

**DESARROLLO DE ONTOLOGÍAS PARA SU USO EN PERFILES DE USUARIO EN EL ENTORNO
IMS**



MONOGRAFÍA

DIEGO FABIÁN GALLEGO FERNÁNDEZ

JONNY ALBERTO CABRERA PAZOS

**Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Servicios Avanzados de Telecomunicaciones
Popayán, Noviembre de 2009**

**DESARROLLO DE ONTOLOGÍAS PARA SU USO EN PERFILES DE USUARIO EN EL ENTORNO
IMS**



DIEGO FABIÁN GALLEGO FERNÁNDEZ

JONNY ALBERTO CABRERA PAZOS

**Trabajo de Grado presentado como requisito para optar al título de
Ingeniero en Electrónica y Telecomunicaciones**

Directora: I.E. Mary Cristina Carrascal Reyes

**Universidad del Cauca
Facultad de Ingeniería Electrónica y Telecomunicaciones
Departamento de Telemática
Servicios Avanzados de Telecomunicaciones
Popayán, Noviembre de 2009**

TABLA DE CONTENIDO

CAPÍTULO 1 INTRODUCCIÓN AL ENTORNO IMS Y LA PERSONALIZACIÓN DE SERVICIOS.....	1
1.1 IMS IP MULTIMEDIA SUBSYSTEM.....	1
1.1.1 Requisitos de IMS.....	3
1.1.2 Arquitectura de IMS.....	3
1.1.3 Descripción de los componentes de IMS.....	4
1.2 PERSONALIZACIÓN Y PERFILES DE USUARIO	16
1.2.1 Contexto de la personalización de servicios	16
1.2.2 Personalización basada en perfil de usuario.	18
1.2.3 Creación de perfiles.	19
1.2.4 Perfil de usuario en el entorno de IMS.....	19
CAPÍTULO 2 ONTOLOGÍAS Y SU USO EN LA PERSONALIZACIÓN DE SERVICIOS.....	21
2.1 DEFINICIONES DE ONTOLOGÍA.....	21
2.2 CARACTERÍSTICAS DE LAS ONTOLOGÍAS	22
2.3 VENTAJAS	22
2.4 DISEÑO Y CONSTRUCCIÓN DE UNA ONTOLOGÍA.....	23
2.5 ONTOLOGÍAS Y LA PERSONALIZACIÓN DE SERVICIOS.....	24
CAPÍTULO 3 ONTOLOGÍA PARA LA GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS.....	26
3.1 REQUISITOS PARA LA GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS.....	26
3.2 REQUERIMIENTOS DE UNA ONTOLOGÍA DE PERFILES DE USUARIO	27
3.3 APORTES DE LAS ONTOLOGÍAS A LOS SERVICIOS EN EL ENTORNO DE IMS.	28

3.4 ARQUITECTURA PARA LA GESTIÓN ONTOLÓGICA DE PERFILES DE USUARIO EN EL ENTORNO DE IMS	29
3.4.1 Servidor de aplicaciones (AS)	30
3.4.2 Ontología de perfiles de usuario	30
3.4.3 Gestor de perfiles de usuario.....	30
3.4.4 Aplicación IMS.....	31
3.4.5 Núcleo de IMS	31
3.4.6 Clientes IMS	32
3.5 DIAGRAMA DE INTERACCIÓN DE COMPONENTES DE LA ARQUITECTURA DE REFERENCIA	32
CAPÍTULO 4 IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS.....	35
4.1 ENTORNO DE DESARROLLO ONTOLÓGICO	35
4.2 ONTOLOGÍA DE PERFILES DE USUARIO	35
4.2.1 Características de la ontología de perfil de usuario base.	36
4.2.2 Características adicionadas a las ontología base.....	38
4.3 IMPLEMENTACIONES DEL NÚCLEO DE IMS	39
4.3.1 Open IMS Core	39
4.3.2 Ericsson Service Development Studio – SDS	40
4.3.3 Selección del entorno para la emulación del Núcleo de IMS.....	41
4.4 ARQUITECTURA PARA EL INTERCAMBIO DE INFORMACIÓN ENTRE SERVICIOS.....	42
4.4.1 RPC	42
4.4.2 SOA	42
4.4.3 REST	43
4.5 IMPLEMENTACIÓN DE REFERENCIA.....	44

4.5.1 Modelo de casos de uso.....	44
4.5.2 Diagrama de paquetes análisis.....	47
4.5.3 Diagrama de paquetes de diseño.....	48
4.5.4 Modelo de Implantación.....	50
4.6 METODOS DE ACCESO A LOS RECURSOS DE LA ONTOLOGÍA	52
4.7 EVALUACIÓN DEL GESTOR DE PERFILES DE USUARIO.....	55
4.7.1 Escenario de evaluación	55
4.7.2 Actividades y organización de las pruebas	56
4.7.3 Selección de las herramientas a utilizar.....	57
4.7.4 Realización de pruebas	58
4.8 INTRODUCCIÓN	73
4.9 DESCRIPCIÓN DEL ESCENARIO IMS.....	73
4.9.1 Configuración del núcleo de red IMS	73
4.9.2 Descripción del escenario para el servicio	74
4.9.3 Descripción del cliente	74
4.9.4 Herramientas utilizadas para análisis y pruebas	75
4.10 VALIDACIÓN DEL SISTEMA DE GESTIÓN.....	75
4.10.1 Registro y establecimiento de sesión	75
4.10.2 Prueba 1: verificar la creación de una instancia en la ontología.....	78
4.10.3 Prueba 2: Verificar la adición de una instancia de la clase LivingConditions a una instancia de la clase persona.	82
4.10.4 Prueba 3: Verificar la recuperación de información de perfil de usuario	85
CAPÍTULO 5 LINEAMIENTOS PARA LA PERSONALIZACION DE SERVICIOS EN EL ENTORNO DE IMS.....	88

CAPÍTULO 6 APORTES, CONCLUSIONES Y TRABAJOS FUTUROS	91
6.1 APORTES	91
6.2 CONCLUSIONES	92
6.3 TRABAJOS FUTUROS.....	94
REFERENCIAS	96

LISTA DE FIGURAS

Figura 1. Arquitectura de IMS.....	4
Figura 2. Entidades del CSCF.	5
Figura 3. Funcionalidad del Proxy-CSCF.	6
Figura 4. Funcionalidad del Interrogating CSCF.....	8
Figura 5. Funcionalidad del Serving CSCF	11
Figura 6. Funcionalidad del HSS.....	13
Figura 7. Tipos de Servidores de Aplicación	15
Figura 8. Relaciones entre identidades de usuario y perfiles de servicio.	20
Figura 9. Perfil de Usuario y perfiles de Servicio	20
Figura 10. Arquitectura propuesta para la gestión de perfiles de usuario.....	29
Figura 11. Diagrama de interacción de componentes	33
Figura 12. Diagrama de casos de uso del gestor de perfiles de usuario	45
Figura 13. Diagrama de paquetes de análisis del gestor de perfiles de usuario	48
Figura 14. Diagrama de paquetes de diseño	48
Figura 15 Modelo de implantación.....	50
Figura 16 Escenario de evaluación del gestor de perfiles de usuario	55
Figura 17 Paquetes HTTP de la prueba 1	58
Figura 18 Flujo TCP de la prueba 1	59
Figura 19 Paquetes HTTP de la prueba 2	60
Figura 20 Flujo TCP de la prueba 2.....	61
Figura 21 Paquetes HTTP de la prueba 3.....	62
Figura 22 Flujo TCP de la prueba 3.....	62
Figura 23 Paquetes HTTP prueba 4.....	64

Figura 24 Flujo TCP de la prueba 4	64
Figura 25 Paquetes HTTP prueba 5	65
Figura 26 Flujo TCP de la prueba 5	66
Figura 27 Paquetes HTTP de la Prueba 6.....	67
Figura 28 Flujo TCP de la prueba 6	67
Figura 29 Tiempos de respuesta a las solicitudes en Wireshark	68
Figura 30 Resumen de las pruebas realizadas en JUnit.	69
Figura 31 Conexiones exitosas.....	70
Figura 32 Tiempo promedio de respuesta.....	71
Figura 33 Número de peticiones respondidas por segundo	72
Figura 34 Escenario IMS	73
Figura 35 Escenario de prueba para el intercambio de información.....	75
Figura 36 Registro y suscripción	76
Figura 37 Establecimiento de sesión.....	77
Figura 38 Envío de mensajes entre entidades en el dominio IMS.....	78
Figura 39 Flujo TCP entre el dominio IMS y el SGPU.....	80
Figura 40 Petición leída desde el lado del SGPU.....	80
Figura 41 Mensaje recibido en cliente.....	81
Figura 42 Respuesta a una solicitud de creación de una persona existente.	81
Figura 43 Mensajes de solicitud y respuesta para agregar una propiedad.....	82
Figura 44 Mensajes asociados a la creación de una instancia.	83
Figura 45 Método PUT capturado en el gestor.	83
Figura 46 Mensajes HTTP para la adición de una propiedad.	84
Figura 47 Flujo TCP para la adición de una propiedad.....	84

Figura 48 Mensajes para la recuperación de datos en el dominio IMS	86
Figura 49 Paquetes HTTP	87
Figura 50 Flujo TCP para la solicitud de datos.....	87

LISTA DE TABLAS

Tabla 1 Tipos de personalización	18
Tabla 2 Clases de alto nivel de la ontología de perfil de usuario	37
Tabla 3 Ejemplo de cómo una jerarquía de intereses puede ser modelada	38
Tabla 4 Analogías entre acciones de RESTful	52
Tabla 5 Representación de los recursos de una clase	53
Tabla 6 Representación para el caso de una propiedad con múltiples valores.....	53
Tabla 7 URIs de acceso a los diversos recursos de la ontología	54
Tabla 8 Componentes de la evaluación.....	56
Tabla 9 Tabla de pruebas y acciones realizadas	57
Tabla 10 Conexiones Exitosas	70
Tabla 11 Tiempo promedio de respuesta.....	71
Tabla 12 Número de peticiones respondidas por segundo.....	72

LISTA DE ANEXOS

Anexo A. ENTORNO DE DESARROLLO ONTOLOGÍCO.

Anexo B. CASO DE ESTUDIO

Anexo C CONFIGURACION DEL SDS PARA LA PROVION DE SERVICIOS

Anexo D. IMPLEMENTACIÓN DEL SERVICIO DE PRUEBA

Anexo E. DESARROLLO DE CLIENTES IMS CON EL SDS

CAPÍTULO 1 INTRODUCCIÓN AL ENTORNO IMS Y LA PERSONALIZACIÓN DE SERVICIOS

Al momento de desplegar y generar servicios convergentes, los operadores comienzan a pensar en algunos elementos con el fin de garantizar una buena experiencia a los usuarios. El primero de tales elementos es la transparencia y ubicuidad, los cuales deben permitir a los diferentes usuarios acceder y poder disfrutar de un conjunto de servicios, en forma independiente de su red y el dispositivo de acceso. El segundo aspecto, no menos importante, es la personalización; esto hace referencia a que el usuario espera datos que sean relevantes para él, acordes con sus necesidades e intereses y enmarcados dentro de su contexto. El propósito de las siguientes secciones es mostrar estos dos elementos en detalle.

Así bien este capítulo se divide en dos partes, la primera parte hace una introducción a IMS (*IP Multimedia Subsystem*, Subsistema Multimedia IP), sus requisitos, su arquitectura y sus componentes principales, en la segunda parte se hace un estudio del concepto de personalización de servicios, las fuentes de información, las técnicas de personalización y los tipos de personalización.

1.1 IMS IP MULTIMEDIA SUBSYSTEM

IMS es una arquitectura creada básicamente para eliminar la barrera entre las tecnologías tradicionales utilizadas en las telecomunicaciones y las tecnologías empleadas en Internet. La pregunta clave para un operador de telecomunicaciones es ¿por qué IMS?. Existen muchas respuestas, no obstante la respuesta que mejor se acomoda a los intereses del operador es que IMS proporciona servicios multimedia innovadores sobre redes fijas y móviles usando estándares abiertos [1]. IMS canaliza temas importantes como convergencia, creación y entrega de servicios, interconexión de éstos y estándares abiertos. En conclusión, IMS permite a un operador mantener sus modelos de negocio actuales o evolucionar hacia los nuevos.

Inicialmente se creó para los operadores móviles ya que el trabajo lo comenzó el 3GPP (*3rd Generation Partnership Project*, Proyecto Asociado de 3ra Generación), grupo encargado de trabajar en la evolución de redes GSM (*Global System for Mobile communications*, Sistema Global para las comunicaciones Móviles). Este desarrollo se produjo para evitar que los operadores móviles pasaran a ser simples canales por donde terceras partes ofrecieran servicios de valor agregado. La creación de IMS, por lo tanto, es dada como consecuencia del lanzamiento de servicios adicionales a la voz. Los operadores, tanto fijos como móviles, han pasado de ofrecer un servicio básico (voz) a adentrarse en la oferta de servicios de datos, cuya integración a la infraestructura existente ha sido compleja en muchos casos. Un ejemplo de esto se da cuando un operador móvil que ya ha establecido su infraestructura de datos, desea lanzar un nuevo servicio, en este escenario el operador debe añadir un nuevo componente a la red e integrarlo con algunos de los sistemas existentes (por ejemplo con el sistema de facturación), no obstante el servicio como tal estará aislado del resto de servicios. Cada servicio es pues una isla dentro de la red. Esto, además de ser poco práctico desde el punto de vista de la administración de los servicios, restringe la interacción de los mismos al momento de ofrecerse conjuntamente al usuario. Para los operadores la importancia de IMS radica en la simplificación de la arquitectura de la red al incorporar nuevos servicios, así

como al crearlos y lanzarlos al mercado. Este beneficio lo obtienen todos los operadores independientemente de si son fijos, móviles o cable operadores.

Aquellos operadores integrados (con negocio fijo y móvil) tienen la ventaja de poder implementar fácilmente la convergencia de servicios a los que el usuario puede acceder a través de cualquiera de las dos redes. Estos operadores integrados son los que más tienden a ganar con IMS, ya que las divisiones de servicios fijos y móviles pueden integrarse dentro de la organización puesto que los mismos servicios en las redes fijas son los que se ofrecerán en las redes móviles y viceversa. La eficiencia que estas grandes organizaciones pueden adquirir con IMS las pone a la cabeza de los grupos que mayores beneficios encontrarán al migrar a este tipo de arquitectura.

Por otro lado, la disponibilidad de una arquitectura IMS operativa permite, desde el punto de vista de aplicación, la utilización de componentes aplicativos estándar y reutilizables (habilitadores), en un entorno abierto de desarrollo y en una mejor integración de las aplicaciones en la red y los sistemas de gestión y negocio. Además de reducir el tiempo requerido para la introducción de nuevas aplicaciones, mejorar la interoperabilidad entre ellas, enriquecer las aplicaciones en capacidades multimedia y hacerlas más intuitivas y combinables para crear nuevos servicios y más complejos. No se puede decir que exista una “*killer application*”, una aplicación estrella, sino una serie de funcionalidades nuevas e innovadoras que permiten al usuario tener comunicaciones más fáciles, eficientes y atractivas, tanto para los usuarios residenciales como para los usuarios empresariales. Debido a la flexibilidad de la arquitectura IMS, estas aplicaciones pueden ser desarrolladas no sólo por los operadores y sus proveedores, sino por terceros gracias a la utilización de Java y SIP (*Session Initiation Protocol*, Protocolo de Inicio de Sesiones).

Hace algunos años, las aplicaciones como directorios personales y corporativos, buzón de voz, correo electrónico y aplicaciones de *softphone* para VoIP (*Voice over Internet Protocol*, Voz sobre el Protocolo de Internet), entre otras, funcionaban de forma independiente sin estar correlacionadas de ninguna forma, era un escenario con una realidad fragmentada. Con la aparición del protocolo SIP, todas estas capacidades pueden ser integradas en un único perfil, de esta forma se representa un único directorio, mensajería instantánea, buzón de voz, etc. Aunque esto está bien para hoy, no es suficiente para el futuro. La tendencia apunta hacia servicios centrados en el usuario y no en el terminal. Este nuevo escenario da la bienvenida a servicios más complejos que a su vez manejan mayor información proveniente del usuario, servicios interactivos donde la personalización cobra mayor importancia.

Los usuarios de telecomunicación actuales también están cada vez más informados y son más exigentes, por lo que, como ha sucedido con los servicios 3G, no siempre se cumplen las expectativas creadas por los operadores y proveedores de infraestructura de telecomunicaciones. Para que los servicios multimedia tengan éxito, no basta con que sean útiles, también es necesario que sean sencillos de utilizar, baratos, accesibles en cualquier momento y lugar y que den importancia a las preferencias y gustos del usuario.

1.1.1 Requisitos de IMS

Con IMS se propusieron los siguientes objetivos:

- Combinar las últimas tendencias en tecnología.
- Hacer realidad el paradigma Internet móvil.
- Crear una plataforma común para desarrollar servicios multimedia.
- Crear un mecanismo que incremente los márgenes por el uso extra de las redes de conmutación de paquetes.

En los requisitos que dirigieron el diseño según el *release 5*, IMS se define como un marco arquitectónico creado con el propósito de proporcionar servicios multimedia IP a los usuarios finales. Dicho marco necesita [2]:

- Soportar el establecimiento de sesiones multimedia IP.
- Soportar un mecanismo que negocie calidad de servicio (QoS).
- Soportar el inter funcionamiento con Internet y con redes de conmutación de circuitos.
- Soportar *roaming* (itinerancia) entre redes.
- Soportar un fuerte control establecido por los operadores respecto a los servicios entregados al usuario final.
- Soportar una creación rápida de servicios sin requerir estandarización.

La versión *release 6* de [2] considera el acceso desde otras redes diferentes a GPRS/UMTS (*General Packet Radio Service*, Servicio General de Paquetes vía Radio /*Universal Mobile Telecommunications System*); permitiendo acceder a IMS desde diferentes redes.

1.1.2 Arquitectura de IMS

Las especificaciones de IMS definen una arquitectura completa de la capa de control (por encima del dominio de conmutación de paquetes) y cubren también todos los elementos necesarios para soportar sesiones multimedia en la red de paquetes. IETF (*Internet Engineering Task Force*, Grupo de Trabajo en Ingeniería de Internet) estandarizó el protocolo SIP, el más apropiado en relación a las soluciones cliente/servidor, pero sin asociarlo con las arquitecturas, 3GPP definió con precisión las arquitecturas y los procedimientos para hacer la capa de control y los núcleos de la red seguros, fiables y que puedan ser gestionados [3].

Los componentes principales relacionados con los servicios de comunicación son (Figura 1):

- CSCF (*Call State Control Functions*, Funciones de Control de Estado de Llamada) son las entidades de control de la sesión. Hay varios tipos de CSCF en función de su papel dentro del control de las sesiones.
- HSS (*Home Subscriber Server*, Servidor de Suscriptor Local) es la base de datos centralizada de la red y de los servicios.

- AS (*Application Server*, Servidor de Aplicaciones) son las plataformas que proporcionan los servicios relacionados con la sesión a los usuarios (presencia, conferencia, etc.).

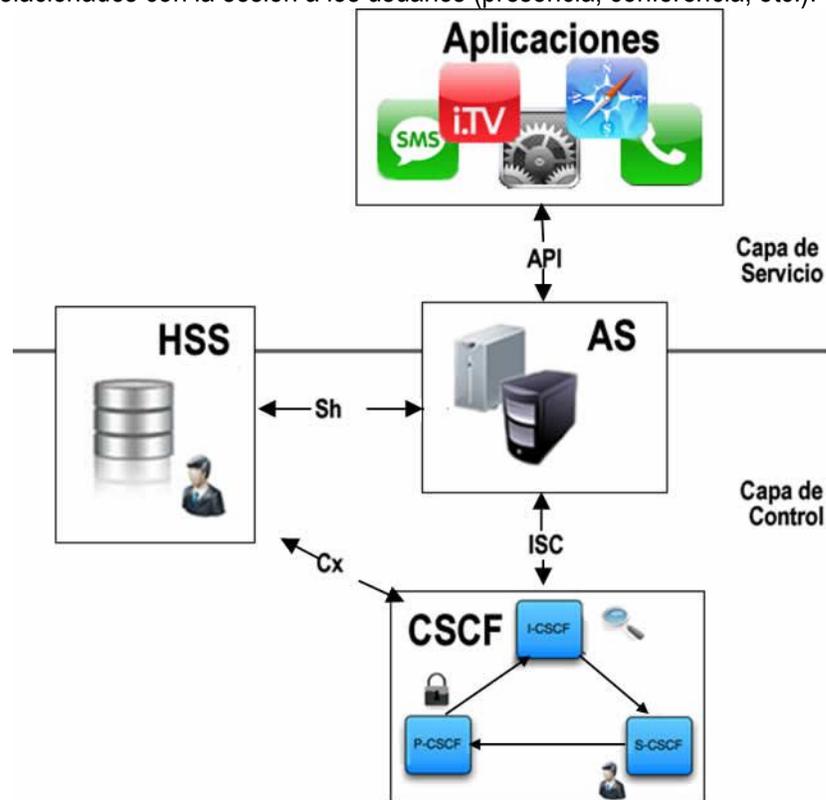


Figura 1. Arquitectura de IMS

Mientras la función CSCF pertenece a la capa de control, HSS y AS están en la divisoria entre la capa de control y la capa de servicio. APIs (*Application Programming Interface*, Interface de Programación de Aplicaciones) como AS/Parlay o SIP servlet son las recomendadas para el desarrollo de aplicaciones.

1.1.3 Descripción de los componentes de IMS

1.1.3.1 Call Session Control Function (CSCF)

En la arquitectura genérica de una red IMS, la entidad funcional clave es el nodo CSCF. Su función principal es el control de sesiones y llamadas. Esta función está distribuida a lo largo de la red buscando la eficiencia y escalabilidad. Existen 3 entidades que son responsables por el control de sesiones y llamadas:[4][5].

- P-CSCF el *Proxy Call Session Control Function*.
- I-CSCF el *Interrogating Call Session Control Function*.
- S-CSCF el *Serving Call Session Control Function*.

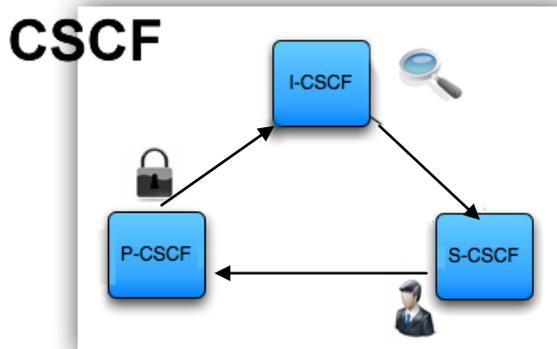


Figura 2. Entidades del CSCF.

Las diferencias entre estas entidades radican en las tareas y procedimientos que realizan, a continuación son mostradas en detalle cada una de ellas.

1.1.3.2 Proxy CSCF (P-CSCF)

El P-CSCF es el punto de entrada a la red IMS. Esta entidad actúa como el punto de acceso al dominio SIP desde una perspectiva de control de sesión. El tráfico portador de datos no se transmite a través de esta parte de IMS, ya que es una red de control y señalización. El tráfico es transmitido utilizando diferentes métodos de acceso para el transporte. Toda la información que fluye a través del P-CSCF es SIP[4][5].

El primer intercambio de información tiene como fin registrar la localización del dispositivo; ella está asociada con la dirección IP de su ubicación actual. Dado que el dispositivo se comunica con otros dispositivos, lo primero a establecer es una sesión, así bien el establecimiento de sesiones son realizadas en primera instancia a través del P-CSCF. Cuando se activa por primera vez el equipo terminal de un suscriptor, la red de servicio le asigna a este una dirección IP, una vez esta ha sido asignada, el equipo buscará el P-CSCF local (o el P-CSCF que haya sido asignado para servir en esta parte de la red). El P-CSCF, así como todas las entidades IMS, tiene una dirección con un formato SIP URI (*Universal Resource Identifier*, Identificador de Recursos Universal) (haciendo más fácil el enrutamiento de los mensajes al P-CSCF apropiado) [4].

Cuando el dispositivo está encendido, envía su dirección IP al HSS y al S-CSCF mediante un proceso de registro. El P-CSCF juega un rol importante en el proceso de registro, el primer rol, sin embargo, es identificar la red de origen y el dominio del suscriptor (utilizando la URI de su dirección). El nombre de dominio de la red de origen es resuelto por un servidor DNS (*Domain Name System*, Sistema de Nombres de Dominio). El DNS identifica la dirección del I-CSCF que será utilizado para acceder a la red de origen. El I-CSCF proporciona la pasarela de acceso a cualquier red. Esto dispone al P-CSCF para que pueda determinar cómo direccionar cualquier mensaje SIP recibido por el equipo terminal. Por ejemplo, cuando el P-CSCF recibe un INVITE, el debe decidir a donde será enviando el mensaje [5].

El P-CSCF actúa como el punto de acceso en IMS pero no en las redes individuales (al menos en términos de mensajes SIP). El I-CSCF proporciona el enrutamiento al S-CSCF adecuado de acuerdo con los procedimientos de registro. Al igual que todas las entidades CSCF al interior de IMS, el P-CSCF genera CDRs (*Call Data Records*, Registro de Datos de Llamada) para todas las sesiones que pasan a través de él. El P-CSCF también adiciona cabeceras a los mensajes de solicitud y respuesta antes de su envío hacia el próximo CSCF [4] [5].

El P-CSCF genera el ICID (*IMS Charging Identifier*, Identificador de Cobro IMS) utilizado en los procedimientos de cobro donde se correlacionan las sesiones y los gastos, esto dado que el P-CSCF como ya se expuso es el punto de entrada en IMS. Estas cabeceras son agregadas con el fin de que las entidades puedan intercambiar datos de facturación dentro de los mensajes SIP, sin tener que considerar el soporte de otra interface para estos fines. Sin embargo también existe una interfaz de facturación separada que es soportada por el protocolo DIAMETER [4][5].

Las cabeceras que son usadas para la facturación no son compartidas con otras redes; el P-CSCF sólo envía estas cabeceras a entidades funcionales dentro de la misma red. Esto impide a otros proveedores tener acceso a información sobre suscriptores y servicios. Esta es otra función del P-CSCF la cual está basada en políticas configuradas por el operador. Desde una perspectiva de seguridad, el P-CSCF tiene una tarea crítica en la prevención de accesos no autorizados a la red (figura 3). El P-CSCF puede ser usado para restringir el acceso a cualquier dispositivo. Sin embargo, el P-CSCF no ejecuta el proceso de autenticación dentro de IMS [5].

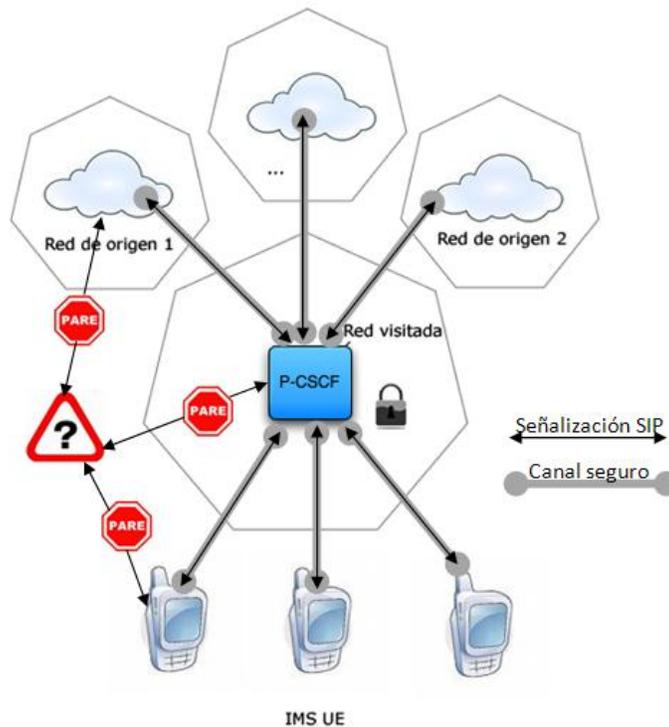


Figura 3. Funcionalidad del Proxy-CSCF.

El S-CSCF es responsable de los dispositivos que intentan establecer una sesión sin encontrarse registrados, o cuando se hace un intento de registro. PDF (*Policy Decision Function*, Función de Políticas de Decisión) puede residir en el P-CSCF y ser usada para determinar la forma como se debe reaccionar ante escenarios específicos. PDF permite a los operadores establecer reglas que son aplicadas para manejar el acceso a las redes. PDF controla la Función de Ejecución de Políticas en la red de portador. Esto permite a los operadores controlar el flujo de acuerdo a los permisos y a las direcciones de origen y destino [4][5].

El P-CSCF también puede comprobar el enrutamiento, verificando que la ruta recibida en el mensaje SIP de solicitud/respuesta sea la misma que fue identificada cuando el dispositivo se registró en la red. Si las cabeceras de enrutamiento no contienen una dirección que concuerde con la dirección almacenada por el P-CSCF durante el proceso de registro, el enrutamiento es modificado por el P-CSCF de acuerdo con la dirección capturada por esta entidad. Esto se hace para prevenir el “*hacking*” y otros escenarios donde un hacker puede capturar un mensaje SIP y usarlo como una copia para obtener servicios desde otra parte de la red. El P-CSCF puede proveer esta funcionalidad porque cuando una entidad se registra en la red, el P-CSCF almacena todas las direcciones proporcionadas en las cabeceras de enrutamiento. Otra información almacenada por esta entidad incluye la dirección del dispositivo (la dirección IP) y las identidades de usuario públicas y privadas. Si un dispositivo pierde su conexión en la red IP, el P-CSCF es notificado y libera todas las sesiones IMS mediante el envío un mensaje CANCEL a cada una de las entidades que hacen parte de la sesión. El P-CSCF conoce todas las sesiones creadas a través de él y a su vez conoce el estado de cada una de estas [4][5].

1.1.3.3 Interrogating CSCF (I-CSCF)

Mientras el P-CSCF es el punto de acceso a la red IMS, el I-CSCF sirve como pasarela o puerta de enlace dentro de cada red IMS. El I-CSCF determina si se concede el acceso a otras redes que envían mensajes SIP hacia el operador. Por esta razón, el I-CSCF puede ser usado para ocultar los detalles de la red a otros operadores, determinando el enrutamiento al interior de un dominio seguro. El I-CSCF ayuda a proteger al S-CSCF y al HSS de accesos no autorizados por parte de otras redes. Cuando el S-CSCF está enviando peticiones o respuestas a otras redes, el mensaje primero es enviado hacia el I-CSCF, el cual a su vez lo envía hacia la red de destino [4][5].

El I-CSCF en la red destino tiene la responsabilidad de identificar la localización del usuario al que ha sido dirigido el mensaje (posiblemente a través del SLF (*Subscription Location Function*, Función de Localización de Suscriptor). La localización hace referencia a la identificación del S-CSCF que ha sido asignado al usuario, así como también el HSS en el que se han almacenado los datos de suscripción. Esta información es suministrada por el SLF. El I-CSCF cumple un rol importante en IMS ayudando tanto en la seguridad en el acceso como en la protección de las identidades (direcciones de enrutamiento) del S-CSCF y el HSS [4][5].

La función de pasarela que cumple el I-CSCF es algo diferente con la misma función que ejecuta el P-CSCF, ya que el P-CSCF actúa como una pasarela hacia redes que no son IMS y como punto de acceso hacia IMS, mientras que el I-CSCF actúa como pasarela entre dos redes IMS. Otra función importante del I-CSCF es la asignación del S-CSCF, esto se hace de acuerdo a la capacidad o a las

políticas del proveedor del servicio. Hay dos opciones disponibles para su implementación, un método es asignar el S-CSCF de acuerdo con los servicios a soportar. Por ejemplo, si una video conferencia está siendo establecida, se asignara un S-CSCF que este suministrando acceso a recursos de video. Este método puede ser favorable para los proveedores de servicio que observan la forma de distribuir multimedia a lo largo de la red o en porciones específicas de la misma. Este modelo funciona bien en redes de datos donde los servicios y las plataformas están localizados de acuerdo al contenido multimedia que deba ser soportado [4][5].

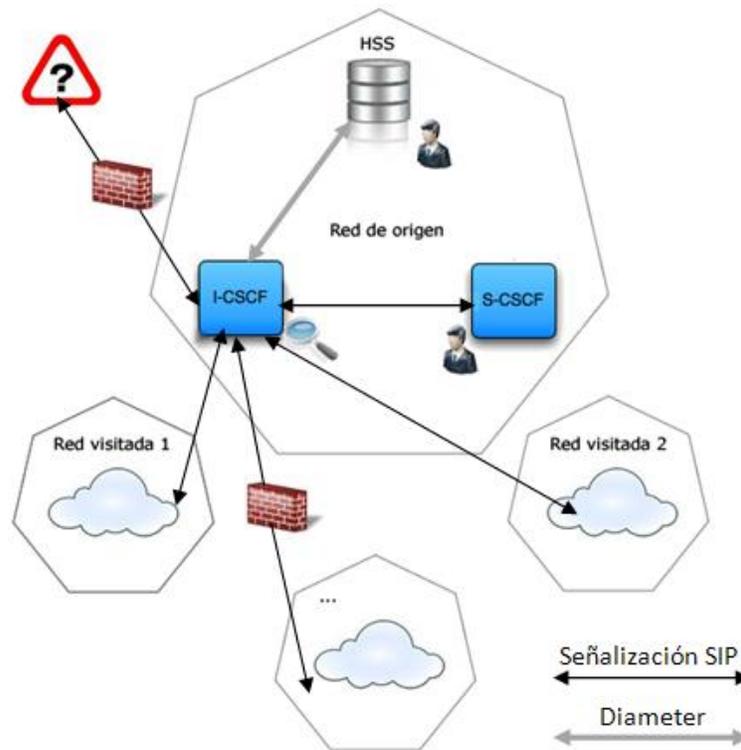


Figura 4. Funcionalidad del Interrogating CSCF.

En un modelo de Proveedor de Servicios de Aplicación, por ejemplo, el servidor de video puede estar localizado al interior de un *data center*, junto con un S-CSCF para así dar soporte a servicios de video. Todas las suscripciones para esos servicios serian asignados al S-CSCF que se encuentra en esa ubicación. La decisión de como asignar el S-CSCF se haría basándose en las suscripciones como tal y no en la ubicación de la suscripción. El otro método es asignar cada S-CSCF de acuerdo a la posición geográfica. Este es el método usado tradicionalmente en las redes de telecomunicaciones que utilizan conmutadores. El modelo se desempeña bien para operadores de telecomunicaciones tradicionales, y para algunas redes puede tener mayor sentido. El S-CSCF en este caso es asignado de acuerdo a la localización del I-CSCF que recibe las peticiones/respuestas. La información del S-CSCF que se asigna es almacenada en el HSS para ser referenciado en el futuro. Para operadores de redes inalámbricas y cableadas, probablemente este modelo tenga más sentido dado que es al que están acostumbrados hoy en día. Los

suscriptores son asignados de acuerdo a la localización de sus hogares y todos los servicios son soportados a lo largo de la red y no por segmentos [4][5].

El S-CSCF es asignado por el I-CSCF cuando un suscriptor se registra en la red. Así el I-CSCF almacena esta información (junto con la información de enrutamiento) durante la existencia del registro. En el momento en que el I-CSCF recibe peticiones o respuestas envía los mensajes al S-CSCF que ha sido asignado según los datos de registro. El registro dura mientras el equipo terminal permanezca en el área de servicio. Por ejemplo, en una red inalámbrica, mientras el equipo terminal continúe recibiendo un servicio desde la misma celda, el registro permanece sin cambios, sin embargo tan pronto como el suscriptor se cambie a otra celda, el registro cambiará (debido a que la dirección de la celda cambió). En el modelo de las redes cableadas, mientras un equipo terminal mantenga la misma dirección IP, el registro se mantiene sin cambios, no obstante si al suscriptor se le asigna otra dirección IP, el equipo terminal debe registrar esta nueva dirección [4][5].

Para todos los registros existe un temporizador asociado, así bien cada registro tiene una expiración y un vez esta es alcanzada, el registro será terminado por el S-CSCF y el HSS. Esto impide a un equipo terminal registrarse en la red y luego desconectarse sin cancelar el registro. Esa situación desde luego proporcionaría una oportunidad para que un hacker se aproveche del registro. Para prevenir la exposición de la topología de red (el número de los S-CSCFs, direcciones de entidades internas, etc.) a redes en las que no se tiene confianza, el I-CSCF proporciona una función conocida como ocultamiento de topología, esto consiste en eliminar determinadas cabeceras SIP para prevenir que otras redes obtengan información valiosa. Por ejemplo la cabecera ROUTE contiene las direcciones de todas las entidades IMS que fueron usadas para encaminar un mensaje. Estas direcciones podrían ser usadas para identificar el número de saltos hasta el S-CSCF, así como información de la red que es restringida y que el operador probablemente no compartiría con otras redes. Si un intruso informático fuese capaz de capturar todos los mensajes SIP usando un analizador de protocolos (o cualquier otro método de *sniffing* sobre la red), estaría en capacidad de utilizar esta información para determinar el tiempo de ida y vuelta en la red y calcular otros parámetros que podrían ser usados para ataques de negación de servicios. El I-CSCF podría ser considerado como el firewall en la red IMS, encargado del enrutamiento entre otras redes y previniendo el acceso no autorizado. Así por consiguiente el I-CSCF juega un rol muy importante en la seguridad de IMS [4][5].

1.1.3.4 Serving CSCF (S-CSCF)

En IMS el S-CSCF cumple el papel de núcleo principal de la red. Esta entidad controla todos los aspectos relacionados con los servicios de un suscriptor, manteniendo el estado de cada una de las sesiones que han sido iniciadas. Una sesión es básicamente cualquier cosa que un dispositivo desee hacer, tal como mensajería instantánea, e-mail, envío de imágenes, listas de correo, voz, etc. El S-CSCF controla la mensajería y la distribución de contenidos. Este proporciona el estado del registro de un suscriptor a otras aplicaciones (servidores de aplicación) y mantiene el control sobre esos servicios mientras el dispositivo este registrado. El S-CSCF es responsable por la autenticación de todos los suscriptores que intentan registrar su ubicación con la red [4][5].

Desde una perspectiva SIP, el S-CSCF es el registrador, responsable de autenticar a todos los usuarios que intentan registrar su ubicación con la red. Cuando esto sucede, el S-CSCF forzará al equipo terminal a enviar otro mensaje REGISTER el cual lleva las credenciales apropiadas y las claves de autenticación antes de permitir el acceso a los servicios.

Esta es una función que no es común en las implementaciones de VoIP actuales. A menudo los *Softswitches* (en los que actualmente se gestiona el control de llamadas) no tienen funciones de seguridad robustas y no hacen una verificación de cada equipo terminal en el momento del acceso. Así bien el área de la seguridad sobre VoIP ha mejorado con especificaciones de la 3GPP [4][5].

Después del registro el S-CSCF almacena la siguiente información relacionada con el dispositivo:

- La dirección del HSS
- El perfil de usuario
- La dirección del P-CSCF (punto de entrada durante el registro)
- El dominio del P-CSCF (en el caso de que el dispositivo haya entrado desde otra red)
- Identidad de usuario pública
- Identidad de usuario privada
- Dirección IP del dispositivo

La duplicación de esta función a través de la red en una configuración de malla sería extremadamente difícil, por no hablar de ineficiente. El espíritu de IMS radica en la utilización de una función principal para la seguridad y el control de acceso, lo cual ha sido probado exitosamente por muchos proveedores. Ubicar estas funciones en los límites de la red tampoco tiene sentido, ya que muchas veces los ataques provienen desde el interior de la red [4][5].

Sin mencionar el asunto de la autenticación en las fronteras de la red. Los dispositivos de frontera aun necesitan conocer la autenticación y las claves de cifrado para cada suscriptor, ellos aun tendrían que tener acceso a alguna función centralizada para esta información. El S-CSCF también tiene la responsabilidad de habilitar servicios mediante el acceso a diversos AS dentro de la red [4][5].

Por ejemplo, si un equipo terminal intenta unirse a una videoconferencia, el S-CSCF proporcionaría la conectividad al AS apropiado de acuerdo con la suscripción (como haya sido definido en el HSS) y los requerimientos multimedia definidos en el protocolo SIP SDP (*Session Description Protocol*, Protocolo de Descripción de Sesión) [4][5].

Esto significa que el S-CSCF necesita conocer los servicios a los cuales se tiene permitido el acceso, y las direcciones de los servidores que proporcionan tales servicios. El S-CSCF accede al HSS para identificar el perfil de la suscripción, este a su vez incluye el perfil del servicio. Esto se explicará en detalle en la sección 1.2.4.

El S-CSCF tiene la capacidad de dividirse o ramificarse para algunos servicios como las conferencias. Por ejemplo, durante una videoconferencia, el equipo terminal invita a múltiples usuarios, en este caso el S-CSCF divide el mensaje INVITE para que llegue la petición a cada una

de las partes direccionadas. Las preferencias de división o ramificación son definidas generalmente por el dispositivo en el momento del registro, no obstante si estas preferencias no se determinan en ese momento, el S-CSCF tiene la capacidad de determinar la ruta de la petición [4][5].

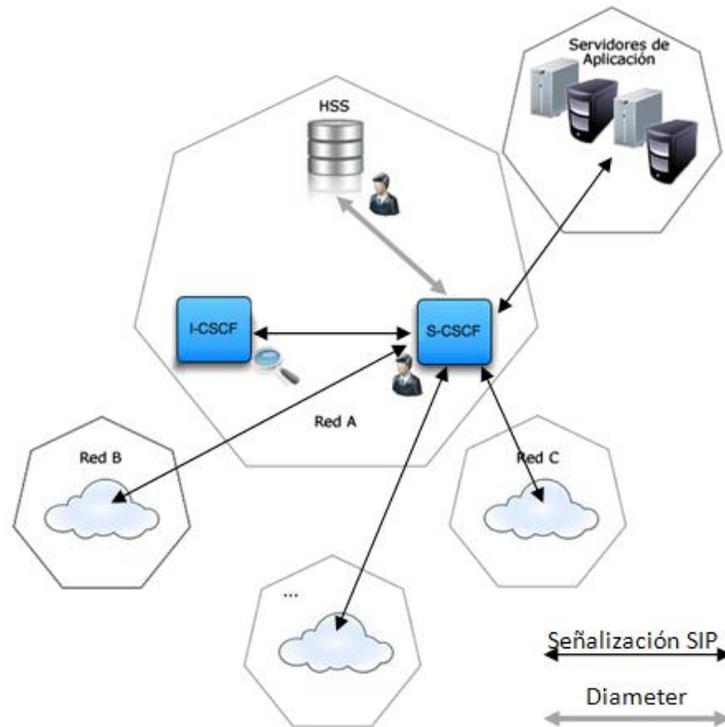


Figura 5. Funcionalidad del Serving CSCF

Otra de las tareas que realiza el S-CSCF es la conversión de direcciones (esta también es realizada por el P-CSCF y el I-CSCF). Dado que las rutas SIP están basadas en SIP URIs, cualquier TEL URIs debe ser traducido o convertido a una SIP URI. Lo mismo ocurre en el sentido opuesto cuando se hace el enrutamiento desde IMS hacia la PSTN (Public Switched Telephone Network, Red Telefónica Pública Conmutada). El S-CSCF tiene acceso a la aplicación ENUM/DNS para convertir las direcciones a SIP URI antes de enviar cualquier petición o respuesta a su destino.

La aplicación ENUM/DNS puede ser ubicada dentro del servidor de aplicación o puede ser una función ENUM independiente. El objetivo de esta es permitir el enrutamiento hacia/desde las redes legadas donde los números E.164 son usados para llegar a los abonados, dando soporte al concepto de SIP URI [4][5].

El S-CSCF siempre debe mantener el estado de todos los registros y sesiones que se encuentren bajo su control, de esta manera al conocer los estados el S-CSCF tiene la capacidad de actualizar a otras entidades que pueden haberse suscrito a la notificación de eventos utilizando el método SIP SUSCRIBE [4][5].

Siempre que el estado de un equipo terminal cambie dentro de la red, el S-CSCF deberá notificar a todas las entidades que participan del cambio de estado. Esto le permite a servicios tales como presencia, ser notificados si el estado del registro de un suscriptor debe cambiar, previniendo también el acceso no autorizado al estado del registro. Sólo entidades autorizadas tienen permitido el uso del método SUBSCRIBE, las mismas han sido validadas por el S-CSCF. Esto también elimina la necesidad de que el equipo terminal por sí mismo envíe el estado del registro a múltiples entidades en la red [4][5].

En resumen, el S-CSCF es el núcleo de IMS. Este es el corazón de la red, el cual proporciona un punto de control que permite a los operadores manejar tanto la distribución de servicios como el establecimiento de sesiones. En redes de gran tamaño deben existir varios S-CSCF, así bien esta es una función distribuida. El S-CSCF debe ser desplegado en la red de acuerdo con su número de suscriptores y según el tipo de servicios que esta entidad deba controlar y soportar [4][5].

1.1.3.5 Home Subscriber Server (HSS)

El *Home Subscriber Server* (HSS) es la base de datos maestra que contiene la información de usuario y de suscriptor y se encarga de apoyar a las entidades de red en el establecimiento tanto de llamadas como de sesiones [4][5]. Ofrece las siguientes funciones:

- manejo de la identificación,
- autorización para el acceso,
- autenticación,
- gestión de la movilidad (seguimiento de las entidades de control de sesión, que están brindando un servicio al usuario),
- soporte para el establecimiento de sesión,
- soporte para la prestación de servicios y
- soporte para la autorización de servicios.

Cuando un usuario se registra en un dominio IMS, el perfil de usuario (definido como la información relevante, relacionada con los servicios que están habilitados para el usuario) es descargado desde el HSS al CSCF. Para el establecimiento de una sesión, el HSS proporciona información del CSCF qué actualmente está prestando sus servicios al usuario. Cuando existe más de un HSS desplegado en la red, entra en acción el SLF, cuya función es localizar al HSS que está almacenando los datos de suscripción de un determinado usuario. Tanto el HSS como el SLF utilizan el protocolo Diameter (bajo las interfaces Cx y Dx) [4][5].

Mientras el S-CSCF actúa como el núcleo de la red, el HSS cumple con el papel de fuente central de los datos del suscriptor. El HSS almacena los datos del usuario tales como los servicios a los que un suscriptor tiene acceso, las diversas identidades (la identidad de usuario privada, y todas las identidades de usuario públicas), las redes a las cuales el usuario tiene acceso para transitar (en el caso de las redes inalámbricas), y la localización del dispositivo del suscriptor [6]. Cuando un suscriptor se registra en la red, el S-CSCF accede al HSS para obtener el perfil de usuario. El perfil de usuario es lo que identifica a todas las identidades públicas y privadas asociadas a la suscripción, así como los perfiles de servicio para cada una de las identidades [4][5].

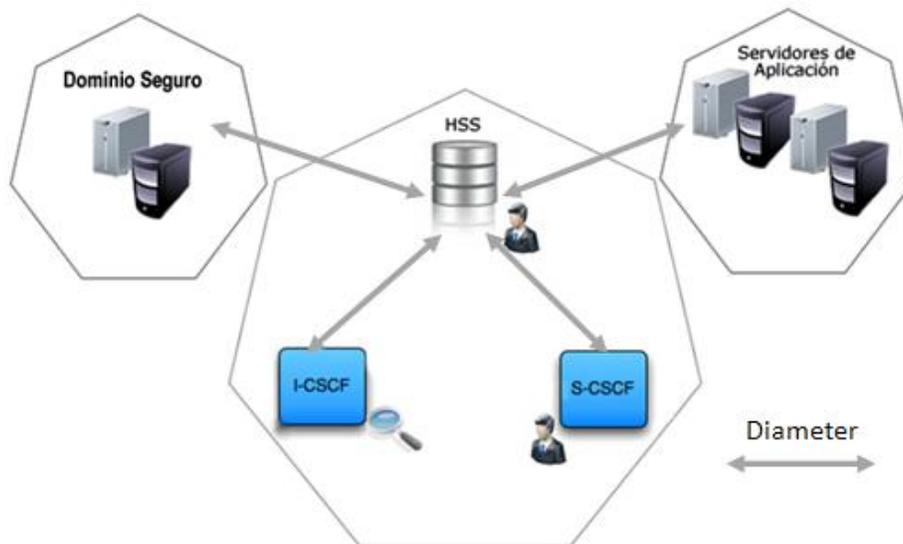


Figura 6. Funcionalidad del HSS

Siempre que hay un cambio en la suscripción de un equipo terminal, la información es llevada al S-CSCF. El HSS envía la totalidad de los datos de suscripción al S-CSCF, y de ninguna manera datos parciales. Esto elimina la posibilidad de que existan datos corruptos o que se encuentren fuera de sincronización con el HSS. A continuación el S-CSCF, sustituye todos los datos de suscripción que tiene almacenados por los datos que son enviados por el HSS [4][5].

El objetivo del registro es proporcionar una ubicación para el equipo terminal. Esto no significa necesariamente la localización exacta. En el caso inalámbrico, esta puede estar dada por las coordenadas de un GPS, pero por lo general está dada por el identificador del sitio donde se encuentra la celda. En las redes fijas, esta información depende de donde esté ubicado el P-CSCF que fue usado para acceder a la red. Esta también puede estar dada por la dirección IP asignada al equipo terminal. En otras palabras, la localización identifica como encaminar las sesiones hacia el suscriptor en cualquier momento [4][5].

Es por esto que el registro es tan dinámico. En cada momento el suscriptor cambia de localización, las direcciones asignadas en el nivel IP deben ser cambiadas y actualizadas en el registro de tal manera que las sesiones se puedan establecer correctamente [4][5].

Los datos del suscriptor también hacen parte de la información de registro, tanto la identidad privada del usuario como las identidades públicas son almacenadas como parte de los datos de registro. Esto le permite al S-CSCF proveer un soporte completo al equipo terminal del suscriptor y a cualquier servicio que desee utilizar [4][5].

Los servicios tienen identificadores y se almacenan asociados a una identidad de usuario pública. Las identidades de usuario públicas pueden ser asignadas a múltiples identificadores de servicio, basándose en la suscripción. Esta información es almacenada en el HSS, y es compartida con el S-CSCF cuando el dispositivo terminal se registra en la red [4][5].

Si un suscriptor no tiene acceso a los servicios, el operador bloquea en el HSS la identidad de usuario pública o privada asociada con la suscripción. Esto proporciona una ubicación central donde la suscripción puede ser controlada. Si el suscriptor se encuentra en otra red, la restricción de los servicios sigue siendo válida, desde la red visitada se consulta al HSS mediante el S-CSCF para así determinar los servicios a los que el suscriptor puede tener acceso. [4][5]

Una de las funciones más críticas del HSS es proporcionar el cifrado y las claves de autenticación para cada suscripción. Cuando el equipo terminal se registra en la red, el S-CSCF asignado pide los datos de identificación almacenados por el HSS. El S-CSCF consulta al HSS cuáles son los datos de identificación que fueron consignados durante el registro, el equipo terminal retorna un mensaje que contiene la información de identificación adecuada.

Sólo el proveedor de red y el equipo terminal conocen las claves, estas se encuentran programadas en el equipo terminal (más específicamente en la tarjeta SIM del equipo) y en el HSS. Esta es una de las razones por las que las funcionalidades ofrecidas por HSS y el S-CSCF se encuentran en el núcleo de la red.

Pueden existir muchos HSS desplegados en una misma red IMS, dependiendo del número de suscriptores que deben ser soportados por la red. El HSS debe tener la capacidad de soportar a los suscriptores que le sean asignados [4][5].

El HSS no está disponible para otras redes. Solamente el S-CSCF alojado en la misma red puede tener acceso a esta entidad. Esto es debido a los datos de suscriptor almacenados dentro de cada HSS. Si se da acceso a un dominio no confiable se podrían dar violaciones de seguridad generándose oportunidades para el robo de información de la identidad de un suscriptor. El P-CSCF y el I-CSCF protegen al S-CSCF y al HSS de accesos no autorizados [4][5].

Todos los servicios a los que un suscriptor tiene acceso están registrados en esta entidad, en el HSS también se registran los cambios que realizados en la suscripción. Esta es una marcada diferencia con las implementaciones de VoIP, donde el suscriptor y sus privilegios son manejados a través de diversos softswitches, en una configuración en malla o usando servidores de aplicación accesibles por el suscriptor [4][5].

Otra ventaja del HSS es la habilidad para manejar múltiples identidades sobre una suscripción común. Una suscripción puede tener sólo una identidad de usuario privada, pero esta puede tener múltiples identidades de usuario públicas. Los identificadores de servicio también pueden ser asignados a cada una de las identidades de usuario públicas basadas en una suscripción [4][5].

1.1.3.6 Servidores de Aplicación (AS)

Los Servidores de Aplicación proveen servicios específicos a los usuarios finales. Estos servicios comprenden: juegos multi-usuario, videoconferencia, mensajería, servicios comunitarios y compartimiento de contenido. IMS define tres tipos de servidores de aplicación:

- Servidores SIP,
- Servidores OSA (Open Service Access, Acceso a Servicios Abiertos)
- Servidores CAMEL (Customise Applications for Mobile Networks Enhanced Logic, Aplicaciones a la Medida para Redes Móviles con Lógica Mejorada)

Los servidores SIP se comunican directamente con los S-CSCF a través del protocolo SIP. Los servidores OSA cumplen la misma función, pero requieren el uso de un servidor SCS (Service Capability Server, Servidor de Capacidades de Servicio) entre el servidor OSA y el S-CSCF para traducir mensajes SIP. El ambiente de servicio CAMEL, es un conjunto de mecanismos que permiten al operador de la red entregar servicios específicos de operador a los usuarios, incluso cuando se encuentran haciendo 'roaming', a través de IM-SSP (IP Multimedia - Service Switching Point, IP Multimedia – Punto de Conmutación de Servicios), que traduce las solicitudes CAMEL a solicitudes SIP. Dependiendo de la implementación, un AS puede contener una o más aplicaciones IMS. En ambos casos, el AS manipula e interpreta los mensajes SIP enviados por el S-CSCF para enviar de vuelta una respuesta a través de este mismo servidor. La arquitectura IMS permite al proveedor establecer diferentes aplicaciones en el mismo dominio. Diferentes ASs pueden ser desplegados para diversas aplicaciones o grupos de usuarios. El S-CSCF decide a cual servidor de aplicación envía una solicitud SIP, esta decisión se basa en información de filtro entregada por HSS.

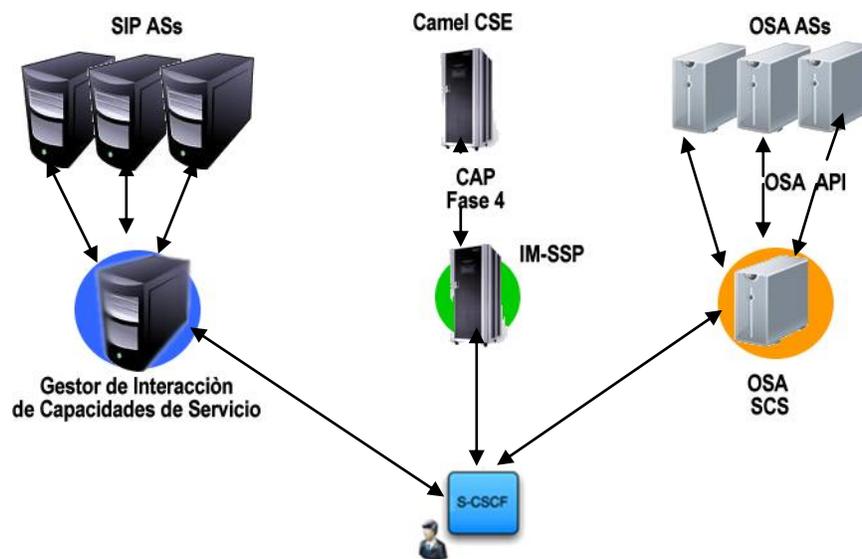


Figura 7. Tipos de Servidores de Aplicación

Cuando el HSS transfiere la información y dirección de más de un AS, el S-CSCF debe contactar cada AS en el orden provisto. El S-CSCF utiliza la primera respuesta de un AS como entrada para el segundo y así sucesivamente. El servidor de aplicación usa reglas de filtrado para decidir que aplicaciones serán servidas al usuario en la sesión. Durante la ejecución del servicio, el servidor puede comunicarse con el HSS a través del protocolo 'DIAMETER' para adquirir información

adicional del suscriptor o aprender de los cambios en el perfil de usuario. Los servidores de aplicación deben cumplir con los siguientes requerimientos:

- Soporte para un amplio rango de servicios para usuarios finales.
- Rápido despliegue y creación del servicio.
- Configuración sencilla del servicio.
- Evolución independiente entre servicios e infraestructura.
- Soporte para ambientes multi-jugador.
- Acceso universal a los servicios.

1.2 PERSONALIZACIÓN Y PERFILES DE USUARIO

Al momento de desplegar y generar servicios convergentes para los usuarios, los operadores deben pensar en algunos elementos con el fin de garantizar una buena experiencia. El primero de tales elementos es la transparencia y ubicuidad, los cuales deben permitir a los diferentes usuarios acceder y poder disfrutar de un conjunto de servicios, en forma independiente de su red y el dispositivo de acceso, lo cual es plenamente soportado por IMS. El segundo aspecto que en la actualidad cobra gran importancia, es la personalización; esto hace referencia a que el usuario espera datos que sean relevantes para él, acordes con sus necesidades e intereses y enmarcados dentro de su contexto [7]

Esta sección del documento se centrará en la descripción del concepto de Personalización de Servicios el cual va de la mano con el concepto de Perfil de Usuario.

1.2.1 Contexto de la personalización de servicios

La personalización de servicios, entendida como la capacidad de alteración dinámica con el fin de proporcionar al usuario la impresión de estar trabajando con una aplicación específicamente diseñada para dar satisfacción a sus necesidades particulares, se puede producir en base a tres tipos principales de fuentes de información [8][9] [10].

- Datos preexistentes, normalmente importados de fuentes externas.
- El perfil y las preferencias de cada usuario individual, que pueden ser proporcionadas de forma explícita por el propio usuario o recogidas por la aplicación de manera implícita, a través de la actividad del usuario en el sistema.
- El contexto en el que se produce la interacción usuario-aplicación (localización, momento, características de la red, etc), que se recoge (si está disponible) de forma implícita por la aplicación como parte integrante de esa interacción.

La integración o no de cada una de estas fuentes de datos en el seno de una aplicación concreta determina el conjunto de técnicas que pueden ser aplicadas con el fin de implementar la política de personalización deseada. En la actualidad, el número y variantes de estas técnicas parecen incontables [11] [11]. No obstante se pueden diferenciar 4 grandes [12][13].

Filtrado Simple. El Filtrado Simple consiste en la definición de grupos de usuarios, a los que se asocian vistas particulares de la aplicación. Estos grupos de usuarios pueden estar predefinidos (por ejemplo en base a roles), o bien crearse en base a determinados atributos del usuario o del contexto (URL (*Uniform Resource Locator*, Localizador de Recurso Uniforme) de conexión, género, edad, etc.). Como ejemplo, se puede diseñar una vista restringida de la aplicación destinada a niños, y determinar que cualquier usuario menor de 12 años acceda a esa vista. Otro ejemplo clásico es el análisis de la URL para determinar el idioma en que se debe presentar la aplicación.

Filtrado por contenido. En este tipo de técnica, partiendo de un conjunto de categorías, se determina la pertenencia o no de los objetos a cada una de dichas categorías, y se muestran aquéllos que entren en el grupo de categorías de interés de cada usuario. Esta técnica tiene como principal inconveniente el amplio grado de objetividad que tiene que darse para que el proceso de clasificación de objetos sea efectivo, objetividad que, dependiendo de la naturaleza de los objetos a clasificar, no siempre es posible. No obstante, se suele utilizar en determinados tipos de sistemas, como son los de clasificación de documentos.

Filtrado colaborativo. Esta técnica, permite aplicar criterios subjetivos y por tanto es muy utilizada como soporte de sistemas de recomendación (ver ej. Amazon [14]), se basa en la actividad de otros usuarios de comportamiento similar al usuario actual para determinar aspectos como cuál será la próxima acción que el usuario realice en el sistema o en qué productos puede estar interesado. El filtrado colaborativo está en el origen del concepto de la Navegación Social [15] que pretende simular los mecanismos de orientación de los que dispone el usuario en el mundo real. Esta técnica, que proporciona una capacidad de adaptación elevada a la aplicación, tiene como principales inconvenientes el alto coste computacional al que puede dar lugar en presencia de un alto volumen de usuarios y el bajo grado de fiabilidad de los algoritmos ante un volumen de información reducido, lo que obliga a establecer estrategias alternativas mientras no se disponga de dichos datos.

Perfilado (Profiling). Por último, existen técnicas que, a partir de un identificador proporcionado por el propio usuario de manera implícita, permiten la personalización de la aplicación en base a la actividad del usuario en otras aplicaciones. El uso de este identificador global limita los problemas asociados a la seguridad y privacidad de datos personales. Sus principales inconvenientes son, por un lado, la necesidad de que el propio usuario obtenga y proporcione este identificador global, y por otro la necesidad de intercambio e integración de esa información proveniente de fuentes externas para su posterior procesado.

El efecto que la elección de una u otra técnica de personalización tiene sobre la aplicación se materializa no sólo en el uso de un tipo u otro de técnica de adquisición (que determina la manera en que se recoge la información necesaria), sino que define la propia capacidad de personalización de dicha aplicación, dando lugar a su clasificación en tres grandes grupos: adaptables, adaptivas o proactivas (ver Tabla 1).

Fuente Información	Información proporcionada explícitamente por el usuario o por el diseñador	Información inferida a partir de la actividad del usuario o del contexto de uso de la aplicación
Causa personalización		
Personalización causada por la interacción explícita usuario-sistema	ADAPTABLES	ADAPTATIVAS
Personalización causada por evento ajeno a la interacción explícita usuario-sistema	PROACTIVAS	

Tabla 1 Tipos de personalización

En las aplicaciones adaptables (*Customizable* [10]), es el usuario el que desencadena la acción de personalización, que además se basa en información introducida por el propio usuario en el sistema. Suelen utilizar por tanto técnicas de filtrado simple, filtrado por contenidos o perfilado. Las aplicaciones adaptativas (*Adaptive* [10]) utilizan por el contrario información recogida y analizada por la propia aplicación de manera transparente para el usuario, aunque sigue siendo éste el que desencadena la acción de personalización. Ejemplos típicos de este tipo de aplicaciones son los sistemas de recomendación utilizados como apoyo en numerosas aplicaciones de comercio electrónico, que utilizan mecanismos de filtrado colaborativo. Por último las aplicaciones proactivas (*Proactive* [9]) son capaces de detectar de forma autónoma cambios en el entorno y reaccionar en consecuencia, actuando por tanto como Observadoras [16] de dicho entorno. La principal diferencia con respecto a las aplicaciones adaptativas es por tanto que, mientras que en éstas cualquier cambio es causado por una acción del usuario, una aplicación proactiva puede cambiar la vista del usuario sin necesidad de que éste realice ninguna acción. Un ejemplo de comportamiento proactivo es la capacidad por parte de una aplicación de detectar un cambio en la localización de un dispositivo de acceso móvil, y la modificación automática de la vista de información ofertada en función de dicha localización.

1.2.2 Personalización basada en perfil de usuario.

Como se mostró en la sección anterior la personalización de servicios, se puede producir en base a tres tipos principales de fuentes de información, entre ellas el perfil y las preferencias de cada usuario individual. Habitualmente, los términos de perfil y modelo del usuario se emplean indistintamente como sinónimos o se utiliza uno de ellos para referirse a ambos. Sin embargo, son conceptos diferentes y es necesario aclararlos. La diferencia entre un perfil y un modelo del usuario radica en que el nivel de sofisticación es diferente, siendo más alta la complejidad del modelo. Incluso algunos autores consideran que el perfil del usuario es un modelo del usuario muy simple. El *perfil* del usuario es una colección de información personal. La información se almacena sin añadir descripciones más detalladas o interpretaciones de dicha información. Se puede comparar a un mecanismo de recuperación-establecimiento (*get/set*) de clases en un lenguaje de programación orientado a objetos, donde los diferentes parámetros se pueden establecer o recuperar. Los perfiles del usuario representan habilidades cognitivas e intelectuales, intenciones, estilos de aprendizaje, preferencias e interacciones con el sistema. Estas propiedades se almacenan después de haberles asignado un valor, que puede ser constante o dinámico. Dependiendo del contexto y de la cantidad de información disponible acerca del usuario, la cual está almacenada en su perfil, el usuario puede ser *modelado*. Por tanto, el perfil de usuario se utiliza para recuperar la información necesaria para

construir el modelo del usuario. El modelo se basa en esta información, por lo que sólo es una pequeña parte del usuario real. No obstante, el modelo del usuario representa las características necesarias de dicho usuario en relación al contexto de la aplicación.

1.2.3 Creación de perfiles.

La analogía entre diseñar una silueta y crear un perfil de usuario es en este caso bastante buena: se puede imaginar que el programa de creación de perfiles sostiene una hoja entre el computador y el usuario, y su tarea es la de construir el perfil del usuario a partir de la sombra en la hoja. No sabe nada de él, pero puede seguir sus acciones. Cuando un perfil es usado para representar “cosas del mundo real”, necesita saber que “características tienen”. Éstas son dadas a través de una ontología. Una ontología es una especificación formal explícita de cómo representar objetos, conceptos y otras entidades, que se asume que existen en una determinada área de interés, y las relaciones que se mantienen entre ellos. El perfil proyecta el modelo del mundo real, hacia el mundo definido por una ontología. Un sistema que hace *profiling*, necesita una ontología conocida y común a los perfiles del sistema, así como de métodos para crearlos. Hay tres métodos principales para crear perfiles: el método explícito o manual; el método colaborativo o de composición a partir de otros perfiles; por último, el método implícito, que utiliza técnicas específicas para extraer las características automáticamente. En el método explícito o de creación manual, los datos son introducidos por el usuario escribiéndolos directamente en su perfil de usuario, respondiendo a formularios, etc. También se puede crear y modificar un perfil a partir de la interacción colaborativa con otros perfiles, con los que se relaciona, recurriendo a conocimiento específico del dominio y heurísticas inteligentes. En el método implícito, se extraen/crean y modifican/actualizan automáticamente los perfiles, recurriendo normalmente a técnicas de Inteligencia Artificial para realizar estas tareas. Es importante señalar que estos muchas veces se utilizan simultáneamente (métodos híbridos), para producir perfiles más precisos y comprensibles.

1.2.4 Perfil de usuario en el entorno de IMS

Cada suscriptor en IMS debe tener un perfil de usuario, esta información define los servicios y las formas como los usuarios acceden a los mismos. El perfil es almacenado en el HSS que se encuentra en la red de origen del suscriptor. El perfil de usuario está ligado a una única identidad de usuario privada y a una colección de identidades públicas de usuario. Una identidad de usuario pública puede ser un SIP URI o un TEL URI. Un SIP URI se representa de la siguiente forma: sip:first.last@operator.com; por otro lado un TEL URI adopta típicamente la siguiente forma: tel:+1-212-555-0293. En IMS, las identidades públicas de usuario son usadas para direccionar peticiones SIP.

Las identidades Privadas de Usuario a diferencia de las identidades públicas no son SIP URIs o TEL URIs, estas identidades toman el formato NAI (*Network Access Identifier*, Identificador de Acceso a la Red), este formato se representa de la forma nombreusuario@operador.com. Las identidades privadas de usuario son usadas exclusivamente para identificación de la suscripción y autenticación. Estas identidades no son usadas para el direccionamiento de peticiones SIP. El perfil de usuario también contiene Perfiles de Servicio los cuales definen condiciones de activación (*triggers*) que se aplican a cierto grupo de identidades públicas de usuario (Figura 8).

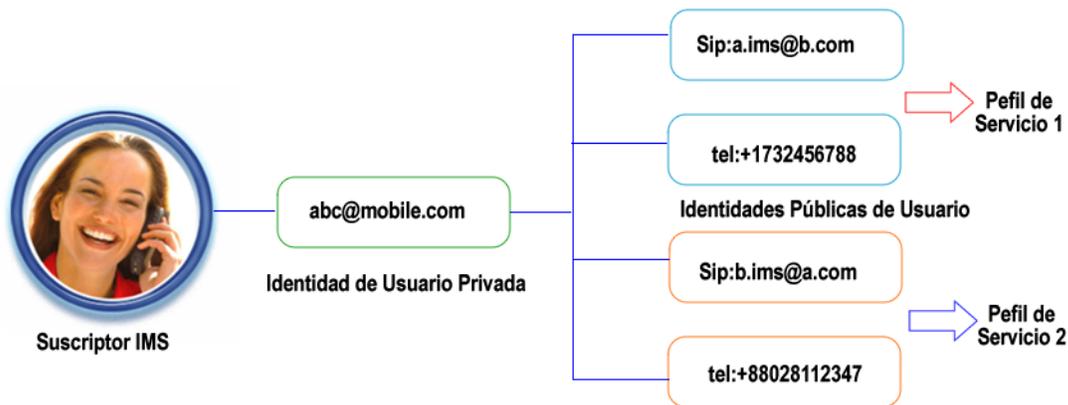


Figura 8. Relaciones entre identidades de usuario y perfiles de servicio.

Un perfil de servicio esta dividido en tres partes: una colección de una o más identificaciones públicas, cero o mas criterios de filtrado (*filter criteria*) y opcionalmente autorizaciones de servicios del núcleo de la red. (Figura 9) [1].

Parte de la información que se maneja al interior de IMS relacionada con perfil de usuario es transparente para los desarrolladores, ya que son las entidades de red las únicas que tienen acceso a estos datos, no obstante las identidades públicas y privadas, son básicas a la hora de hacer pruebas bajo un entorno de simulación. Cabe entonces resaltar que en la información necesaria para construir un perfil de usuario para la personalización de un servicio, debe participar parte de la información visible que es manejada por el perfil de usuario de IMS.

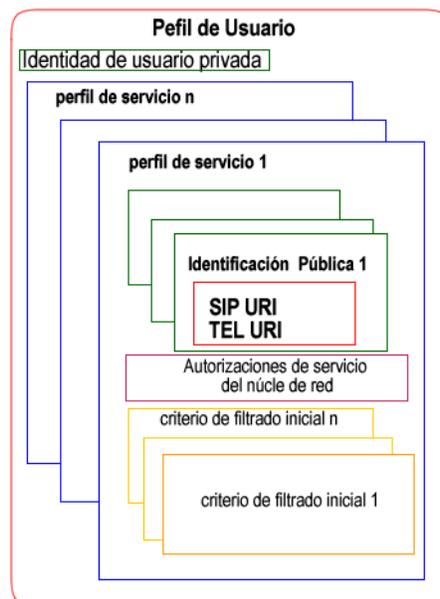


Figura 9. Perfil de Usuario y perfiles de Servicio

CAPÍTULO 2 ONTOLOGÍAS Y SU USO EN LA PERSONALIZACIÓN DE SERVICIOS

Hoy en día, el uso de Internet ha hecho que los datos cobren mucha importancia. Sin embargo, estos datos por sí solos, sin tener una semántica asociada, no son útiles, ya que resultan ambiguos, y por esto, es necesario documentarlos para dotarlos de un significado, es decir, agregarles una descripción a los datos por medio de más datos, a los que se les ha denominado metadatos. Los metadatos permiten estructurar contenidos al describir las partes de un recurso. Sin embargo, si lo que se quiere es dar una descripción formal de contenidos mediante el modelado de conceptos y relaciones, hace falta algo más potente que los metadatos, es aquí donde surgen las ontologías [17].

En este capítulo se da una definición del concepto de Ontología, a la vez que se describen sus características y ventajas. Posteriormente, se enumeran los pasos básicos para su diseño y construcción, y se finaliza el capítulo con una descripción del papel que desempeñan las ontologías en la personalización de servicios.

2.1 DEFINICIONES DE ONTOLOGÍA

Existen múltiples definiciones de lo que son las ontologías, como la dada por Neches [18] que la define como “los términos básicos y relaciones de un vocabulario dentro de un área o tema, así como las reglas para combinar términos y relaciones que sirvan para extender el vocabulario”; la de Gruber [19] que la define como “una especificación explícita de una conceptualización”, es decir, que proporciona una estructura y contenidos de forma explícita que codifica las reglas implícitas de una parte de la realidad; la de Borst [20] quien modificó esta última definición, diciendo que “las ontologías se definen como una especificación formal de una conceptualización compartida”. Partiendo de estas definiciones una ontología puede definirse como una conceptualización basada en un conjunto de conocimientos expresados formalmente, los cuales tienen una visión subjetiva de un dominio, cuyo esquema conceptual facilita la comunicación y el intercambio de la información entre sistemas [21].

Desde el punto de vista de las telecomunicaciones, las ontologías se pueden ver como teorías que especifican un vocabulario común tanto para personas como para aplicaciones que trabajan en un dominio determinado. Este vocabulario define entidades, clases, propiedades, restricciones y, las relaciones entre estos componentes. Las clases son el centro de la mayoría de las ontologías, dado que estas describen conceptos de un dominio y poseen subclases que representan conceptos que son más específicos que la superclase. Las propiedades llamadas también *slots* describen las características de las clases e instancias de las mismas. Las restricciones también denominadas *facetas*, o restricciones de rol son aplicadas a los *slots*. [22].

2.2 CARACTERÍSTICAS DE LAS ONTOLOGÍAS

Algunas de las características más representativas de las ontologías son las siguientes:

- Pueden existir ontologías múltiples: El propósito de una ontología es hacer explícito algún punto de vista, lo cual, en algunas ocasiones es necesario combinar dos o más ontologías. Cada una de ellas va a introducir sus conceptualizaciones específicas [22].
- Multiplicidad de la representación: Un concepto puede ser representado de muchas formas, por lo que pueden coexistir múltiples representaciones de un mismo concepto [22].
- Mapeo de ontologías: Permiten crear relaciones entre los elementos de una o más ontologías, para establecer conexiones, especializaciones, generalizaciones, etc. [22].
- Modelado: Las ontologías permiten definir diferentes niveles de generalización o abstracción, los cuales conforman una topología de ontologías. Dado que es muy complejo tener una descripción completa del mundo, se define una red de ontologías usando multiplicidad y abstracción, que permite crear una estrategia de construcción gradual de abajo hacia arriba, donde la definición de un área específica de conocimiento puede ser reutilizada [23].
- Colaboración: Las ontologías proporcionan una base de conocimiento unificado que puede ser usado como un dominio público y compartirse como referencia para impulsar el desarrollo y la participación [23].
- Interoperabilidad: Las ontologías facilitan la integración de diferentes fuentes de información [23].
- Formación: Las ontologías proveen una fuente confiable y objetiva de información convirtiéndola en un buen medio de publicación y referencia [23].

2.3 VENTAJAS

Entre las ventajas que nos ofrecen las ontologías se destacan:

La disminución de la confusión semántica. Reduce la ambigüedad terminológica al considerar sinónimos (dos o más palabras con el mismo significado) y polisemias (una palabra tiene más de un significado), mejorando notablemente la comunicación [22].

La posibilidad de reutilizar conocimientos, dado que las ontologías realizadas sobre cualquier área pueden ser aprovechadas, gracias a que su desarrollo refleja formas concretas de ver el mundo [22].

Los aspectos colaborativos garantizados por las ontologías, son útiles para hacer posible la creación de servicios definidos por diferentes partes. Un proveedor de servicios necesita compartir de forma dinámica la lógica del servicio (la manera como el servicio será ejecutado y entregado al usuario), pero las partes no necesariamente tienen el mismo modelo de información que el proveedor maneja.

Sin embargo, si los servicios estuviesen descritos en un lenguaje de ontologías, su mapeo sería tan simple como comparar información de un modelo de datos o incluso archivo XML [23].

La construcción de ontologías se presenta como una buena alternativa para alcanzar la interoperabilidad semántica entre diferentes desarrollos, característica que las formas de organización actuales no han logrado satisfacer, convirtiéndose en una importante mejora en la representación de la información, que conlleva a la creación de mejores sistemas de personalización de servicios [22].

En general las ontologías permiten conceptualizar el conocimiento generando formas comunes y compartidas de ver el mundo, que ayudan a los sistemas a gestionar más información que datos, consecuencia de que el experto ha cedido parte de su conocimiento a los sistemas [22].

2.4 DISEÑO Y CONSTRUCCIÓN DE UNA ONTOLOGÍA

La construcción de ontologías no responde a una única aproximación lógica, sino que depende el contexto en el que se construyen. Además se debe tener en cuenta que una ontología especifica una conceptualización, es decir una forma de ver el mundo. Por lo cual cada ontología incorpora un punto de vista. Una ontología contiene definiciones que se proveen del vocabulario para referirse a un dominio y éstas dependen del lenguaje que se usa para describirlas [22][26].

Para el diseño de una ontología se debe tener en cuenta los siguientes aspectos [22][26]:

- Claridad: una ontología debe comunicar de manera efectiva el significado de sus términos. Las definiciones deben ser objetivas y comentadas en lenguaje natural.
- Coherencia: debe permitir hacer inferencias que sean consistentes con las definiciones.
- Extensible: debe anticipar usos y permitir extensiones y especializaciones.
- Mínimo compromiso ontológico: debe hacer la menor cantidad posible de "pretensiones" acerca del mundo modelado.

Existen diferentes estructuras para la construcción una ontología, una de estas sugiere los siguientes pasos a seguir [22][26]:

- Identificación del propósito y alcance (usuarios potenciales).
- Captura:
 - Identificación de los conceptos y relaciones claves en el dominio de interés.
 - Producción de definiciones no ambiguas de conceptos y de sus relaciones.
 - Identificación de términos para referirse a estos conceptos y relaciones.

- Codificación: Representación explícita de la conceptualización en un lenguaje formal:
 - Términos básicos de especificación (a veces llamado meta ontología).
 - Lenguaje de representación adecuado.
 - Codificación de este lenguaje.
- Integración de ontologías existentes: cómo, cuáles y si se va a usar alguna ontología existente.
- Evaluación: Se considera que la ontología construida va a ser reutilizada por lo tanto debe seguir unos principios básicos:
 - Abstracción: lo más abstracto posible pero suficientemente concreto.
 - Desarrollo Modular: permite aislar conceptos.
 - Representación jerárquica: debe seguir un orden.
 - Estandarización.
- Documentación: Debe de hacerse de forma paralela a los puntos anteriores y debe de contener esta clase de puntos:
 - Tener el tipo de mapeo en que se basa la nueva teoría.
 - Contener diferencias semánticas con las ontologías seleccionadas.
 - Justificación de las decisiones tomadas.
 - Evaluación.
 - Conocimiento adicional para usarla, etc.

Además la ontología construida debe ser indexada y ordenada con las ontologías existentes para su posterior reutilización.

2.5 ONTOLOGÍAS Y LA PERSONALIZACIÓN DE SERVICIOS

En los últimos años se ha reconocido la necesidad de crear sistemas software que se adapten automáticamente a sus usuarios, lo cual ha hecho que la investigación de perfiles de usuario y su contexto se haya extendido a muchas disciplinas que se preocupan por el desarrollo de mecanismos para la personalización de servicios [27].

El desarrollo de mecanismos para la personalización de servicios, se basan en la construcción de perfiles de usuario, los cuales recolectan información que permita adaptarse a sus preferencias,

mostrar sugerencias, aprender de la experiencia de usuario, etc. En esta labor, se utilizan dos fuentes de información: los datos que explícitamente introduce el usuario (nombre y dirección de correo electrónico en los casos más sencillos) y los datos que pueden derivarse indirectamente de la conducta del usuario cuando accede al servicio (frecuencia de visitas, duración de la estancia, enlaces seguidos, etc.) [28]. Sin embargo, los diseñadores de aplicaciones modelan los perfiles de usuario de manera específica, lo que impide la interoperabilidad de las aplicaciones a nivel de perfiles de usuario, incrementado el trabajo para su desarrollo y la posibilidad de cometer errores u omisiones en el modelo del perfil. Es aquí donde las ontologías desempeñan un papel importante debido a la facilidad que ofrece la posibilidad de generar un dominio compartido para de la gestión de perfiles de usuario [27].

Hoy en día, si bien existen un sin número de ontologías publicadas en el área de la personalización de servicios, son muy pocas las que han sido objeto de estandarización, por lo cual, su desarrollo y actualización ha estado a cargo de los investigadores que las crearon [28]. Una de estas ontologías es FOAF del acrónimo en inglés de *friend of a friend*, y aunque inicialmente sólo se uso para describir personas, actualmente también puede usarse para describir otras entidades, como proyectos, organizaciones o grupos. FOAF se ha usado ampliamente para la creación de redes sociales, sin embargo su estructura no está enfocada propiamente para el desarrollo de un perfil de usuario [29].

Otro trabajo relevante para modelar el perfil de usuario se plantea por Golemati, Katifori, Vassilakis, Lepouras y Halatsis en [27], cuyo objetivo principal ha sido el crear una ontología general y entendible que pueda adaptarse a las necesidades de cada aplicación, manteniendo una estructura genérica que permita la portabilidad y comunicación entre diferentes aplicaciones. Para su desarrollo se han incorporado los conceptos y propiedades más relevantes usados en la personalización de servicios, basándose en la literatura existente, aplicaciones y ontologías relacionadas al dominio del contexto del usuario y su perfil.

La ontología planteada en [27] ha servido de base para diferentes desarrollos, uno de estos trabajos se especifica en [30], en el cual se propone el desarrollo de un conjunto de herramientas para la gestión de información personal denominado OntoPIM. En este trabajo se estudia el potencial que tienen las ontologías para indexar información como documentos, fotografías, correos electrónicos o cualquier otro tipo de datos, de acuerdo a dominios creados a partir de los intereses del usuario, facilitándole así su consulta [23].

CAPÍTULO 3 ONTOLOGÍA PARA LA GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS

El presente trabajo realiza una propuesta para la gestión de los perfiles de usuario en el entorno de red IMS, tratando de establecer un acuerdo entre los conceptos utilizados en su arquitectura y otros mecanismos para la personalización de servicios, por medio de la definición de una ontología con un dominio unificado y coherente que permita la reutilización del perfil de usuario a través de diferentes servidores de aplicación.

En este capítulo se exponen los requisitos para dicha propuesta, junto con los requerimientos de su ontología y los aportes que las ontologías ofrecen a los servicios en el entorno de IMS. Luego se expone la arquitectura propuesta como referencia para la solución desarrollada y se describe la interacción de cada uno sus componentes por medio de un diagrama.

3.1 REQUISITOS PARA LA GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS

En la formulación de los requisitos para el establecimiento de un gestor de perfiles de usuario en el entorno de IMS, se ha teniendo en cuenta, que la base de conocimiento utilizada es una ontología de perfiles de usuario cuyos requerimientos se especificaran en la sección 3.2 .

Los requisitos establecidos son los siguientes:

- Es necesario definir un dominio ontológico unificado y coherente de perfiles de usuario que permitan la reutilización de la información de personalización por parte de diferentes aplicaciones en la arquitectura de IMS.
- Se debe especificar una ontología cuya información sobre el perfil de usuario este acorde con los estándares utilizados en la arquitectura de IMS, y la vez, que tenga presente las recomendaciones de la W3C para su desarrollo, dado que dicha organización ha publicado estándares como RDF (*Resource Description Framework*, Marco de Descripción de Recursos) y OWL (*Ontology Web Language*, Lenguaje de Ontologías Web) que proporcionan una infraestructura para la integración empresarial y la compartición y reutilización de datos basados en ontologías.
- Es indispensable tener la capacidad de gestionar todas las tareas relacionadas con el manejo de una ontología, es decir que la solución propuesta debe ser capaz de cargar, modificar y guardar los cambios realizados en una ontología.
- Se debe definir un servicio que permita el intercambio de información del perfil de usuario, de una manera confiable y de fácil acceso.
- Permitir a las aplicaciones IMS, la manipulación del perfil de usuario alojado en la ontología, ofreciéndoles los métodos para gestionarla, es decir generar, actualizar, consultar y eliminar la información de personalización.

- Buscar un manejo eficiente del ancho de banda disponible y estudiar los tiempos de respuesta de las operaciones realizadas en la ontología.

El servicio a desarrollar debe usar protocolos estandarizados, aceptados internacionalmente, basados en IP y que sean de libre uso, a la vez que debe tener en cuenta las recomendaciones establecidas para el desarrollo de una ontología y así lograr una buena interoperabilidad con otros servicios.

3.2 REQUERIMIENTOS DE UNA ONTOLOGÍA DE PERFILES DE USUARIO

Para cumplir con el objetivo de proponer una ontología en el dominio de la personalización de servicios en el entorno de IMS, se estudio la literatura existente, las aplicaciones y las ontologías relacionadas al dominio del contexto de usuario y su perfil, con lo cual se pueden establecer algunos de los requerimientos para el desarrollo de un modelo de usuario general, integral y extensible.

Aplicaciones como la búsqueda en la red [31][32] y gestión de información personal [30] han propuesto el uso de ontologías para modelar el perfil del usuario, ofreciendo modelos que permiten establecer varios de los requerimientos que debe cumplir una ontología en el dominio de la personalización de servicios. Sin embargo, hay que tener en cuenta que en estas así como en otras aplicaciones, las ontologías creadas son modeladas de acuerdo a un dominio particular, específico de cada aplicación.

Una ontología debe contar con las propiedades necesarias para resolver problemas particulares del mundo real. Por esta razón podemos ver que los intereses [27], [31], [33], [34] y preferencias [27] son consideradas propiedades importantes por la mayoría de las aplicaciones que cuentan con una ontología para la descripción de sus perfiles de usuario, convirtiéndolas en uno de los principales requisitos a la hora de personalizar servicios. Tanto los intereses como las preferencias pueden ser representados con una taxonomía adecuada y capturados ya sea por medio de la consulta directa con el usuario o mecanismos automáticos.

Por otra parte, características del usuario como género, nivel educativo, ocupación, habilidades, tanto físicas como mentales, suelen desempeñar un rol importante a la hora de interactuar con un sistema y por ende deberían considerarse en la ontología. Aquí algunos conceptos sugeridos para la creación de perfil de usuario son: La información personal (nombre, cumpleaños, dirección), características generales (los factores físicos: el peso y altura, invalidez física) y su interrelación con otras personas. Otros conceptos como la actividad actual, el terminal usado y su ubicación, también pueden ser relevantes, los cuales harían parte de un perfil dinámico dado que esta información puede variar con el tiempo [27].

La experiencia del usuario y su competencia, así como su interacción con las computadoras u otros dominios similares son conceptos requeridos por muchas aplicaciones de personalización. Aquí es fundamental reconocer cuales son las propiedades más relevantes que deben incluirse en la ontología dada la complejidad que conlleva el definir una medida de experiencia universal y objetiva, con las categorías claramente definidas [27].

Otro de los requisitos que debe cumplir una ontología que pretenda servir de base para la personalización de servicios es que permita la instanciación de conceptos de acuerdo a las necesidades de cada aplicación.

La ontología diseñada debe capturar aquellas propiedades con información de carácter estática y perdurable, así como establecer algún mecanismo que permita tener presente aquellos aspectos temporales que sean relevantes.

Dado que se pretende diseñar una ontología que pueda usarse en la personalización de servicios dentro de la arquitectura de IMS, hay que tener en cuenta las características asociadas al usuario y posiblemente tener en cuenta los servicios a los cuales este está suscrito. Aquí los estándares de la 3GPP brindan un conjunto de propiedades relevantes, como por ejemplo la identidad privada del usuario (*Private User Identity*) de amplio uso en la red de IMS.

3.3 APORTES DE LAS ONTOLOGÍAS A LOS SERVICIOS EN EL ENTORNO DE IMS.

Por lo general el desarrollo de un servicio se realiza en diferentes contextos, puntos de vista y suposiciones de acuerdo a su área de aplicación. Dada la importancia que hoy en día cobra la personalización del servicio prestado, cada servicio, crea sus propios perfiles de usuario, y por ello pueden existir conceptos con significados que, a veces, se solapan y, pueden tener diferentes métodos y estructuras [23]. Por tanto, se crean problemas de comunicación por falta de entendimiento compartido que limita la interoperabilidad y, por consiguiente, el potencial de reutilizar y compartir información, y es aquí donde las ontologías toman un papel clave en la resolución de interoperabilidad semántica entre sistemas de información y su uso, dado que fomentan el desarrollo de un vocabulario consensuado tanto para personas como para aplicaciones de un dominio.

Los NGS (*Next Generation Services*, Servicios de Nueva Generación) son un conjunto de funcionalidades que están organizadas de acuerdo a una lógica del servicio. Estas funcionalidades son reutilizables para definir varios servicios [23]. En este sentido el desarrollo de una ontología para la gestión del perfil de usuario, ofrece un repositorio de información que puede ser reutilizado e incluido dentro del desarrollo de nuevos servicios.

El usuario final no necesita conocer quien genera ni de donde se obtiene la información usada para personalizar sus servicios, sino, obtener un servicio acorde con sus preferencias. Aquí las ontologías aportan la interoperabilidad entre servicios, necesaria para la integración de múltiples fuentes de información a nivel de perfiles de usuario, permitiendo la personalización [23].

La colaboración al interior del proveedor de servicio debe basarse en un buen medio de publicación y una buena fuente de referencia, aquí las ontologías proveen información confiable y objetiva para quienes quieren aprender más acerca de los servicios, perfiles de usuario o consumidores a fin de mejorar sus sistemas. Los expertos en telecomunicaciones también pueden compartir su comprensión de los conceptos y la estructura del dominio [23].

3.4 ARQUITECTURA PARA LA GESTIÓN ONTOLÓGICA DE PERFILES DE USUARIO EN EL ENTORNO DE IMS

En el presente apartado se describe claramente la organización fundamental del sistema propuesto, enfatizando en sus componentes, las relaciones entre ellos y su entorno y los principios que orientan su diseño. La propuesta de arquitectura del sistema de gestión de información de personalización parte de las tecnologías y conceptos que se estudiaron en los capítulos previos y hace uso de patrones como el “factory” y el “singleton” para su concepción. En el patron "Factory" Un objeto especializado se encarga de instanciar objetos para clientes, y estos utilizan los métodos que proporciona la interfaz del objeto recibido, pero sin saber nada de las interioridades del objeto, ni siquiera qué tipo de objeto es (=a partir de qué clase está instanciado)[24]. El patrón de diseño singleton (instancia única) está diseñado para restringir la creación de objetos pertenecientes a una clase o el valor de un tipo a un único objeto. Su intención consiste en garantizar que una clase sólo tenga una instancia y proporcionar un punto de acceso global a ella.[25]. El patrón “factory” simplifica el manejo de la información que ofrece la ontología por medio de una clase constructora que brinda los métodos necesarios para el acceso a los datos del perfil de usuario, y el patrón “singleton” garantiza el acceso global al repositorio ontológico ofrecido.

En la Figura 10 se muestra la arquitectura propuesta acorde a los servicios IMS nativos, en donde se hacen evidentes algunos de sus componentes, los cuales se explican a continuación

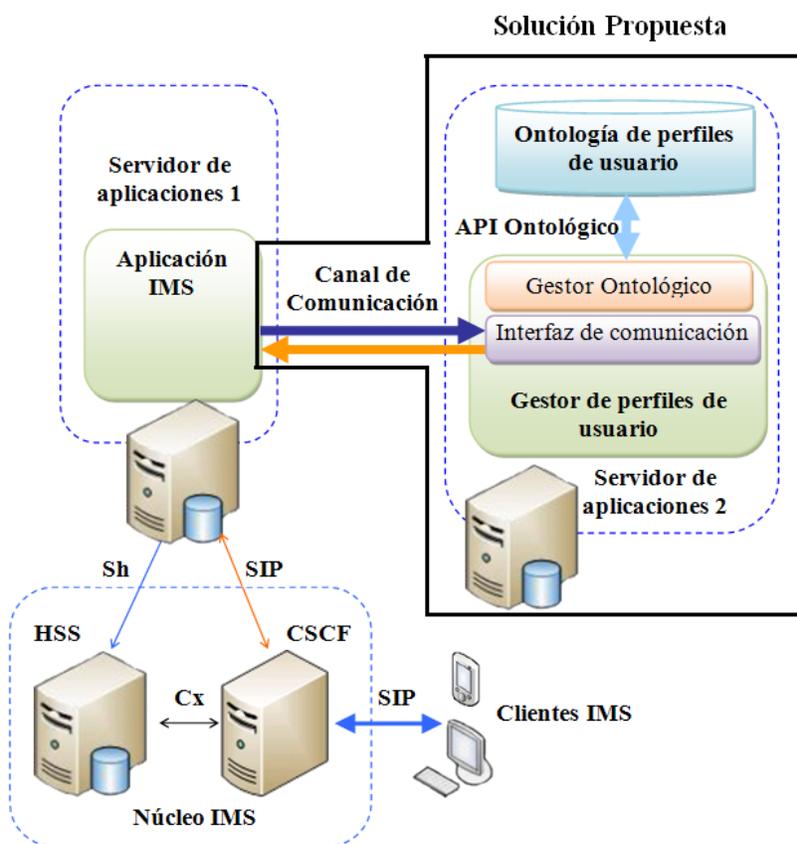


Figura 10. Arquitectura propuesta para la gestión de perfiles de usuario

Como se explicó en el capítulo I, en la arquitectura de IMS se puede hacer uso de tres tipos Servidores de Aplicación los cuales son: SIP AS, OSA-SCS, IM-SSF. Dado que la mayoría de servicios nuevos que serán ofrecidos en IMS se basan en SIP, la arquitectura expuesta en este documento, se enfoca en los Servicios IMS Nativos, prestados por los SIP AS.

3.4.1 Servidor de aplicaciones (AS)

Es la plataforma que sirve de contenedor de las diversas aplicaciones relacionadas con la sesión, y dentro del cual se aloja el gestor de perfiles de usuario así como el sistema que gestiona la ontología. Los servidores de aplicación se comunican con el HSS dentro del núcleo de IMS por medio de la interfaz sh, la cual provee la funcionalidad de guardar y recuperar información desde el HSS, responsable de la información suministrada a cada AS.

Los servidores de aplicaciones, también se comunican con el S-CSCF por medio de la interfaz ISC (*IMS Service Control*, Control de servicios IMS) basada en el protocolo SIP, la cual transporta información relacionada con el registro de los usuarios y las capacidades y características del equipo del usuario.

3.4.2 Ontología de perfiles de usuario

Es un conjunto de términos asociados al dominio del perfil de usuario estructurados de acuerdo a una jerarquía de clases y sus propiedades, con el propósito de servir de fuente de conocimiento para distintas aplicaciones en el entorno de IMS, por medio de su interacción con el gestor de perfiles de usuario.

Esta ontología se comunica con el gestor, por medio de los APIs provistos por las herramientas de edición de ontologías cuya selección se especifica en la sección **¡Error! No se encuentra el origen e la referencia.** y una descripción más detallada se aborda en el anexo A.4. Es así como brinda la posibilidad de crear, editar y eliminar las instancias de una clase, a la vez, que se pueden modificar las propiedades de las mismas.

3.4.3 Gestor de perfiles de usuario

Es una aplicación alojada en el AS, encargada de proveer las interfaces necesarias para crear, modificar, eliminar y consultar instancias de la ontología propuesta para modelar el perfil de usuario, estableciéndose como un mediador entre las diferentes aplicaciones que quieran acceder a los datos de personalización de la ontología.

En la arquitectura propuesta, el gestor de perfiles de usuario actúa de forma similar a un *Middleware*, el cual se define como una capa software que se encuentra ubicada sobre el recurso que se desea manejar (la ontología de perfiles de usuario), y por debajo de la aplicación que lo manejará (Aplicaciones IMS), para lograr esto un *middleware* provee herramientas como librerías y APIs, que reducen significativamente las labores de desarrollo y evitan de alguna forma que se cometan errores [35].

Dentro de la arquitectura de referencia este componente está conformado por el gestor ontológico y la interfaz de comunicación. El gestor ontológico, se encarga de gestionar la información de personalización que se encuentra almacenada en una ontología, con la ayuda de los APIs provistos por las herramientas de edición ontológica. La interfaz de comunicación es una capa software, la cual interactúa con otros servicios en el entorno de IMS, ofreciéndoles un canal de comunicación por medio del cual intercambiar información del perfil de usuario.

La definición de las interfaces de comunicación entre aplicaciones IMS no se encuentra estandarizada, por lo cual el gestor de perfiles de usuario se basa en las tecnologías que ya se emplean para el intercambio de información entre diferentes servicios, con el objetivo de ofrecer una interfaz cuya generalidad permita su uso, no solo en el entorno de IMS sino, en otros entornos.

3.4.4 Aplicación IMS

Es una aplicación alojada en el AS, que interactúa con diversos clientes de su servicio, por medio del núcleo de red IMS. Dicha interacción puede generar información de personalización, y es aquí donde el gestor de perfiles de usuario, juega un rol importante, dado que le permite a la aplicación IMS gestionar dicha información, por medio del uso de sus interfaces de intercambio de datos.

Esto convierte a la aplicación IMS en un consumidor de los recursos que el gestor de perfiles de usuario ofrece, permitiéndole la reutilización de la información de personalización que esta u otras aplicaciones hayan depositado en la ontología, con las ventajas que esto trae consigo, como la reducción en el tiempo de desarrollo de una estructura para almacenar los datos del usuario y la captura de dicha información.

Cualquier servicio estandarizado o no estandarizado estaría en la capacidad de acceder a los recursos de personalización que el gestor de perfiles de usuario ofrece.

Cabe destacar que entre los servicios que ya se encuentran estandarizados en IMS, podemos mencionar los siguientes:

- Telefonía multimedia IP, estandarizado por 3GPP / TISPAN VoIP
- *Push to talk Over Celular (PoC)*, estandarizado por el OMA PoC (*Open Mobile Alliance PoC*).
- Mensajería IMS, estandarizada por el OMA SIMPLE IM.
- Presencia, Estandarizada por el *OMA Presence*.
- *Combinational Services (CSI, CS call with any IP)*, estandarizado por el 3GPP.

3.4.5 Núcleo de IMS

Dentro de la arquitectura de referencia se observa que básicamente está conformada por el CSCF y el HSS componentes que describen a continuación:

3.4.5.1 CSCF

Como se especificó en el capítulo I, son el conjunto de entidades que hacen parte del núcleo de IMS encargadas del control de sesión, los cuales son de tres tipos I-CSCF, P-CSCF y el S-CSCF.

El S-CSCF se encarga de seleccionar el SIP AS adecuado para la prestación del servicio que el usuario esté demandando. Para cumplir con esta tarea el S-CSCF hace uso de la información contenida en el Perfil de Usuario que se encuentra guardado en el HSS y que es descargado al S-CSCF en el momento del registro de dicho usuario.

La interacción con el HSS se realiza por medio de la interfaz Cx, la cual se utiliza para realizar diferentes funciones asociadas a los usuarios. El protocolo que maneja dicha interfaz es "DIAMETER". Por otro lado la comunicación con los servidores de aplicación la realiza por medio del protocolo SIP.

3.4.5.2 HSS

Hace parte del núcleo de IMS, y como se vio en capítulo I, es la principal base de datos que contiene la información de usuario y de suscripción, encargada de apoyar a las entidades de red en el establecimiento tanto de llamadas como de sesiones.

El HSS interactúa por medio de interfaz Cx con el CSCF y con la Sh con los servidores de Aplicación, Mediante la interfaz Sh, se pueden obtener algunas de las propiedades que hacen parte de la ontología de perfiles de usuario, y permiten su adecuada gestión y futura consulta..

3.4.6 Clientes IMS

Son un conjunto de dispositivos con la capacidad de interactuar con el núcleo de IMS, y consumir los servicios ofrecidos por el servidor de aplicaciones. Dicha interacción se realiza utilizando el protocolo SIP, el cual les permite el inicio, modificación y finalización de sesiones en las diferentes aplicaciones de IMS, además, son una de las principales fuentes de información sobre el perfil de usuario.

3.5 DIAGRAMA DE INTERACCIÓN DE COMPONENTES DE LA ARQUITECTURA DE REFERENCIA

En la Figura 11 se muestra el diagrama de interacción de los componentes que hacen parte de la arquitectura de referencia, que se describe a continuación.

El cliente IMS tras haber iniciado sesión y establecido una ruta de comunicación, por medio del núcleo de IMS con la aplicación IMS, envía a este último un mensaje SIP, en cuyo contenido estructura una solicitud asociada con la información sobre el perfil de usuario. Dicha solicitud le permite al cliente realizar acciones para la gestión de información de personalización, basadas en la interacción del usuario con la aplicación cliente o simplemente por una consulta directa.

El núcleo de IMS se encarga de direccionar dicho mensaje, remitiéndoselo a la aplicación IMS, la cual al recibirlo, lo interpreta y procesa para generar una nueva solicitud que se envía al gestor. En este caso, la aplicación IMS se vale del servicio que el gestor ofrece para acceder a los recursos que se encuentran en la ontología de perfiles de usuario.

La aplicación IMS realiza la solicitud al gestor, que al procesarla, determina que acción realizar sobre la ontología. Aquí el gestor de perfiles de usuario se vale de los APIs provistos por las herramientas de manipulación de ontologías para realizar la correspondiente acción solicitada.

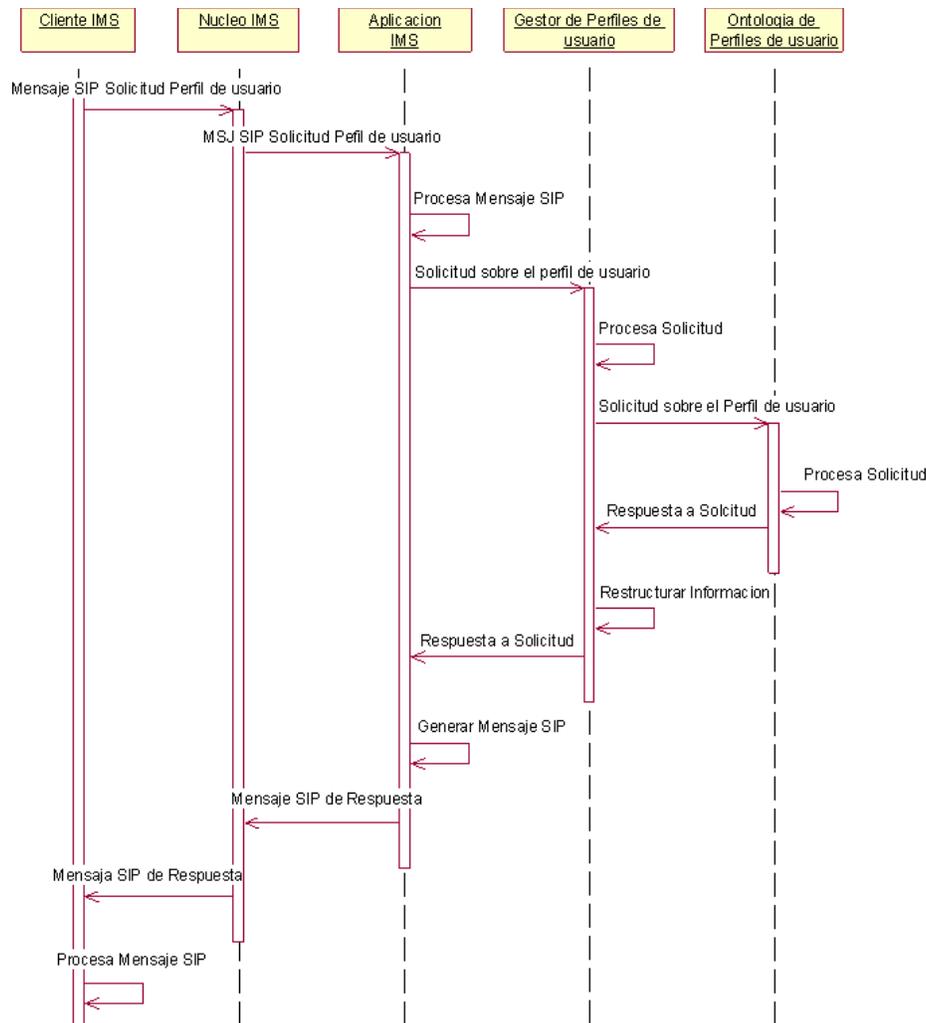


Figura 11. Diagrama de interacción de componentes

Enviada la solicitud a la ontología de perfiles de usuario, esta la procesa para generar una respuesta, basada en su modelo de clases, instancias y propiedades. Esta respuesta es entregada al gestor de perfiles de usuario, el cual la interpreta y reestructura de tal manera que pueda ser enviada por el canal de comunicación establecido con la aplicación IMS.

La aplicación IMS recibe respuesta a su solicitud, la cual se adiciona a un mensaje SIP, que por medio del núcleo de IMS es enviado al Cliente IMS como respuesta a la solicitud realizada en un principio.

El cliente de IMS procesa el mensaje SIP recibido y utiliza la información de su contenido para llevar a cabo acciones de personalización o simplemente realizar la notificación específica de cada caso, por ejemplo el éxito en la actualización de los datos del usuario.

CAPÍTULO 4 IMPLEMENTACIÓN DEL SISTEMA DE GESTIÓN DE PERFILES DE USUARIO EN EL ENTORNO DE IMS

En este capítulo se describe el entorno de desarrollo ontológico utilizado y las fuentes usadas en la implementación de ontología de perfiles de usuario, a la vez que se hace un breve recuento de las implementaciones del núcleo de IMS y se explican las razones para la selección de una en particular. También en este capítulo se explican algunas de las arquitecturas para el intercambio de la información entre servicios, para luego describir la implementación de referencia del sistema y los métodos para acceder a los recursos que el gestor de perfiles de usuario ofrece. Para finalizar el capítulo se realizan un conjunto de pruebas que permiten la evaluación del sistema de gestión y que dan fe de su buen funcionamiento.

4.1 ENTORNO DE DESARROLLO ONTOLÓGICO

La representación mediante ontologías como se explica en detalle en el anexo A.1, es posible por medio de lenguajes como RDFS, DAML+OIL (DARPA's *Agent Markup Language + Ontology Inference Layer*), OWL, etc. De este conjunto de lenguajes, OWL es la recomendación de la W3C (*World Wide Web Consortium*) para especificar ontologías y compartir el conocimiento, por lo cual, varias herramientas actualmente lo soportan, convirtiéndolo en un estándar a nivel industrial.

En cuanto a las herramientas más usadas para la edición de ontologías, en el anexo A.4 se presenta un compendio de éstas, donde se resumen sus características más relevantes. De todos los editores de ontologías expuestos en ese apartado, para este proyecto se seleccionó a Protégé, dada la amplia gama de trabajos desarrollados con este editor y de que brinda un entorno gráfico que facilita en gran medida el diseño de ontologías. Además, Protégé cuenta con un API que permite el manejo de una ontología desde cualquier aplicación Java y realizar operaciones como consulta e inferencia.

Una de las funcionalidades más importante de Protégé para este proyecto, fue la generación automática de código Java, utilizado para la manipulación de la ontología de perfiles de usuario, sin embargo, dicha funcionalidad sólo está disponible para ontologías definidas en el lenguaje OWL. Por esta razón y por lo expuesto anteriormente, se determinó que para el desarrollo de la ontología de perfiles de usuario, el lenguaje OWL era la mejor alternativa.

4.2 ONTOLOGÍA DE PERFILES DE USUARIO

En el desarrollo de una ontología es relevante conocer los trabajos anteriores y verificar si es posible refinarlos y extender sus recursos para nuestro dominio y tarea particular. Muchas ontologías ya están disponibles en forma electrónica y pueden ser importadas dentro un entorno de desarrollo. Aquí el formalismo en el cual una ontología esta expresado a menudo no interesa, puesto que muchos sistemas de representación de conocimiento pueden importar y exportar ontologías en diferentes formatos. Aun si el sistema de representación de conocimiento no funciona directamente con un formalismo particular, se puede asumir la tarea de traducir una ontología a otro lenguaje.

Dado el análisis realizado, durante el transcurso del proyecto se estableció que una de las mejores soluciones para el desarrollo de la ontología para la personalización de servicios, era tomar como punto de partida una ya existente y a ésta, realizarle las modificaciones necesarias para adaptarla apropiadamente al entorno de IMS.

Existen diversos trabajos que aplican el uso de ontologías para la creación de perfiles de usuario, sin embargo, ninguna está enfocada específicamente a la arquitectura de IMS y dado que el propósito del presente trabajo es la creación de una ontología que pueda ser reutilizada por diferentes aplicaciones, se requiere que el perfil de usuario que se utilice sea lo más genérico posible, lo cual coincide con uno de los objetivos planteados en el proyecto [27] y cuyo desarrollo previo garantiza un alto grado de satisfacción de las características con las que cuenta un usuario, siendo este el punto de partida, desde donde se adicionaron los conceptos que se determinaron como esenciales dentro de la arquitectura de IMS y de gran utilidad para las aplicaciones que pueden ofrecerse dentro su entorno, las cuales se mencionan en la sección 4.2.2 .

4.2.1 Características de la ontología de perfil de usuario base.

La ontología utilizada como base para el establecimiento del perfil de usuario, está directamente relacionada con la información que es principalmente estática y perdurable. En ella las características dinámicas como la posición actual del usuario, mientras se mueve aun no han sido incluidas, sin embargo, el aspecto temporal de algunas de las clases de la ontología se han tenido en cuenta. Es así como la ontología permite la existencia de múltiples instancias de clases que representan características que pueden cambiar con el pasar del tiempo, como por ejemplo la ciudad donde se vive, aquí a estas clases particulares se les incluye un intervalo de tiempo que representa el período de validez de la instancia, por ejemplo, el usuario vivió en Popayán desde 12/3/2005 - hasta 18/8/2009 [27]. En la Tabla 1 se presenta un resumen de las clases de nivel superior de la ontología, junto con su descripción.

La clase central en la ontología es "*Person*", la cual contiene todas las características de perfil de usuario. Éstas pueden ser de tipo simple, como el nombre del usuario ("*name*") o fecha de nacimiento ("*date of birth*"), o pueden ser instancias de otras clases de la ontología, como las características físicas ("*physical characteristics*"), sus contactos o conocidos ("*contacts*"), etc. [27].

El resto de las clases se usan para describir las características complejas del usuario, como las condiciones de vida ("*Living Conditions*"), sus contactos ("*Contact*"), Educación ("*Education*"), nivel de experiencia ("*Expertise*"), Actividades ("*Activity*") y Profesión ("*Profesión*") incluyendo un juego de *slots* que describen aspectos de la vida del usuario, así como períodos de tiempo que representan la duración de ese aspecto en particular. Por ejemplo, un usuario puede haber tenido un "*Contact*" de tipo "*Friend*" desde 2005 al 2009. El *slot* "*person*" de la clase "*Contact*" es de tipo instancia de la clase "*Person*". De esta manera, las relaciones entre diferentes usuarios también pueden ser modeladas [27].

Nombre de la clase	Descripción de la clase
<i>Person</i>	Contiene la Información del Usuario básica como su nombre, fecha de nacimiento, el correo electrónico.
<i>Characteristic</i>	Las características generales del usuario, como el color de ojos, altura, peso, etc.
<i>Ability</i>	Habilidades y discapacidades mentales y físicas del usuario.
<i>Living Conditions</i>	Información relevante acerca del lugar de residencia y tipo de vivienda.
<i>Contact</i>	Otra persona, con la cual la persona esté relacionada, incluyendo parientes, amigos y colaboradores
<i>Preference</i>	Preferencias del usuarios, por ejemplo “le gustan los gatos”, “le gusta el color azul”, o “le desagrada la música clásica”
<i>Interest</i>	Intereses en pasatiempos o trabajos relacionados. Por ejemplo “interés en los deportes” o “interés en la cocina”
<i>Activity</i>	Actividades, o pasatiempo o trabajo relacionados. Por ejemplo “coleccionar estampillas” o “investigar artículos históricos”
<i>Education Level</i>	Nivel educativo, incluyendo por ejemplo los diplomas universitarios y lenguajes.
<i>Profesion</i>	La ocupación del usuario
<i>Expertise</i>	Incluye toda clase de experiencias, como la experiencia en el uso de computadoras.
<i>Thing</i>	Cosas del hogar o no que el usuario posee o relacionadas a estas, como un automóvil, una casa, un libro o una mascota.

Tabla 2 Clases de alto nivel de la ontología de perfil de usuario

“*Interest*” (intereses), “*Preference*” (preferencias), “*Ability*” (habilidades), “*Characteristic*” (características) y “*Thing*” (cosas) contienen sólo tres tipos de slots: “*type*” (tipo), “*name*” (nombre) y “*score*” (puntuación) o “*valore*” (valor) en el caso de “*Thing*”. “*Thing*” tiene dos subclases las clases “*Living Thing*” y “*Non Living Thing*” En el caso de los intereses, aparte del tipo del slot “*type*”, el cual es una cadena de texto, un slot llamado “*interest type*” (tipo de interés) del tipo “*Interest*” se ha agregado para permitir la creación de jerarquías de interés, en la Tabla 3 se muestra un ejemplo [27].

Jerarquía de intereses	Instancia de "Interest" (Type, Name)
Negocios	(<Raiz>, Negocios)
Inversiones	(Negocios, Inversiones)
Reservas y Certificados de inversión	(Inversiones, Reservas y Certificados de inversión)
Deportes	(<Root>, Deportes)
Basquetbol	(Deportes, Basquetbol)
Profesional	(Basquetbol, Profesional)
Universidades	(Profesional, Universidades)

Tabla 3 Ejemplo de cómo una jerarquía de intereses puede ser modelada

La experiencia del usuario se ha definido mediante la combinación de tres dimensiones: La amplitud, que representa la magnitud o variedad de herramientas diferentes, habilidades y conocimiento que el usuario puede poseer; la profundidad, que es la integridad del conocimiento actual del usuario de un dominio particular; y la afinidad, la cual se refiere a la innovación y la creatividad. La amplitud y la profundidad se desarrollan con el tiempo a través de una combinación de estudio y el uso práctico, sin embargo la afinidad está más relacionada a la personalidad del usuario. Estas propiedades son incluidas en la ontología del usuario como atributos [27].

La noción de "átomos de experiencia" se introduce por medio de la definición de unidades elementales de experiencia como resultado de la actividad en un dominio particular. Los átomos de experiencia pueden expresarse en la ontología del usuario como instancias individuales de la clase "Expertise", a las cuales se les asigna un puntaje o nivel expresado a través del slot "score" [27].

La clase "Expertise", tiene como objetivo el coleccionar las características de usuario que pueden servir como indicaciones o factores durante la valoración del nivel de experiencia del usuario. Esta clase se ha creado como un contenedor para las medidas de experiencia y la valoración de la experiencia con el propósito de adecuarse a las necesidades particulares de aplicaciones individuales que hacen uso del perfil de usuario. Sin embargo la definición de los niveles de experiencia y su especialización no se han especificado en la ontología dado la complejidad del sistema necesario para dicha evaluación [27].

4.2.2 Características adicionales a las ontología base.

Para establecer una conexión entre el perfil de usuario del entorno IMS y la ontología base, se adicionó la información de las identidades tanto públicas como la privada del usuario. La identidad privada fue agregada a la clase "Person" como un dato simple y las identidades públicas como un conjunto de datos. Cabe resaltar que estas identidades han sido utilizadas por el gestor de perfiles de usuario como las claves principales para identificar al usuario, y no como normalmente se

realizaría dentro de la ontología por medio del *localname*, el cual es asignado a cada instancia de la clase persona. Es así como el uso de dicha información se ve reflejado en el servicio desarrollado, puesto que las URI's se valen de dichas identidades para acceder a la información del perfil de usuario, de acuerdo a como se especifica en la sección **¡Error! No se encuentra el origen de la referencia.**

4.3 IMPLEMENTACIONES DEL NÚCLEO DE IMS

4.3.1 Open IMS Core

Existen aspectos fundamentales que hacen del Open IMS Core una implementación de referencia, el primero de ellos es el hecho de estar disponible sin costo alguno en virtud de una licencia *open source* (código abierto), esto atrae tanto a grupos académicos de I+D como a compañías comerciales que complementan sus demostraciones de escenarios NGN. Este aspecto además de ayudar con la distribución del software permite establecer un proceso de retroalimentación, algo sumamente importante ya que la información suministrada por estos grupos es clave en el proceso de mejora continua en áreas como el desempeño del software y la estandarización, para este fin se han dispuesto de páginas web y listas de correo que son monitoreadas por un grupo de desarrolladores encargados de dar soporte a los cientos de visitantes que diariamente utilizan estos canales.

Un segundo aspecto importante es, que en base a los requerimientos que provienen de los diferentes socios industriales del Open IMS *Playground* y desde los usuarios del software, El Open IMS Core está respondiendo a las necesidades actuales que establecen las especificaciones para tráfico IMS (y especialmente con respecto a la autenticación) en los dominios de las redes fijas móviles y cableadas. Actualmente las organizaciones de estandarización consideran al proyecto como una referencia y como la mejor herramienta para la experimentación tanto de las funcionalidades nuevas como las convergentes.

El desarrollo del Open IMS Core [36] [37], fue financiada con fondos públicos, el apoyo de proyectos de I + D, así como otros proyectos de la industria. A continuación se presentan escenarios en los que el Open IMS Core ha sido utilizado:

- **ETSI TISPAN IMS evaluación comparativa del estándar.** Al interior del grupo de trabajo 6 del ETSI TISPAN y desde el comienzo de la estandarización [38], el Open IMS Core sirvió de referencia para la evaluación comparativa de los componentes principales de IMS. El estándar define por primera vez la metodología, las métricas, reportes, y el proceso para encontrar y certificar el rendimiento de las redes NGN.
- **Usado en escenarios de interoperabilidad NGN.** El software es ampliamente usado en escenarios de interoperabilidad por una gran cantidad de vendedores y socios de I+D. Cualquier socio puede utilizar el software en su centro de pruebas antes de pasar a un escenario de interoperabilidad real. A partir de las pruebas de interoperabilidad IMS originadas en el sector comercial el proyecto ha recibido algunos comentarios bastante positivos, estos provienen de entidades de investigación muy respetadas, tal como el

Laboratorio de Interoperabilidad de la Universidad de Nueva Hampshire [39], el cual es el anfitrión de las pruebas del *IMS Forum* y el *MultiService Forum's* entre otros. Para este caso el *Open IMS Core* es parte de las configuraciones que ellos tienen por defecto para las redes IMS. Otra organización importante que también confió en el *Open IMS Core* es el ETSI TISPAN WG6, estableciendo al software como una solución que puede ser usada en sus IMS Plug Tests.

- **Utilizado como núcleo para ambientes de desarrollo de IMS y otros proyectos de código abierto.** Aunque el *Open IMS Core* durante mucho tiempo ha creado servicios y ha desarrollado componentes tales como el núcleo central para el *Open IMS Playground*, existen ambientes de desarrollo que no se relacionan puramente con IMS pero que pueden beneficiarse con el software en aspectos como: el enfoque del proyecto en áreas como la investigación de nuevas plataformas que trabajan bajo los principios de una arquitectura orientada al servicio, el uso del software integrado estrechamente con la representación de escenarios específicos IPTV junto con aplicaciones para la investigación y el soporte de futuras distribuciones multimedia sobre redes IMS.

Por otro lado el software recibe comentarios y respuestas muy positivas de proyectos de I + D, universidades y vendedores de componentes IMS. Además el *Open IMS Core* es usado en algunos de los proyectos FP6 de la Unión Europea así como también en muchos de los departamentos de I + D de operadores en sus laboratorios NGN (ej. *Portugal Telecom Inovacao*, *Orange Labs* y *Slovak Telekom*)[40]. Adicionalmente el proyecto recibe comentarios positivos de centros independientes de I+D que hacen del *Open IMS Core* una parte fundamental de sus pruebas rutinarias en laboratorio. En el campo académico gran cantidad de expertos están trabajando con el software como parte de su trabajo doctoral, algunos otros están construyendo sus tesis de maestría alrededor del mismo, también es utilizado como parte de conferencias o para ilustrar conceptos de IMS y para enseñar a estudiantes acerca de las tecnologías NGN.

Como efecto indirecto, el *Open IMS Core* ha dado inicio al desarrollo de proyectos *open source* en el dominio del cliente IMS [41][42]. El *Open IMS Core* hoy en día está habilitando a diferentes entes dentro de la comunidad de investigación en NGN, ya sea para crear prototipos, hacer mediciones o generar desarrollos alrededor del mismo (servicios o componentes). Aunque todavía la mitad de los visitantes del sitio web del proyecto son de Europa, el interés por este software especializado es a nivel mundial registrando visitantes de 120 países.

4.3.2 Ericsson Service Development Studio – SDS

Ericsson Service Development Studio (SDS) permite a los operadores, proveedores de software independientes y desarrolladores diseñar sus propias aplicaciones IMS cliente-servidor. SDS ha sido utilizado para realizar múltiples aplicaciones IMS con la ejecución de ensayos y demostraciones al cliente. SDS se ejecuta en un entorno Windows y soporta la creación y pruebas de aplicaciones IMS de extremo a extremo tanto del lado del cliente como del servidor, usando emuladores para: la red IMS, habilitación del servicio, dispositivos de usuario y servidores SIP. SDS brinda una gran cantidad de plantillas y asistentes para ayudar a los desarrolladores a acortar los tiempos de ejecución del proyecto, a su vez que pueden usar APIs de alto nivel para controlar y acceder a

capacidades avanzadas como Presencia y Gestión de Grupos, Voz sobre IP (VoIP) , *Push-To-Talk* (PTT), mensajería IMS (IMS-M) , IPTV y en próximos lanzamientos soportará servicios adicionales tales como telefonía IP Multimedia y servidores Media.

4.3.2.1 Capacidades del SDS

Desarrollo y depuración de aplicaciones IMS cliente-servidor

- IDE basado en Eclipse, EclipseME y WTP (*Web Tools Platform*, Plataforma de Herramientas Web), siendo estos entornos muy poderosos y versátiles.

Pruebas de extremo a extremo con la emulación de una red IMS

- Presencia y Grupos, emulador PGM
- *Push-to-Talk* , emulador PTT-AS
- Mensajería IMS, emulador IMS-M
- Televisión sobre el Protocolo de Internet, emulador IPTV
- Servicios combinados, llamadas CS +PS; "WeShare" (3GPP CSI)
- Voz sobre IP; P-2-P

Soporte para acceso a dispositivos fijos y móviles

- Acceso móvil y fijo de banda ancha y WLAN
- PC(Windows)
- Teléfonos móviles con sistema operativo Symbian (UIQ y S60 UI)
- Teléfonos con el estándar Java ME para servicios IMS, basados en el JSR 281

Soporte para servidores Java EE/SIP

- Servidor de aplicaciones SailFin establecido por defecto
- El SDS ha sido probado con servidores de aplicación comerciales tales como Oracle OCCAS 4.0 y SUN SGCS 1.5

El SDS provee un ambiente de simulación donde los nodos (CSCF y HSS) dentro del núcleo de una red IMS son simulados. Esto permite al SDS actuar como una red IMS virtual para el desarrollador. Este ambiente de simulación soporta un subconjunto de funcionalidades de la implementación del núcleo IMS de Ericsson.

4.3.3 Selección del entorno para la emulación del Núcleo de IMS

La herramienta seleccionada para la emulación del núcleo de IMS es el SDS de Ericsson. A continuación se enumeran las razones que permitieron esta elección:

- El SDS facilita la implantación de servicios, ya que cuenta con un gran número de herramientas de soporte, para la creación y pruebas de aplicaciones IMS de extremo a extremo, tanto del lado del cliente como del servidor.

- La integración del SDS con el entorno de desarrollo Eclipse permite encontrar todo en una misma herramienta.
- Hasta el momento ningún grupo de investigación ha realizado pruebas ni desarrollos al interior de la universidad apoyados sobre el entorno SDS de Ericsson, de esta forma el proyecto abre las puertas a una herramienta con un gran soporte y respaldo.

4.4 ARQUITECTURA PARA EL INTERCAMBIO DE INFORMACIÓN ENTRE SERVICIOS

Dado que uno de los requerimientos establecidos para la solución, es la definición de un servicio que intercambia información del perfil de usuario con otros servicios en IMS, es necesario adoptar algún patrón arquitectónico orientado a la comunicación entre estos, convirtiendo a los servicios Web, en un buen punto de referencia. Un servicio Web es un conjunto APIs Web que pueden ser accedidas dentro de una red y son ejecutados en el sistema que lo aloja. Entre sus arquitecturas se encuentran las siguientes:

4.4.1 RPC

En RPC (*Remote Procedure Calls*, llamadas a Procedimientos Remotos) los Servicios Web poseen una interfaz de llamada a procedimientos y funciones distribuidas, comúnmente, representada por medio de un documento WSDL (*Web Services Description Language*, Lenguaje de Descripción de Servicios Web). Dado que fue una de las primeras herramientas para Servicios Web, es un estilo muy extendido, pero a la vez, muy criticado por no ser débilmente acoplado, ya que por lo general su implementación se realiza por medio del mapeo de servicios directamente a funciones específicas de un lenguaje o de llamadas a métodos [43] [44].

4.4.2 SOA

Los Servicios Web también pueden ser implementados siguiendo los conceptos de la arquitectura SOA (*Service-Oriented Architecture*, Arquitectura Orientada a Servicios), donde la unidad básica de comunicación es el mensaje, más que la operación, por lo que comúnmente se los denomina servicios orientados a mensajes [45].

Los Servicios Web basados en SOA son ampliamente conocidos, y al contrario que los Servicios Web basados en RPC, este estilo es débilmente acoplado, dado que se centra en el “contrato” proporcionado por el documento WSDL, más que en los detalles asociados a su implementación [45].

Las aplicaciones basadas en SOA utilizan tecnologías totalmente estándar tales como XML y servicios Web para la mensajería. Estándares como SOAP (*Simple Object Access Protocol*, Protocolo Simple de Acceso a Objetos), WSDL y BPEL (*Business Process Execution Language*, Lenguaje de Ejecución de Procesos de Negocio con Servicios Web), estandarizan así la compartición de información, el modelo de integración de procesos y la cooperación entre aplicaciones a la hora de crear aplicaciones orientadas a servicio [45].

4.4.3 REST

Los Servicios Web basados en REST (*REpresentation State Transfer*, Transferencia de Estado Representacional) intentan emular al protocolo HTTP (*HyperText Transfer Protocol*, Protocolo de transferencia de hipertexto) o protocolos similares mediante la restricción de establecer la interfaz a un conjunto conocido de operaciones estándar (por ejemplo GET, PUT, POST...). Por tanto, este estilo se centra más en interactuar con recursos con estado, que con mensajes y operaciones [43].

Cabe destacar que REST no es un estándar, ya que es tan sólo un estilo de arquitectura, sin embargo se basa en estándares como HTTP, URL, Representación de recursos y los tipos MIME.

REST se rige por cuatro principios de diseño fundamentales [43]

- El primer principio es el utilizar los métodos HTTP de manera explícita, de forma consistente con la definición de dicho protocolo. Este principio de diseño básico establece una asociación uno-a-uno entre las operaciones de crear, leer, actualizar y borrar y los métodos HTTP. De acuerdo a esta asociación [43]:
 - Se usa POST para crear un recurso en el servidor.
 - Se usa GET para obtener un recurso.
 - Se usa PUT para cambiar el estado de un recurso o actualizarlo.
 - Se usa DELETE para eliminar un recurso.
- El segundo principio es el de no mantener un estado, lo cual hace que los servicios RESTful (nombre dado a los API que implementan REST), sean más simples de diseñar, escribir y distribuir a través de múltiples servidores. Un servicio sin estado no sólo funciona mejor, sino que además mueve la responsabilidad de mantener el estado al cliente de la aplicación[44].
- Como tercer principio se tiene que cada recurso es identificado con una dirección única a través de una sintaxis universal basada en identificadores uniforme de recursos (URI). Todos los recursos son compartidos uniformemente a través de una interfaz simple que consta de un conjunto de operaciones y tipos de datos bien definidos, basados típicamente en el protocolo HTTP [44].
- El cuarto principio es que la información entre el servidor y el cliente es transferida a través de XML, JSON (*JavaScript Object Notation*, Notación de Objetos de JavaScript) o ambos [43].

REST ha ganado amplia adopción en toda la web como una alternativa más simple a SOAP y a los servicios web basados en el WSDL. Ya varios grandes proveedores de Web 2.0 están migrando a esta tecnología, incluyendo a *Yahoo*, *Google* y *Facebook*, quienes marcaron como obsoletos a sus servicios SOAP y WSDL y pasaron a usar un modelo más fácil de usar, orientado a los recursos [43].

Para el desarrollo de este proyecto se optó por el uso de REST como patrón de intercambio de información, dadas sus ventajas en cuanto al menor consumo de ancho de banda necesario para el envío y recepción de mensajes, una mejor tiempo de respuesta a las solicitudes tal como lo demuestran en los resultados de la investigación realizada en [46]. Además la amplia acogida por parte de los desarrolladores de servicios además de que su interacción con sus clientes puede ser descritos mediante un escaso conjunto de operaciones, en donde las acciones de sus usuarios pueden ser determinadas como un CRUD (*Create Read Update and Delete*, Crear Leer Actualizar y Eliminar).

4.5 IMPLEMENTACIÓN DE REFERENCIA

4.5.1 Modelo de casos de uso

La arquitectura de IMS de la cual hacen parte componentes como las aplicaciones IMS, el cliente IMS y el núcleo de IMS, no necesitan la intervención del gestor perfiles de usuario para su normal funcionamiento, sin embargo, al ser adicionado éste brinda a los servicios una base compartida de información sobre el usuario, que puede ser muy útil en su personalización.

Para la implementación de referencia se ha tenido en cuenta que las aplicaciones IMS, son los principales consumidores de los recursos que ofrece el gestor de perfiles de usuario, por lo tanto, los modelos desarrollados se enfocan en su interacción, sin olvidar que quienes se ven beneficiados en realidad son los clientes de IMS, quienes recibirán un servicio más acorde a sus preferencias.

A partir de la arquitectura propuesta y con los requisitos expuestos en el apartado 3.1 se han especificado los casos de uso que se muestran en la **¡Error! No se encuentra el origen de la referencia.**, donde se distingue como uno de los principales actores a una aplicación IMS, la cual accede a las diferentes funcionalidades que provee la solución propuesta, a la vez que se muestra al operador de IMS quien es el encargado de las tareas de gestión del servicio.

A continuación se describen los casos de uso para cada uno de los actores mencionados.

Caso de uso Eliminar Instancia	
Iniciador	Aplicación IMS
Propósito	Eliminar de la ontología de perfiles de usuario una instancia de una Clase.
Resumen	Cuando la aplicación IMS establece que alguna instancia de una clase ya no es de utilidad en el perfil de usuario, envía una solicitud al gestor de perfiles de usuario, el cual la procesa y elimina la instancia solicitada, modificando la ontología, y envía una notificación de éxito en la operación.

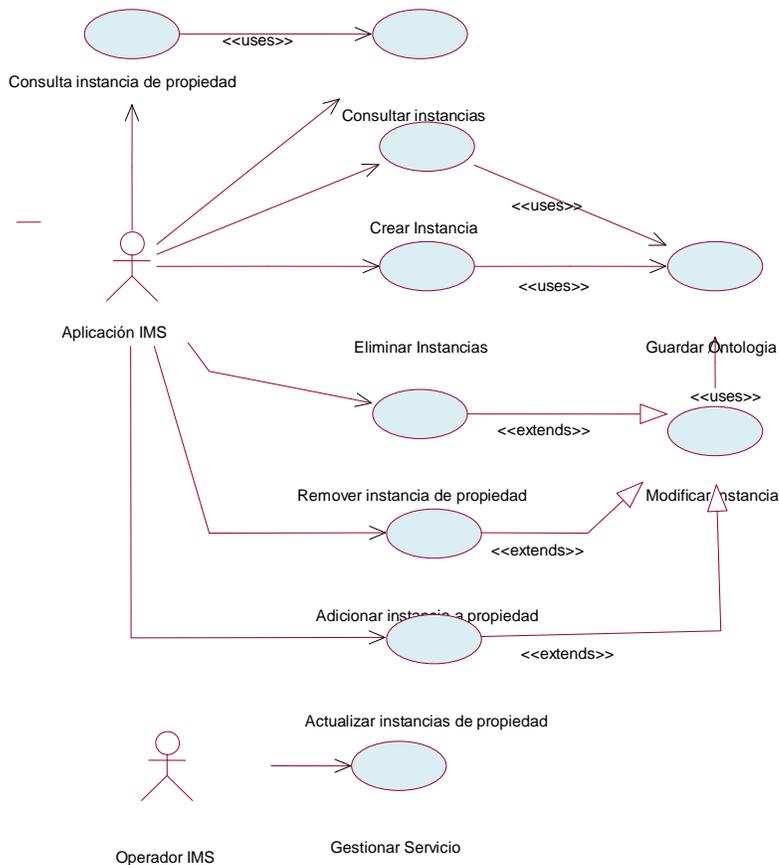


Figura 12. Diagrama de casos de uso del gestor de perfiles de usuario

Caso de uso Crear Instancia	
Iniciador	Aplicación IMS
Propósito	Crear instancias de cualquiera de las clases establecidas en la ontología de perfiles de usuario.
Resumen	La aplicación de IMS determina que es necesario crear una nueva instancia de una clase, como por ejemplo de una Persona (<i>Person</i>), y realiza una solicitud al gestor de perfiles de usuario utilizando la interfaz de comunicación que éste ofrece. Dicha solicitud se procesa para generar la correspondiente respuesta y almacenar los cambios en la ontología.

Caso de uso Adicionar Instancia a Propiedad	
Iniciador	Aplicación IMS
Propósito	Permitir a la aplicación IMS la adición de una instancia al valor de una propiedad
Resumen	<p>La aplicación de IMS luego de haber creado instancias, puede asociarlas como el valor de una propiedad de otra instancia. Por ejemplo, en el caso de una instancia de la clase "<i>Person</i>" (persona), esta cuenta con la propiedad llamada <i>prefereces</i> (preferencias), la cual es un conjunto de instancias de la clase <i>Preference</i> (preferencia), que será incrementado cuando se adicionan nuevas instancias a la propiedad.</p> <p>Para esto la aplicación IMS envía al gestor de perfiles de usuario la solicitud de adición de instancia junto con la información necesaria para realizar la operación. El gestor procesa la solicitud y realiza los cambios correspondientes en la ontología, que al ser exitosos, éste retorna una respuesta de confirmación a la aplicación IMS.</p>

Caso de uso Remover Instancia de Propiedad	
Iniciador	Aplicación IMS
Propósito	Permitir a la aplicación IMS remover una instancia de la colección de instancias asociadas a una propiedad.
Resumen	<p>La aplicación de IMS tras haber seleccionado la instancia que desea remover de las agrupadas en una propiedad, invoca a los métodos establecidos por el gestor de perfiles de usuario para removerla de dicho grupo.</p> <p>El gestor procesa la solicitud y realiza los cambios correspondientes en la ontología para remover la instancia de la propiedad, proceso que al ser exitoso, retorna una confirmación a la aplicación IMS</p>

Caso de uso Consultar Instancias de Propiedad	
Iniciador	Aplicación IMS
Propósito	Permitir a la aplicación IMS acceder al conjunto de instancias que están asociadas a una propiedad.
Resumen	<p>La aplicación IMS realiza una consulta del conjunto de instancias que hacen parte de la propiedad de un usuario, valiéndose de los métodos ofrecidos por el gestor.</p> <p>El gestor de perfiles de usuario procesa la solicitud y luego de consultar el conjunto de instancias de la propiedad, le envía a la aplicación IMS dichos datos.</p>

Caso de uso Consultar Instancia	
Iniciador	Aplicación IMS
Propósito	Consultar la información de una instancia de la ontología de perfiles de usuario.
Resumen	La aplicación IMS puede consultar toda la información de una instancia de cualquier clase que hace parte del perfil de usuario, recuperando ya sea información que ella misma ha generado o generada por otras aplicaciones. En dicho caso esto envía una solicitud, invocando los métodos ofrecidos por el gestor de perfiles de usuario para realizar la consulta de una instancia. El gestor procesa la solicitud y tras consultar en la ontología los datos requeridos de la instancia, retorna dicha información.

Caso de uso Gestionar Servicio	
Iniciador	Operador IMS
Propósito	Permitir a la entidad responsable del dominio de IMS establecer la configuración del servicio ofrecido por el gestor de perfiles de usuario.
Resumen	El Operador de IMS que es el responsable del servidor de aplicaciones donde se encuentra alojado el gestor de perfiles de usuario y la ontología, se encarga de la configuración inicial del entorno así como los de los procesos de iniciar y parar el servicio.

Caso de uso Actualizar Instancias de Propiedad	
Iniciador	Aplicación IMS
Propósito	Permitir a la aplicación IMS modificar en una sola acción todo el conjunto de instancias asociadas a una propiedad
Resumen	Dado que una propiedad puede tener como valores un conjunto de instancias, el gestor de perfiles de usuario permite modificarlas realizando una solicitud de actualización, con el nuevo conjunto de instancias que serán asociadas a la propiedad. Como respuesta a la solicitud, el gestor realiza las correspondientes asociaciones dentro de la ontología y envía una notificación de éxito a la aplicación de IMS

4.5.2 Diagrama de paquetes análisis

El diagrama de análisis del gestor de perfiles de usuario se muestra en la Figura 13 y se describe a continuación:

Comunicación: Este paquete contiene todas las clases que brindan una interfaz de comunicación para el intercambio de información, entre el gestor de perfiles de usuario y las aplicaciones IMS, tal como se muestra en la arquitectura de referencia.

Gestor Ontología: Este paquete contiene todas las clases con las cuales se implementa la lógica necesaria para la manipulación de la ontología de perfiles de usuario, así como aquellas que atienden a las solicitudes recibidas a través del paquete de comunicación y que generan la correspondiente respuesta.

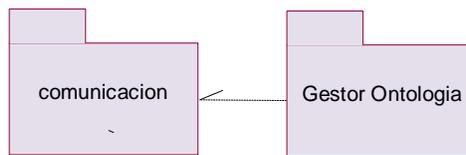
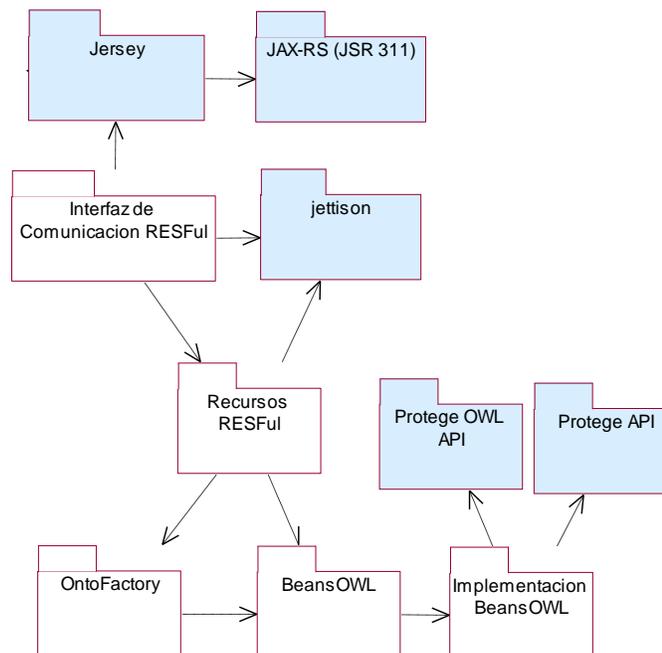


Figura 13. Diagrama de paquetes de análisis del gestor de perfiles de usuario

4.5.3 Diagrama de paquetes de diseño

El diagrama de paquetes de diseño se muestra en la Figura 14, cuyos componentes se describen a



continuación.

Figura 14. Diagrama de paquetes de diseño

Comunicación: Este está conformado por los siguientes paquetes:

- **JAX-RS (JSR 311):** (*The Java API for RESTful Web Services, API Java para Servicios Web RESTful*) es un API Java que contiene las clases para la implementación de servicios Web RESTful, este es necesario para utilizar la implementación de Jersey.
- **Jersey:** Es un API de código abierto que basándose en el JAX-RS (JSR 311) ofrece una implementación de referencia para la construcción de servicios Web RESTful. Además, puede ser extendido de acuerdo a las necesidades de cada aplicación.

- **Jettison:** Es una colección de APIs Java el cual permite leer y escribir con JSON, que es un formato ligero para el intercambio de información. Este permite de manera casi transparente para el desarrollador, la implementación de servicios web basados en JSON.
- **Interfaz de comunicación RESTful:** Es la encargada de dar acceso a las aplicaciones IMS con el Servicio Web RESTful prestado por el gestor de perfiles de usuario, cuya implementación se basa el API de Jersey y el API JAX-RS (JSR 311). Esta interfaz recibe las peticiones HTTP provenientes de las aplicaciones IMS, las cuales son respondidas en formato JSON, estructurado con el API de Jettison.

Gestor Ontológico: Este paquete está conformado por:

- **Recursos RESTful:** Contiene todas las clases que procesan las peticiones recibidas por medio de la interfaz de comunicación, quienes hacen uso del paquete OntoFactory y de las interfaces definidas en el paquete BeansOWL, para manipular la ontología.
- **OntoFactory:** Este paquete contiene la clase encargada de la construcción de objetos para el manejo de la ontología, basándose en las interfaces definidas en el paquete BeansOWL, lo cual hace innecesario el conocimiento del modelo ontológico que soporta, dado que se encarga de realizar todas las operaciones sobre objetos, aun desconociendo por completo de que tipo son.
- **BeansOWL:** Es el paquete que alberga todo el conjunto de interfaces definidas para la manipulación de cada una de las clases y propiedades con que cuenta la ontología de perfiles de usuario. Las interfaces aquí definidas obedecen las convenciones de nomenclatura de métodos, construcción, y comportamiento de la notación de JavaBean, lo cual facilita la manipulación de la ontología, haciendo innecesario el total conocimiento del modelo ontológico que se está soportando.
- **Implementación BeansOWL:** En este paquete se realiza la implementación específica para protégé de las interfaces definidas en el paquete BeansOWL. En sí este paquete es quien usa las librerías del API de Protégé para el manejo de la ontología de perfiles de usuario.
- **Protégé OWL API:** Es un librería de código abierto Java para los lenguajes ontológicos OWL y RDF(s). Este API provee las clases y los métodos para cargar y guardar archivos OWL, para consultar y manipular modelos de datos en OWL y realizar razonamiento. Adicionalmente, el API esta optimizado para la implementación de interfaces gráficas de usuario.
- **Protégé API:** Es una librería de código abierto Java en el cual se especifican los modelos básicos para la manipulación de una ontología que la convierte en indispensable a la hora de utilizar el API Protége OWL.

Los paquetes de Ontofactory, BeansOWL, y la implementación de BeansOWL fueron generados a partir de la ontología base, haciendo uso de las funcionalidades que ofrece Protégé para la generación automática de código para la manipulación de una ontología en el lenguaje OWL.

4.5.4 Modelo de Implantación

En la Figura 15 se representa el modelo de implantación para el sistema de gestión de perfiles de usuario, que se describe a continuación:

4.5.4.1 Equipo 1

En este equipo se ha establecido una la red IMS emulada por medio del SDS, compuesta por un servidor de aplicaciones Sailfin, el núcleo de IMS y los clientes IMS. El equipo cuenta con las siguientes características:

- Sistema Operativo Windows XP SP3 versión en Inglés
- *Java Development Kit* versión 1.6 (jdk 1.6)
- *Ericsson Service Development Studio 4.1*

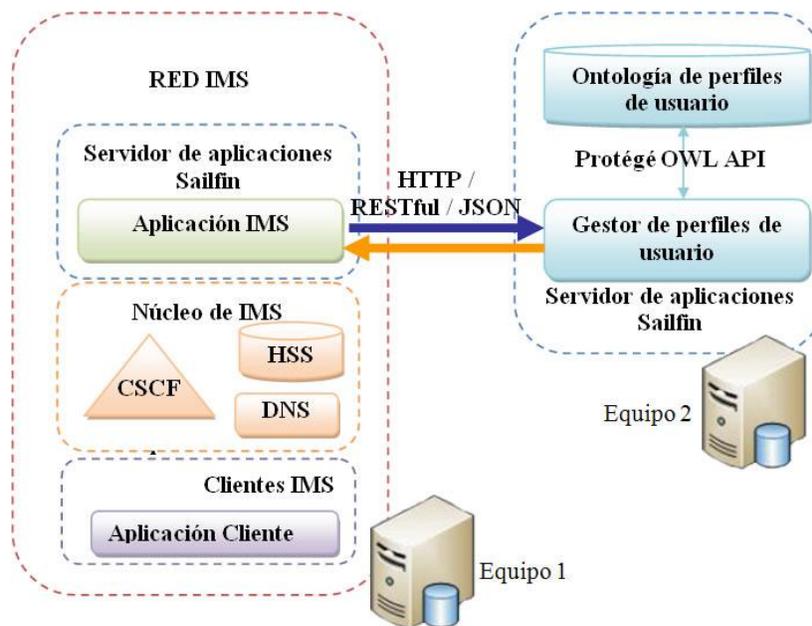


Figura 15 Modelo de implantación

Servidor de aplicaciones Sailfin: Es el servidor de aplicaciones que por defecto es instalado por el SDS. Este alberga a la aplicación IMS, que además de ser un SIPServlet, también utiliza la interfaz

de comunicación RESTful, por lo que para su correcto funcionamiento necesita de los siguientes paquetes:

- Jettison API. Para el manejo del Formato JSON.
- JSR 311. Especificación de Servicios RESTful.
- Jersey API. Implementación de Jersey para un cliente de un servicio RESTful
- JSR 289: SIP-Servlet v1.1

Núcleo de IMS: Compuesto principalmente por el CSCF, HSS y un servidor de nombres de dominio DNS, cuya simulación se realiza por medio del SDS en el Equipo 1.

Clientes IMS: El SDS permite la emulación de múltiples clientes para aplicaciones IMS. La simulación del cliente IMS se lleva a cabo en el mismo equipo donde se encuentra todo el entorno de red de IMS. Más detalles de la implementación del cliente puede verse en el Anexo E.

4.5.4.2 Equipo 2

En el segundo equipo se tiene también un servidor de aplicaciones Sailfin, el cual fue instalado al momento de instalarse en el equipo el SDS, y que alberga tanto al gestor de perfiles de usuario y así como a la ontología. Las características del equipo son las siguientes:

- Sistema Operativo Windows XP SP3 versión en Inglés
- *Java Development Kit* versión 1.6 (jdk 1.6)
- *Ericsson Service Development Studio 4.1*

Ontología de perfiles de usuario: Esta se basa el trabajo expuesto en [27], el cual fue adaptado para que fuese coherente con la información manejada en el perfil de usuario dentro de entorno de IMS, a la vez, que se modificó el lenguaje en el que estaba desarrollado a OWL. Todas estas adaptaciones, se llevaron a cabo utilizando del editor de ontologías Protégé.

Gestor de perfiles de usuario: Es una aplicación alojada en un servidor, la cual ofrece un servicio web RESTful que permite el intercambio de información de perfiles de usuario con aplicaciones IMS, por lo cual hace uso de las siguientes librerías:

- Protégé OWL API. Api para la manipulación de ontologías en lenguaje OWL
- Protégé API y sus dependencias. Núcleo básico de protégé requerido por el Protégé OWL API.
- Jersey API, Implementación de Jersey para el despliegue de un Servicio RESTful.

- JSR 311, Especificación de Servicios RESTful
- Jettison API, Para el manejo de formato JSON

4.6 METODOS DE ACCESO A LOS RECURSOS DE LA ONTOLOGÍA

El servicio prestado por el gestor de perfiles de usuario, se implementó bajo los principios de RESTful, por lo tanto, se vale de los métodos PUT, GET, POST y DELETE de HTTP para realizar sus principales operaciones.

Estos métodos suelen ser comparados con las operaciones asociadas a la tecnología de base de datos, denominada CRUD (*CREATE, READ, UPDATE, DELETE*). Otras analogías también pueden ser hechas como con el concepto de copiar-y-pegar (*Copy&Paste*). Todas las analogías se representan en la Tabla 4.

Acción	HTTP	SQL	Copy&Paste	Unix Shell
Create	PUT	Insert	Pegar	>
Read	GET	Select	Copiar	<
Update	POST	Update	Pegar después	>>
Delete	DELETE	Delete	Cortar	Del/rm

Tabla 4 Analogías entre acciones de RESTful

Las acciones (verbos) CRUD se diseñaron para operar con datos atómicos dentro del contexto de una transacción con la base de datos. REST se diseña alrededor de transferencias atómicas de un estado más complejo, tal que puede ser visto como la transferencia de un documento estructurado de una aplicación a otra [44].

Cuando se utiliza REST, HTTP no tiene estado, por lo cual cada mensaje contiene toda la información necesaria para comprender la petición cuando se combina con el estado del recurso. Como resultado, ni el cliente ni el servidor necesitan mantener ningún estado en la comunicación, sin embargo, cualquier estado mantenido por el servidor debe ser modelado como un recurso [44].

Cada clase de la ontología junto a algunas de las propiedades que manejan múltiples valores, han sido identificadas mediante una URI, convirtiéndolas en objetos lógicos a los que se les envía peticiones (GET, PUT, POST, DELETE), sin embargo, dichos recursos no pueden ser directamente accedidos o modificados, sino que se trabaja con representaciones de ellos, que para el caso del gestor, son mensajes en formato JSON o un conjunto de parámetros con la información del recurso. Cuando se utiliza el método POST para actualizar valores de una instancia, es como si se enviara una representación de lo que se quiere que el estado del recurso fuese. Internamente dicho estado puede almacenarse en una base de datos relacional, un fichero de texto o como en este proyecto en una ontología.

Cada instancia dentro de la ontología tiene asignada un URI la cual está conformada por un prefijo y un *localname* (nombre local). Por ejemplo, el URI de una instancia de habilidad es la siguiente:

http://www.owl-ontologies.com/unnamed.owl#Ability_2

Aquí el prefijo es <http://www.owl-ontologies.com/unnamed.owl> y el *localname* vendría siendo *Ability_2*. Cabe recalcar, que para simplificar el manejo de instancias dentro de la ontología, la gran mayoría de sus operaciones se realizan utilizando solamente el *localname*.

El servicio de gestión de perfiles de usuario combina tanto los URI asignados a cada clase y a algunas propiedades con múltiples valores, con el *localname* de las instancias de la ontología para dar acceso a cualquiera de sus recursos. El acceso se puede hacer de muchas formas, ya sea recibiendo una representación del recurso (GET), añadiendo o modificando una representación (POST o PUT) y eliminando algunas o todas las representaciones (DELETE).

No sólo es necesario conocer qué tipo de representación va a ser retornada, también es necesario enumerar los códigos de estado HTTP típicos que podrían ser devueltos. En las tablas 5 y 6 se muestran las representaciones utilizadas en cada acción, junto a sus posibles códigos de estado.

HTTP	Representación	Códigos de estado
PUT	Parámetros con los datos del nuevo recurso que va a ser creado.	200, 410
GET	Cadena de texto en formato JSON que representa al recurso	200, 400, 410
POST	Parámetros con los nuevos datos que actualizaran el recurso	200, 400
DELETE		200, 204

Tabla 5 Representación de los recursos de una clase

HTTP	Representación	Códigos de estado
PUT		200, 410
GET	Cadena de texto en formato JSON que representa a colección de valores asociados a la propiedad	200, 400, 410
POST	Cadena de texto en formato JSON que representa la nueva colección de valores asociados a la propiedad	200, 400
DELETE		200, 204

Tabla 6 Representación para el caso de una propiedad con múltiples valores

En la Tabla 7 más adelante se hace un resumen de las URIs dispuestas para los recursos el gestor:

URI del Recurso	Descripción
/private_user_id}	Acceso a la clase Person (Persona)
/Ability	Acceso a la clase Ability (Habilidad)
/CognitiveAbiliy	Acceso a la clase Cognitive_Ability (Habilidad cognitiva)
/PhysicalAbility	Acceso a la clase Physycal_Ability (Habilidad física)
/Activity	Acceso a la clase Activity (Actividad)
/Contact	Acceso a la clase Contact (Contacto)
/Education	Acceso a la clase Education (Educación)
/Expertise	Acceso a la clase Expertise (Experiencia)
/ComputerExpertise	Acceso a la clase Computer_Expertise (Experiencia en computadores)
/Interest	Acceso a la clase Interest (Interes)
/LivingConditions	Acceso a la clase Living_Conditions (Condiciones de vida)
/Preference	Acceso a la clase Preference (Preferencias)
/Profession	Acceso a la clase Profession (Profesion)
/Thing	Acceso a la clase Thing (Objetos)
/private_user_id}/abilities	Acceso a la colección de habilidades del usuario
/private_user_id}/activities	Acceso a la colección de actividades del usuario
/private_user_id}/cellularphone	Acceso a la colección de teléfonos celulares del usuario
/private_user_id}/contacts	Acceso a la colección de contactos del usuario
/private_user_id}/email	Acceso a la colección de correos electrónicos del usuario
/private_user_id}/education	Acceso a la colección de educación del usuario
/private_user_id}/expertise	Acceso a la colección de experiencia del usuario
/private_user_id}/interests	Acceso a la colección de intereses del usuario
/private_user_id}/livingconditions	Acceso a la colección de condiciones de vida del usuario
/private_user_id}/posseses	Acceso a la colección de posesiones del usuario
/private_user_id}/preferences	Acceso a la colección de preferencias del usuario
/private_user_id}/profession	Acceso a la colección de profesiones del usuario
/private_user_id}/website	Acceso a la colección de sitios web del usuario

Tabla 7 URIs de acceso a los diversos recursos de la ontología

Por ejemplo, para creación una instancia de la clase habilidad en la ontología, se debe realizar una solicitud que invoque a la URI de dicha clase, es decir “/Ability”, utilizando el método PUT, además de enviar un conjunto de parámetros con los datos de la nueva habilidad. Si la operación es exitosa se retornará un código 200 OK junto con el *localname* asignado a la nueva instancia creada. Si se desea recuperar los valores de la habilidad creada, se utilizaría el método GET nuevamente con la URI de la clase, enviándole como parámetro el *localname* de la instancia que se desea recuperar. Si la operación es exitosa retorna un código 200 OK junto a una representación en formato JSON de la habilidad solicitada.

4.7 EVALUACIÓN DEL GESTOR DE PERFILES DE USUARIO

Para evaluar la funcionalidad establecida en los requerimientos para la gestión ontológica de perfiles de usuario, se realizaron un conjunto de pruebas unitarias que comprobaron el correcto funcionamiento de cada uno de los métodos desarrollados para gestionar cada una de las clases de la ontología.

En este apartado se describirán las pruebas realizadas específicamente para los recursos que ofrece la clase habilidad (*Ability*), no obstante las mismas pruebas se realizaron con cada una de las clases que hacen parte de la ontología de personalización, por medio de la implementación de test unitarios que desarrollan un conjunto de actividades que conforman las pruebas.

4.7.1 Escenario de evaluación

En la Figura 16 se representa el escenario de pruebas, que se utilizó para constatar el correcto funcionamiento del gestor de perfiles de usuario. En esta se aprecian a dos servidores de aplicación que albergan por un lado a la aplicación IMS y por otro al gestor de perfiles de Usuario.

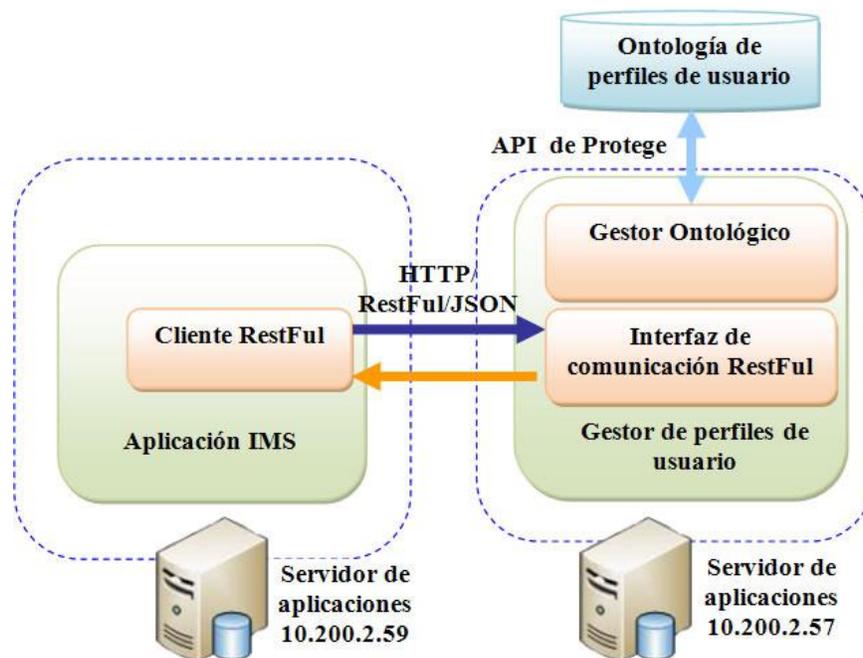


Figura 16 Escenario de evaluación del gestor de perfiles de usuario

Componente	Alojado en AS Dirección	Descripción
Cliente RestFul	10.200.2.59	El cliente RestFul es un módulo que hace parte de la Aplicación IMS y cuya función es la comunicarse con el gestor de perfiles de usuario utilizando el protocolo HTTP, enviando solicitudes acordes con el estilo de arquitectura RESTful, y recibiendo información en formato JSON. Dentro de esta evaluación es el encargado de realizar las peticiones que conforman las pruebas unitarias de funcionamiento.
Gestor de perfiles de usuario	10.200.2.57	Es el gestor de perfiles de usuario quien se encarga de manipular la ontología de acuerdo a las peticiones hechas por el cliente y que son recibidas por medio de su interfaz de comunicación RESTful.

Tabla 8 Componentes de la evaluación

4.7.2 Actividades y organización de las pruebas

De acuerdo a los casos de uso, la aplicación IMS puede realizar las siguientes actividades sobre el gestor de perfiles de usuario:

- Crear una instancia
- Consultar una instancia
- Actualizar una instancia
- Eliminar una instancia
- Adicionar una instancia al conjunto de valores de una propiedad
- Consultar instancias presentes en el conjunto de valores de una propiedad
- Actualizar instancias que conforman el conjunto de valores de una de propiedad
- Remover instancia del conjunto de valores de una propiedad

Todas estas actividades han sido combinadas para estructurar diversas pruebas, que garanticen el correcto funcionamiento del gestor de perfiles de usuario. Las pruebas son secuenciales dado que unas dependen de otras, tal es el caso de la prueba 3, en donde se utiliza la instancia creada en la prueba 1, así como en la prueba 6 se elimina una de las instancias creadas en la prueba 5.

En la siguiente tabla hace un resumen de las pruebas realizadas en conjunto con las acciones que integran cada prueba:

Prueba	Acción Realizada
Prueba 1 creación y consulta de una instancia	PUT Crear instancia
	GET Consultar instancia
Prueba 2 Actualizar una instancia	PUT Crear instancia
	GET Consultar
	POST Actualizar
	GET Consultar
Prueba 3 Eliminar una instancia:	DELETE Eliminar
	GET Consultar
Prueba 4 Adicionar y consultar instancias de una propiedad	PUT Adicionar un valor a propiedad
	GET Consultar valores de propiedad
Prueba 5 Actualizando la colección de instancias que hacen parte de una propiedad.	PUT Crear instancia
	PUT Crear instancia
	POST Actualizar valores de propiedad
	GET Consultar valores de propiedad
Prueba 6 Remover instancia de propiedad	DELETE Remover valores de propiedad
	GET Consultar valores de propiedad

Tabla 9 Tabla de pruebas y acciones realizadas

4.7.3 Selección de las herramientas a utilizar

Las herramientas utilizadas en las pruebas son las siguientes:

WireShark: Es el analizador de protocolos más utilizado alrededor del mundo, el cual permite examinar la información de una red en tiempo real o de un archivo de captura guardado en el disco, a través de detalles y sumarios por paquete. Wireshark incluye un completo lenguaje para filtrar paquetes y la habilidad de mostrar el flujo reconstruido de una sesión TCP, capacidades que son aprovechadas en el desarrollo de las pruebas.

JUnit: Es un *framework* que permite realizar la ejecución de clases Java de manera controlada, para poder evaluar si el funcionamiento de cada uno de los métodos de la clase se comporta como se espera. Es decir, en función de algún valor de entrada se evalúa el valor de retorno esperado; si la clase cumple con la especificación, entonces JUnit devolverá que el método de la clase pasó exitosamente la prueba; en caso de que el valor esperado sea diferente al que regresó el método durante la ejecución, JUnit devolverá un fallo en el método correspondiente [47].

JUnit es también un medio de controlar los cambios realizados a las aplicaciones, dado que cuando se modifica una parte del código y se desea verificar que el nuevo código cumple con los requerimientos anteriores, se pueden utilizar los test unitarios para constatar que no se ha alterado su funcionalidad después de la nueva modificación [47].

Dado que el SDS es un IDE basado en Eclipse, cuenta con los plug-ins que permiten la generación de las plantillas necesarias para la creación de las pruebas unitarias de una clase Java, que facilitan al programador enfocarse en la prueba y el resultado esperado, y dejando a la herramienta la creación de las clases que permiten coordinar las pruebas.

4.7.4 Realización de pruebas

4.7.4.1 Prueba 1 Creación y consulta de una instancia

En esta prueba se llevaron a cabo dos actividades: una de ellas fue la creación de una instancia de la clase Habilidad, y la otra, fue la consulta de los datos de la instancia creada. Estas actividades en conjunto, permiten verificar que los métodos de creación y consulta funcionan correctamente, dado que se conoce de antemano la información asignada a la nueva instancia y por ende es posible compararla con la recibida en la consulta.

En la primera actividad se invocó al recurso de clase la Habilidad por medio de su URI “/Ability” y se utilizó el método PUT, para solicitar la creación de una nueva instancia. Esta solicitud fue enviada con varios parámetros que representan los valores de las propiedades para la nueva habilidad tales como el nombre (“name”), valoración (“score”) y tipo (“type”), tal y como puede apreciarse en la Figura 17 y en el primer mensaje del flujo TCP mostrado en la Figura 18.

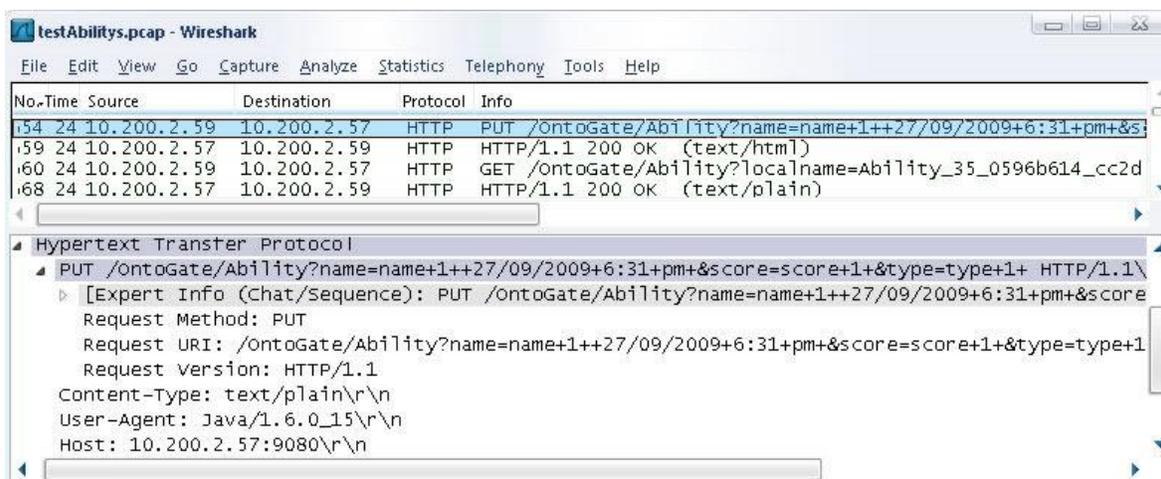


Figura 17 Paquetes HTTP de la prueba 1

La respuesta a la solicitud anterior es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene el *localname* asignado al nuevo recurso, Ability_35_0596b6b14_cc2d, tal y como puede verse en la Figura 18.

```
Follow TCP Stream
Stream Content
PUT /OntoGate/Ability?name=name+1+27/09/2009+6:31+pm+&score=score+1+&type=type+1+ HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, */*; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/html
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:22:58 GMT

18
Ability_35_0596b614_cc2d
0

GET /OntoGate/Ability?localname=Ability_35_0596b614_cc2d HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, */*; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/plain
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:22:58 GMT

73
{"name": "name 1 27/09/2009 6:31 pm ", "score": "score 1 ", "type": "type 1 ", "localname": "Ability_35_0596b614_cc2d"}
0
```

Figura 18 Flujo TCP de la prueba 1

En la segunda actividad, se realizó una consulta de la instancia creada anteriormente, invocando de nuevo al recurso de la clase Habilidad por medio de su URI “/Ability” y utilizando el método GET, para realizar una solicitud de consulta, enviando como parámetro el *localname* retornado en la primera actividad. Dicha solicitud puede apreciarse en el flujo TCP mostrado en la Figura 18.

La respuesta a la consulta fue un mensaje HTTP con código de estado 200 OK, en cuyo contenido había una representación de la habilidad creada en el primera actividad, que como puede apreciarse en la Figura 18 está en formato JSON. El mensaje de respuesta al ser interpretado, permite obtener los valores de nombre, valoración y tipo, además del *localname* pertenecientes a la nueva instancia de la clase Habilidad dentro de la ontología.

Como puede apreciarse en la Figura 18, al comparar los valores de los parámetros nombre, valor y tipo enviados para crear la habilidad y los que hacen parte de la respuesta en formato JSON, se puede verificar que son los mismos, y por tanto, la prueba realizada fue exitosa.

4.7.4.2 Prueba 2 Actualizar una instancia

Dentro de esta prueba se realizaron cuatro actividades, dos de las cuales se desarrollaron de forma similar a la prueba 1 invocando a los métodos PUT y GET, para crear y consultar los valores de una instancia, y así verificar que fue creada exitosamente.

Las siguientes actividades llevadas a cabo fueron la actualización y consulta de la información de la instancia creada en la primera actividad de esta prueba. Dichas actividades permiten verificar que el método de actualización está funcionando correctamente, dado que se conoce de antemano los

datos con los cuales se actualizará la instancia y por ende se pueden comparar con los recibidos en su consulta.

En la actividad de actualización se invocó al recurso de clase Habilidad por medio de su URI “/Ability”, y se utilizó el método POST, para realizar una solicitud de actualización. Esta solicitud fue enviada junto con varios parámetros que representan los valores de las propiedades que serán actualizadas, siendo estas el nombre, la valoración y tipo. Otro de los parámetros enviados, y a la vez necesario para realizar la actualización, es el *localname*, dato que se obtuvo al realizar la primera actividad en prueba 2, después de crear una nueva instancia cuyo valor es Ability_36_0596b614_cc2d. En la Figura 19 se resalta la llamada al método POST para la actualización al igual que puede verse en el primer mensaje del flujo TCP mostrado en la Figura 20.

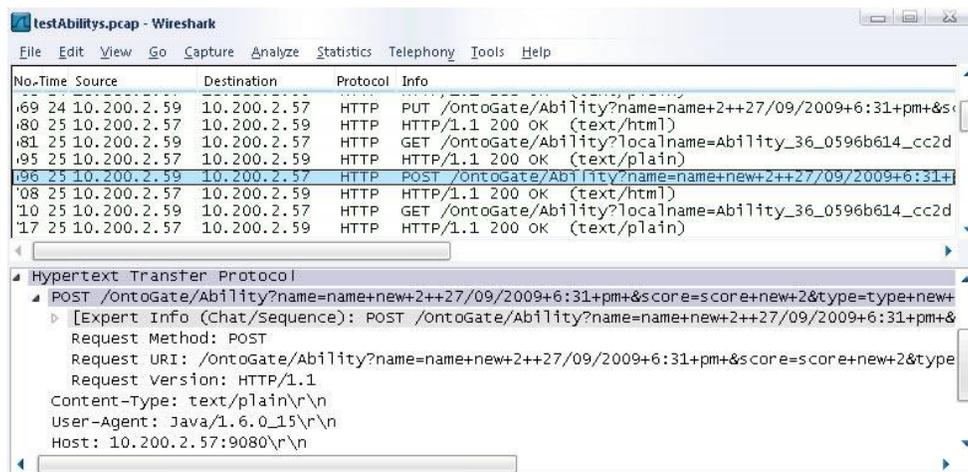
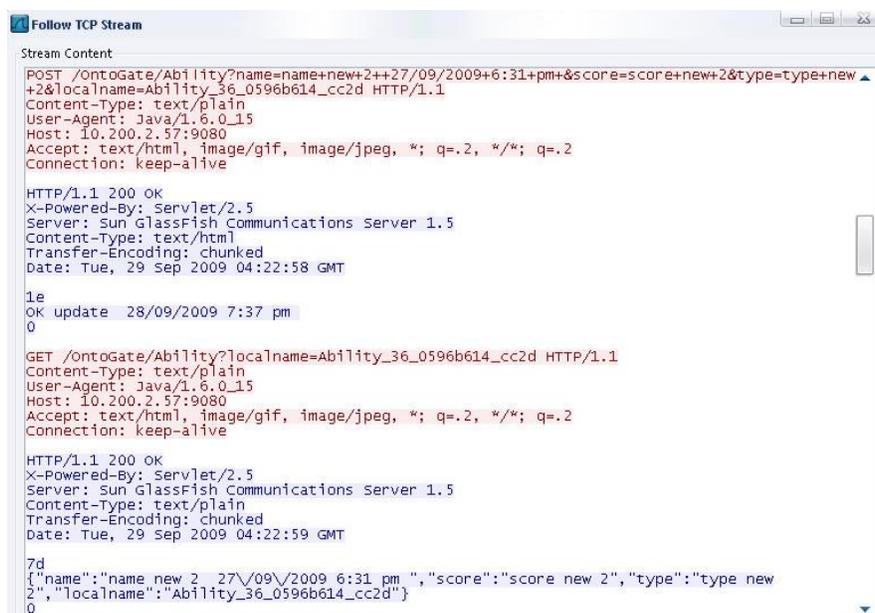


Figura 19 Paquetes HTTP de la prueba 2

La respuesta a la solicitud enviada es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene un mensaje de éxito en la operación (“OK update”), tal y como puede verse en la Figura 20

Como última actividad de la prueba 2, se realizó una consulta de los datos de la instancia que se actualizó, utilizando el método GET e invocando nuevamente al recurso de la clase habilidad por medio de su URI “/Ability”, agregándole como parámetro el *localname* retornado en la primera actividad, y así, realizar la solicitud que se puede apreciar en el flujo TCP mostrado en la Figura 20.



```
Stream Content
POST /OntoGate/Ability?name=name+new+2++27/09/2009+6:31+pm+&score=score+new+2&type=type+new
+2&localname=Ability_36_0596b614_cc2d HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, */*; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/html
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:22:58 GMT

1e
OK update 28/09/2009 7:37 pm
0

GET /OntoGate/Ability?localname=Ability_36_0596b614_cc2d HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, */*; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/plain
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:22:59 GMT

7d
{"name":"name new 2 27\09\2009 6:31 pm ","score":"score new 2","type":"type new
2","localname":"Ability_36_0596b614_cc2d"}
0
```

Figura 20 Flujo TCP de la prueba 2

La respuesta a la petición hecha, es un mensaje HTTP con código de estado 200 OK, en el cual se retorna la representación de la habilidad actualizada con sus nuevos valores, que como puede apreciarse en la Figura 20 es una respuesta en formato JSON, la cual contiene los valores enviados al solicitar la actualización, además del *localname* que le fue asignado a la instancia.

Como puede apreciarse en la Figura 20, al comparar los valores de los parámetros nombre, valor y tipo enviados para actualizar la habilidad y los que hacen parte del mensaje en formato JSON, se puede verificar que son los mismos y por lo tanto la prueba realizada fue exitosa.

4.7.4.3 Prueba 3 Eliminar una instancia

Dentro de esta prueba se llevaron a cabo las actividades de eliminar una instancia de habilidad junto a una consulta para constatar fue eliminada. Dichas actividades en conjunto permiten verificar que el método para eliminar una instancia está funcionando correctamente, dado que se debe obtener una respuesta de error cuando se quiere acceder a un recurso que ya no existe.

Esta prueba depende de la prueba 2 dado que la instancia creada en dicha prueba, es la que se elimina en la prueba 3. Si la prueba 2 no ha sido ejecutada entonces la prueba 3 no será satisfactoria.

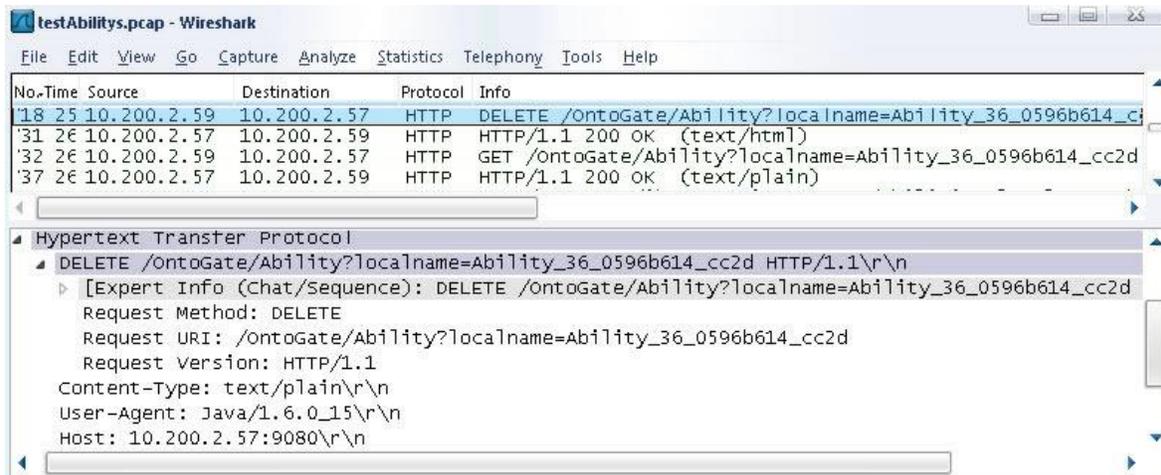


Figura 21 Paquetes HTTP de la prueba 3

En la primera actividad se invocó al recurso de clase Habilidad por medio de su URI “/Ability”, utilizando el método DELETE, enviando así una la solicitud de eliminación. A dicha solicitud se le agrega como parámetro el *localname* de la instancia que se quiere eliminar que en este caso es Ability_36_0596b614_cc2d, tal y como puede apreciarse en la Figura 21 y en el primer mensaje del flujo TCP mostrado en la Figura 22.

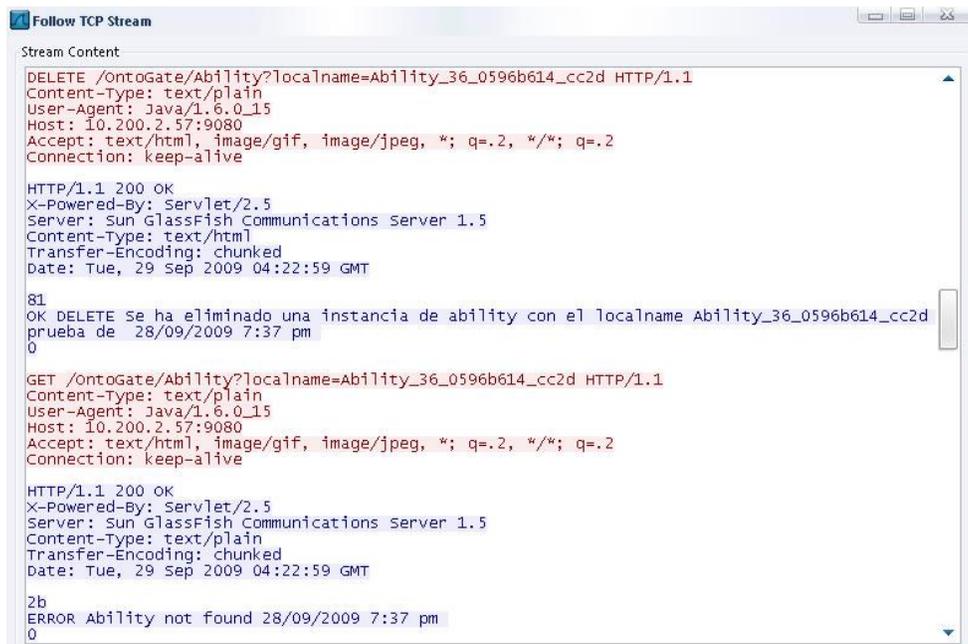


Figura 22 Flujo TCP de la prueba 3.

La respuesta a la solicitud enviada es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene un mensaje de éxito en la operación (“OK DELETE”), que puede apreciarse en el flujo TCP de la Figura 22.

En la segunda actividad, se realizó una consulta sobre la instancia eliminada, por medio del método GET e invocando recurso de la clase Habilidad utilizando su URI “/Ability”, enviándole como parámetro el *localname* (Ability_36_0596b614_cc2d) de la instancia que fue eliminada, y así, realizar la solicitud que se puede apreciar en el flujo TCP mostrado en la Figura 22.

La respuesta a la petición hecha, es un mensaje HTTP con código de estado 200 OK, que en su contenido se devuelve un Error, el cual se estaba esperando dado que la habilidad que se consultó ya no debía hacer parte de la ontología.

En las pruebas no siempre se espera un valor de éxito en las transacciones realizadas, un ejemplo de esto es esta prueba, donde el valor esperado es un error, que determina que la prueba ha culminado con éxito.

4.7.4.4 Prueba 4 Adicionar y Consultar instancias de una propiedad

Aquí se utiliza la instancia creada en la prueba 1, identificada con el *localname* Ability_35_0596b14_cc2d, para ser adicionada a la colección de instancias que hacen parte de la propiedad habilidades (*abilities*) de un usuario, que previamente ha sido creado para facilitar el correcto desempeño de las pruebas.

La prueba realizada está compuesta por dos actividades, una ellas la adición de una instancia a la colección de valores que hacen parte la propiedad habilidades de un perfil de usuario identificado como *jhonny@ericsson.com*; y la otra su consulta. Dichas actividades en conjunto permiten verificar que los métodos de adición y consulta están funcionando correctamente, puesto que se conoce de antemano la instancia a adicionar y podemos distinguir si hace parte de la colección retornada en la consulta.

En la primera actividad se invocó el URI que identifica a la colección de habilidades de un usuario, “*jhonny@ericsson.com/abilities*” y se utilizó el método PUT, para solicitar la adición de una instancia. Esta solicitud fue enviada agregado como parámetro el *localname* (Ability_35_0596b14_cc2d) de la habilidad ha adicionar, tal y como puede apreciarse en la Figura 23 y en el primer mensaje del flujo TCP mostrado en la Figura 24.

La respuesta a la solicitud anterior es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene un mensaje de operación exitosa (OK Habilidad adicionada), tal y como puede verse en la Figura 24.

En la segunda actividad se consultó la colección de habilidades del usuario a la que anteriormente se le adicionó una instancia, para ello se invocó nuevamente el URI que la identifica a la colección “*jhonny@ericsson.com/abilities*”, y se utilizó el método GET, para realizar una solicitud de consulta. Dicha solicitud puede apreciarse en el flujo TCP mostrado en la Figura 24.

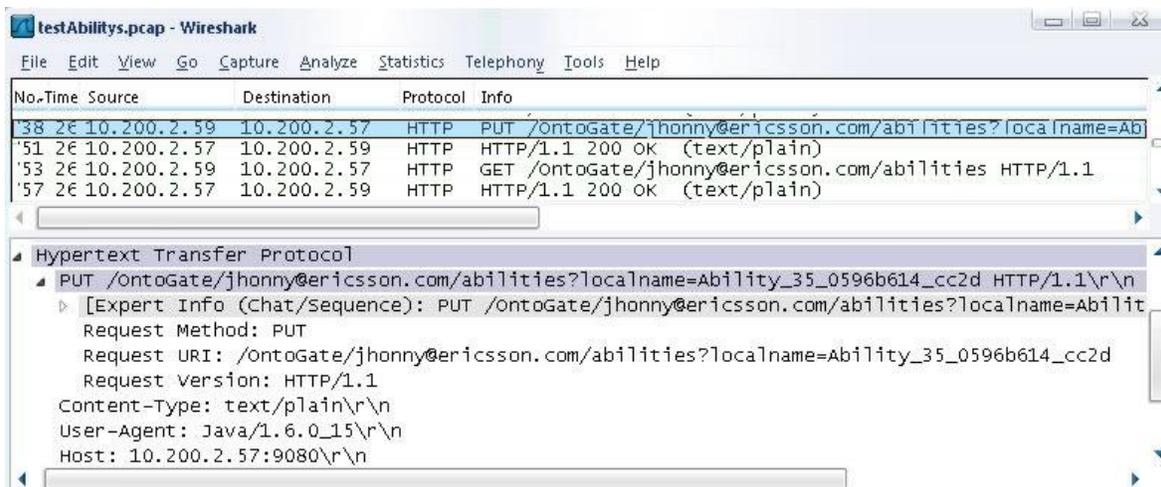


Figura 23 Paquetes HTTP prueba 4

La respuesta a la consulta es un mensaje HTTP con código de estado 200 OK, en cuyo contenido hay una representación de la colección de habilidades del usuario, que como puede apreciarse en la Figura 24 está en formato JSON. Dicho mensaje de respuesta al ser interpretado, permite obtener cada una de las instancias que hacen parte de la colección, en donde podemos observar se encuentra la instancia adicionada en la primera actividad, y por lo cual la prueba es exitosa.

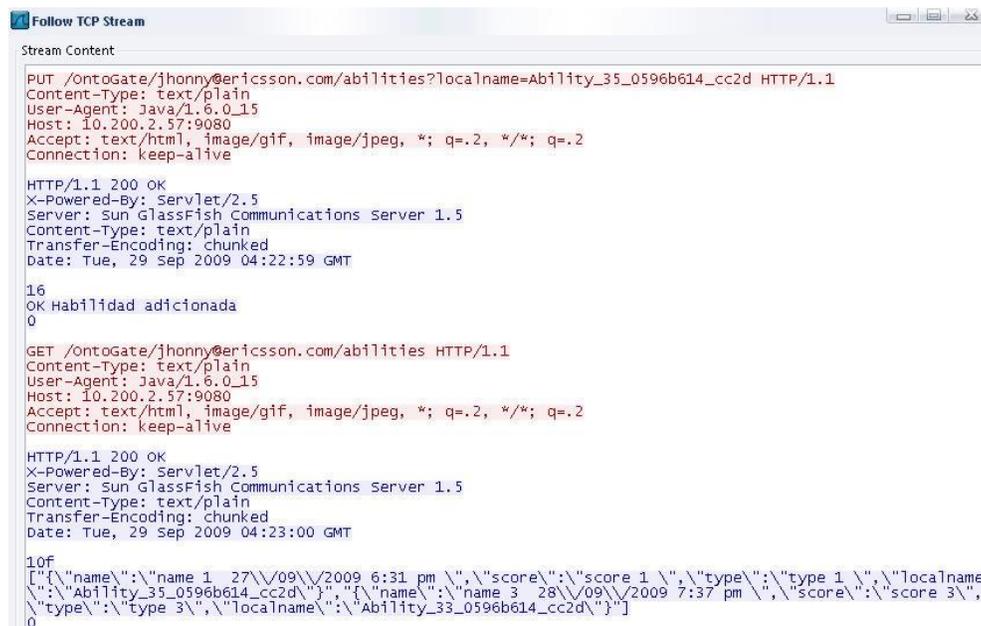


Figura 24 Flujo TCP de la prueba 4

4.7.4.5 Prueba 5 Actualizando la colección de instancias de una propiedad.

Dentro de esta prueba se realizaron actividades como: la creación de dos nuevas instancias, la actualización de la colección habilidades y su consulta. La creación de instancias se desarrolla de forma similar a la prueba 1 utilizando el método PUT y enviando como parámetros los valores de nombre, valoración y tipo. Dichas solicitudes se pueden apreciar la Figura 25.

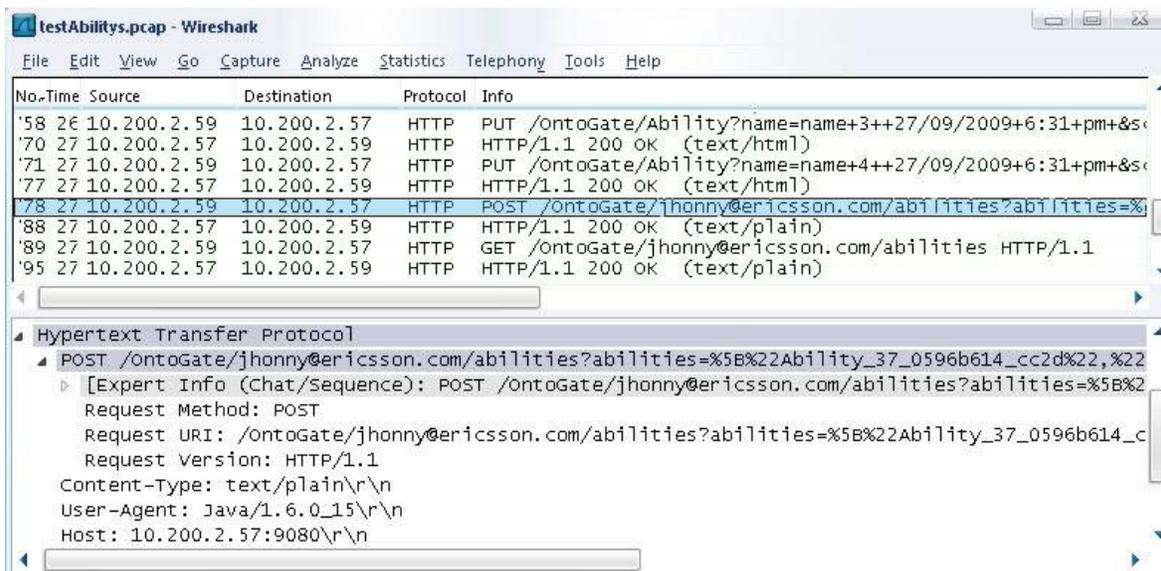
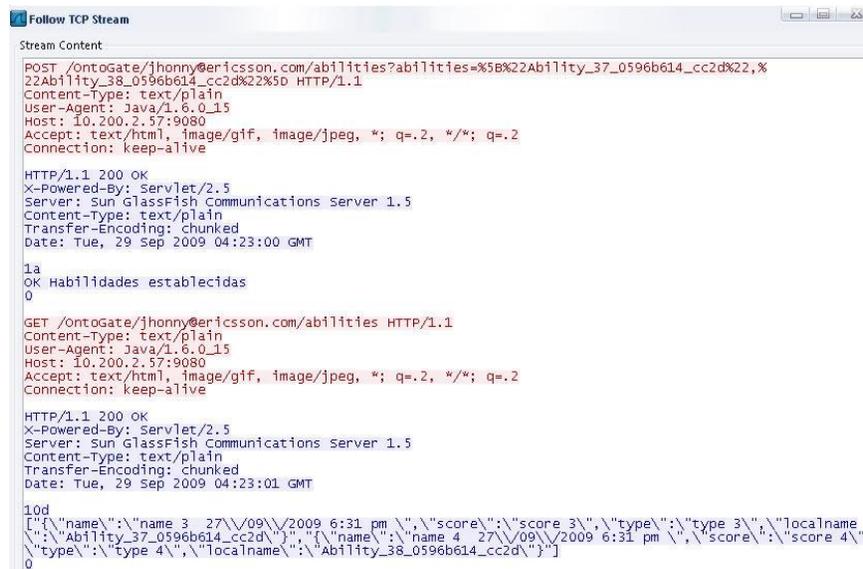


Figura 25 Paquetes HTTP prueba 5

Para la actualización de las habilidades, se utilizan los nombres locales de las nuevas instancias creadas al inicio de la prueba, para conformar un arreglo en formato JSON, el cual es enviado como parámetro en la solicitud de actualización. Esta solicitud se realizó invocando al URI de la colección de habilidades del usuario *“jhonny@ericsson.com/abilities”* y utilizando el método PUT, tal y como puede verse en la Figura 25

La respuesta a la solicitud anterior es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene un mensaje de operación exitosa (OK Habilidades establecidas), tal y como puede verse en la Figura 26.

Como segunda actividad, se consultó la colección de habilidades del usuario a la que anteriormente se le realizó la actualización, para ello se invocó nuevamente el URI que la identifica a la colección *“jhonny@ericsson.com/abilities”*, y se utilizó el método GET, para realizar una solicitud de consulta. Dicha solicitud puede apreciarse en el flujo TCP mostrado en la Figura 26.



```
Stream Content
POST /OntoGate/jhony@ericsson.com/abilities?abilities=%5B%22Ability_37_0596b614_cc2d%22,%
22Ability_38_0596b614_cc2d%22%5D HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/plain
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:23:00 GMT

1a
OK Habilidades establecidas
0

GET /OntoGate/jhony@ericsson.com/abilities HTTP/1.1
Content-Type: text/plain
User-Agent: Java/1.6.0_15
Host: 10.200.2.57:9080
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive

HTTP/1.1 200 OK
X-Powered-By: Servlet/2.5
Server: Sun GlassFish Communications Server 1.5
Content-Type: text/plain
Transfer-Encoding: chunked
Date: Tue, 29 Sep 2009 04:23:01 GMT

10d
[{"name":"name 3 27\09\2009 6:31 pm \\", \"score\":\"score 3\", \"type\":\"type 3\", \"localname
\": \"Ability_37_0596b614_cc2d\"}, {"name":"name 4 27\09\2009 6:31 pm \\", \"score\":\"score 4\",
\"type\":\"type 4\", \"localname\":\"Ability_38_0596b614_cc2d\"}]
```

Figura 26 Flujo TCP de la prueba 5

La respuesta de la consulta es un mensaje HTTP con código de estado 200 OK, que contiene una representación de la colección de habilidades del usuario, que como puede apreciarse en la Figura 26 está en formato JSON. Dicho mensaje de respuesta al ser interpretado, permite obtener cada una de las instancias que hacen parte de la colección, en donde se puede observar se encuentran tan sólo las dos nuevas instancias con las que se actualizó la colección, cuyos *localnames* son Ability_37_0596b614_cc2d y Ability_38_0596b614_cc2d, los cuales son los mismos que hacen parte de el arreglo enviado en la actualización, tal como se puede constar en la Figura 26

4.7.4.6 Prueba 6 Remover Instancia de propiedad

Esta prueba requiere de la prueba 5 para su correcto funcionamiento, dado que una de las instancias creadas en ella y que hace parte de la colección de habilidades del usuario, será removida, por lo tanto, dicha prueba debe ser ejecutada previamente para que los resultados sean satisfactorios.

Las actividades realizadas durante esta prueba son dos: la eliminación de una de las instancias que hacen parte de la colección de habilidades de un usuario, y la consultad de dicha colección, las cuales permiten constatar que la funcionalidad de remover una instancia del una propiedad trabaja de la manera deseada.

En la primera actividad se invocó el URI que identifica a la colección de habilidades de un usuario, “/jhony@ericsson.com/abilities” y se utilizó el método DELETE, para solicitar la remoción de la instancia identificada con el *localname* (Ability_38_0596b614_cc2d), dato que se envía como parámetro de la solicitud, tal y como puede apreciarse en la Figura 27 y en el primer mensaje del flujo TCP mostrado en la Figura 28.

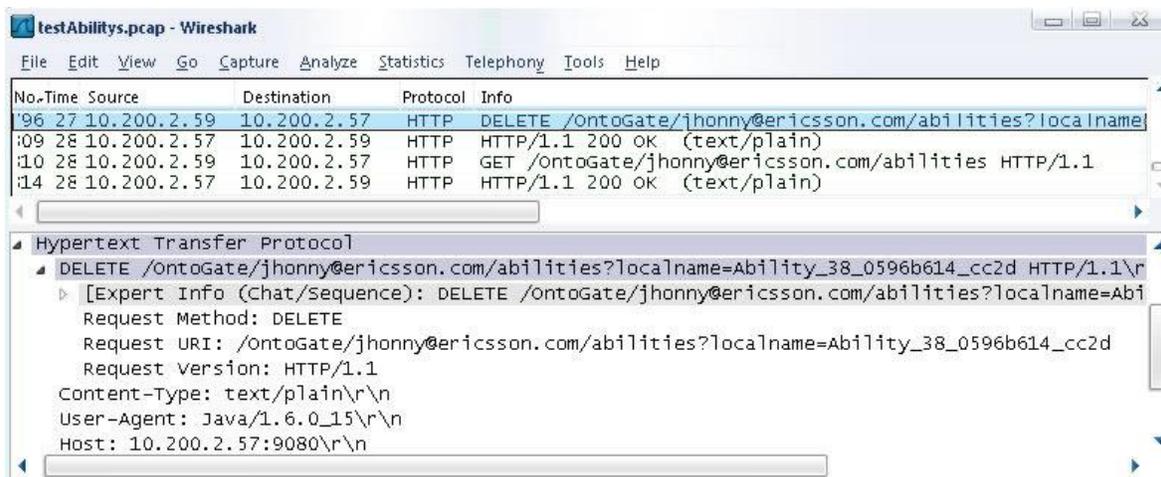


Figura 27 Paquetes HTTP de la Prueba 6

La respuesta a la solicitud anterior es un mensaje HTTP cuyo código de estado es 200 OK, y que contiene un mensaje de operación exitosa (OK Hability removed), tal y como puede verse en la Figura 28.

Dentro de la segunda actividad de la prueba 6, se consultó nuevamente la colección de habilidades del usuario, para ello se invocó el URI que identifica a la colección “*jhonny@ericsson.com/abilities*”, y se utilizó el método GET, para realizar una solicitud de consulta. Dicha solicitud puede apreciarse en el flujo TCP mostrado en la Figura 28

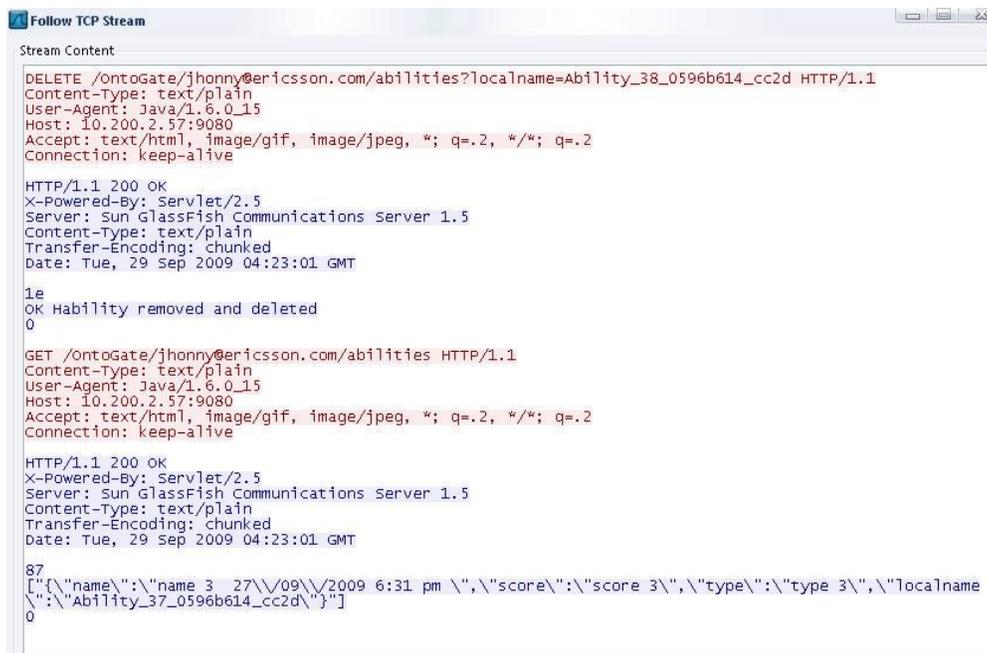
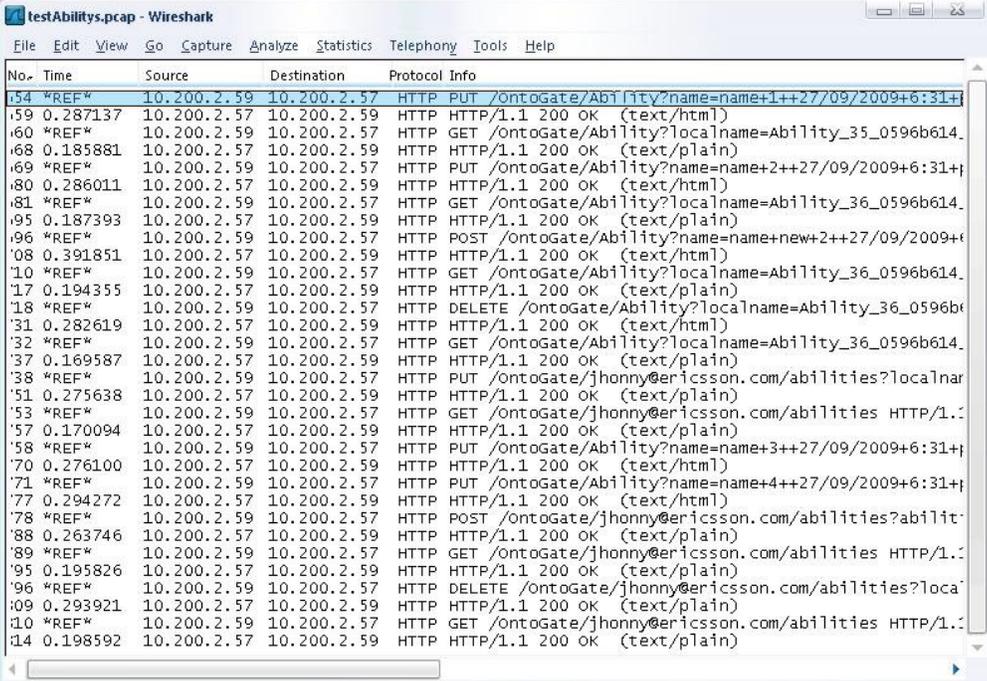


Figura 28 Flujo TCP de la prueba 6

La respuesta de la consulta es un mensaje HTTP con código de estado 200 OK, que contiene una representación de la colección de habilidades del usuario, que como puede apreciarse en la Figura 28 está en formato JSON. Dicho mensaje de respuesta al ser interpretado, permite obtener cada una de las instancias que hacen parte de la colección, en donde se observa que se encuentra tan sólo una de las instancias con las que anteriormente se actualizó la colección en la prueba 5, cuyo *localname* es Ability_37_0596b614_cc2d. De esta forma se puede constatar que la habilidad identificada con el *localname* Ability_38_0596b614_cc2d ya no hace parte de colección de habilidades del usuario, tal como se puede constar en la Figura 28, y por lo cual se puede decir que la prueba fue exitosa.

4.7.4.7 Tiempos de respuesta

Wireshark y los test de JUnit permiten medir de tiempo de respuesta de cada una de las actividades realizadas en las pruebas. Cabe anotar que en los tiempos que ofrecen no son los mismos, dado Wireshark brinda la medida del tiempo entre una petición y su respuesta, mientras que el test unitario nos ofrece el tiempo en realizar una actividad desde que esta es invocada. En la Figura 29 se muestra los tiempos de respuesta que ofrece Wireshark.



No.	Time	Source	Destination	Protocol Info
54	*REF*	10.200.2.59	10.200.2.57	HTTP PUT /OntoGate/Ability?name=name+1++27/09/2009+6:31+...
59	0.287137	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
60	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/Ability?localname=Ability_35_0596b614...
68	0.185881	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
69	*REF*	10.200.2.59	10.200.2.57	HTTP PUT /OntoGate/Ability?name=name+2++27/09/2009+6:31+...
80	0.286011	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
81	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/Ability?localname=Ability_36_0596b614...
95	0.187393	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
96	*REF*	10.200.2.59	10.200.2.57	HTTP POST /OntoGate/Ability?name=name+new+2++27/09/2009+...
108	0.391851	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
10	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/Ability?localname=Ability_36_0596b614...
17	0.194355	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
18	*REF*	10.200.2.59	10.200.2.57	HTTP DELETE /OntoGate/Ability?localname=Ability_36_0596b...
31	0.282619	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
32	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/Ability?localname=Ability_36_0596b614...
37	0.169587	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
38	*REF*	10.200.2.59	10.200.2.57	HTTP PUT /OntoGate/jhonny@ericsson.com/abilities?localnar...
51	0.275638	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
53	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/jhonny@ericsson.com/abilities HTTP/1.1...
57	0.170094	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
58	*REF*	10.200.2.59	10.200.2.57	HTTP PUT /OntoGate/Ability?name=name+3++27/09/2009+6:31+...
70	0.276100	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
71	*REF*	10.200.2.59	10.200.2.57	HTTP PUT /OntoGate/Ability?name=name+4++27/09/2009+6:31+...
77	0.294272	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/html)
78	*REF*	10.200.2.59	10.200.2.57	HTTP POST /OntoGate/jhonny@ericsson.com/abilities?abilit...
88	0.263746	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
89	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/jhonny@ericsson.com/abilities HTTP/1.1...
95	0.195826	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
96	*REF*	10.200.2.59	10.200.2.57	HTTP DELETE /OntoGate/jhonny@ericsson.com/abilities?loca...
109	0.293921	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)
110	*REF*	10.200.2.59	10.200.2.57	HTTP GET /OntoGate/jhonny@ericsson.com/abilities HTTP/1.1...
114	0.198592	10.200.2.57	10.200.2.59	HTTP HTTP/1.1 200 OK (text/plain)

Figura 29 Tiempos de respuesta a las solicitudes en Wireshark

Por otra parte JUnit nos ofrece un resumen donde se muestra si cada uno de los test realizados fue exitoso o no, junto a sus respectivos tiempos de respuesta. Este resumen también muestra el número de fallos y errores encontrados, así como el tiempo total en la ejecución de todos los test. En la Figura 30 se muestra uno de estos resúmenes.

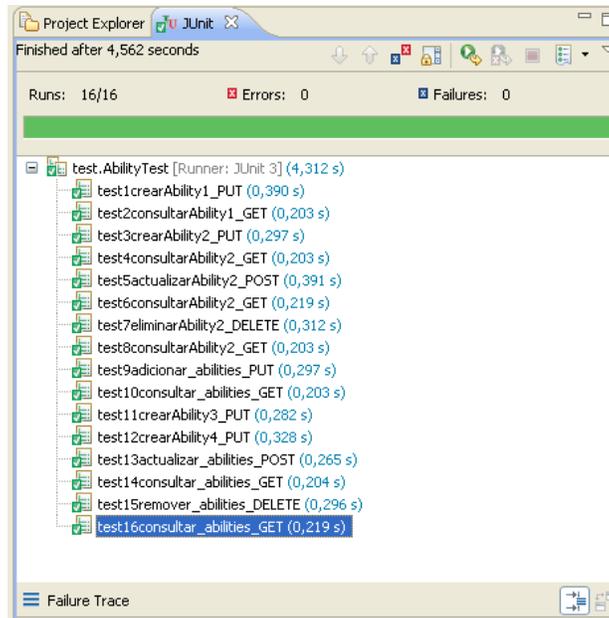


Figura 30 Resumen de las pruebas realizadas en JUnit.

4.7.4.8 Pruebas de rendimiento.

Las pruebas de rendimiento son realizadas con el fin de comprobar el desempeño del gestor de perfil de usuario con respecto a un número determinado de clientes que realizan peticiones simultáneas.

Para el desarrollo de estas pruebas se elaboró una clase encargada de crear varios clientes que simultáneamente realizan una petición cuya respuesta ya se conoce de antemano y por ende se pueden determinar cuándo ocurre un error o simplemente los clientes despliegan en consola los errores que ocurren en el transcurso de las peticiones.

En la Tabla 10 y Figura 31 se muestran los resultados obtenidos a los intentos de conexión simultáneos al servicio, es así que se puede apreciar que a partir de las 300 conexiones solicitadas tan solo 273 obtienen una respuesta exitosa, número que de ahí en adelante decae con respecto al aumento de las solicitudes de conexión realizadas.

Conexiones Solicitadas	Conexiones Exitosas
1	1
3	3
5	5
8	8
10	10
20	20
30	30
50	50

80	80
100	100
150	150
200	200
250	250
280	280
300	273
400	253
500	243

Tabla 10 Conexiones Exitosas

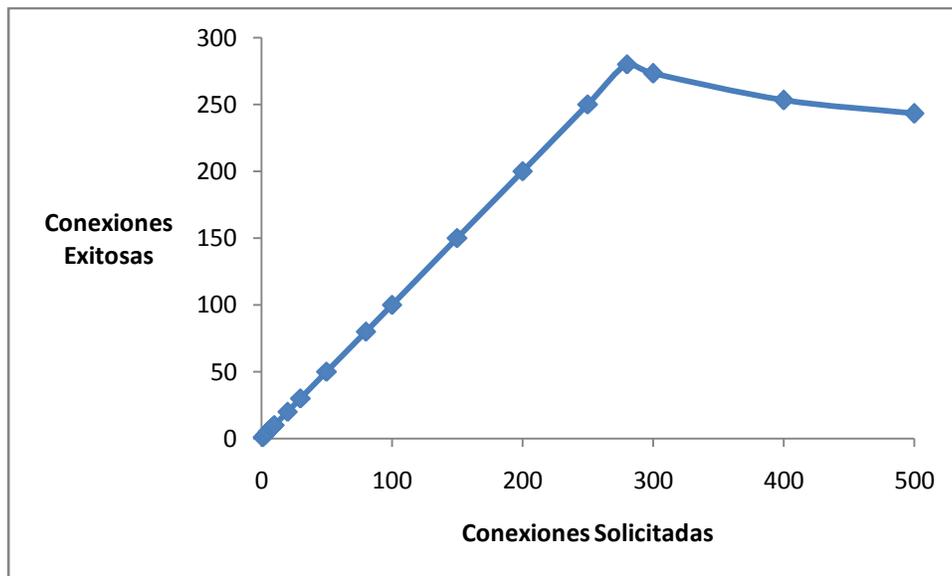


Figura 31 Conexiones exitosas.

En la Tabla 11 se muestran los obtenidos del tiempo promedio de respuesta a las solicitudes consecutivas que realizan un determinado número de clientes del servicio ofrecido por el gestor de perfiles de usuario.

Número de Clientes	tiempo promedio de respuesta en milisegundos
1	22
3	57
5	94
8	150
10	186
20	370
30	551

50	950
80	1739
100	2233
150	8378
200	11680
250	13768
300	15357
400	15497
500	14854

Tabla 11 Tiempo promedio de respuesta

En la Figura 32 se puede apreciar que conforme aumentan el número de clientes que realizan peticiones consecutivas, el tiempo de respuesta va aumentando y empiezan a decaer cuando se alcanza el número máximo de solicitudes que se pueden responder, dado que en ese momento las conexiones comienzan a ser rechazadas.

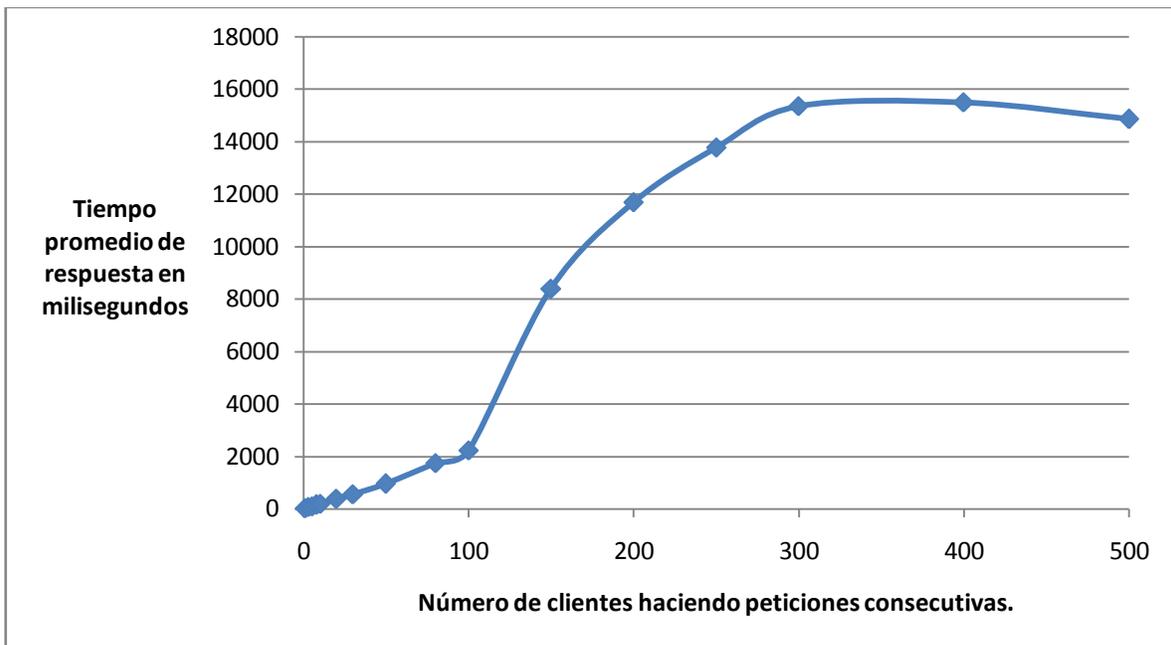


Figura 32 Tiempo promedio de respuesta.

En la Tabla 12 se muestran los valores obtenidos de las peticiones que obtuvieron una respuesta correcta en un segundo.

Número de Clientes	peticiones/segundo
10	57

20	58
30	59
50	60
80	61
100	63
150	65
200	68
250	71
280	72
300	72
400	64
500	58

Tabla 12 Número de peticiones respondidas por segundo

En la Figura 33 se puede apreciar que cuando se alcanza el número máximo de clientes a los que se le puede responder sus peticiones exitosamente se sobrepasa de los 70 peticiones por segundo, sin embargo al aumentar el número de clientes que realizan solicitudes simultaneas el número de peticiones que obtienen una respuesta exitosa comienza a decaer, esto sucede porque el número de conexión rechazas comienzan incrementarse.

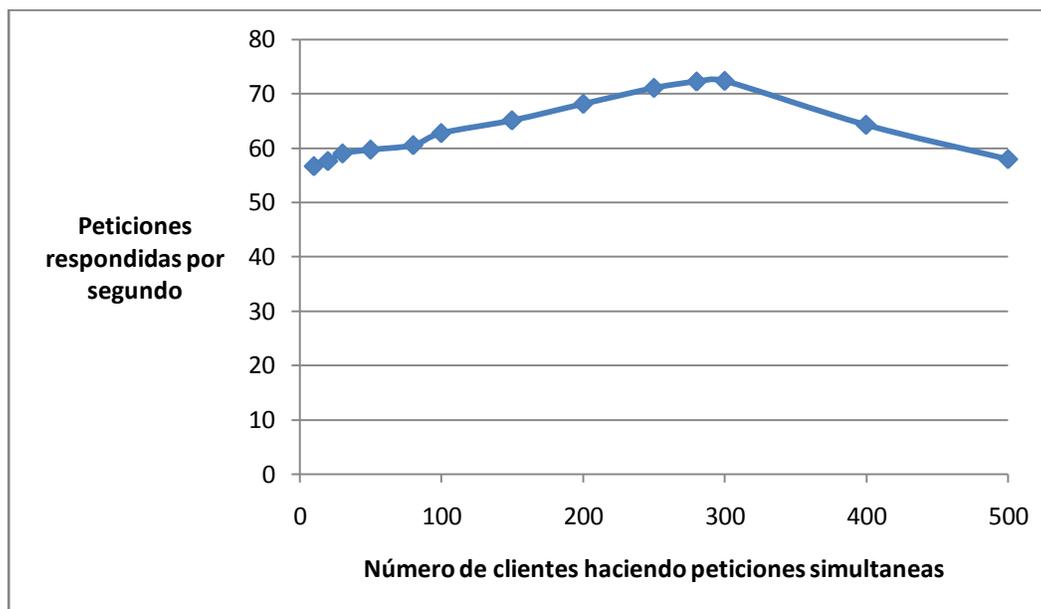


Figura 33 Número de peticiones respondidas por segundo

SERVICIO DE PRUEBA

4.8 INTRODUCCIÓN

El objetivo de este capítulo es evidenciar la interacción entre el Sistema de Gestión de Perfiles de Usuario (SGPU) y un servicio dentro del dominio IMS.

Utilizando el conjunto de herramientas que el SDS de Ericsson ofrece para la generación de servicios y contando con un ambiente de pruebas para la simulación de aplicaciones en el dominio IMS, se desarrolló un escenario que permite validar la arquitectura propuesta y con el cual se verifica la gestión de los datos del perfil de usuario, para la personalización de servicios.

4.9 DESCRIPCIÓN DEL ESCENARIO IMS

El escenario creado está conformado por los siguientes elementos:

- el núcleo de red IMS,
- un servicio que corre sobre un AS e interactúa con el SGPU
- una aplicación cliente
- y las herramientas para el análisis de tráfico.



Figura 34 Escenario IMS

Cada uno de estos elementos será descrito en las siguientes secciones.

4.9.1 Configuración del núcleo de red IMS

Es necesario seguir ciertos pasos para la provisión de servicios sobre la plataforma proporcionada por el SDS. Dentro de la configuración del núcleo de IMS, se deben establecer algunos valores para determinadas entidades que son básicas en el desarrollo de soluciones de extremo a extremo. Una entidad a la cual se debe prestar gran atención es el HSS. La configuración de esta entidad involucra:

- La configuración del *Initial Filter Criteria (iFC)* y los *Service Point Triggers (SPT)*,
- la configuración del *Service Profile*
- la configuración del *User Profile* y
- la configuración del *Public Service Identity (PSI)*

Por otro lado es necesario definir dominios virtuales dentro de un DNS para los iFC del CSCF y establecer las direcciones y puertos del P-CSCF, I-CSCF y S-CSCF. Para esto el SDS de Ericsson facilita una herramienta completa para la provisión de servicios, los detalles y los pasos que se siguieron se encuentran en el Anexo C.

4.9.2 Descripción del escenario para el servicio

El servicio desarrollado se publicó sobre un AS Sailfin v1, el cual viene pre configurado en el SDS para el despliegue de servicios. La solución generada se realizó a partir del JSR 289: SIP-Servlet v1.1. Por otro lado se utiliza el cliente RESTful de la implementación Jersey, para establecer un canal de comunicación HTTP entre el dominio IMS y el SGPU.

La captura de datos para la generación de perfiles de usuario partió de la interacción explícita usuario-sistema a través de formularios de consulta, se debe aclarar que, dependiendo de las características y ambiciones de cada servicio, variarán los mecanismos para la obtención de los datos, alcanzando de esta forma soluciones Adaptables, Adaptativas o Proactivas según sea el caso, como se explicó en el capítulo 1 sección 1.2.1.

El servicio de prueba interactúa con el SGPU mediante la interfaz RESTful atendiendo a las solicitudes del cliente, dichas solicitudes están enfocadas en la creación, adición, edición, recuperación y eliminación de información. Como resultado de esta interacción se obtiene una base de conocimiento enriquecida para la conformación de perfiles de usuario.

Los detalles de la implementación del servicio se encuentran en el Anexo D.

4.9.3 Descripción del cliente

El SDS facilita la generación de clientes de escritorio y móviles (con soporte para J2ME o para el sistema operativo Symbian), mediante el uso de la herramienta ICP (*IMS Client Platform*, Plataforma Cliente IMS). Para el escenario de pruebas se decidió implementar un cliente de escritorio desarrollado en JAVA, ya que permite la creación de interfaces gráficas de mayor calidad y sin las

limitaciones evidentes de los emuladores para equipos móviles, no obstante en el Anexo E se describen los pasos necesarios para la generación de este tipo de cliente.

La solución desarrollada permite la gestión de la información a través de formularios de consulta y campos para el despliegue de los datos retornados por el SGPU. Los detalles de la implementación se encuentran en Anexo E.

4.9.4 Herramientas utilizadas para análisis y pruebas

Dentro del escenario de pruebas fueron utilizadas las siguientes herramientas:

Visual Traffic Flow VTF: Es una funcionalidad proporcionada por el SDS para el seguimiento de tráfico en el dominio IMS, el VTF genera diagramas de secuencia a partir de un archivo de registro o por el procesamiento en tiempo real de los mensajes enviados entre el cliente, el núcleo de red y el servidor de aplicación, en una sesión SIP. El VTF sólo procesa tráfico UDP/SIP.

Wireshark: Este analizador de protocolos fue utilizado para el seguimiento del tráfico TCP/HTTP entre el dominio IMS (a través del cliente RESTful) y el SGPU.

4.10 VALIDACIÓN DEL SISTEMA DE GESTIÓN

La Figura 35 muestra el escenario de prueba, de un lado se encuentra el dominio IMS conformado por los clientes, el núcleo de red y el servidor de aplicación y del otro el SGPU con el gestor y la ontología de perfiles de usuario. Cada uno de los pasos realizados, comenzando con el registro e inicio de sesión hasta la creación y modificación de información serán abordados. Para cada una de las pruebas se establece que el usuario está previamente registrado en la plataforma IMS, tiene asociado un perfil de usuario, un perfil de servicio y unos criterios de filtrado iniciales.

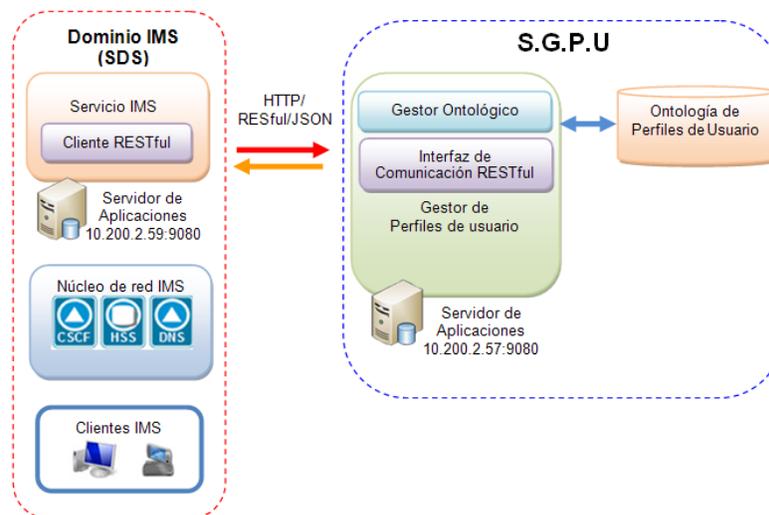


Figura 35 Escenario de prueba para el intercambio de información

4.10.1 Registro y establecimiento de sesión

El primer paso es el registro del usuario en la plataforma y el establecimiento de la sesión para acceder al servicio. El tráfico existente entre el cliente y el AS es visualizado mediante el Visual Traffic Flow, el cual captura en tiempo real la interacción entre las entidades en el dominio IMS.

En la Figura 36 y Figura 37 se muestra claramente el intercambio de mensajes SIP tanto en el registro y la suscripción como en el establecimiento de la sesión, después de estos pasos previos el servicio en el AS puede establecer comunicación con el SGPU para hacer intercambio de información según las peticiones del cliente.

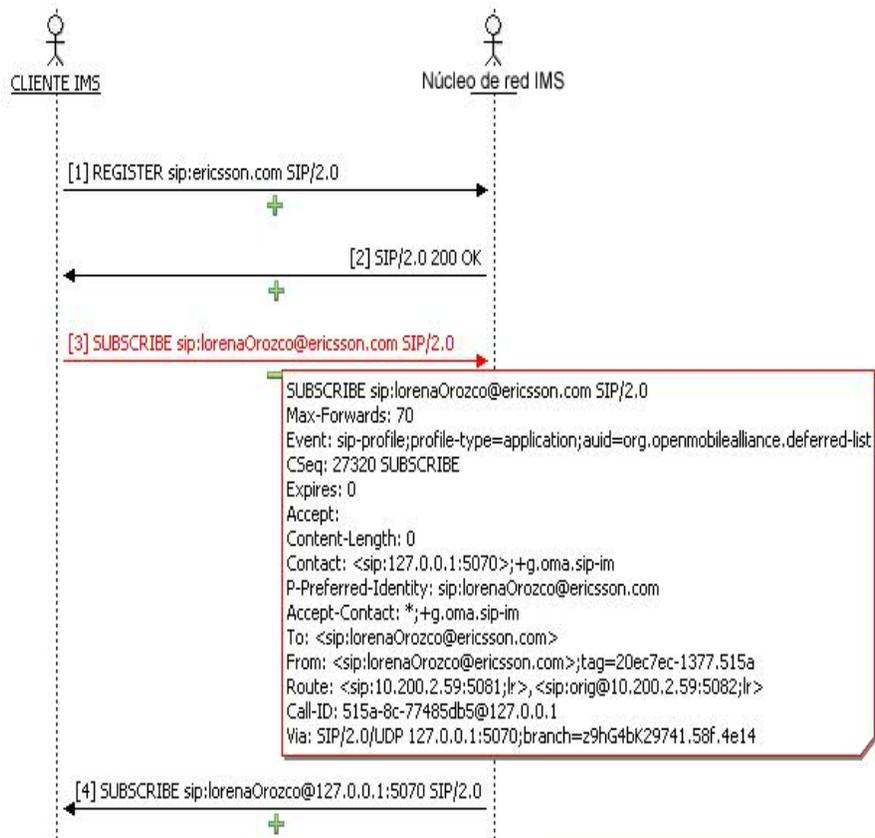


Figura 36 Registro y suscripción

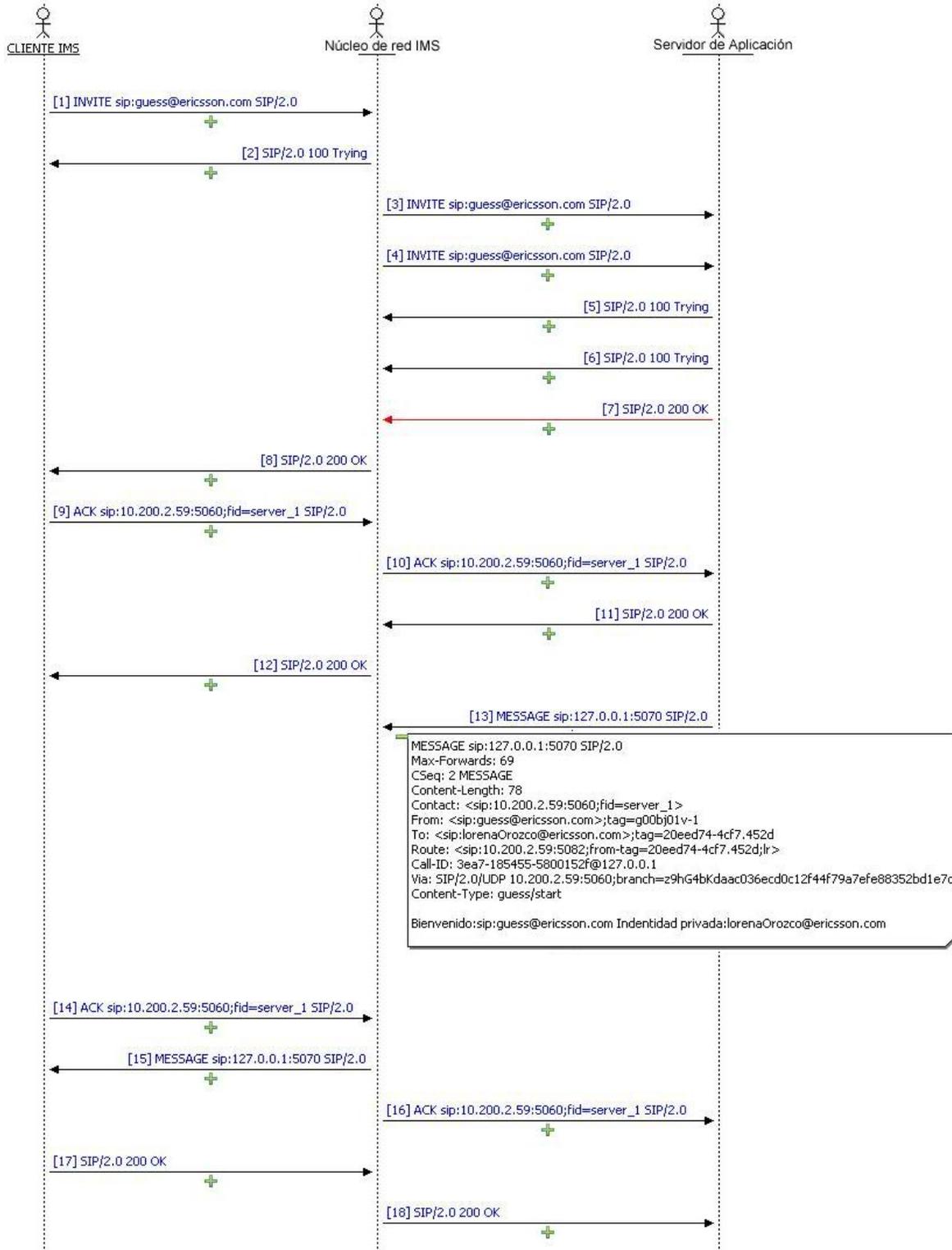


Figura 37 Establecimiento de sesión

4.10.2 Prueba 1: verificar la creación de una instancia en la ontología

El objetivo de esta prueba es verificar el intercambio de información entre el servidor de aplicaciones en el dominio IMS y el gestor que maneja la ontología. Como se ha mostrado, la ontología es una base de conocimiento que en principio está completamente vacía. Esta debe ser alimentada a partir de la interacción con los usuarios. La prueba consiste en crear una instancia de persona con cada uno de los datos que la componen. Esta información será capturada desde un formulario en el cliente.

La Figura 38 presenta un diagrama de secuencia donde se muestran los mensajes capturados por el Visual Traffic Flow. El proceso se origina en el cliente, cuando se hace la petición para crear una instancia de persona en la ontología y termina en el momento que se recibe el *localname* de la instancia creada. En caso fallido se recibe un error explicando la causa del mismo.

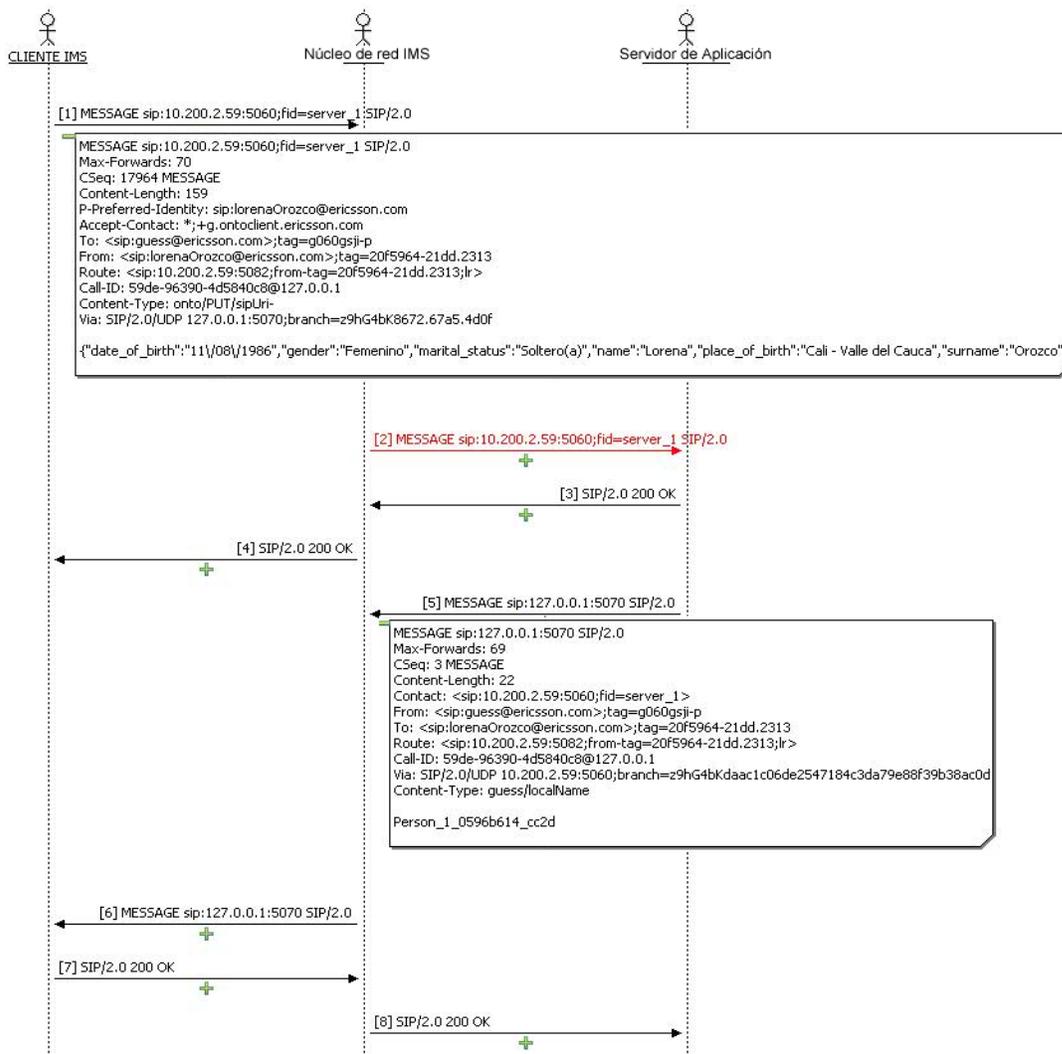


Figura 38 Envío de mensajes entre entidades en el dominio IMS.

Para verificar la creación de la instancia, se realizaron dos observaciones, una desde el AS y otra desde el SGPU. En el dominio IMS fueron capturados dos mensajes que evidencian el intercambio de información relacionada con el proceso de crear una instancia.

El primer mensaje, ubicado en la posición número 1 del diagrama de secuencia ([1] MESSAGE sip:10.200.2.59:5060), contiene la información que se envía desde el cliente hacia el AS. En la aplicación se utilizó el campo **Content-Type** de la cabecera SIP, con el fin de asociar los datos transportados con la operación que se desea realizar en la ontología. Para este fin el campo adoptó la siguiente estructura: `servicio/operación/clasePropiedad-localname`. De esta forma la cadena **onto/PUT/sipUri-**, se interpreta del lado del AS como una petición para realizar una operación PUT (creación o agregación) en la ontología, utilizando la identidad pública de usuario para crear una instancia de persona.

El campo **To** es utilizado para la validación en la plataforma IMS, el valor de este campo debe coincidir con el valor que posee un Service-Point-Trigger de tipo Request URI, que hacen parte de un iFC, de este modo, cuando llega un Request URI con valor guess@ericsson.com el *trigger* es activado y el mensaje es enviado al AS correspondiente para ser procesado. También es necesario que el valor del campo **From** coincida con el public user ID registrado en el perfil de usuario dentro del HSS. Este perfil de usuario debe tener asignado un perfil de servicio que a su vez está ligado con el iFC, nombrado anteriormente. Si el usuario está registrado pero no tiene asignado el perfil de servicio adecuado, no podrá utilizar el *trigger* que le permitirá llegar al servicio. En el diagrama de secuencia se puede observar que el mensaje llega al núcleo de red IMS y es reenviado al servidor de aplicación (mensaje número 2 del diagrama de secuencia).

Los datos que se envían para ser procesados se encuentran en formato JSON con el fin de manejar de forma ágil y estructurada el intercambio de información. Como se ve en la figura los datos que se enviaron son: los nombres, los apellidos, el género, la fecha de nacimiento, el lugar de nacimiento y el estado civil.

El segundo mensaje analizado, se ubica en la posición número 5 del diagrama de secuencia ([5] MESSAGE sip:127.0.0.1:5070), contiene la respuesta que el AS ha generado a partir de la petición del cliente. En este punto es importante aclarar el papel realizado por el servicio en el AS y la interacción de este con el gestor de la ontología ubicado en el SGPU. La primera acción que realiza el servicio es analizar la cabecera Content-Type para establecer la operación que se realizará en la ontología, el paso siguiente es crear el canal de comunicación y finalmente el reenvío de los datos al gestor de perfiles de usuario. Es de recordar que el diagrama de secuencia sólo muestra los mensajes en el dominio IMS. La Figura 39 y Figura 40 muestran el flujo de información entre los dos dominios (IMS y SGPU).

En Figura 39 se puede ver una petición PUT enviada al gestor y el mensaje de respuesta enviada al AS con el *localname* de la instancia creada en la ontología cuyo valor es Person_1_0596b614_cc2d.



Figura 39 Flujo TCP entre el dominio IMS y el SGPU.

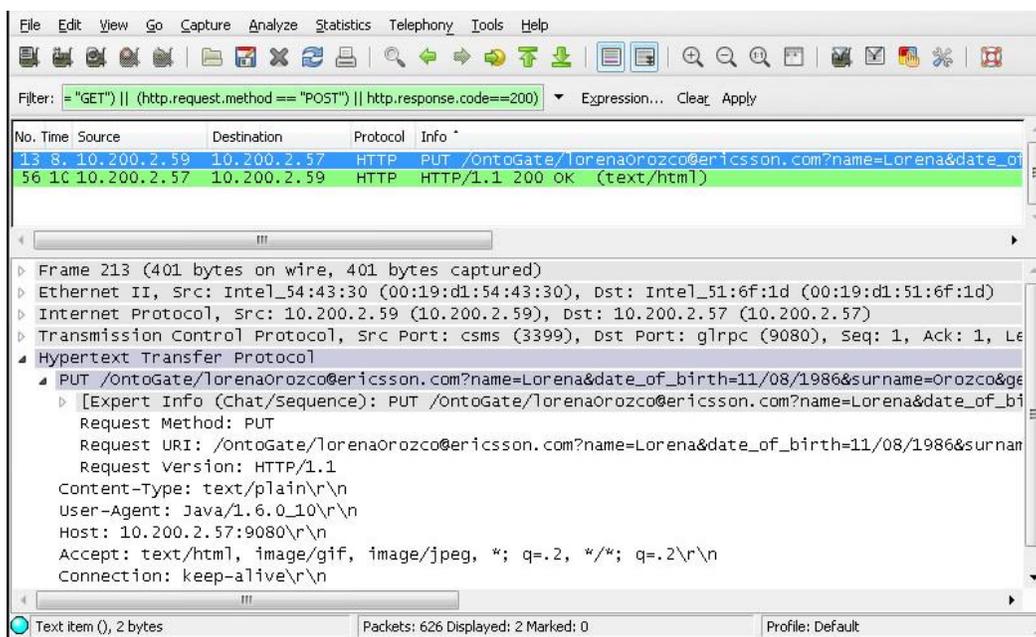


Figura 40 Petición leída desde el lado del SGPU

En la Figura 41 se puede ver el mensaje que llega finalmente al cliente, el campo Content-Type es nuevamente utilizado para identificar el contenido de la respuesta.

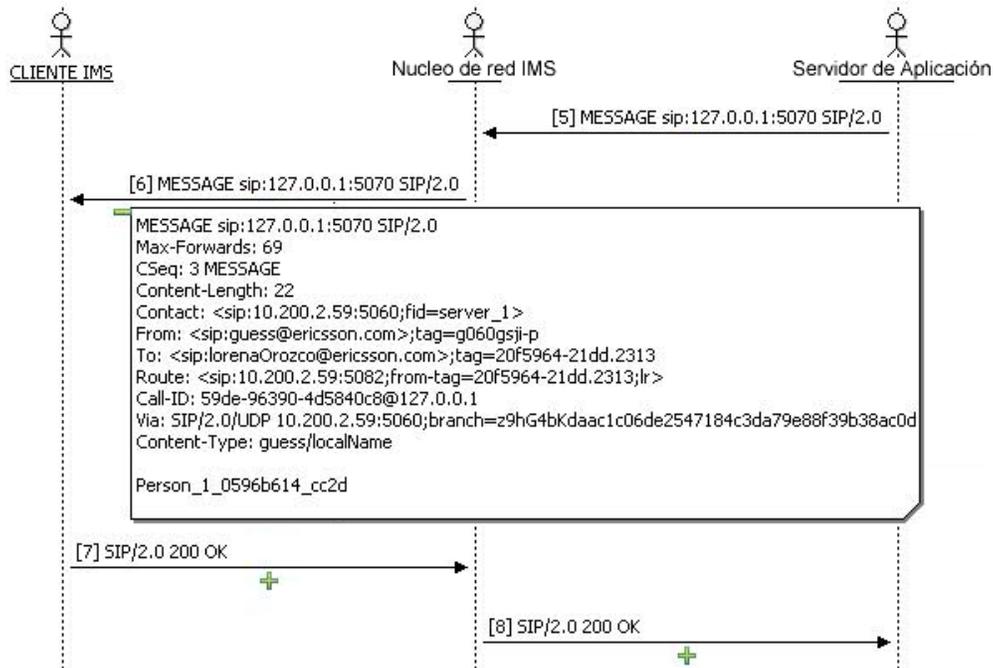


Figura 41 Mensaje recibido en cliente

Figura 42 muestra el mensaje de error que se obtiene para el caso donde se intenta crear una persona ya creada.



Figura 42 Respuesta a una solicitud de creación de una persona existente.

4.10.3 Prueba 2: Verificar la adición de una instancia de la clase LivingConditions a una instancia de la clase persona.

Esta prueba se encarga de verificar la adición de una propiedad a una persona registrada en la ontología. La propiedad llamada LivingConditions agrupa datos tales como, el lugar de residencia, números telefónicos, el tipo de vivienda entre otros. Esta prueba difiere de la prueba número 1, ya que además de crear un elemento, vincula el nuevo conjunto de datos a un perfil de usuario. La Figura 43 muestra el diagrama de secuencia para este caso. El mensaje inicial envía los datos de la propiedad en formato JSON y utiliza la cadena onto/PUT/livingConditions dentro del campo Content-Type indicando la operación que se va a realizar. La respuesta que se obtiene es un mensaje informando que la acción ha sido realizada con éxito.

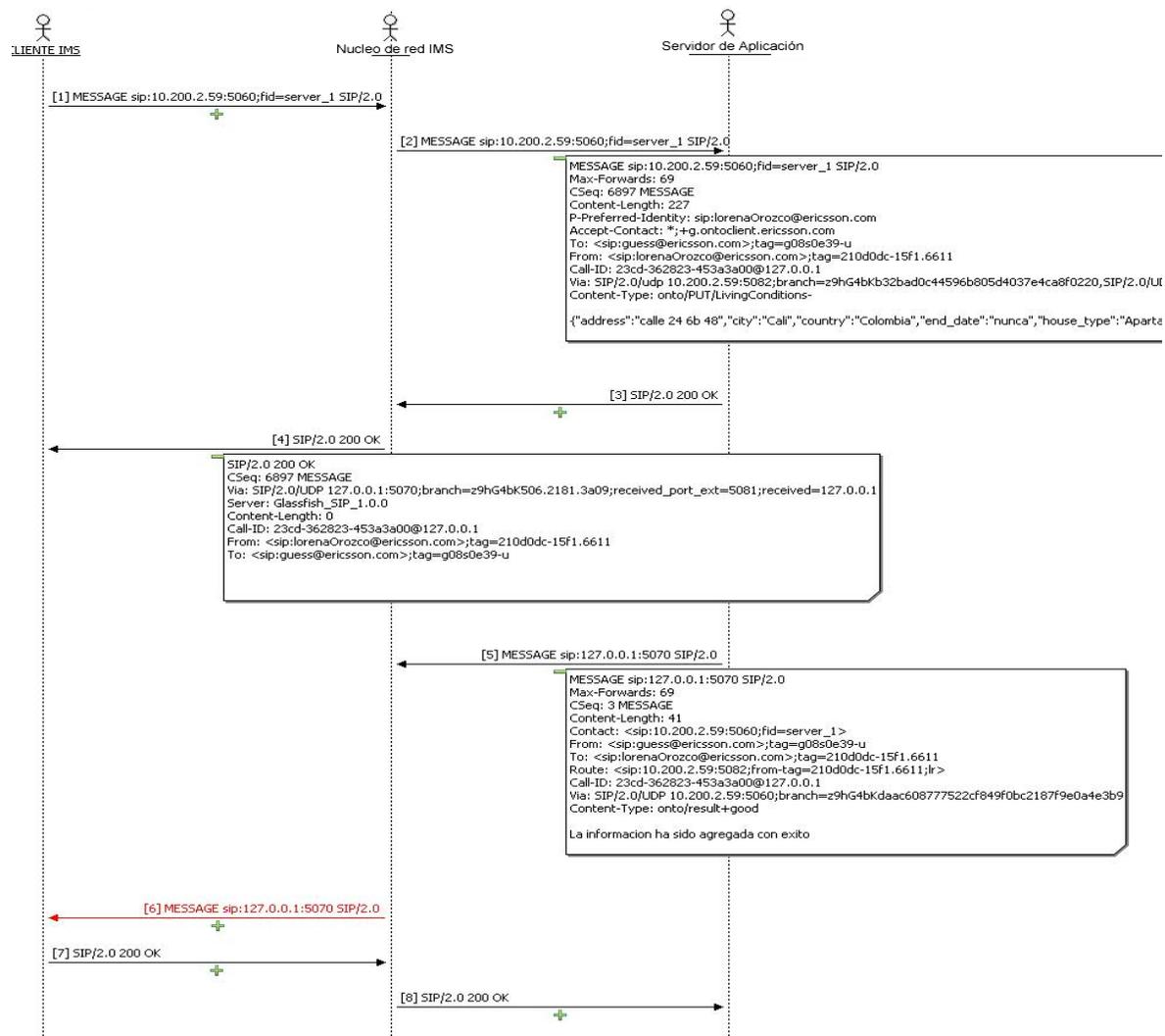


Figura 43 Mensajes de solicitud y respuesta para agregar una propiedad.

El flujo de datos entre el dominio IMS y el SGPU, es observado con el Wireshark. Para agregar una propiedad se realizan dos pasos. El primer paso se encarga de enviar desde el AS la petición para crear una nueva instancia dentro de la ontología, del lado del gestor se realiza la operación y se entrega como resultado un *localname*. La Figura 44 muestra una petición PUT con el tipo de propiedad que se desea crear y los valores que tiene asociados. En seguida se recibe un 200 OK con la respuesta generada por el gestor, en la cual el dato Living_Conditions_15_ac213c0b_5749 corresponde al *localname* de la nueva instancia.

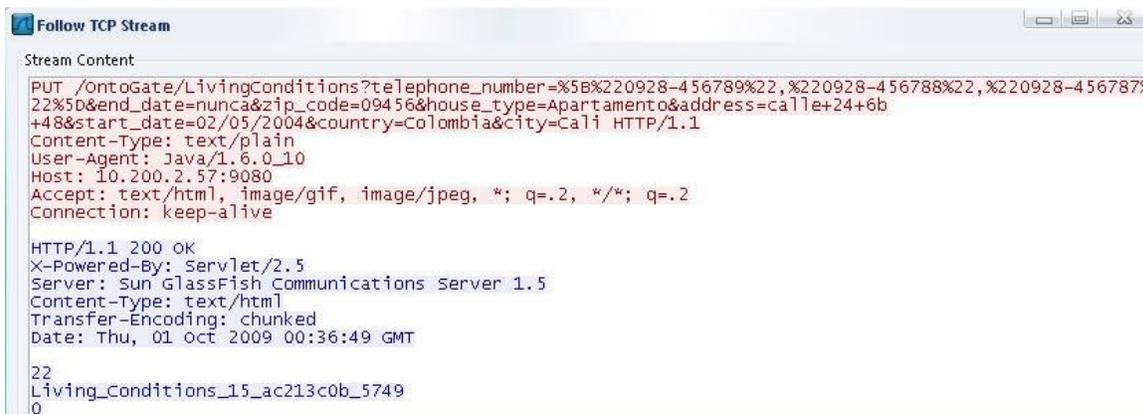


Figura 44 Mensajes asociados a la creación de una instancia.

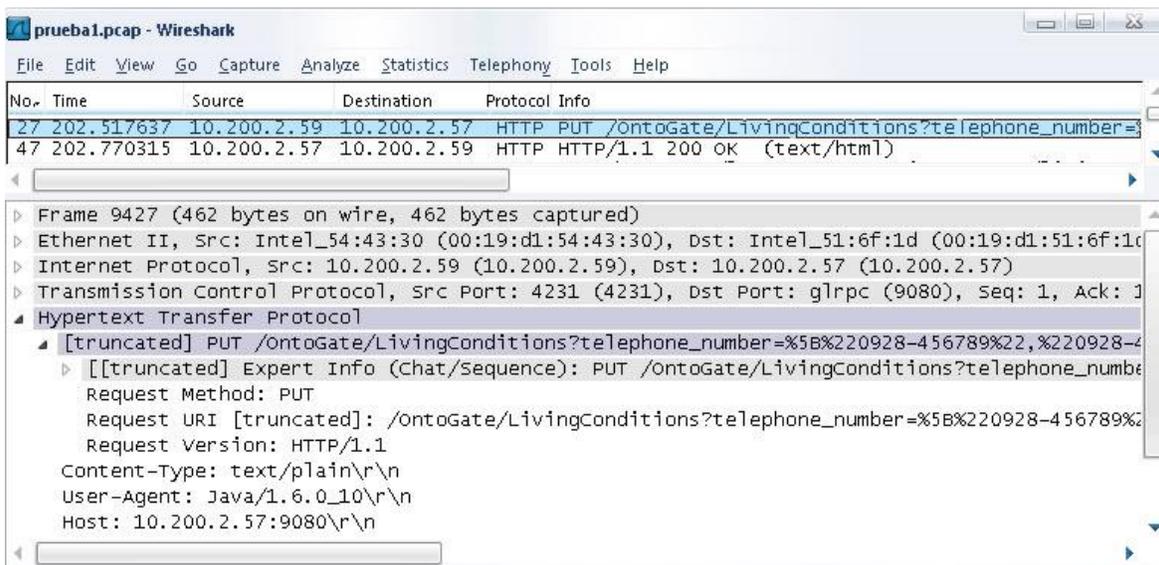


Figura 45 Método PUT capturado en el gestor.

El segundo paso consiste en establecer una asociación entre la propiedad y el perfil de usuario. Para esto es necesario el *localname* retornado por el gestor y la identidad pública de usuario del dominio IMS. Estos datos conforman una *request* URI con la que se creará el nuevo recurso. El resultado de este proceso es un perfil enriquecido que permitirá en el futuro abastecer de información a distintas aplicaciones.

La Figura 46 y la Figura 47 muestran la información que procede del AS y el mensaje que se envía al cliente informando sobre éxito de la operación. La Figura 46 destaca el tiempo de respuesta para la solicitud, el origen y destino de los mensajes y el protocolo utilizado.

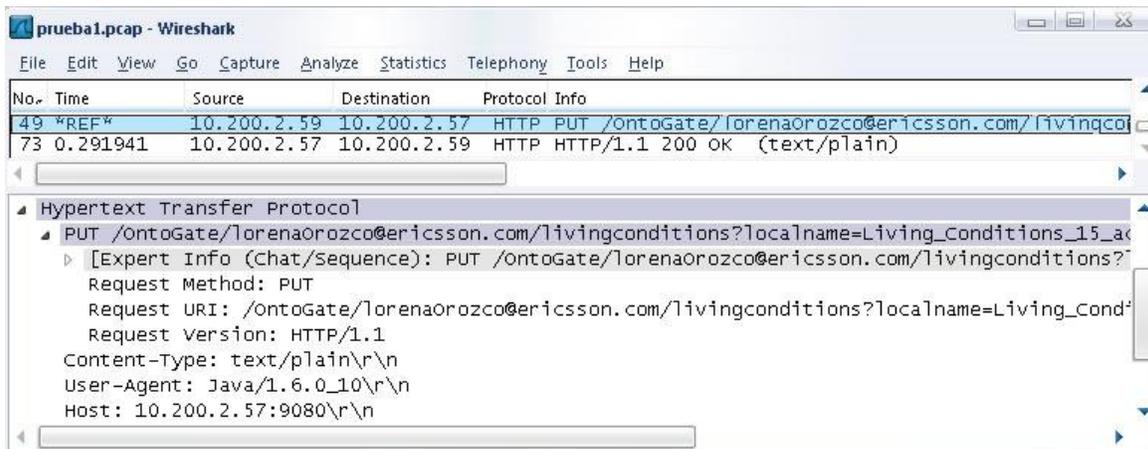


Figura 46 Mensajes HTTP para la adición de una propiedad.

En la Figura 47 se captura el flujo TCP entre los dominios, una solicitud de PUT realizada por el AS y los datos enviados para la misma, entre los que se encuentra: la identidad pública de usuario, el tipo de propiedad que es adicionada y el *localname* que la identifica. En seguida se muestra la respuesta "OK Living_Conditions adicionado" para el caso exitoso.



Figura 47 Flujo TCP para la adición de una propiedad.

4.10.4 Prueba 3: Verificar la recuperación de información de perfil de usuario

Con las pruebas anteriores se verificó la interacción entre el AS y el SGPU (creando y agregando información de perfil de usuario), el objetivo de esta prueba es verificar el acceso a la información almacenada en la ontología con el fin de personalizar un servicio.

Como se dijo anteriormente el campo *Content-Type* del mensaje SIP es utilizado para caracterizar el tipo de petición realizada al AS. En este escenario se crearon tres tipos de peticiones para acceder a los datos almacenados.

- a. onto/GET/sipUri
- b. onto/GET/NombreDeClase
- c. onto/GET/ NombreDeClase-*localname*

El formato de la petición a) es utilizado para obtener todos los datos del perfil que estén asociados a la identidad pública del usuario registrado.

El formato de la petición b) permite recuperar datos simples o múltiples (colecciones de datos) de una misma clase, asociados a un perfil.

El último formato (c) permite acceder a una propiedad de un tipo específico, suministrando el *localname* que la identifica.

Esta prueba utiliza el formato c). La Figura 48 muestra la petición enviada desde el cliente hacia el AS. El mensaje SIP en la posición 1 del diagrama de secuencias contiene entre otros datos el valor del campo *Content-Type* para: onto/GET/LivingConditions-Living_Conditions_15_ac213c0b4.2d4f, de esta forma lo que se espera recuperar es una propiedad del tipo *LivingConditions* cuyo *localname* es: Living_Conditions_15_ac213c0b4.2d4f.

El diagrama de secuencia de la Figura 48 también muestra la respuesta retornada al cliente IMS. El resultado es una cadena en formato JSON que contiene todos los datos de la propiedad *LivingCotions*, de esta forma se puede verificar el resultado de la operación GET.

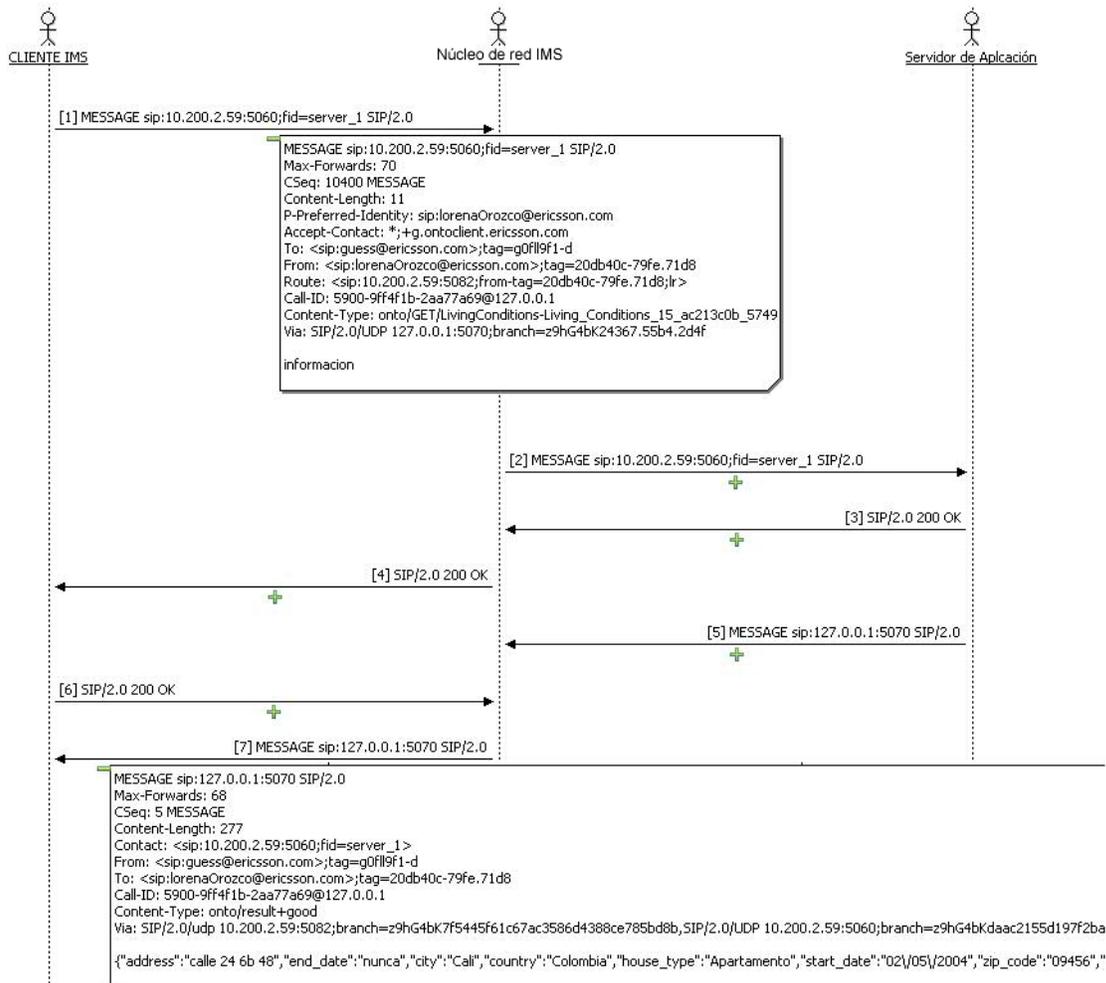


Figura 48 Mensajes para la recuperación de datos en el dominio IMS

La Figura 49 y la Figura 50 muestran la interacción entre los dominios. La Figura 49 se centra en la comunicación HTTP, los tiempos (petición y respuesta) y las direcciones IP de las entidades (AS y el gestor). La Figura 50 por su parte muestra de forma extendida el contenido de los flujos TCP, en primera instancia se puede observar una petición GET realizada desde el AS, con su respectiva respuesta cuyos datos están en formato JSON.

Cada uno de estos diagramas permite comprobar el funcionamiento del SGPU en la recuperación de datos de perfil de usuario. El diagrama de secuencia muestra el flujo SDP/SIP en el dominio IMS, resaltado los datos enviados y los recibidos por el cliente IMS. Los imágenes capturadas con el Wireshark muestran el flujo TCP/HTTP entre el dominio IMS y el SGPU evidenciando el tránsito de datos a través de la interfaz RESTful y comprobando la gestión de datos desde el servicio realizado.

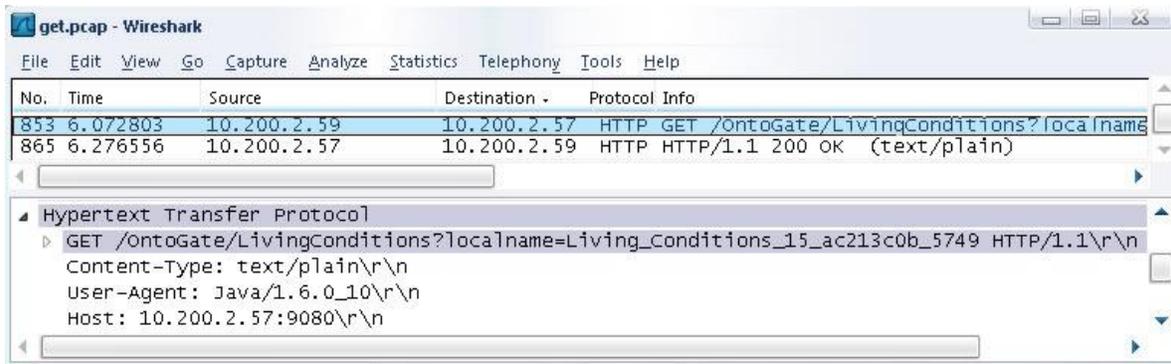


Figura 49 Paquetes HTTP



Figura 50 Flujo TCP para la solicitud de datos.

Los datos y el manejo de los mismos en la personalización de servicios recaen sobre el desarrollador de aplicaciones, el escenario propuesto muestra de forma práctica el funcionamiento del SGPU y comprueba la gestión de información desde un ambiente IMS.

CAPÍTULO 5 LINEAMIENTOS PARA LA PERSONALIZACION DE SERVICIOS EN EL ENTORNO DE IMS.

Dentro de este proyecto se entiende como lineamiento a las directrices, orientaciones o tendencias que establecen algunos aspectos o rasgos, que permiten llevar a cabo una tarea, que específicamente en este desarrollo, se trata de la personalización de servicios en el entorno de IMS [48],[49].

Acorde con esta definición y con lo planteado durante el desarrollo de este proyecto se han determinado los siguientes lineamientos:

- De acuerdo a lo establecido por [26],[27],[28],[31] para la personalización de servicios es necesario el desarrollo de un perfil de usuario que describa sus características, gustos y afinidades. Esta información es clave a la hora de desarrollar servicios que satisfagan de manera particular las necesidades de cada usuario.
- Según lo expuesto en el Capítulo 2, una ontología permite agrupar la semántica asociada al perfil de usuario, haciendo de esta una de las herramientas más poderosas para la captura de la información relacionada con la persona y por lo cual debería tenerse en cuenta a la hora de personalizar un servicio.
- Dado que existen varias ontologías que ya han explorado la creación de perfiles de usuario para la personalización de servicios, la selección de una de las ya existentes, se presenta como una de las mejores alternativas, por cual se debería tener en cuenta las siguientes características:
 - Que su generalidad garantice un alto grado de usabilidad, o en lo posible que realice un consenso entre diversos trabajos para que abarque diferentes puntos de vista.
 - Que cuente con los datos personales relativos a la identidad del usuario, tales como el nombre, apellidos, lugar y fecha de nacimiento, etc.
 - Que realice alguna descripción de los rasgos físicos del usuario como pueden ser edad, altura, peso, sexo, discapacidades, etc.
 - Que ofrezca la posibilidad de capturar algunos datos geográficos del usuario como su lugar de residencia actual y previa, otros lugares visitados, próximos destinos, etc.
 - Se deberían tener presente los datos profesionales como por ejemplo la profesión actual y profesiones anteriores, vinculación actual y pasada con empresas, nivel de responsabilidad, etc.
 - Otra característica de la ontología seleccionada es que debe permitir modelar las relaciones con otros individuos como por ejemplo su familia, amigos, personas a cargo, compañeros de trabajo, vecinos, etc.

- Se deben tener presente los intereses del usuario así como sus preferencias, dado que son de vital importancia a la hora de personalizar un servicio, pues permiten ofrecerlo de una manera más acorde con los gustos de la persona.
 - La experiencia que tiene un usuario en el manejo de una aplicación, es relevante dado que permiten determinar el tipo de interfaz más adecuada a sus habilidades, haciendo la interacción con las aplicaciones algo mucho más amigable.
 - El nivel educativo brinda una idea de los conocimientos que el usuario tiene y del lenguaje que puede ser utilizado, por ejemplo puede usarse para dar mayor profundidad a los contenidos desplegados por una aplicación, o incluso refinar las búsquedas de información que el usuario requiera.
- El lenguaje en el cual este especificada la ontología no es relevante, dado que las herramientas de manipulación de ontologías permiten el intercambio de un lenguaje a otro, en cuyo caso se debería tener en cuenta el uso del lenguaje OWL dado que es el lenguaje recomendado por la W3C para la especificación de ontologías.
 - Dentro de la ontología seleccionada es indispensable establecer algún esquema para los valores de la información que se van a capturar, ya sea por medio del enriquecimiento de conceptos de la ontología o por medio la definición de facetas para cada una de las propiedades. Por ejemplo en el caso de la propiedad nivel de educación se pueden establecer valores como educación primaria, secundaria, universitaria, maestría, etc. para que sean asociados con dicha propiedad, por otra parte para la definición de las características de persona podría enriquecerse dicho concepto por medio de la creación de una clase llamada "Características" cuyas subclases pueden ser del tipo "Físicas" o "Intelectuales".
 - De acuerdo a los trabajos expuestos en [25],[26],[27] se puede establecer que la clase más importante dentro de la definición del perfil de usuario es la persona a la cual se ligan las diferentes características que pueden ser relevantes a la hora de personalizar un servicio.
 - En una ontología o aplicación que gire en torno a la personalización de servicios en el entorno de IMS, es necesario tener presente que la identidad privada y las identidades públicas del usuario son la base fundamental para reconocer al usuario dentro de dicho entorno.
 - Es importante tener en cuenta que los mecanismos para la adquisición de datos del perfil de usuario dependen única y exclusivamente del desarrollador de servicios.
 - Para facilitar la tarea de elaborar un perfil de usuario en el entorno de IMS, se debería hacer uso de algún mecanismo que permita su gestión y distribución entre las diversas aplicaciones que forman parte de dicho entorno.

- Un sistema que pretenda ser usado para la gestión de perfiles de usuario debería contar con las siguientes características: la definición de unas interfaces para el intercambio de información que garanticen la integridad de los datos, una alta disponibilidad, un bajo consumo de ancho de banda, así como el establecimiento de una arquitectura que permita una rápida respuesta a las peticiones recibidas.
- Para una aplicación que haga parte de la arquitectura de IMS, el consumo de un servicio que gestione los perfiles de usuario debería no afectar su normal funcionamiento, para lo cual es indispensable delegar dicha tarea a un módulo específico dentro de la aplicación, que permita mantener la independencia con respecto a terceros y que brinde la posibilidad de realizar un intercambio por mejores fuentes de información de personalización que en un futuro seguramente harán parte del núcleo de IMS.
- Los servicios ofrecidos en el entorno de IMS, se comunican básicamente por medio del intercambio de mensajes SIP, los cuales pueden ser utilizados para enviar y recibir información de personalización entre los clientes y aplicaciones, por medio de la definición de un tipo de contenido (Content-Type) especializado para dicha actividad.

CAPÍTULO 6 APORTES, CONCLUSIONES Y TRABAJOS FUTUROS

6.1 APORTES

El desarrollo de este proyecto trajo los siguientes aportes:

- La definición de una arquitectura de referencia que se describe en la sección 3.4 , que permite la interacción de una ontología con el dominio de IMS.
- La arquitectura propuesta abre las puertas para que ontologías desarrolladas en otras áreas puedan ser utilizadas del mismo modo como se hizo en este proyecto.
- Durante el desarrollo del proyecto se construyó un conjunto de interfaces para el acceso a las clases que hacen parte de la ontología.
- Las interfaces desarrolladas brindan la posibilidad de ser reutilizadas, dado que es posible realizar una implementación de dichas interfaces con otros APIs para la manipulación de ontologías o incluso el desarrollo de un modelo basado en bases datos, que permiten actualizar y mejorar el sistema.
- El servicio web RESTful desarrollado para la gestión de perfiles de usuario, puede ser usado tanto en entorno de IMS, como en otros ambientes basados en el protocolo HTTP, por lo cual se aporta un servicio que permite el acceso ubicuo a la información del perfil del usuario.
- Durante la implementación de referencia del proyecto se desarrollaron aplicaciones y servicios valiéndose de diversas herramientas, cuya experiencia en configuración y manejo son de gran aporte a la comunidad académica.
- El uso del entorno de desarrollo ofrecido por el Ericsson Service Development Studio (SDS), durante el desarrollo de este proyecto, abre las puertas a una herramienta con un gran soporte y respaldo, que hasta el momento ningún grupo de investigación al interior de la facultad ha aprovechado en sus pruebas y desarrollos.
- Otro de los aportes realizados por el proyecto es el desarrollo de un sistema que facilita la gestión del perfil de usuario, el cual juega un papel fundamental en la creación de aplicaciones personalizadas las cuales mejoran la experiencia del usuario.
- El desarrollo de este trabajo permitió la identificación de algunos lineamientos para la personalización de servicios en el entorno de IMS, los cuales son una herramienta/guía útil para la creación de aplicaciones personalizadas dentro de dicho entorno.
- Para este proyecto se creó un cliente IMS totalmente funcional desde cero, se creó un servicio desplegado sobre un servidor de aplicaciones en el dominio IMS utilizando como base la tecnología SIP-Servlet, se realizó la comunicación entre servidores de aplicación,

complejizando el servicio propuesto y se accedió a una base de conocimiento para la personalización del servicio.

6.2 CONCLUSIONES

Después del desarrollo de este proyecto se han llegado a las siguientes conclusiones:

- En el capítulo 1 tras realizar una descripción del estado de arte de la personalización de servicios y abordar su estudio en el entorno de IMS, se determinó que aunque en esta arquitectura existe un perfil de usuario, la información que se maneja no realiza la captura de datos como los intereses, preferencias o la experiencia de un usuario.
- El desarrollo de este proyecto se enmarco en la siguiente pregunta de investigación: ¿Pueden las ontologías facilitar la gestión de la información de perfiles de usuario en un entorno IMS de manera tal que se permita su reutilización?, la cual tras el estudio realizado, y la construcción de un servicio para la gestión de perfiles de usuario basado en una ontología, se comprobó que estas herramientas pueden ser utilizadas para facilitar la gestión de información de personalización y brindar una base de conocimiento compartida entre diversas aplicaciones que fácilmente puede ser reutilizada.
- Durante el desarrollo de este proyecto se adaptó la ontología propuesta en [27] agregándole las características adecuadas para su correcto desempeño y uso dentro del entorno de IMS, cumpliendo con el objetivo de proponer una ontología en el dominio de la personalización de servicios.
- La ambigüedad que puede presentarse en el dominio de la personalización de servicios en el entorno de IMS, se reduce con la introducción de una ontología de perfiles de usuario, dado que la información a capturar cuenta con una estructura organizada y de fácil comprensión, lo cual facilita la realización de consultas más puntuales y rápidas.
- Un repositorio Ontológico del perfil del usuario facilita la captura de sus datos haciendo de esta una tarea menos ardua, dado que dicho trabajo será distribuido entre las diversas aplicaciones que hacen uso del servicio propuesto.
- El servicio ofrecido por el gestor de perfiles de usuario propuesto, brinda un repositorio de información que permite a las aplicaciones dentro del entorno de IMS, compartir la información que cada una genera, por medio de la definición de un dominio compartido basado en ontologías.
- El uso de ontologías dentro del entorno de telecomunicaciones así como en otras áreas permite la creación de modelos con diferentes niveles de abstracción lo cual facilita la tarea de enriquecerlas de acuerdo al dominio donde quieran ser aplicadas.
- La implementación de servicios basados en la arquitectura REST, tiene un mínimo de complejidad, y las peticiones que se realizan, contienen toda la información necesaria para

obtener una respuesta. Esta característica debería tenerse en cuenta a la hora de implementar un servicio usado por otras aplicaciones dentro del entorno de IMS, pues permite ofrecer un servicio ágil y dinámico con un bajo consumo de ancho de banda.

- El Gestor de perfiles de usuario desarrollado en este proyecto, ofrece unas interfaces de comunicación acordes con la arquitectura REST, por lo cual el servicio ofrecido brinda un conjunto de métodos que pueden ser reutilizados por otras aplicaciones que formen o no parte de la arquitectura de IMS, permitiendo así que sistemas desarrollados para otras plataformas también tengan a su disposición la información del perfil de usuario, por ejemplo, una aplicación en un dispositivo móvil por medio de la utilización de librerías como JSON y una conexión HTTP estaría en capacidad de manipular los datos que ofrece el servicio desarrollado.
- Se determinó que Protégé es una de las mejores herramientas para la manipulación de ontologías dado que ofrece un completo juego de utilidades que permiten la integración de ontologías en diferentes procesos.
- JUnit permite la organización de test que las aplicaciones deben pasar, convirtiéndolo en un API fundamental a la hora de continuar con el desarrollo del gestor de perfiles de usuario propuesto, dado que ya se ha establecido su funcionalidad y cualquier modificación futura puede valerse de los test que ya se estructuraron para verificar que los cambios realizados no afectan el comportamiento normal del servicio ofrecido.
- Con respecto a los tiempos de respuesta que se obtuvieron en las pruebas realizadas al gestor de perfiles de usuario, se puede afirmar que aun no son comparables con las que se pueden obtener con una base de datos, dado que es necesario un mayor procesamiento de la información, además que el desarrollo de una estructura de almacenamiento masiva para las ontologías aun se encuentra en proceso de desarrollo.
- La utilización del SDS de Ericsson en el desarrollo del proyecto, permitió aclarar conceptos teóricos relacionados con las entidades que se encuentran al interior del Núcleo de red IMS, así como también conceptos aplicados en el momento de configurar un servicio y en tareas como: crear un perfil de usuario, un perfil de servicio, los criterios de filtrado iniciales, los iniciadores de punto de servicio, etc.
- La arquitectura planteada en la sección 3.4 abre el camino para el desarrollo de otras aplicaciones que requieran la interacción con otros servicios dentro de la arquitectura de IMS, dado que el prototipo propuesto se desempeña de la forma esperada sobre el servidor de aplicaciones utilizado.
- Es importante notar que la arquitectura propuesta para lograr la integración de una ontología que gestione la información de perfiles de usuario, no realiza alteraciones de la arquitectura de IMS, dado que se define como un servicio que puede o no ser utilizado por otras aplicaciones.

- El patrón arquitectónico utilizado por Protégé para la generación de los Beans disminuye enormemente el trabajo de la manipulación de una ontología, pues ofrece métodos para la creación de instancias de cada una de las Clases, a la vez que permite acceder a los valores de cada una de sus propiedades. Dicho patrón, permite la implementación del modelo ontológico por medio de otras APIs e incluso hace posible el uso de bases de datos. Sin embargo, aún no se realiza un buen manejo de datos booleanos así como de algunas excepciones, por lo cual es necesaria la modificación del código generado, haciendo de su mantenimiento una tarea de cuidado.

6.3 TRABAJOS FUTUROS

A continuación se presentan algunos trabajos que se pueden considerar para continuar y aportar a la propuesta aquí presentada.

- Establecer un sistema que permita generar una representación ontológica del contexto para adaptar dinámicamente la selección de atributos según las preferencias del usuario. La información relativa al contexto del usuario puede ayudar a mejorar la comunicación hombre-máquina, adaptar los contenidos a las habilidades personales de los usuarios y al tipo de dispositivo, tener en cuenta factores como la localización del usuario y los recursos que tiene cercanos, sacar provecho de información temporal y espacial para el cálculo de rutas en entornos controlados, etc.
- Entre las mejoras que se pueden realizar al presente proyecto se encuentra el establecimiento de un mecanismo para la publicación del servicio, aquí tecnologías como OWL-S (Semantic Markup for Web Services, Marcado Semántico para servicios Web) pueden jugar un papel fundamental a la hora fortalecer el presente desarrollo ya que se podría realizar un descubierto automático, se podrían componer servicios más complejos y se podría realizar una invocación del servicio con un grado alto de automatización.
- Crear clientes IMS para servicios estandarizados y no estandarizados capaces de capturar información de forma automática asociada al contexto y preferencias del usuario, donde no sea necesaria la interacción explícita usuario-sistema.
- El gestor de perfiles de usuario puede mejorarse por medio del desarrollo de un esquema de funcionamiento en el cual este se adapte de manera automática a los cambios que pueden realizarse a la ontología en cuanto a sus clases, propiedades y relaciones.
- El presente desarrollo se centro en el servidor de aplicaciones Sailfin, como trabajo futuro puede proponerse la puesta a prueba del sistema propuesto dentro de otros servidores de aplicación.
- Adicionar un mecanismo que permita la sincronización de la información del perfil de usuario, con aquellas aplicaciones suscritas a dicho servicio.
- Las ontologías siempre pueden ser adaptadas y mejoradoras para diferentes entornos, por lo cual la ontología propuesta en este proyecto puede servir de base para el desarrollo de nuevas

soluciones y aplicaciones que se centren en los perfiles de usuario, en particular en el dominio de IMS y la personalización de servicios de telecomunicaciones.

- Agregar elementos de seguridad al servicio RESTful, por medio de la definición de un mecanismo de autenticación y una comunicación basada en el protocolo HTTPS. A la vez se puede pensar en un mecanismo para establecer diferentes permisos que restrinjan el acceso a la información del perfil de usuario, permitiendo que solo aplicaciones confiables realicen cambios en la misma.

REFERENCIAS

- [1] Camarillo, G., and Garcia-Martin, M.A. *"The 3G IP Multimedia Subsystem (IMS), Merging the Internet and the Cellular Worlds"*, Wiley, 2006
- [2] 3GPP, *"Service requirements for the Internet Protocol (IP) multimedia core network subsystem"*, Stage 1, TS 22.228, 3rd Generation Partnership Project (3GPP)
- [3] Bertin, E., *"Stakes of Next-Generation Communication Services"*, aict-sapir-elete, páginas 15-20, Advanced Industrial Conference on Telecommunications/Service Assurance with Partial and Intermittent Resources Conference/ELearning on Telecommunications Workshop (AICT/SAPIR/ELETE'05), 2005.
- [4] Travis Russell. *"The IP Multimedia Subsystem (IMS):Session Control and Other Network Operations"*. Editorial McGraw-Hill. 2008.
- [5] *IP Multimedia Subsystem (IMS) Handbook*. Editado por Syed A. Ahson, Mohammad Ilyas. Editorial Taylor & Francis Group.2009
- [6] 3GPP. *"Network architecture"*, Release 8, TS 23.002, 3rd Generation Partnership Project (3GPP)
- [7] Juan Carlos Montoya Mendoza, Edwin Montoya Múnera. *"Servicios convergentes en redes de próxima generación"* Departamento de Informática y Sistemas. Universidad EAFIT, 2006.
- [8] S. Ceri, P. Fraternali, and S. Paraboschi. *"Data-Driven One-to-One Web Site Generation for Data- Intensive Applications"*, Procedente de el VLDB'99, 02 1999.
- [9] P. Fraternali. *"Tools and Approaches for Developing Data-Intensive Web Applications: a Survey"*. ACM Computing Surveys, 2000.
- [10] G. Kappel, W. Retschitzegger, and W. Schwinger. *"Modeling Customizable Web Applications – A Requirement's Perspective"*, 2000 Conferencia Internacional de Kioto sobre librerías digitales, Noviembre 2000.
- [11] ACM, *"Communications of the ACM: Special issue on Personalization"*, volume 43, Agosto 2000.
- [12] J. Pavón. *"Personalización de servicios en la Web"*. ZOCO 2001. <http://tdg.lsi.us.es/zoco/res/ppt/pavon.zip>, 2001.
- [13] Personalization Consortium, *"Personalization Consortium Home Page"*. <http://www.personalization.org>, Enero 2002.
- [14] Amazon, *"Web de Amazon"*. <http://www.amazon.co.uk>, 2009
- [15] M. Svensson, K. Höök, J. Laaksohlahti, and A. Waern. *"Social Navigation of Food Recipes"*, SIGCHI'01, 03 2001
- [16] E. Gamma, R. Helm, R. Johnson, and J. Vlissides. *"Design Patterns: Elements of Reusable Object-Oriented Software"*. Addison Wesley, 1995.
- [17] Francisco Javier Honrubia López. *"Introducción a las Ontologías"*. Escuela Universitaria Politécnica de Albacete. España. 2002.
- [18] R. Neches, R.E. Fikes, T. Finin, T.R Gruber. T. Senador. W.R. Swartout *"Enabling technology for knowledge sharing"*. Magazín de Inteligencia Artificial. Páginas 36-56, 1993.
- [19] Gruber, T. R. *"A Translation Approach to Portable Ontologies"*. Cit. pp 199-220, Knowledge Acquisition vol. 5, 1993.
- [20] Borst, W.N. *"Construction of engineering ontologies for knowledge sharing and reuse"*, CTIT. Universidad de Twente, Países Bajos, Enschede. 1997

- [21] Juan Pablo Palacios Escalona. "Modelo de unificación semántica de ontologías, aplicado al dominio de los archivos digitales " Departamento de Ingeniería de Sistemas Telemáticos. 2005
- [22] Natalya Fridman Noy, Deborah McGuinness. "Desarrollo de Ontologías-101: Guía para crear tu Primera Ontología". Reporte técnico KSL-01-05. Laboratorio de Sistemas de Conocimiento de Stanford, Universidad de Stanford. Septiembre 2005
- [23] Imen Grida Ben Yahia, Emmanuel Bertin, Noel Crespi. "Ontology-based Management Systems for the Next Generation Services: State-of-the-Art," icns, página 40, Conferencia Internacional sobre Redes y Servicios (ICNS '07, *International Conference on Networking and Services*), 2007.
- [24] Patron Factory disponible en: <http://davidasorey.net/80/patrones-diseno-para-torpes-1>.
- [25] Patron Singleton disponible en <http://es.wikipedia.org/wiki/Singleton>.
- [26] José Javier Samper Zapater. "Ontologías para servicios web semánticos de Información de tráfico". Departamento de Informática. Universidad de Valencia. Junio 2005
- [27] Maria Golemati, Akriki Katifori, Costas Vassilakis, George Lepouras, Constantin Halatsis "Creating an Ontology for the User Profile: Method and Applications". Primera Conferencia Internacional IEEE sobre los desafíos en la Investigación en Ciencias de la Información (RCIS -Research Challenges in Information Science), Marruecos 2007.
- [28] Diego Berruta. "Especificación de ontologías avanzadas para la descripción del perfil de usuario". Fundación CTIC (Centro Tecnológico de la información y la Comunicación). Madrid. Noviembre 2006.
- [29] Dan Brickley, Libby Miller. "FOAF vocabulary specification 0.91". Noviembre 2007. Disponible en: <http://xmlns.com/foaf/spec/>
- [30] Vivi Katifori,, Antonella Poggi,. Monica Scannapieco, Tiziana Catarci, Yannis Ioannidis. "OntoPIM: how to rely on a personal ontology for Personal Information Management". Primer taller sobre Semántica de escritorio.. Atenas Grecia. 2005.
- [31] Joana Trajkova, Susan Gauch, "Improving Ontology-based User Profiles", RIAO (*Recherche d'Information Assistée par Ordinateur*), Universidad de Avignon (Vaucluse), Francia. Páginas 380-389. Abril 26-28, 2004.
- [32] Steve Lawrence. "Context in web search". IEEE Boletín de Ingeniería de Información, Volumen 23, número 3. Páginas 25-32.
- [33] Susan Gauch, Jason Chaffee, Alexander Pretschner. "Ontology-Based User Profiles for Search and Browsing". Modelado de usuario e interacción adaptada: El Diario de investigación en la personalización, Edición especial sobre modelado de usuarios para la web y de recuperación de información hipermedia. 2003
- [34] Jaime Teevan, Susan T. Dumais, Eric Horvitz. "Personalizing Search via Automated Analysis of Interests and Activities". Acta del SIGIR (*Special Interest Group on Information Retrieval*) 2005, Agosto 2005.
- [35] David E. Bakken. "Middleware", Enciclopedia de sistemas distribuidos, Kluwer Academic Press, 2003
- [36] P. Weik, D. Vingarzan, T. Magedanz. "Design and implementation of an open IMS core. Mobility Aware Technologies and Applications", Segundo taller Internacional MATA . Berlin 2005.
- [37] P. Weik, D. Vingarzan, T. Magedanz. "Towards an open source IMS core system enabling rapid prototyping of NGN services" 3er Taller Internacional sobre *Middleware* en redes de nueva generación. Portugal. Mayo 2006.
- [38] ETSI. "Telecommunications and Internet Converged Services and Protocols for Advanced Networking (TISPAN)" .TS 186 008. IMS/NGN Pruebas comparativas de rendimiento.
- [39] Laboratorio de Interoperabilidad. Universidad de Nueva Hampshire. <http://www.iol.unh.edu>
- [40] IMS Advanced Research Cluster for Services. Disponible en <http://www.ims-arcs.org>

- [41] The UCT IMS client. Disponible en <http://uctimsclient.berlios.de>
- [42] The IMS communicator. Disponible en <http://imscommunicator.berlios.de>
- [43] Leonardo De Seta, "Introducción a los Servicios Web RESTful", Septiembre 2009 <http://www.dosideas.com/>
- [44] Rafael Navarro Maset, "REST vs Web Services", 2007. Disponible en:
<http://users.dsic.upv.es/~rnavarro/NewWeb/docs/RestVsWebServices.pdf>
- [45] Isaac Gutiérrez Gómez y Salvador Otón Tortosa, "Arquitecturas Orientadas a Servicios". Universidad del Alcalá. Departamento de Ciencias de la Computación. España. Disponible en:
<http://ftp.informatik.rwth-aachen.de/Publications/CEUR-WS/ol-132/paper09.pdf>
- [46] Elena Sánchez, Martín Ruiz y Jorge Rodríguez, "Acceso Dinámico a Servicios de una Infraestructura Web desde Teléfonos Móviles". Disponible en :
<http://www.w3c.es/Eventos/2007/MWeb/Comunicaciones/Papers/p3.pdf>
- [47] Guillermo Gonzales. JUnit – Testeo de proyectos. Universidad Nacional de Asunción. Agosto 2008.
- [48] Real Academia de la lengua Española. Disponible en <http://rae.es/rae.html> [Accedida el 20 de Octubre de 2009]
- [49] Qué significa lineamiento. <http://definicion.de/lineamiento/> [Accedida el 20 de Octubre de 2009]