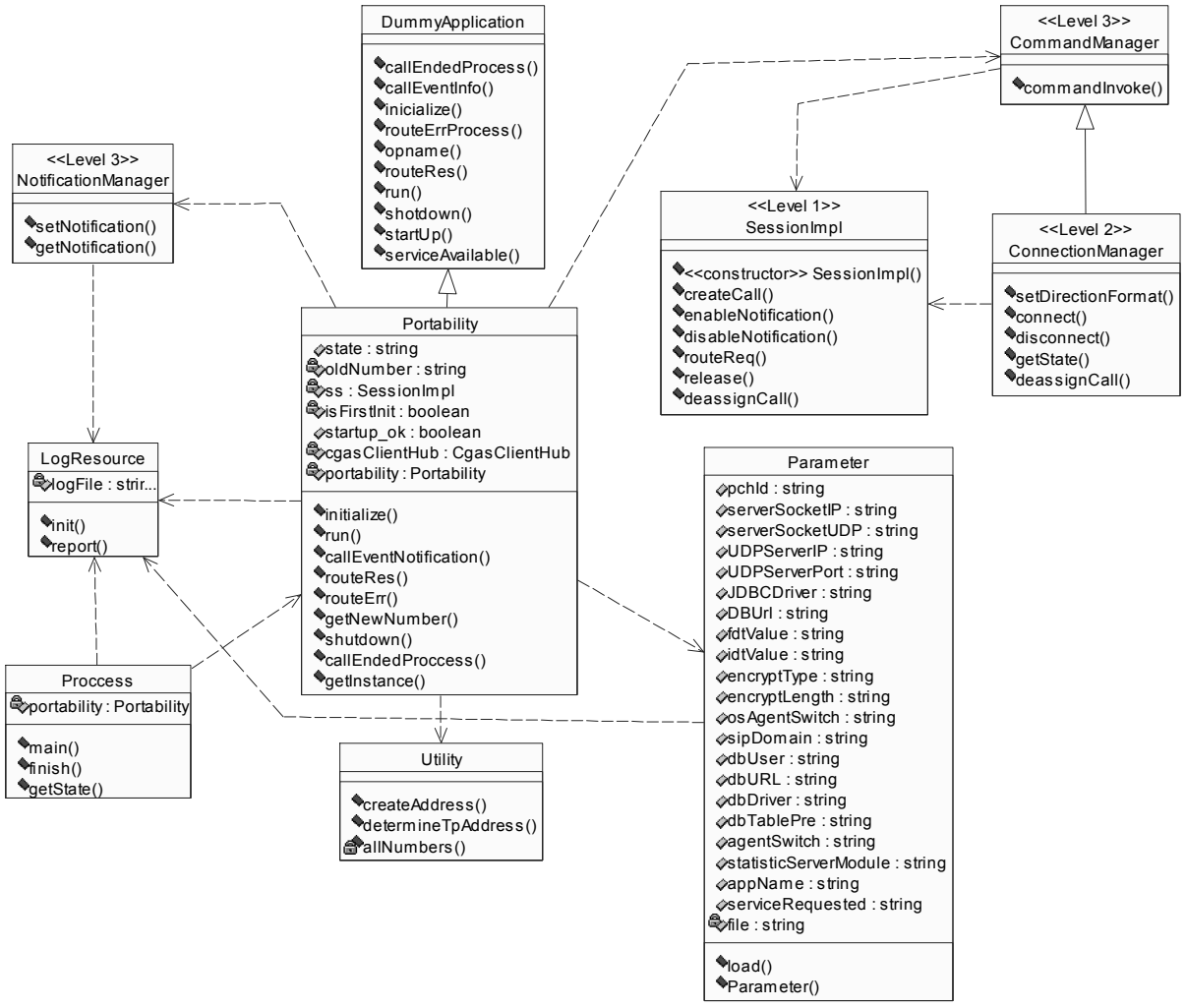


Figura 3. Diagrama de clases de la aplicación

3.1.1. Capacidad de Nivel 3

CommandManager. Esta capacidad ofrece las siguientes operaciones:

- *commandInvoke(command : String, paramerters: String) : String*. Recibe el nombre del comando a invocar con sus parámetros y devuelve una cadena con el resultado. Realiza la invocación del comando mediante la llamada al método asociado a nivel 1 (en este caso un método de *SessionImpl*) y devuelve el resultado del mismo.

  Argumentos: *command*. Recibe uno de las siguientes opciones: r*outeReq, release, clean* entre otros. *parameters*. Cadena de texto que contiene los parámetros necesarios para cada comando según la especificación de PCH.

- *getCommands()*: *String*. Da acceso a una lista de comandos soportados por una capacidad.

3.1.2. Capacidad de Nivel 2

ConnectionManager. Ofrece las siguientes operaciones:

- *connect(address : String) : String*. Llama al método *routeReq* de la capacidad de servicio de nivel 1 *SessionImpl*.

  Argumentos: *address*. En este argumento se deben concatenar los siguientes parámetros: *TpCallSessionId* (identificador de la sesión), *TpAddress* (dirección de destino), *TpAddress* (dirección de origen), *TpCallAppInfo* (Información de la aplicación), *TpCallLegConnectionProperties* (propiedades de la conexión); para lo cual se debe usar como separador entre parámetros los caracteres %$%.

- *disconnect(idr : String) : String*. Llama al método *release* de la capacidad de servicio de nivel 1 *SessionImpl*.

  Argumentos: *Idr*. En este caso null

- *getState(id : String) : String*. Realiza una llamada al método *geInfoReq* de la capacidad de servicio de nivel 1 *SessionImpl*.

  Argumentos: *id*. Se deben concatenar los siguientes parámetros: *TpcallSessionID* (identificación de la sesión) y *TpcallChargePlan* (Plan de carga a usar)

3.1.3. Capacidad de Nivel 1

La capacidad de nivel 1 corresponde a la capacidad *SessionImpl* propia de PCH que ofrece las siguientes operaciones: *createCall*, *enableNotification*, *disableNotification*, *routeReq*, *release*, *deassignCall*.

3.2 Capa de Descripción de Servicios

La descripción de las capacidades de servicio de la aplicación NP se hace en código GSDL, el cual se muestra a continuación. Esta descripción permite el descubrimiento y uso de las capacidades ofrecidas por el servicio por las aplicaciones.

```
<?xml version=»1.0" encoding=»UTF-8"?>
<GSDL xmlns=»http://ingenieria.uao.edu.co/telematica/GSDLSchema»
xmlns:xsi=»http://ingenieria.uao.edu.co/telematica/XMLSchema-
instance»>
<service name= «NP» id=»1">
    <level3>
        <NotificationManager>
        </NotificationManager >
        <CommandManager>
            <CommandInvoke>
              <Commands type=»String»>
              <Command name=»GetcallSessionID»/
        >
                    <Command
                    name=»routeReq»/>
                    <Command
                    name=»release»/>
                    <Command
                    name=»clean»/>
                    <Command
                    name=»deassignCall»/>
                  </Commands>
                    <Parameters type=»String»/>
                    <Result type=»String»/>
            <GetCommands>
                    <Result type=»String»/>
            </GetCommands>
        </CommandManager >
    </level3>
      <level2>
        <Repository/>
        <FrameworkAccess/>
        <InformationManager/>
        <UserInformationManager/>
        <ConnectionManeger>
            <Connect>
                    <Address type=»String»/>
                    <Result type=»String»/>
            </Connect>
            <Disconnect>
                    <Idr type=»String»/>
                    <Result type=»String»/>
            </Disconnect>
            <GetState>
                    <Id type=»String»/>
                    <Result type=»String»/>
            </GetState>
        </ConnectionManager>
    </level2>
    <level1>
        <Servicio nombre=»SessionImpl»
        URL=»http://ingenieria.uao.edu.co /telematica /
    SessionImpl»/>
    </level1>
</service>
</GSDL>
```

3.3 Capa de Creación de Servicios

Esta capa permite la creación rápida, flexible e intuitiva de los servicios a través de una interfaz gráfica en la cual un usuario define la máquina de estados del servicio. Esta máquina de estados se traduce en un archivo en lenguaje GSCL que es almacenado. Cuando se ejecute el servicio, el archivo GSCL se convierte en código Java ejecutable. La figura 4 muestra la máquina de estados para el servicio de portabilidad numérica.

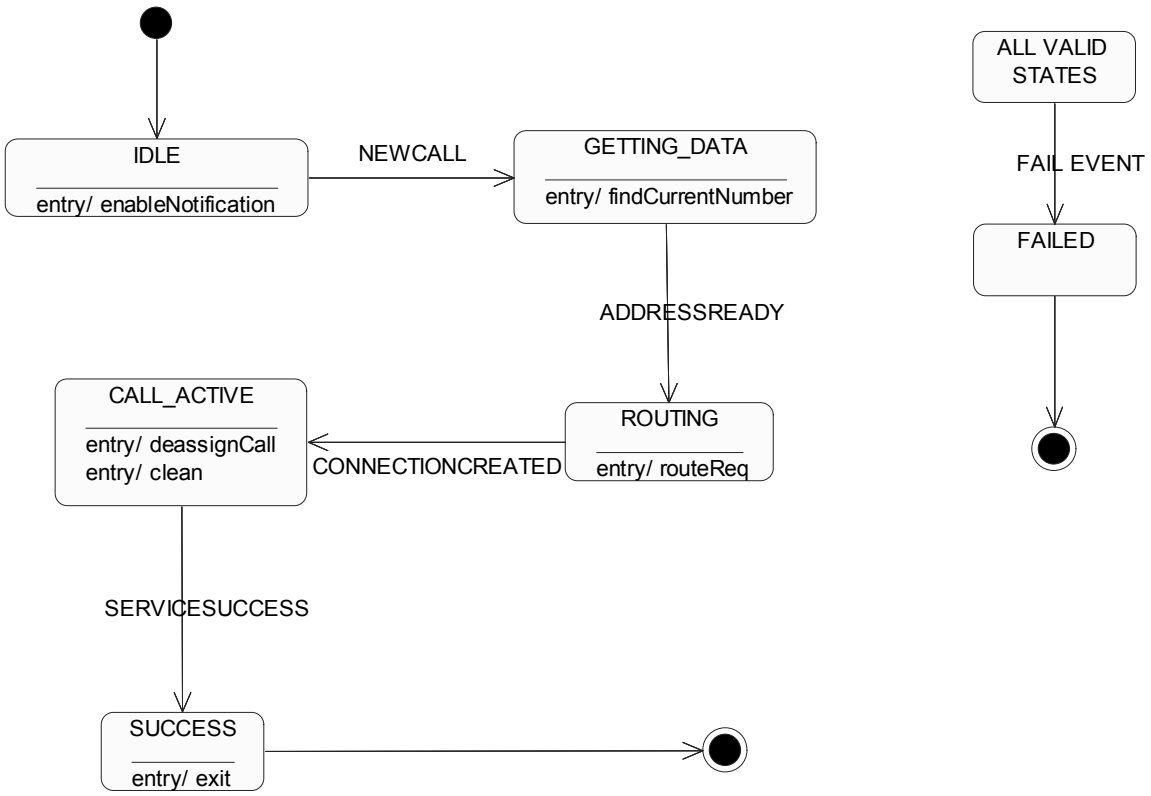


Figura 4. Máquina de estados del servicio de portabilidad numérica

Inicialmente el servicio entra en un estado IDLE donde se realiza el proceso de habilitar en el Framework la notificación, a través del método *enableNotification*, para así registrar el objeto de *CallBack* al cual se notificará cuando se presente una nueva llamada de interés para el servicio.

Posteriormente, cuando ocurra la nueva llamada se genera el evento NEWCALL y se pasa al estado GETTING_DATA, aquí se genera una acción para buscar el nuevo número del usuario llamado a través del método *findCurrentNumber*.

Una vez obtenido el nuevo número, se genera el evento ADDRESSREADY que permite iniciar el estado ROUTING en el cual se enruta la llamada a través del método *routeReq*.

Cuando el usuario llamado contesta se genera el evento CONNECTIONCREATED que inicia el estado CALL_ACTIVE.

Una vez la llamada se active, la aplicación se desentiende de su desenlace y la libera, es decir la deja de monitorear esto se logra a través de los métodos *deassignCall* y *clean*.

En este momento se considera exitosa la ejecución del servicio y se genera el estado SUCCESS.

Una vez en el estado SUCCESS se ejecuta la acción de salir de la aplicación a través del método *exit*.

Cada estado valido tiene un listado de eventos aceptables que pueden ocurrir mientras el servicio se encuentra en el estado. En cualquier momento si ocurre un evento inesperado se genera un evento de error y se pasa al estado FAILED.

3.4 Diagrama de secuencia

En la figura 5 se muestra el diagrama de secuencias para el servicio de portabilidad numérica para el caso de una llamada exitosa.

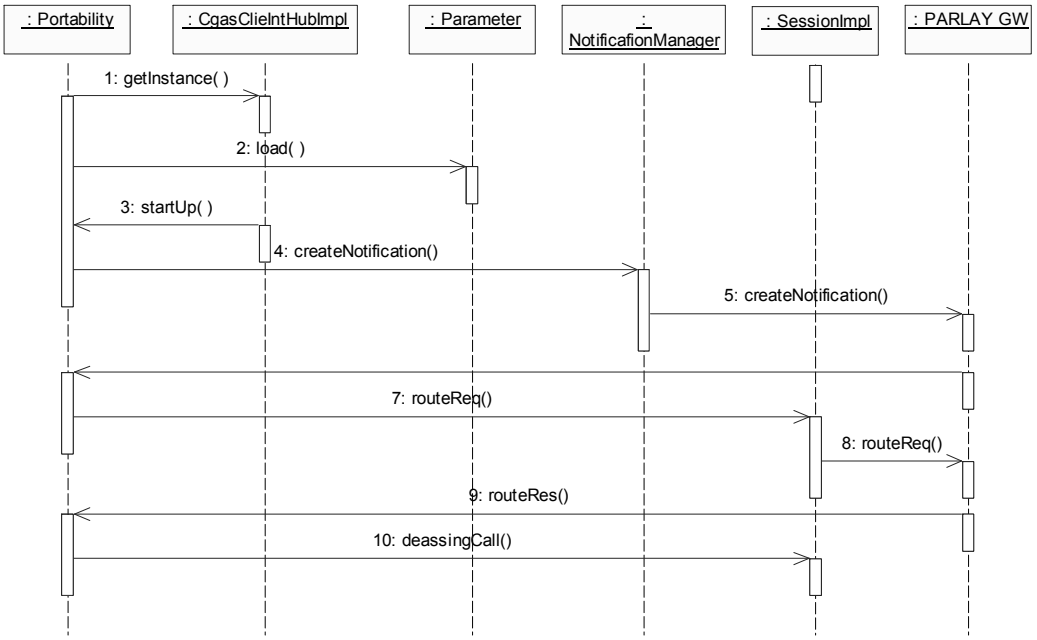


Figura 5. Diagrama de Secuencias para el servicio de portabilidad numérica

Al iniciarse la aplicación la primera función que realiza, durante su etapa de inicialización, es acceder al Framework para solicitar acceso a los servicios, esto se realiza a través de la clase *CgasClientHubImpl*. Una vez se obtienen los servicios la aplicación carga de su archivo de configuración parámetros relevantes e información de la notificación a habilitar.

Posteriormente la aplicación habilita la notificación y pasa a un estado de espera.

A partir de ese momento, las llamadas que cumplen con los criterios especificados en la notificación generan una notificación a la aplicación. Esa notificación ocasiona que el Parlay Gateway invoque el método *callEventNotificationInfo(),* este método de callback entrega un parámetro que tiene información sobre la llamada actual. En ese momento la aplicación tiene control sobre la llamada.

Utilizando la información del parámetro entregado la aplicación busca el número al que debe diseccionarse la llamada. Cuando se obtiene el número a direccionar se invoca el método *routeReq(),* este método se traducirá en una orden que hará que el softswitch redireccione la llamada al número especificado.

En caso de que se produzca un enrutamiento exitoso, se invoca el método *routeRes(),* donde la aplicación invoca el método *deassignCall(),* este método le indica al softswitch que la aplicación no está interesada en la llamada y libera los recursos que la aplicación esté usando.

En caso de que se presente un error en el enrutamiento, se invoca el método *routeErr()* y la aplicación intentará enrutar nuevamente.

IV. CONCLUSIONES Y RECOMENDACIONES

De las tecnologías analizadas osa/parlay ofrece un conjunto más amplio de capacidades de la red, lo que facilita la implementación de una gran variedad de servicios avanzados de telecomunicaciones. Además, por ser una tecnología ya establecida garantiza un alto nivel de estandarización y adopción por parte de la comunidad desarrolladora de servicios y por los proveedores de los mismos.

Facilitar el proceso de creación de servicios permitirá que Emcali aumente su portafolio y se posicione en la región como una empresa fuerte en el área de las telecomunicaciones y explote al máximo la plataforma NGN con que cuenta en este momento.

Para el diseño e implementación de nuevos servicios sobre la plataforma de aplicaciones de la red multiservicios de Emcali se recomienda utilizar las herramientas del Parlay Client Hub (PCH) ya que este se encuentra integrado con el hardware de la plataforma de aplicaciones y sus bases de datos. Igualmente este API implementa funciones que no fueron especificadas en el API de Osa/Parlay y que son necesarias para el despliegue de servicios y aplicaciones funcionales.

En el contexto nacional este es uno de los primeros proyectos en abordar el tema de la creación de servicios para redes de nueva generación, por lo cual se constituye en una base importante para que a partir del mismo surjan nuevos proyectos que impulsen esta área tanto a nivel local como nacional que redunden en un beneficio para los diferentes actores que intervienen en el sector de las telecomunicaciones.

A través de este proyecto se han fortalecido las relaciones entre la Universidad Autónoma de Occidente y las Empresas Municipales de Cali - Emcali, empresa con la cual se han desarrollado y están en curso una buena cantidad de proyectos de investigación y pasantías institucionales, con excelentes experiencias, que han contribuido a incrementar el saber de nuestros estudiantes y la productividad de la empresa.

Este tipo de proyectos fortalece los lazos entre la academia y la industria, vínculo de gran importancia que debe mantenerse para el avance de ambos sectores posibilitando el fortalecimiento y desarrollo del sector de las telecomunicaciones. Por tanto es recomendable generar vínculos formales, tales como convenios, grupos de investigación, laboratorios compartidos, etc., que dinamicen y mantengan vivas las sinergias creadas.

V. REFERENCIAS

[1] Bakker J.L., 2001. A Service Creation Markup Language for Scripting Next Generation Network Services. Disponible en Internet: http://www.cs.columbia.edu/sip/drafts/draft-bakker-jain-scml-00.txt

[2] Bray T., 2008. Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation. Disponible en Internet: http://www.w3.org/TR/2008/REC-xml-20081126/

[3] Di Benedetto, L., 2005. Las Telecomunicaciones y la Movilidad en la Sociedad de la Información. Edición: División de Relaciones Corporativas y Comunicación de Telefónica I+D.

[4] Falcarin P.; Licciardi, C. A., 2003. Analysis of NGN service creation technologies . IEC Annual Review of Communications, Vol.56. Disponible en Internet: http://softeng.polito.it/falcarin/docs/Annals03.pdf

[5] Kryvinska N., 2008. Conceptual Framework for Services Creation/ Development Environment in Telecom Domain. IiWAS.

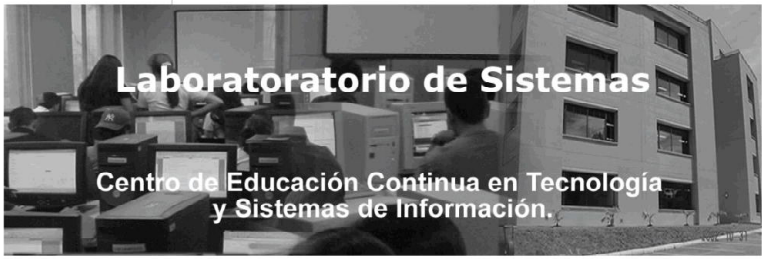
[6] Muñoz Organero, M.,2003. SCMM: Metamodelo para la creación de aplicaciones en redes de siguiente generación. Tesis doctoral (Ingeniería telemática). Universidad Carlos III de Madrid. Departamento de Ingeniería Telemática. 276 P.

[7] Popoff, J.,2005. Parlay/OSA. Boston: The Parlay Group. Disponible en Internet: http://www.parlay.org

[8] ZTE, 2008. Introduction to Parlay Client Hub (PCH). Santiago de Cali.

[9] ZTE, 2005. Plataforma Unificada de Servicios ZXUP10. Santiago de Cali.

Universidad Nacional de Colombia Sede Medellín
Facultad de Minas



Misión

Ofrecer servicios de apoyo a la docencia en cuanto a la operació n de los computadores y del software adecuado con miras al desarrollo int egral de los futuros ingenieros.

Visión

Avanzamos en la búsqueda de convertir el Laboratorio de Sistemas e Informática en una dependencia ágil, moderna, facilitadora de procesos y cambios, atenta a las necesidades de otras dependencias de la Universidad, cuya labor apoyamos y articulamos. Serviremos con dinamismo, amabilidad y efectividad a todos los integrantes de la comunidad universitaria y a la sociedad en gen eral. Uniremos esfuerzos para construir un ambiente de trabajo cada vez más pos itivo que propicie la participación, la creatividad y el desarrollo profesional de los integrantes del equipo de trabajo. Propendemos por un Laboratorio como instrumen to gestor y generador de proyectos de investigación sustentado en un equipo interdisciplinario de trabajo en torno a la informática aplicada a la ingeniería.

Cursos

- Lenguajes de Programación: Diseño de Páginas Web en ASP.NET con VB.NET, Programación Web PHP y MYSQL, MS-Visual Basic Básico y Avanzado, Java
- Generales: Latex, ARCGIS, MS -Office (Word, Excel y PowerPoint), Excel Financiero, Excel Avanzado, Mantenimiento de Hardware y Software Niveles I y II, MS -Access Básico, MS -Project Básico (Programación y Gerencia de Proyectos), AUTOCAD 2 D Básico y 3D, Matlab, Moodle de Apoyo a los Procesos de Enseñanza y Aprendizaje.

Para mayor información, por favor comunicarse a los teléfonos: 4 255313, 4255312, 4255355. Calle 59A No. 63 – 020 Medellín (Colombia), Bloque M7, quinto piso.
Email: labsis@unalmed.edu.co http://xue.unalmed.edu.co/cursos