ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL.

Facultad de Ingeniería en Electricidad y Computación

“Diseño e implementación de un Web API para el Sistema Interactivo de Desarrollo para el Web (SIDWeb)”

INFORME DE PROYECTO DE GRADUACIÓN

Previa a la obtención del Título de:

INGENIERO EN CIENCIAS COMPUTACIONALES

ESPECIALIZACIÓN SISTEMAS DE INFORMACIÓN

Presentado por:

Giancarlo Vera Rivera

GUAYAQUIL – ECUADOR

2012

AGRADECIMIENTO

Agradezco a mi familia por siempre haberme brindado el apoyo necesario para conseguir mis metas, al Dr. Enrique Peláez Jarrín y a todo el equipo que conforma el Centro de Tecnologías de Información por el apoyo brindado en el desarrollo de este trabajo

TRIBUNAL DE SUSTENTACIÓN

Ing. Jorge Aragundi R.

Subdecano de la FIEC

PRESIDENTE

Dr. Xavier Ochoa C.

DIRECTOR DEL PROYECTO DE GRADUACIÓN

Dra. Katherine Chiluiza

VOCAL PRINCIPAL

DECLARACIÓN EXPRESA

"La responsabilidad del contenido de este trabajo de Grado, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL"

(Reglamento de Graduación de la ESPOL)

_____

Giancarlo Vera Rivera

RESUMEN

El presente trabajo presenta el diseño y la implementación de una interfaz de comunicación entre SIDWeb 4 con otras aplicaciones, con la cuál se espera proveer un mecanismo de interacción con los contenidos y funcionalidades del software en cuestión, se presenta también las motivaciones y estructuras de algunas soluciones implementadas por compañías reconocidas en el medio. Como parte de las pruebas de la implementación este trabajo presenta el funcionamiento de un aplicativo que hace uso de la interfaz de comunicación desarrollada.

INDICE GENERAL

INDICE DE FIGURAS

INDICE DE TABLAS

ABREVIATURAS

API	Application Program Interface
AWS	Amazon Web Services
CAS	Central Authentication Service
CTI	Centro de Tecnologías de Información
ESPOL	Escuela Superior Politécnica del Litoral
FTP	File Transfer Protocol
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
iOS	iPhone Operating System
JSON	JavaScript Object Notation
LDAP	Lightweight Directory Access Protocol
LMS	Learning Management System
OAuth	Open Authorization
ORM	Object Relational Mapper
PAAPI	Product Advertising API
PHP	Hypertext Preprocessor
REST	Representational State Transfer
SIDWeb	Sistema Interactivo de Desarrollo para el Web
SOAP	Simple Object Access Protocol
SSO	Single Sign On
URI	Uniform Resource Identifier

INTRODUCCIÓN

La idea de implementación de este trabajo se origino luego de una consulta realizada al Dr. Xavier Ochoa sobre la posibilidad de implementar una versión móvil del Sistema Interactivo de Desarrollo para el Web (SIDWeb), sobre lo cual el Dr. Ochoa propuso la implementación de un API que expusiera las operaciones del sistema para posteriormente utilizarlo en implementaciones de aplicativos móviles.

SIDWeb es un software de aprendizaje electrónico, de código abierto desarrollado por el Centro de Tecnologías de Información, este software es usado ampliamente por la comunidad de profesores y estudiantes de la ESPOL para el manejo de contenidos y recursos adicionales de clases dictadas por la universidad.

El objetivo principal es implementar un API de comunicaciones que permita la interacción de aplicaciones externas con el software SIDWeb 4.

El desarrollo de este trabajo ha sido basado en la versión 4 de SIDWeb, para el momento del desarrollo SIDWeb 4 está en etapa Beta lo que significa que se encuentra en fase de prueba de funcionalidades con un grupo reducido de usuarios.

CAPITULO 1

1. ANÁLISIS DEL PROBLEMA

En este capítulo se realiza un análisis de las limitaciones de SIDWeb con respecto al acceso de sus operaciones en dispositivos móviles y en otras plataformas.

1.1. Acceso de SIDWeb en dispositivos móviles

Actualmente el acceso al sitio Web de SIDWeb mediante dispositivos móviles posee limitaciones inherentes al ancho de banda, formato de presentación de páginas, compatibilidad de navegadores y usabilidad de la interfaz.

Entre algunas de las limitaciones en el acceso sobre dispositivos móviles se pueden mencionar las siguientes:

- Aumento de tiempos de espera para cargar contenidos, esto debido a que la velocidad de la conexión de los dispositivos móviles es comúnmente menor, otro factor que afecta el tiempo

de respuesta es la limitada capacidad de procesamiento de los dispositivos móviles, esto provoca que el renderizado de la página sea más lento (1).

- Los tamaños de las pantallas en dispositivos móviles son reducidos. En general las interfaces móviles son más restrictivas que las interfaces de escritorio en relación con capacidad de procesamiento, tamaño de pantalla, y atención del usuario (2), en la figura 1.1 se muestra el tamaño reducido de las pantallas de dispositivos móviles esto reduce la usabilidad de la interfaz, puesto que es necesario realizar más desplazamientos verticales, horizontales y acercamientos de la pantalla para acceder a los diferentes elementos (3).

Figura 1.1: SIDWeb 4 desde Safari para iPhone

- Muchos de los navegadores en teléfonos móviles pueden carecer de soporte para ciertas características tales como ejecución de código JavaScript, u objetos Flash [1].
- Los planes de datos para dispositivos móviles limitan la cantidad de datos que se pueden transferir, de modo que, si se traspasa el límite contratado se puede incurrir en costos adicionales [2] [3].

1.2. Acceso de SIDWeb en otras plataformas

Actualmente las operaciones de SIDWeb no están expuestas para plataformas diferentes a navegadores, debido a que, los contenidos generados por el software solo son legibles por navegadores Web, esto provoca que no sea posible que otros sistemas se comuniquen con el SIDWeb para propósitos de integración o intercambio de información, esto representa una limitante si se considera la opción del desarrollo de aplicaciones externas que interactúen con SIDWeb

---

[1] Publicación de Steve Jobs acerca de la negativa de incluir Flash en iPhone y iPad http://www.apple.com/hotnews/thoughts-on-flash/

[2] Tarifas de planes Claro http://claro.com.ec/wps/portal/ec/pc/personas/movil/planes/postpago/planes-ideal-datos

[3] Tarifas de planes MoviStar http://www.movistar.com.ec/site/movil-personas/correo-e-internet/internet-en-el-movil.html

1.3. Justificación

Dado el auge de los dispositivos móviles, tales como teléfonos inteligentes, computadoras Tablet, entre otros, es preciso contar con una interfaz que brinde acceso a las operaciones del sistema con una arquitectura de comunicaciones estándar, lo cual amplía la posibilidad a desarrolladores de desenvolver aplicaciones que se comuniquen con SIDWeb independientemente de la plataforma donde se desarrolle la aplicación, esta interfaz debería también permitir el desarrollo de aplicaciones externas que se integren con las operaciones del sistema.

Otra motivación importante del desarrollo de una interfaz de comunicaciones es la posibilidad del desarrollo de Mashups, estas aplicaciones Web usan más de un recurso de información para crear nuevos servicios o mejorar la experiencia del usuario (4), generalmente los Mashups recogen información de las fuentes usando Web APIs.

1.4. Soluciones Existentes

Existen un sin número de compañías que han puesto a disposición de sus clientes lo que se conoce como Interfaz de Programación de Aplicaciones (API por sus siglas en inglés), un API es una interfaz por la cual dos aplicaciones se comunican en un lenguaje que ambas conocen, los APIs son utilizados para exponer las operaciones de sus

sistemas sin la necesidad de utilizar una interfaz Web, compañías como Google, Facebook, Amazon, Twitter, eBay entre otras han desarrollado APIs de sus productos.

Las motivaciones de estos APIs han sido la obtención directa de utilidades a través de costos aplicados al uso, motivar el desarrollo de aplicaciones basadas en los servicios prestados por el API, aumento de visibilidad de la marca, o en algunos casos como respuesta a mecanismos de hacking para incorporar funcionalidades de los servicios a otros sitios de manera irregular (5).

Uno de los aspectos importantes de la apertura de un API son la documentación y herramientas que se provean para facilitar el uso del API, entre la documentación necesaria se encuentra la especificación de las operaciones del API, ejemplos de consumo de las operaciones, y descripción de escenarios de error, con respecto a las herramientas se puede mencionar consolas de pruebas que permiten realizar invocaciones al API [1].

[1] Google Code Playground http://code.google.com/apis/ajax/playground/

A continuación se exploran un grupo de soluciones existentes clasificadas en soluciones para sistemas de manejo del conocimiento (LMS por sus siglas en inglés) y otras soluciones para sistemas con distintos propósitos.

1.4.1. Soluciones para sistemas LMS

Los sistemas LMS son sistemas de manejo del conocimiento que permiten realizar tareas de documentación, rastreo, y generación de reportes de programas de aprendizaje, aulas de clase, eventos en línea, aprendizaje electrónico y contenidos de entrenamientos [1].

Sistemas LMS populares como Moodle y Canvas han implementado APIs para exponer parte de las operaciones del software que ofrecen a sistemas externos.

1.4.1.1 Moodle API

Moodle es un software popular de aprendizaje electrónico (e-learning) para manejo de cursos presenciales, semi-presenciales o virtuales.

[1] Documentación Canvas REST API (https://canvas.instructure.com/doc/api/index.html)

El API de Moodle expone ciertas funciones del software para ser utilizadas por aplicaciones externas, entre las funciones expuestas por defecto en el API constan operaciones para: crear/eliminar grupos, crear/eliminar/actualizar usuarios, consultar usuarios por su identificación, entre otras, sumado a esto el software da la posibilidad de ampliar las operaciones expuestas o crear funciones personalizadas para el API.

Las operaciones por defecto en Moodle API requieren como mecanismo de autenticación incluir como parámetros el usuario y contraseña del cliente para el cual se realiza el requerimiento.

1.4.1.2 Canvas API

Canvas al igual que Moodle es un sistema de aprendizaje electrónico, este sistema está desarrollado en Ruby on Rails y también cuenta con un API para exponer algunas de sus operaciones a servicios externos.

El API de Canvas esta implementado bajo las normas de diseño REST, el API de Canvas en comúnmente referido como REST API dada la utilización de REST y para evitar confusiones con el API nativo de la plataforma.

Con respecto a la seguridad, Canvas REST API utiliza el protocolo Oauth para manejar autenticación, por defecto todas las

invocaciones deben ser realizadas por medio de canales seguros usando HTTPS.

Las operaciones que son expuestas a través del API incluyen mecanismos para acceder a tareas, asignaciones, información de cursos, registros, roles, usuarios, tópicos de discusión, entre otros [1].

1.4.2. Soluciones en otros sistemas

Los sistemas que han implementado APIs varían desde sistemas dedicados a búsquedas en la Web (Google, Bing), redes sociales (Facebook, Twitter, hi5), compañías de comercio electrónico (amazon.com, eBay), el auge de creación de estas interfaces se debe al crecimiento del mercado de dispositivos que usan el Internet y con ello la necesidad de estos sistemas de mantener comunicaciones con otras aplicaciones basadas o que hacen uso de sus servicios.

A continuación se presenta un vistazo de algunos APIs utilizados por aplicaciones cuyo propósito es diferente al de los LMS, entre estos se encuentran APIs de Google, Twitter, y Amazon.

[1] Documentación Canvas REST API (https://canvas.instructure.com/doc/api/index.html)

1.4.2.1 Google APIs

Google es el buscador más usado en Internet, según Hitwise Experian compañía que realiza sondeos en Internet, Google tiene un poco más del 62% de participación en el mercado de búsquedas en Internet.

Ranking	Motor de Búsqueda	Porcentaje
1	www.google.com	62.30%
2	search.yahoo.com	15.96%
3	www.bing.com	14.80%
4	www.ask.com	3.89%
5	search.aol.com	2.50%
Fuente: Experian Hitwise		

Tabla I: Volumen de búsquedas en diferentes motores

Actualmente Google ofrece un API para casi todos sus servicios como por ejemplo Búsquedas, Youtube, Mapas, Google+, Charts [1], algunas de estas APIs requieren la obtención de una llave (API key) previo a la utilización de las mismas (6), mediante esta llave Google puede identificar las aplicaciones desde las cuales se están generando las requisiciones y así tener registro de uso del API con fines de monitoreo.

[1] Directorio de APIs de Google (http://code.google.com/more/)

En general las APIs ofrecidas por Google tienen restricciones en cuanto a cantidad de invocaciones por día, si alguna aplicación planea generar más invocaciones de las especificadas en las restricciones puede ser que sea objeto de facturación, por ejemplo en API de Búsquedas limita el acceso a 1000 peticiones por día (7).

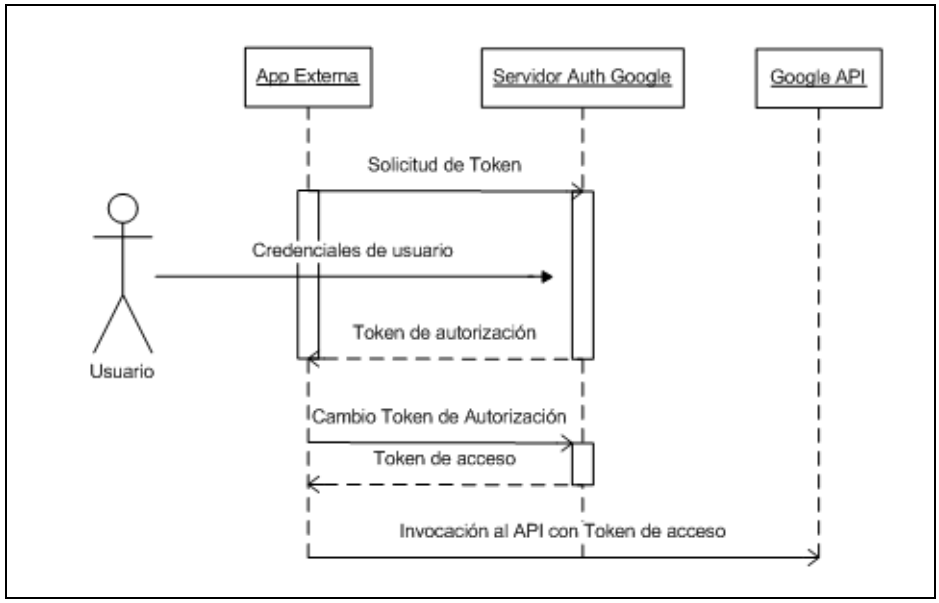

Figura 1.2: Diagrama de secuencia para autorización con Google APIs

Para operaciones que requieren autenticación de usuario, Google API usa el protocolo OAuth 1.0 (8), este protocolo permite a la aplicación externa obtener una cadena de caracteres llamado Token de acceso para acceder a operaciones protegidas sin que el usuario revele sus credenciales, el proceso mostrado en la figura 1.2 puede ser resumido en los siguientes pasos:

1. La aplicación externa redirecciona al usuario a un sitio de autenticación manejado por Google para obtener un Token de autorización.
2. El usuario entrega sus credenciales a Google, de ser exitosa la autenticación Google devuelve a la aplicación externa un Token de autorización.
3. La aplicación externa se comunica con un servicio de Google para intercambiar el Token de autorización por un Token de acceso.
4. La aplicación externa usa el Token de acceso para las requisiciones en el API.

Un aspecto importante a considerar en el planeamiento de una aplicación que va a utilizar los servicios del API es el hecho que Google se reserva a cambiar las condiciones de los servicios, según esto, por ejemplo puede dejar de ofrecer lo servicios de un API [1] o convertirlos en servicios facturados [2]

---

[1] Entrada en el blog oficial de Google Code donde se detalla que APIs dejarán de prestar servicios (http://googlecode.blogspot.com/2011/05/spring-cleaning-for-some-of-our-apis.html)

[2] Entrada en el blog oficial de Google Code informando la apertura de una versión pagada del API de traducciones (http://googlecode.blogspot.com/2011/08/paid-version-of-google-translate-api.html)

El API de búsquedas de Google (Custom Search API) puede ser usado para realizar las mismas tareas que en la versión de búsquedas de Web, con la ventaja de que al usar el API se pueden crear soluciones más avanzadas y personalizadas. El API incluye métodos para realizar búsquedas de términos en el Web y devuelve estructuras de datos que pueden ser utilizadas por aplicaciones terceras para realizar integraciones con el servicio.

1.4.2.2 Amazon Product Advertising API (PAAPI)

Amazon es una multinacional dedicada al comercio electrónico, se inició en el año 1995 como una tienda electrónica de libros pero con el tiempo se ha diversificado su oferta ahora incluye CDs, DVDs, descargas MP3, software computacional, mobiliario, artículos electrónicos, vestuario, entre otros.

Amazon se ha constituido en la tienda en línea más grande del mundo, con ramas separadas para países como Reino Unido, Canadá, Polonia, Alemania, Francia, Italia, España, Japón y China.

Para el año 2002 la compañía lanzo Amazon Web Services (AWS) destinado a proveer servicios a otros sitios Web o para aplicaciones, la mayoría de estos servicios no están expuestos para usuarios finales sino para desarrolladores, entre los servicios expuestos se encontraba el Amazon E-Commerce Service, para el año 2008

conocido como Amazon Associates Web Service [1], y finalmente para el año 2009 como Amazon Product Advertising API (PAAPI) [2]. Para enero del 2011 Amazon Web Services contiene más de 260 billones de objetos almacenados (9).

El PAAPI provee operaciones para:

- Obtener información de productos y precios
- Acceder a revisiones de productos por parte de clientes y vendedores.
- Obtener imágenes de productos
- Realizar búsquedas simples y avanzadas
- Acceder a listas de deseos de clientes (wishlists).

La principal motivación del desarrollo del API fue ampliar la visibilidad de los productos ofertados por Amazon, lo que hace atractiva a esta interfaz para los desarrolladores es que las aplicaciones desarrolladas en base a este API reciben un porcentaje de las ganancias por las ventas generadas.

[1] Notas de lanzamiento detallando cambio de nombre del servicio (http://aws.amazon.com/releasenotes/Product-Advertising-API/1022)

[2] Notas de lanzamiento detallando cambio de nombre del servicio (http://aws.amazon.com/releasenotes/Product-Advertising-API/2431)

Figura 1.3: Componentes del negocio de PAAPI

Como se muestra en la figura 1.3, las aplicaciones clientes interactúan con el PAAPI para realizar búsquedas en los catálogos, obtener imágenes de los productos, comentarios, valoraciones entre otros, luego si al redireccionar a los clientes a la página de Amazon la venta se concreta, la aplicación cliente obtiene un porcentaje de la venta.

1.4.2.3 Twitter API

Twitter es una red social por la cual sus clientes pueden publicar o leer mensajes cortos de 140 caracteres llamados tweets, Twitter cuenta con gran popularidad en Internet, para el verano del 2008, dos años después de su creación las cuentas de usuarios pasaban

los 3 millones, 6 millones para Enero del 2009 (10), para el 2011 se reportan 300 millones de usuarios [1].

Twitter API fue lanzado como una iniciativa para el desarrollo de aplicaciones externas, para el año 2009 el 44% del tráfico se generaba por la interfaz Web (11) con lo que el restante 56% era generado por recursos externos usando Twitter API. Para el 2011 el API de Twitter recibe 15 billones de llamadas por día (12).

Entre las operaciones que se pueden realizar con Twitter API se encuentran funciones para obtener tweets públicos usuarios, realizar búsquedas de términos usados en tweets, obtener los tópicos más populares en los tweets, modificar información de cuentas, entre otros.

Para acceder al uso de Twitter API es necesario obtener un API Key, el cual deberá ser enviado con todas las requisiciones al API, las operaciones que requieren autenticación de usuario tales como: publicación de tweets, modificación de información personal entre otros, es necesario realizar autenticación por medio del protocolo OAuth.

[1] Artículo CNN, Junio 2011 (http://articles.cnn.com/2011-06-27/tech/limits.social.networking.taylor_1_twitter-users-facebook-friends-connections)

El protocolo OAuth permite a la aplicación externa obtener una cadena de caracteres llamado Token de acceso para acceder a operaciones protegidas sin que el usuario revele sus credenciales.

Twitter API usa normas de diseño REST para la organización de sus operaciones, esta arquitectura da principal importancia a la representación de recursos mediante URIs, y a los verbos (GET, POST, DELETE) utilizados en las requisiciones, esto se refiere a que se use el verbo GET para lectura de recursos, POST para creación/actualización de recursos, y DELETE para eliminación de recursos. (13)

Twiter API implementa códigos de respuesta HTTP significativos, esto significa que, de al resultado de la petición se retorna un código de estado, así por ejemplo para peticiones exitosas se retorna el código 200, para peticiones no autorizadas 401, estos códigos normalmente van acompañados con mensajes descriptivos en el cuerpo del mensaje HTTP de respuesta.

Código	Estatus	Descripción
200	OK	Éxitosa.
304	Not Modified	No hubo nueva data para devolver.
400	Bad Request	Petición inválida, un mensaje de error explicará la razón.
401	Unauthorized	Credeciales de autorización inválidas o no presentes.
403	Forbidden	Petición entendida pero rechazada, un mensaje de error explicará la razón.
404	Not Found	El URI solicitado es inválido, o el recurso solicitado no existe.
406	Not Acceptable	Retornado por el Search API cuando la invocación se encuentra en un formato inválido.
420	Enhance your calm	Retornado por el Search API y el Trends API cuando se le esta aplicando limites de invocaciones al cliente.
500	Internal Server Error	Error interno del API.
502	Bad Gateway	Retornado cuando el servicio no está disponible debido a fallas o mantenimiento.
503	Service Unavailable	Retornado cuando el servicio esta sobrecargado.
Fuente: Documentación Twitter API		

Tabla II: Códigos HTTP retornados por Twitter API

Twitter limita el número de llamadas al API, para requisiciones que no han sido autorizadas por medio de OAuth, el límite es de 150 invocaciones por hora, en cambio para requisiciones autorizadas por medio de OAuth el límite es de 350 invocaciones por hora, las invocaciones no autorizadas son monitoreadas por medio de el IP del servidor o dispositivo que realiza la invocación, para invocaciones autorizadas el monitoreo se lo realiza por medio del Token de acceso que es usado en la invocación, para evitar sobrepasar la cuota Twitter recomienda usar mecanismos de cache[1].

[1] Documentación Twitter (https://dev.twitter.com/docs/rate-limiting)

CAPÍTULO 2

2. ANÁLISIS DE LA SOLUCIÓN

En este capítulo se realiza un análisis la solución propuesta, la cual es una interfaz que expone las operaciones más críticas de SIDWeb de manera que aplicaciones externas puedan interactuar con las funcionalidades que provee el software SIDWeb.

2.1. Descripción de SIDWeb API

SIDWeb API ha sido implementado en el mismo entorno que el software de SIDWeb 4.0, la programación se la realizó con el lenguaje PHP bajo el Framework Symfony, toda la implementación fue desarrollada de manera modular de tal forma que el mantenimiento o puesta de baja del API no afecte al resto de la implementación de SIDWeb 4.0.

Con relación al Framework Symfony bajo el cual SIDWeb API está implementado, el API aprovecha la característica de enrutamiento (14) para dirigir invocaciones al API a los métodos correspondientes a través

de un archivo que mapea patrones de URIs con los métodos apropiados, así por ejemplo, una invocación realizada a "/sidwebapi/tickets/get" moverá el requerimiento al método que se encarga de la generación de tickets, esto refuerza el uso de REST y provee de más limpieza al URI. Otra de las características que son aprovechadas por el API es la utilización del ORM Doctrine (15), el cual permite la utilización de los recursos de base de datos como objetos relacionales.

SIDWeb API utiliza el criterio de diseño REST (16) para la organización y tratamiento de las operaciones, esta arquitectura prioriza la identificación de recursos por medio de representaciones accedidas por los URIs, y la manipulación de las representaciones por medio de verbos específicos (GET, POST, PUT, DELETE), el uso de esta arquitectura es explicado con mayor detalle en el capítulo 4 de este trabajo.

Considerando la seguridad de la interfaz, todas las interacciones cliente-servidor son realizadas por medio de un canal seguro de comunicación "https", un canal seguro utiliza métodos de encriptación y de identificación para evitar que terceros tengan acceso a información sensible, tal como usuarios, contraseñas, datos de usuarios, etc.

Para el uso de SIDWeb API es necesario realizar un registro de la aplicación que va hacer uso del API, este registro genera una Llave de

Aplicación que será necesaria para acceder a las funcionalidades del API, la interfaz de registro estará disponible para usuarios registrados en la versión Web de SIDWeb.

2.2. Funcionalidades

Las funcionalidades ofrecidas por SIDWeb API abarcan operaciones para realizar autenticación, obtención de información y planificación de cursos, visualización de calendario de actividades, e interacción con foros, en la figura 2.1 se muestra un diagrama con las funcionalidades de SIDWeb API.

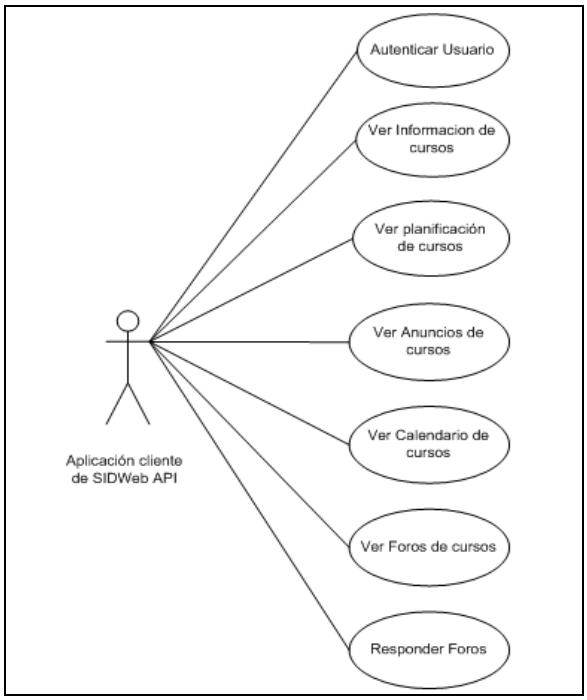

Figura 2.1: Diagrama de casos de uso de SIDWeb API

Dada la estructura de SIDWeb 4.0 en la que se define una entidad llamada “Sitio Web” la cual sirve para agrupar a diferentes secciones de un curso que compartan un mismo espacio virtual dentro del software, esto es necesario para cuando un curso es dividido en paralelos y se desea que los distintos paralelos compartan un mismo espacio virtual en el software.

Es importante mencionar que para los métodos de SIDWeb API (exceptuando métodos de autenticación) es necesario proveer un parámetro que contenga un ticket de acceso, la generación de tickets de acceso se explica en detalle en la sección de autenticación.

2.2.1. Autenticación

La operación de autenticación se refiere a confirmar que una invocación al servicio sea realizada por un usuario autorizado para acceder al sistema, SIDWeb API utiliza un mecanismo de tickets de acceso para este propósito, el API provee dos operaciones para la generación de tickets de acceso: Generación de Ticket de acceso por medio de credenciales de usuario y Generación de Ticket de acceso por medio del uso del servicio CAS de ESPOL, estas operaciones se encargan de validar la identidad de un usuario y de proveer un ticket de acceso temporal y que puede ser revocado para realizar operaciones con SIDWeb API.

El método de generación de Ticket de acceso por medio de credenciales de usuario como el nombre lo indica, genera tickets de acceso utilizando como medio de autenticación las credenciales de usuario.

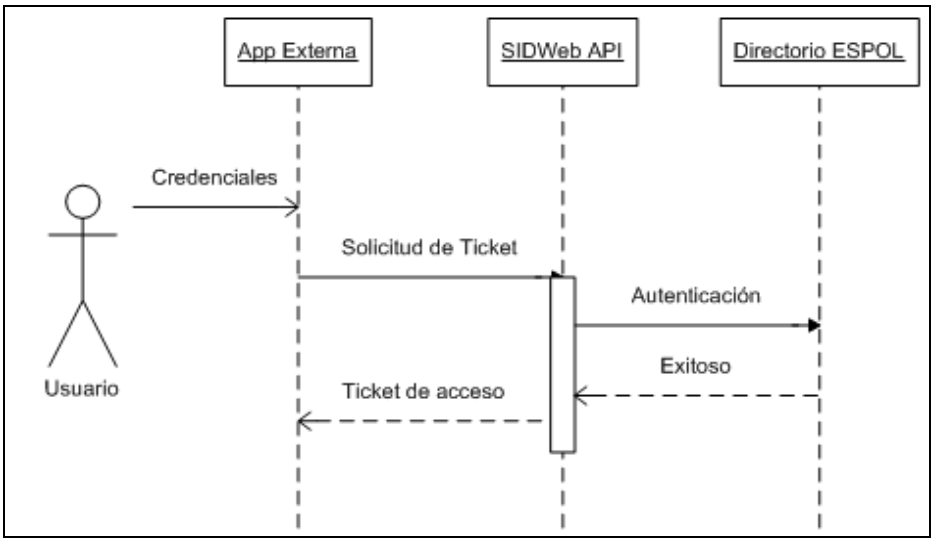

Figura 2.2: Diagrama de secuencia para generación de ticket de acceso por medio de credenciales de usuario

En la figura 2.2 se muestra un diagrama de secuencia para este método, el cual puede ser descrito en los siguientes pasos:

1. La Aplicación externa obtiene las credenciales del usuario.
2. La Aplicación externa realiza la solicitud de ticket de acceso, la requisición realizada debe incluir las credenciales de usuario (usuario y contraseña).

3. SIDWeb API verifica las credenciales entregadas por la aplicación externa realizando una invocación al servicio de Directorio ESPOL [1].
4. En el caso de ser un usuario válido SIDWeb API devuelve un Ticket de acceso el cual deberá ser usado para invocaciones a métodos que requieran Ticket de acceso.

El método de generación de ticket de acceso por medio del servicio CAS de ESPOL, provee de tickets de acceso mediante la verificación de tickets generados mediante CAS de ESPOL [2] el cual es un servicio mantenido por el Centro de Servicios Informáticos de ESPOL, CAS de ESPOL se trata de un mecanismo de SSO (Single Sign On) mediante el protocolo CAS, el propósito de este servicio es proporcionar un punto único de verificación de credenciales de usuario de ESPOL para distintas aplicaciones.

[1] Documentación Servicio Web: Directorio ESPOL
https://www.academico.espol.edu.ec/services/directorioespol.asmx

[2] Central de Autenticación de ESPOL
https://auth.espol.edu.ec/login

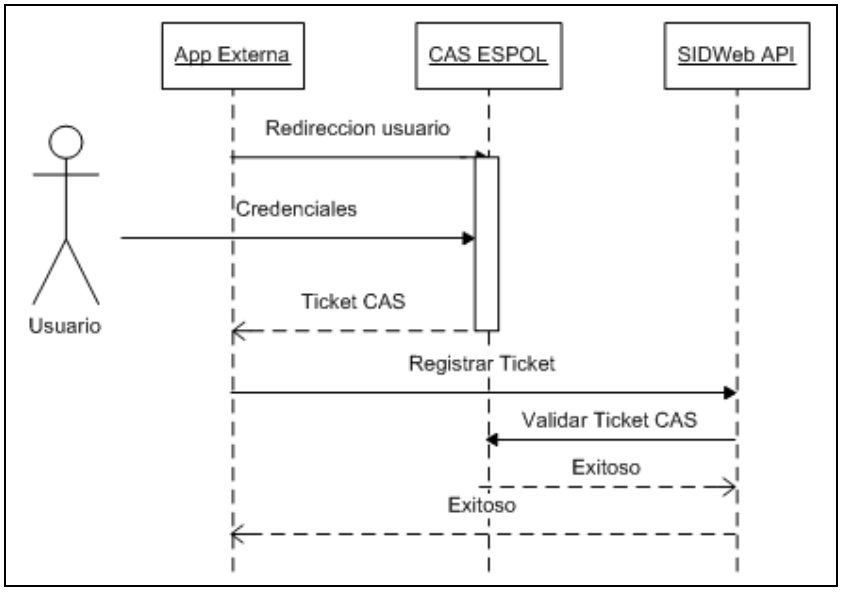

Figura 2.3: Diagrama de secuencia para generación de tickets por medio del servicio CAS de ESPOL

En la figura 2.3 se muestra un diagrama de secuencia para el método de generación de tickets por medio de CAS de ESPOL, el cual puede ser descrito en los siguientes pasos:

1. La aplicación externa redirige al usuario al sitio de ESPOL que provee el servicio CAS, especificando como parámetro “service” una cadena de caracteres que identifique a la aplicación externa con el servicio de autenticación.
2. Una vez realizada la autenticación por medio de CAS la aplicación externa debe capturar el ticket generado por CAS e invocar el método “sidwebapi/tickets/registrar” de SIDWeb API para registrar el ticket generado por CAS.

3. SIDWeb API se comunica con CAS para comprobar la validez del ticket generado, en caso de ser válido SIDWeb API registra el ticket en su base de datos para que pueda ser usado como ticket de acceso.

2.2.2. Información y planificación de cursos

Esta funcionalidad permite recoger información y la planificación de los cursos en los que se encuentra registrado un usuario de SIDWeb, la operación que lleva a cabo la función de recoger información requiere como parámetros de entrada el ticket de acceso de SIDWeb API y una cadena de caracteres que identifica al curso del que se requiere obtener la información. La información entregada por la funcionalidad al ser invocada es:

- Resumen: Incluye información que haya sido definida como saludo de bienvenida del curso
- Horarios: Estructura por la cual se puede acceder a los horarios de los distintos paralelos relacionados con el curso.
- Objetivos generales: Conocimientos habilidades y valores que se espera que adquiera el estudiante al finalizar el curso.
- Texto y otras referencias: Recursos necesarios en el proceso enseñanza-aprendizaje tales como: texto-guía, referencias bibliográficas, bases de datos, links online, páginas Web, etc.

- Recursos: Describe las facilidades logísticas y de equipamiento destinadas para el desarrollo de este curso.
- Nombre Materia: Nombre de la materia asociada con el curso.
- Código Materia: Código de materia manejado por la Facultad.
- Siglas Facultad: Siglas de la facultad a la cuál pertenece la materia.

2.2.3. Calendario de Actividades

La funcionalidad de visualización del calendario de actividades permite obtener una estructura que contiene la información del calendario de actividades de un curso en que se encuentra registrado un usuario de SIDWeb, los tipos registros entregados por esta funcionalidad incluyen ítems que representan Tareas, Clases y Anuncios para los cuales se especifican fechas de publicación y validez para que puedan ser agrupados en un calendario.

2.2.4. Visualizar Anuncios

Esta funcionalidad permite obtener una estructura cuyos ítems representan los anuncios asociados a un Sitio Web, la información es organizada en diferentes secciones:

- **Vigentes**: Anuncios que han sido clasificados como visibles y que no han alcanzado la fecha de expiración.

- **Pasados**: Anuncios que han sido clasificados como visibles y que han alcanzado la fecha de expiración.
- **No Disponibles**: Anuncios que han sido clasificados como no visibles
- **Borrados**: Anuncios que han sido clasificados como borradores

La información entregada por esta funcionalidad en cada una de las diferentes secciones incluye:

- Token de anuncio: Cadena de texto que identifica de manera única al recurso
- Nombre de anuncio: El nombre asignado al anuncio
- Nombre de sección: El nombre de la sección en la cual se encuentra el anuncio
- Visibilidad de anuncio: Determina si el anuncio ha sido clasificado como visible
- Fecha de disponibilidad: Especifica la fecha a partir de la cual el anuncio debe estar disponible
- Fecha de expiración: Especifica la fecha de expiración del anuncio
- Descripción de anuncio: Cadena de texto que contiene la descripción que le fue asignada al anuncio

2.2.5. Interacción con foros

La interacción con foros implementada por SIDWeb API consiste en la posibilidad de visualizar foros asociados con un sitio Web, visualización de respuestas asociadas a foros, la creación de nuevos foros, y la creación de respuestas a foros existentes.

La visualización de foros requiere un parámetro que identifique al sitio Web al que están asociados los foros que se desean visualizar, entre la información de los foros que se entrega a través de esta funcionalidad se encuentra:

- Cadena de identificación del foro
- Título del foro
- Descripción de foro
- Nombre completo del usuario quien creo el foro
- Nombre de usuario del usuario quien creo el foro
- Fecha de creación del foro

La visualización de respuestas de foros requiere como parámetro el número de identificación del foro del que se desean obtener las respuestas, entre la información de cada una de las respuestas que se entrega a través de esta funcionalidad se encuentra:

- Cadena de identificación de la respuesta
- Mensaje contenido en la respuesta
- Nombre completo del usuario quién creo la respuesta

- Nombre de usuario del usuario quién creo la respuesta
- Fecha de creación de la respuesta
- Cadena de identificación del padre de la respuesta, esto se refiere al elemento que esta asociado a la respuesta

La creación de nuevos foros introduce nuevos elementos de tipo foro, de realizar una inserción exitosa el procedimiento devuelve una estructura indicándolo, esta funcionalidad requiere parámetros que indiquen:

- Token de sitio Web que indique el sitio Web al que va a ser asociado el foro a ser insertado
- Título del foro
- Descripción foro
- Valor binario que indique si el foro va a ser catalogado como un borrador
- Valor binario que indique si se debe notificar por correo electrónico a los miembros asociados al sitio Web acerca de la creación del foro
- Fecha de expiración del foro, la cual indica hasta que fecha debe estar publicado el foro
- Fecha de disponibilidad del foro, la cual indica desde que fecha debe estar publicado el foro

2.3. Alcance

SIDWeb API permitirá realizar invocaciones a operaciones del software SIDWeb, estas operaciones comprenden mecanismos de autenticación, la interacción con elementos del sistema como lo son la sección de foros, anuncios, calendario e información de curso. Todas las operaciones implementadas devuelven respuestas en formato JSON y con códigos HTTP consistentes con el tipo de respuesta. SIDWeb API requiere.

A continuación se detalla las limitaciones de SIDWeb API:

- Las operaciones implementadas por SIDWeb API no cubren las funciones del rol de administrador como lo es administración de los sitios Web, modificación de información de cursos, entre otras funciones de administración.
- SIDWeb API no es un producto dirigido a usuarios finales, SIDWeb API es un producto para uso de desarrolladores interesados en construir aplicaciones basadas en SIDWeb o que integren sus funciones a SIDWeb.
- Para las pruebas de SIDWeb API se ha desarrollado un pequeño aplicativo capaz de ser ejecutado en dispositivos móviles, sin embargo este aplicativo fue desarrollado con el solo propósito de probar las funcionalidades de SIDWeb API y no debe ser

considerado como un producto listo para un ambiente de producción.

- SIDWeb API ha sido desarrollado en la versión 4.0 de SIDWeb, SIDWeb API no es compatible con versiones previas de SIDWeb.
- La disponibilidad de métodos de autenticación de SIDWeb API es dependiente de la disponibilidad de los servicios de CAS ESPOL y Directorio ESPOL.

CAPÍTULO 3

3. DISEÑO DE LA SOLUCIÓN

En este capítulo se provee de información más detallada acerca de los componentes de SIDWeb API y sus interacciones, un diagrama de flujo de requisiciones para diferenciar el flujo de una requisición al API versus una requisición a otro módulo de SIDWeb, diagramas de secuencia para describir el orden de las operaciones para los casos de uso más notables de SIDWeb API, también se presenta un diagrama con las tablas incluidas en el esquema del software SIDWeb y la descripción de códigos http usados por SIDWeb API.

3.1. Componentes

SIDWeb API ha sido implementado como un módulo del software SIDWeb 4, esto facilita las interacciones entre el API con los demás módulos del sistema, aprovecha las características del Framework

Symfony y al tener características modulares permite que si el módulo es desconectado el resto del software SIDWeb 4 no sea afectado.

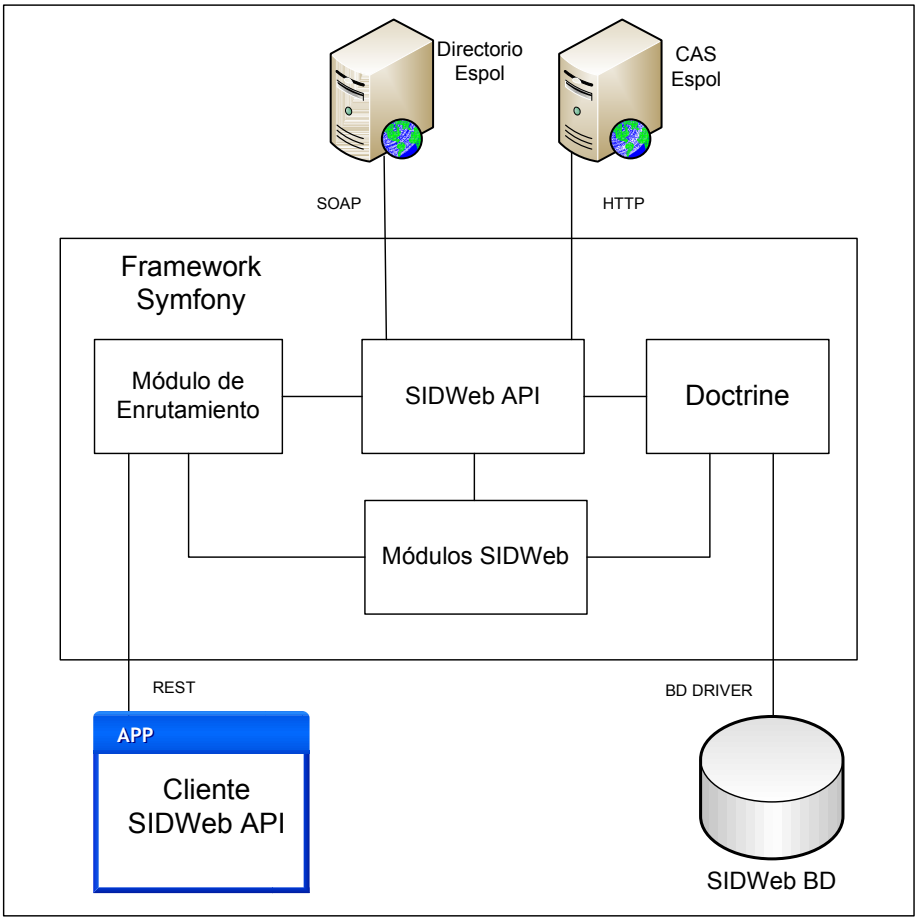

Figura 3.1: Diagrama de componentes de la solución

En la figura 3.1 se muestran los componentes y los enlaces de interacción entre los mismos, el Cliente SIDWeb API es cualquier aplicación que haya sido registrada para el uso del API y que realice requisiciones al sistema, el módulo de enrutamiento es un componente de Symfony que se encarga de analizar patrones de URLs en las

requisiciones y re-direccionarlas a un método específico para que sean procesadas, esta operación se la realiza de acuerdo a un archivo de mapeo el cual define los patrones y a que modulo se deben enviar, SIDWeb API se encarga de procesar requisiciones enviadas por el módulo de enrutamiento, en sí SIDWeb API pertenece al conjunto de Módulos de SIDWeb pero para efectos del diagrama se lo ha separado para hacer explícitas las relaciones entre SIDWeb API con los demás módulos de SIDWeb, Doctrine es un ORM integrado al Framework de Symfony que provee una capa de abstracción para la comunicación con la base de datos (SIDWeb BD), esto facilita que la aplicación sea escrita sin dependencias a un motor de base de datos específico, además de proporcionar un entorno orientado a objetos para el manejo de los datos en la base de datos. Los componentes Directorio ESPOL y CAS ESPOL son consumidos por SIDWeb API en las operaciones de autenticación de usuarios, para ambos las comunicaciones son realizadas por medio del protocolo SOAP, para un escenario de autenticación usando CAS ESPOL también sería necesaria la interacción del Cliente SIDWeb API con CAS ESPOL para la obtención de un ticket de CAS.

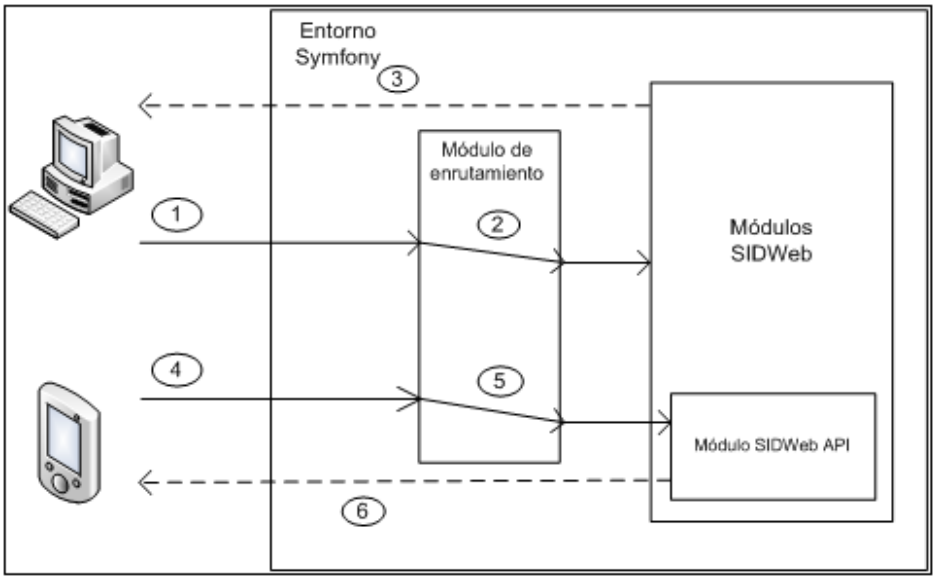

Figura 3.2: Flujo de requisiciones a través de componentes de SIDWeb

En la figura 3.2 se muestra el flujo de requisiciones, para las requisiciones dirigidas a módulos diferentes a SIDWeb API el flujo es el siguiente:

1. Un cliente Web, tal como un navegador origina una requisición a SIDWeb.
2. Una vez en el entorno de Symfony, la requisición es analizada por el módulo de enrutamiento en el cual se redirecciona de acuerdo al mapeo que se haya especificado, con esto por ejemplo una requisición a la URL: “http://beta.sidweb.espol.edu.ec/AuthCAS/logout” es redirigida al módulo “AuthCas” a la ejecución del método “logout”, la información de patrones y a que módulos/métodos se debe redirigir es contenida en un archivo de mapeo.

3. Luego de procesar la requisición el módulo devuelve una respuesta al cliente Web.

Las requisiciones dirigidas a SIDWeb API contienen Urls que siguen el patrón: [URL_BASE]/sidwebapi/*, para estas requisiciones el flujo es el siguiente:

1. Un cliente del API, tal como una aplicación en un dispositivo móvil, genera una requisición con el patrón mencionado.
2. La requisición en redirigida por el módulo de enrutamiento al módulo de SIDWeb API mediante el mapeo especificado.
3. SIDWeb API procesa la requisición y devuelve una respuesta en formato JSON.

3.2. Prototipos de funciones

Las operaciones expuestas en SIDWeb API reciben parámetros por medio de la requisición HTTP y devuelven respuestas con códigos HTTP congruentes con el estado de la operación y mensajes en formato JSON con los datos devueltos.

GET sidwebapi/:ticket_acceso/sitiosweb/:token_sitioweb/calendario
Obtener el calendario asociado a un sitioweb
URL
https://[URL-BASE]/sidwebapi/:ticket_acceso/sitiosweb/:token_sitioweb/calendario

Parametros:

ticket_acceso *requerido*	Ticket de acceso válido generado o registrado con SIDWeb API
token_sitioweb *requerido*	Token del sitioweb del que se desea obtener el calendario, se pueden obtener token de sitiosweb a los que un usuario está registrado mediante el método: **GET sidwebapi/:ticket_acceso/usuarios/:nombre_usuario/registros**

Figura 3.3: Referencia de función para operación GET de obtención de calendarios

En la figura 3.3 se muestra la referencia de una operación de obtención de calendario en SIDWeb API, al ser una operación de solo lectura se la debe invocar usando el método GET de HTTP, esto para asegurar el cumplimiento con las normas de diseño nombradas por REST.

POST sidwebapi/:ticket_acceso/posts/:id_post/responder
Responder a un foro / post
URL
https://[URL-BASE]/sidwebapi/:ticket_acceso/posts/:id_post/responder

Parametros:

ticket_acceso *requerido*	Ticket de acceso válido generado o registrado con SIDWeb API
id_post *requerido*	Id del foro/post a responder.

Figura 3.4: Referencia de función POST para responder foro/post

La figura 3.4 muestra la referencia de una función que debe utilizar el método http POST puesto que se trata de una operación de creación de una respuesta para un foro/post, toda operación por la cual se va a crear un nuevo recurso debería utilizar el método POST según lo define REST.

3.3. Diagramas de secuencia

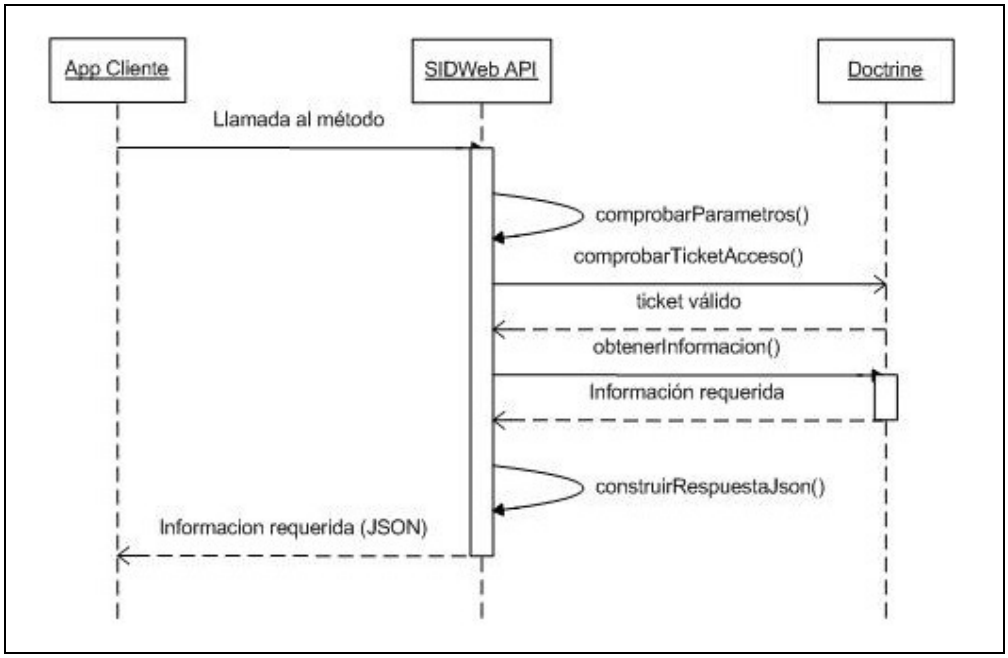

Figura 3.5: Diagrama de secuencia genérico para operaciones de SIDWeb API

La figura 3.5 muestra un diagrama de secuencia genérico que representa la mayoría de las operaciones que se realizan en SIDWeb API a excepción de las funciones que realizan autenticación. La secuencia genérica se podría describir en los siguientes pasos:

1. La Aplicación cliente genera una requisición a uno de los métodos de SIDWeb API.
2. SIDWeb API comprueba los parámetros entregados en la requisición, esto incluye comprobar si el parámetro es requerido y si el tipo es compatible con el contenido recibido.

3. SIDWeb API comprueba que el ticket de acceso entregado exista y sea válido, esta operación requiere consultas en base de datos, es por eso que implica interacción con Doctrine.
4. Doctrine devuelve la información que sirve para verificar validez de un ticket.
5. SIDWeb API llama los métodos necesarios para recolectar la información solicitada del requerimiento.
6. Doctrine entrega la información solicitada.
7. SIDWeb API construye una respuesta en formato JSON
8. SIDWeb API entrega la respuesta a la aplicación cliente.

Las operaciones relacionadas con la autenticación en SIDWeb API poseen secuencias diferentes puesto que requieren comunicación con servicios externos como Directorio ESPOL y el servicio de ESPOL CAS, la figura 3.6 muestra un diagrama de secuencia para la autenticación por medio del servicio de Directorio ESPOL.

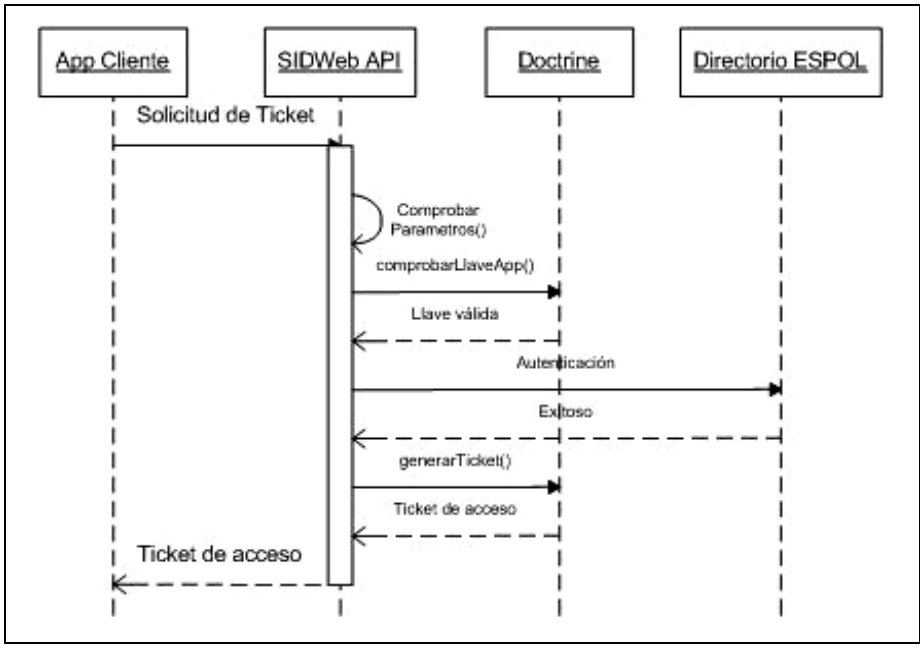

Figura 3.6: Diagrama de secuencia para autenticación en SIDWeb API por medio del servicio de Directorio ESPOL.

3.4. Diseño de la Base de datos

Para la implementación de SIDWeb API fue necesario agregar al esquema de base de datos 2 tablas que permitan guardar información acerca de las aplicaciones registradas para el uso del API y los tickets de acceso generados, estas tablas contienen referencias hacia la tabla de usuarios del software SIDWeb con lo que se identifica que usuario ha inscrito una aplicación para el uso del API y el usuario para el que se ha garantizado un ticket de acceso.

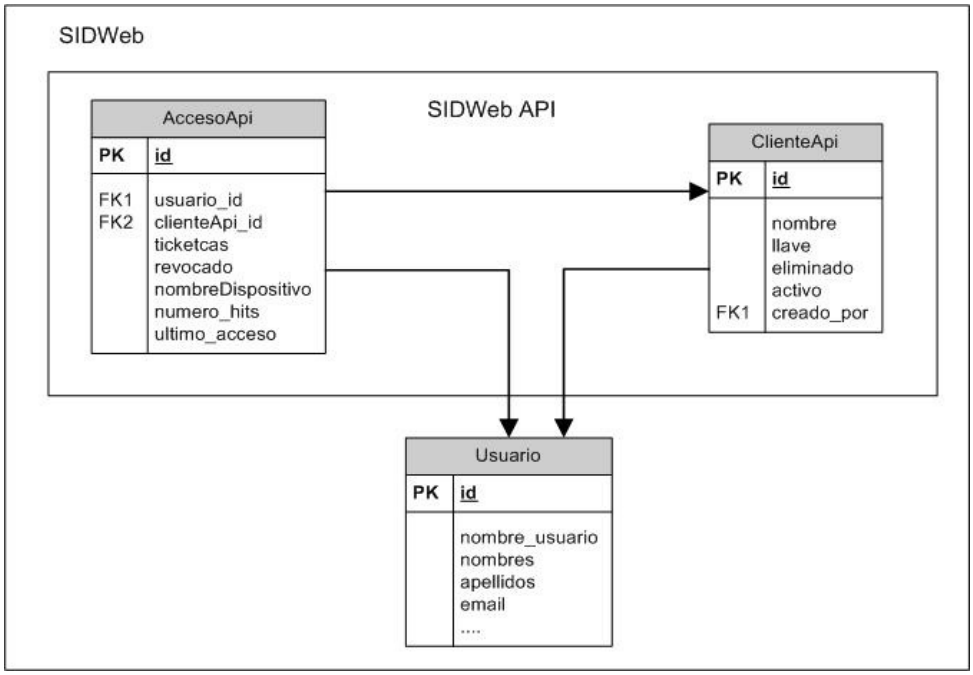

Figura 3.7: Diagrama de base de datos para SIDWeb API

La figura 3.7 muestra un diseño enfocado solamente en las tablas que han sido agregadas al esquema del software SIDWeb, la tabla "ClienteAPI" contiene los datos de las aplicaciones que han sido registradas para hacer uso de las funciones del API, en tanto que la tabla "AccesoApi" contiene los registros de los tickets de acceso generados para acceder a información relacionada con usuarios del sistema.

La tabla "AccesoApi" incluye campos para control de los tickets de acceso generados, por ejemplo un campo que contiene el número de veces que se ha usado el ticket para futuras implementaciones de control de cuota, también incluye un campo que contiene la fecha del

último acceso al ticket para la posible implementación de scripts que invaliden tickets que no hayan sido usados recientemente.

3.5. Códigos HTTP de respuesta

Las respuestas de requisiciones que entrega SIDWeb API incluyen códigos HTTP de respuesta [1], estos códigos representan el status de la operación realizada, así una respuesta con código 200 (“OK”) significa que la operación fue completada exitosamente, en cambio un código entre los rangos 3xx, 4xx, 5xx significa que la requisición no fue procesada con éxito, estos códigos son útiles para comprobar rápidamente del lado del cliente si hubo algún error al procesar la requisición.

Código	Estatus	Descripción
200	OK	Exitosa.
400	Bad Request	Petición inválida, un mensaje de error explicará la razón.
401	Unauthorized	No autorizado, un mensaje de descripcion explicará el error
404	Not Found	El URI solicitado es inválido, o el recurso solicitado no existe.
500	Internal Server Error	Error interno del API.

Tabla III: Códigos de respuesta devueltos por SIDWeb API.

[1] Lista de códigos de respuesta HTTP definidos por World Wide Web Consortium http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html

La tabla III muestra los códigos de respuesta http usados por SIDWeb API, los posibles valores son:

- 200 – Para requisiciones procesadas exitosamente.
- 400 – para requisiciones inválidas, esto puede suceder por ejemplo si uno de los parámetros requeridos no es entregado.
- 401 – para requisiciones que no han presentado un ticket de acceso o llave de aplicación válida y por tanto son no autorizadas.
- 404 – para requisiciones que solicitan recursos que no existen.
- 500 – para requisiciones que no se han podido completar por errores del lado del servidor.

CAPÍTULO 4

4. IMPLEMENTACIÓN DE SIDWEB API

Este capítulo presenta detalles importantes de la implementación de SIDWeb API, tales como el uso de la librería CURL para invocaciones a servicios externos, uso de enrutamiento de requisiciones, esquema de base de datos agnóstico, uso de formato JSON y de canales seguros.

4.1. PHP – Symfony

SIDWeb API fue desarrollado enteramente en PHP bajo el Framework Symfony dado que fue el lenguaje y el Framework facilitaban la integración del API como un módulo del software SIDWeb 4.

El módulo del API fue agregado como un componente del "frontend" de SIDWeb 4, que es el conjunto de módulos que se encargan de las funcionalidades que interactúan directamente con el usuario, por otra

parte el “backend” son un conjunto de módulos que se encargan de las funcionalidades de administración del software.

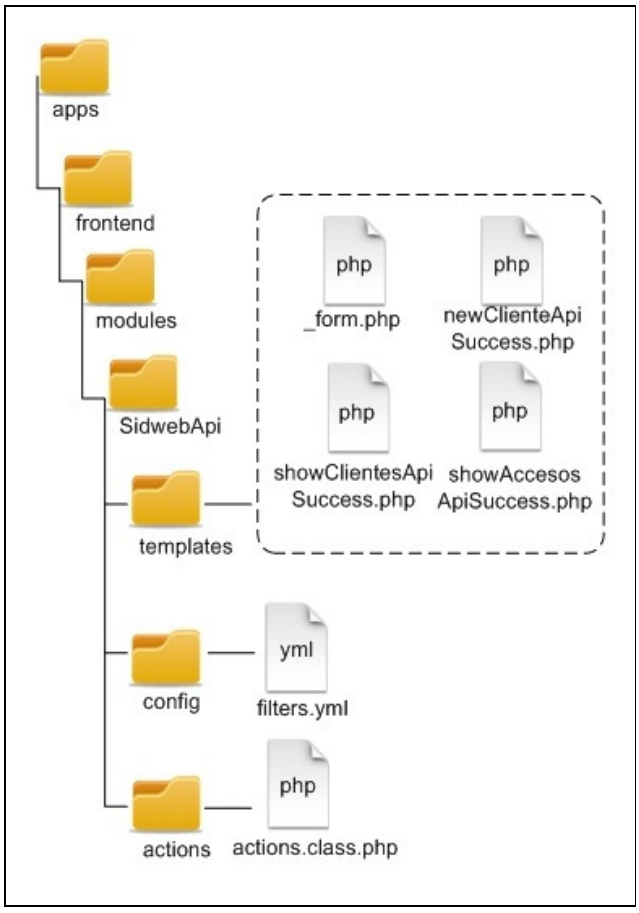

Figura 4.1: Estructura de directorio de SIDWeb API

La figura 4.1 muestra la estructura del directorio del módulo de SIDWeb API. La carpeta “templates” contiene los formularios y pantallas que implementan las pantallas de registro de aplicación para el uso del API y la pantalla de control de accesos al API por parte del usuario que los autorizo, la carpeta “config” contiene el archivo “filters.yml” que contiene configuraciones de filtros aplicados por SIDWeb 4, para el caso de

SIDWeb los filtros usados tienen que ver con procesos de autenticación que para SIDWeb API han sido desactivados puesto que estos no son compatibles para el uso de un API además que el API posee su propio mecanismo de autenticación, la carpeta “actions” contiene el archivo “actions.class.php” el cual contiene la implementación de los métodos que se exponen a través del API.

4.1.1 Invocaciones a CAS-ESPOL

CURL es una librería que permite conectarse y comunicarse con servidores con diferentes tipos de protocolo, la librería de PHP soporta http, https, FTP, gopher, Telnet, dict, file y LDAP [1].

```
$url = 'https://auth.espol.edu.ec/serviceValidate'; //URL CAS-ESPOL
$query = 'ticket='.$token.'&service='.$servicio;    //Parámetros de query

$curlObj = curl_init($url); //Inicialización objeto CURL

curl_setopt($curlObj, CURLOPT_RETURNTRANSFER, true);    //Retornar respuesta como cadena de texto
curl_setopt($curlObj, CURLOPT_POST, true); //Setear método POST
curl_setopt($curlObj, CURLOPT_POSTFIELDS, $query); //Seteo de parámetros de query

$respuesta = curl_exec($curlObj);
```

Figura 4.2: Código de ejemplo de uso de CURL por SIDWeb API

[1] Documentación de la librería cURL en PHP
http://www.php.net/manual/en/intro.curl.php

SIDWeb API realiza invocaciones al servicio CAS-ESPOL haciendo uso de esta librería, para este caso SIDWeb API se comunica con el servidor que contiene a CAS-ESPOL con el objetivo de verificar tickets entregados por una aplicación cliente que se dicen provenir de CAS-ESPOL, para esto SIDWeb API realiza una requisición a CAS-ESPOL por medio de CURL, la figura 4.2 muestra un ejemplo de requisición usando CURL.

4.1.2 Invocaciones de Directorio ESPOL

Directorio ESPOL es un servicio que provee el Centro de Servicios Informáticos (CSI), este servicio permite realizar operaciones de autenticación y de obtención de datos de estudiantes. El servicio esta implementado usando el protocolo SOAP.

```
$wsdl = 'https://www.academico.espol.edu.ec/
        services/directorioEspol.asmx?wsdl';
    //Comprobar disponibilidad del servicio
    if(!@file_get_contents($wsdl)) {
        throw new SoapFault('Server', 'No WSDL found at ' . $wsdl);
    }
    //Instanciar la clase
    $client = new SoapClient($wsdl);
    //Invocación al método del servicio
    $result =  $client->autenticacion(
                array('varUser'=>$user,
                'varContrasenia'=>$password));
```

Figura 4.3: Código de ejemplo de uso de la extensión SOAP por SIDWeb API

SIDWeb API realiza invocaciones a este servicio por medio de la extensión SOAP de PHP, el uso de esta extensión al igual que CURL

facilita el proceso puesto que hace transparente el uso de sockets y control de flujos, el procedimiento esta reducido a crear una instancia de la clase SoapClient y llamar al método del servicio SOAP que se desee utilizar, la figura 4.3 muestra un ejemplo de uso de la extensión por parte de SIDWeb API.

4.1.3 Generación de esquema de Base de datos

La generación del esquema de base de datos se la realizó mediante el uso de una funcionalidad del Framework Symfony que permite especificar la estructura de la base de datos en un archivo de configuración, esto brinda independencia de motor de base de datos y facilita migraciones a otros motores ya que primariamente el esquema reside en la aplicación.

```
ClienteApi:
  actAs:
      Timestampable:
  columns:
    id: {type: integer(4), autoincrement: true,
         primary: true, notnull: true, unique: true}
    nombre: {type: string(255), notnull:true }
    llave: {type: string(255), notnull:true }
    eliminado: {type: boolean, default: 0, notnull: false}
    activo: {type: boolean, default:1}
    creado_por: {type: integer(4), notnull:  false }
  relations:
    Usuario: {local: creado_por, foreign: id,
             foreign Alias: ClienteApiUsuario}
```

Figura 4.4: Especificación de tabla “ClienteAPI” en archivo de configuración de Symfony.

La figura 4.4 muestra la especificación de la tabla “ClienteAPi” en el archivo de configuración “schema.yml” que es donde se encuentran todas las especificaciones de las tablas que conforman el software SIDWeb 4. En la porción de código mostrada en la figura se puede apreciar el uso de la palabra reservada “TimeStampable”, esta se usa para que al crearse el esquema de base de datos se incluyan dos campos que contienen uno la fecha de creación del registro y otro la fecha de actualización del registro.

4.1.4 Enrutamiento de requisiciones

SIDWeb API hace uso de la característica de enrutamiento de requisiciones proveída por Symfony, el proceso de enrutamiento de la requisición comienza al ingresar la requisición, se analiza el URL con los patrones especificados en un archivo de mapeo y se redirecciona la requisición al módulo especificado por el patrón al que pertenece el URL. El uso de esta característica facilita el cumplimiento de la recomendación de limpieza de URL y la correspondencia de recursos con URLs especificada por REST.

```
sa_foro_crear:
  url:  sidwebapi/:ticket_acceso/foros/nuevoForo
  param:  {module: SidwebApi, action: crearForo}

sa_posts_responder:
  url:  sidwebapi/:ticket_acceso/posts/:id_post/responder
       param:  {module: SidwebApi, action: responderPost}
```

Figura 4.5: Porción de código de archivo de mapeo de enrutamiento

La figura 4.5 muestra el mapeo especificado para las acciones de nuevo foro y de responder foro/post, los parámetros “url” especifican el patrón de la URL y los parámetros “param” especifican a que módulo y método se debe redireccionar una requisición que cumpla con el patrón especificado por “url”.

4.2. JSON

La notación de objetos JavaScript “JSON” por sus siglas en inglés es un formato liviano de rápido intercambio de datos (17), independiente de plataforma, leíble por humanos dado que es un formato de texto para serialización de estructuras de datos. JSON puede representar cuatro tipos primitivos de datos:

- Cadenas de texto: una secuencia de cero o más caracteres Unicode (figura 4.6).

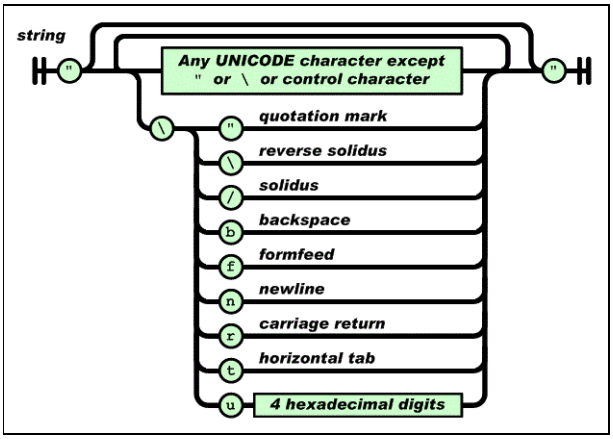

Figura 4.6: Representación de una cadena de texto en JSON [1].

- Números (figura 4.7)

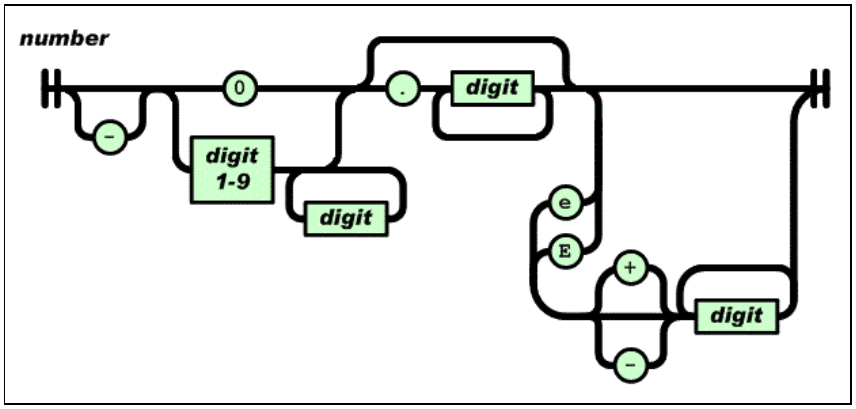

Figura 4.7: Representación de un número en el formato JSON [1].

- Booleanos (true, false).
- Nulos (null).

[1] Tomado de documentación JSON
http://www.json.org

También puede representar dos tipos estructurados:

- Objetos: colección de cero o más pares nombre-valor, donde nombre es una cadena de texto y valor puede ser cualquiera de los cuatro tipos primitivos soportados por JSON (figura 4.8).

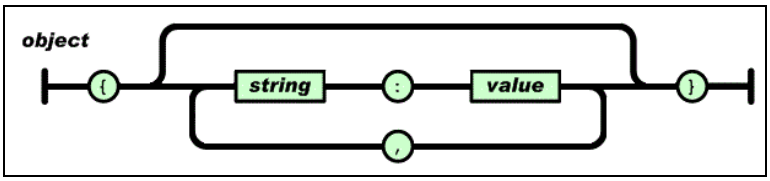

Figura 4.8: Representación de un objeto en el formato JSON [1].

- Arreglos: es una secuencia ordenada de cero o más valores (figura 4.9).

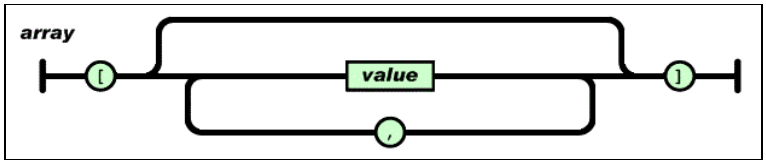

Figura 4.9: Representación de un arreglo en el formato JSON [1].

La ventaja de JSON sobre XML al momento de intercambio de datos, es su simplicidad y la poca sobrecarga en las representaciones en comparado con XML (18), esto lo hace ideal para intercambio de datos entre cliente – servidor.

[1] Tomado de documentación JSON
http://www.json.org

Twitter y Foursquare han dejado de dar soporte para XML en sus APIs y han optado por el uso de JSON exclusivamente [1]. Actualmente existen herramientas para el uso de JSON para los lenguajes de programación más populares.

Los métodos de SIDWeb API entregan las respuestas en formato JSON, para la construcción de las estructuras JSON se hace uso de la función de PHP json_encode [2], esta función toma como argumento un arreglo de datos y devuelve la representación JSON del arreglo de datos en formato de texto, esto facilita la construcción de las estructuras JSON dado que el proceso se reduce en construir un arreglo de datos con la información necesaria y luego usar la función "json_encode" para obtener la representación JSON.


```
[
    {
        "rol":"Estudiante",
        "materia_nombre":"Aplicaciones Multimedia Interactivas",
        "sitioweb_token":"88e1980cad6e694ffd0e20d7da1619c4b5ac339a"
    }
]
```


Figura 4.10: Respuesta a requisición de registros en SIDWeb API

[1] Artículo de Blog ReadWrite Hack
http://www.readwriteweb.com/hack/2010/11/json-vs-xml.php

[2] Documentación de PHP sobre función json-encode
http://www.php.net/manual/en/function.json-encode.php

La figura 4.10 muestra el cuerpo de la respuesta en formato JSON de SIDWeb API a una requisición de adquisición de información acerca de los registros asociados a un usuario.

4.3. Canales seguros

Las invocaciones a SIDWeb API deben ser realizadas utilizando canales seguro dado que involucran información sensible, el uso de canales seguros previene principalmente ataques del "hombre en la mitad" [1]. Para la implementación de canales seguros se hace uso del protocolo SSL/TLS el cual provee comunicaciones encriptadas e identificación segura de un Web Server.

Para el uso del canal seguro en el entorno de desarrollo se generó un certificado para el servidor mediante la herramienta OpenSSL [2] y se modifico la configuración del servidor Apache para el uso de https.

---

[1] Artículo de Wikipedia acerca del ataque del hombre en la mitad
http://en.wikipedia.org/wiki/Man-in-the-middle_attack

[2] Sitio Web del proyecto OpenSSL
http://www.openssl.org/

CAPÍTULO 5

5. PRUEBAS

En este capítulo se describe el ambiente de desarrollo en el cual fue desenvuelto SIDWeb API y los instrumentos usados para realizar las pruebas de la interfaz, se presentan figuras del aspecto del aplicativo SIDWeb Móvil en diferentes plataformas, también se muestran los resultados obtenidos de pruebas realizadas mediante el aplicativo SIDWeb Móvil a cinco usuarios del sistema, y finalmente se presenta un análisis de los resultados obtenidos mediante el desarrollo de este trabajo.

5.1. Entorno de pruebas

Las pruebas de SIDWeb API se realizaron en un entorno con la siguiente configuración de software / hardware:

- Servidor Web: Apache 2.2.17.
- Lenguaje de programación: PHP 5.3.4 / Symfony 1.4.8.

- Sistema Operativo: Windows 7 Home Premium 64 bits.
- Navegador Web: Google Chrome 16.0

Para las pruebas con el aplicativo SIDWeb Móvil además de lo anterior se utilizó:

- Framework: jQuery Mobile 1.0
- Ripple Mobile Environment Emulator 0.9 (beta)
- Android Virtual Device Manager (Emulador)
- iPhone 4, 8 GB, iOS 5, WI-FI
- iPad 2, 16 GB, iOS 5, WI-FI

5.2. Consola de pruebas de SIDWeb API

La consola de pruebas de SIDWeb API es una pequeña aplicación basada en PHP, mediante la consola de pruebas es posible realizar las siguientes acciones:

- Realizar invocaciones a los métodos expuestos por SIDWeb API.
- Obtener el formato de texto JSON de la respuesta.
- Obtener el código de respuesta de la invocación.

Figura 5.1: Configuración de invocación en consola de pruebas

Figura 5.2: Resultado de invocación en consola de pruebas

La figura 5.2 muestra el resultado de la invocación al método de obtención de registros asociados a un usuario, los resultados de las invocaciones en la consola de pruebas muestran información acerca de la requisición enviada, el código de respuesta recibido, el texto JSON y la decodificación de JSON adquirida mediante el método “var_dump” [1] de PHP.

[1] Documentación de PHP sobre var_dump
http://php.net/manual/en/function.var-dump.php

5.2. Aplicativo SIDWeb Móvil

Con el propósito de probar las funcionalidades de SIDWeb API en un ambiente multiplataforma que sea compatible con despliegue en dispositivos móviles se desarrolló un aplicativo llamado SIDWeb Móvil basado en JavaScript que hace uso del framework jQuery Mobile.

jQuery Mobile es un conjunto de plugins y widgets enfocados a proveer un API multi-plataforma para crear aplicaciones Web (19). Mediante jQuery Mobile se accede al uso de transiciones, estilos, menús, diálogos, eventos de toques de pantalla (touch events) dirigidos a dispositivos móviles.

El aplicativo fue probado bajo el emuladores Ripple, el cual es un emulador multi-plataforma para realizar pruebas de aplicaciones basadas en HTML5, también fue probado con un emulador de la plataforma Android y bajo dispositivos con el sistema operativo iOS (iPhone 4, iPad 2).

Figura 5.3: Pantalla de Login de SIDWeb Móvil

La figura 5.3 muestra la pantalla de Login del aplicativo SIDWeb Móvil, al realizarse la operación de login el aplicativo realiza una requisición de autenticación a SIDWeb API con las credenciales del usuario y la llave de la aplicación para obtener un ticket de acceso para las consecuentes invocaciones.

Figura 5.4: Pantalla de registros de SIDWeb Móvil

La figura 5.4 muestra la pantalla de registros, la cual es poblada mediante una invocación al método de obtención de registros asociados a un usuario, los parámetros necesarios para realizar la invocación son: ticket de acceso, y nombre de usuario del que se desean los registros.

Figura 5.5: Pantalla de opciones por Curso de SIDWeb Móvil

La figura 5.5 muestra la pantalla de opciones por curso, esta es accedida luego de escoger una de las materias asociadas con el usuario, esta pantalla es el acceso a las opciones de anuncios, información de curso, foros, calendario y tareas.

Figura 5.6: Pantalla de foros por Curso de SIDWeb Móvil

La figura 5.6 muestra la pantalla con los foros asociados a un curso en SIDWeb Móvil, esta pantalla es poblada mediante una invocación al método de obtención de foros asociados a un curso, los parámetros necesarios para realizar la invocación son: ticket de acceso, token de sitio Web al que esta asociado el usuario, los tokens de los sitios Web asociados al usuario se pueden obtener al realizar la invocación al método de obtención de registros asociados a un usuario.

Figura 5.7: Pantalla de hilos de foro de SIDWeb Móvil

La figura 5.7 muestra la pantalla de hilos asociados con un foro, esta pantalla es poblada mediante la invocación al método de obtención de hilos de un foro, los parámetros que requiere la invocación son: ticket de acceso, y token del foro padre.

5.2.1 SIDWeb Móvil en otras plataformas

El aplicativo SIDWeb Móvil al estar desarrollado sobre jQuery Mobile provee soporte multiplataforma, es así que también fue probado en iPad 2 (figura 5.8), iPhone 4 (figura 5.9) ambos bajo iOS 5, además se corrió el aplicativo en un emulador de la plataforma Android (figura 5.10).

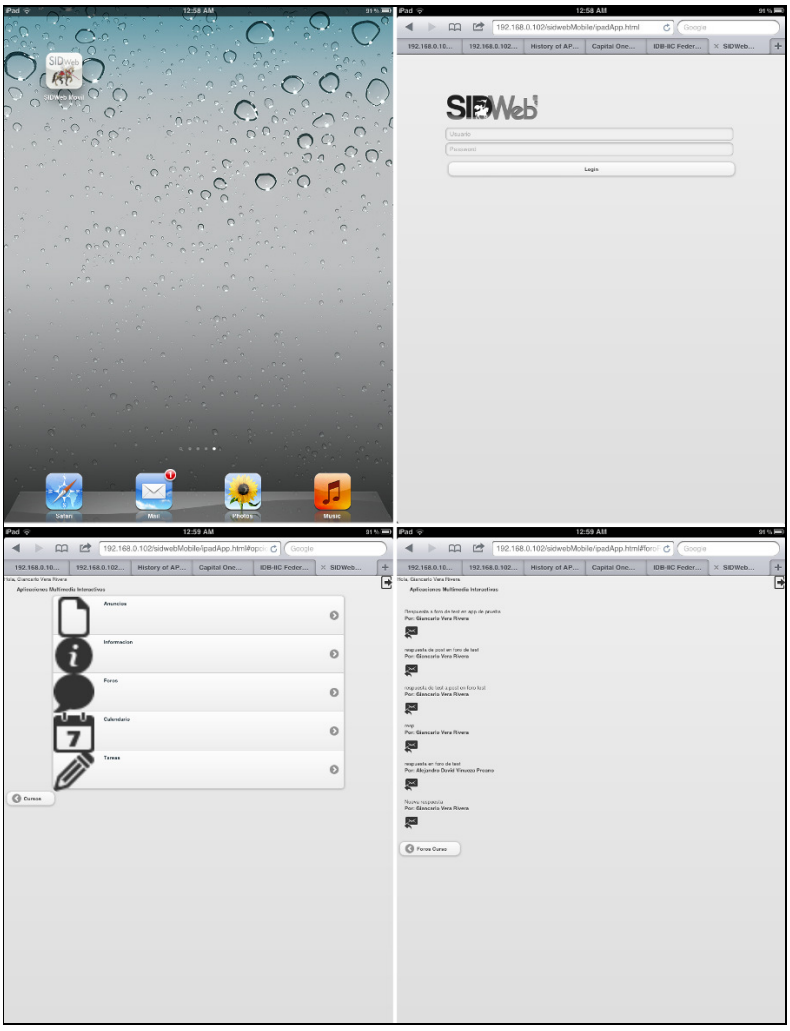

Figura 5.8: SIDWeb Móvil sobre iPad 2

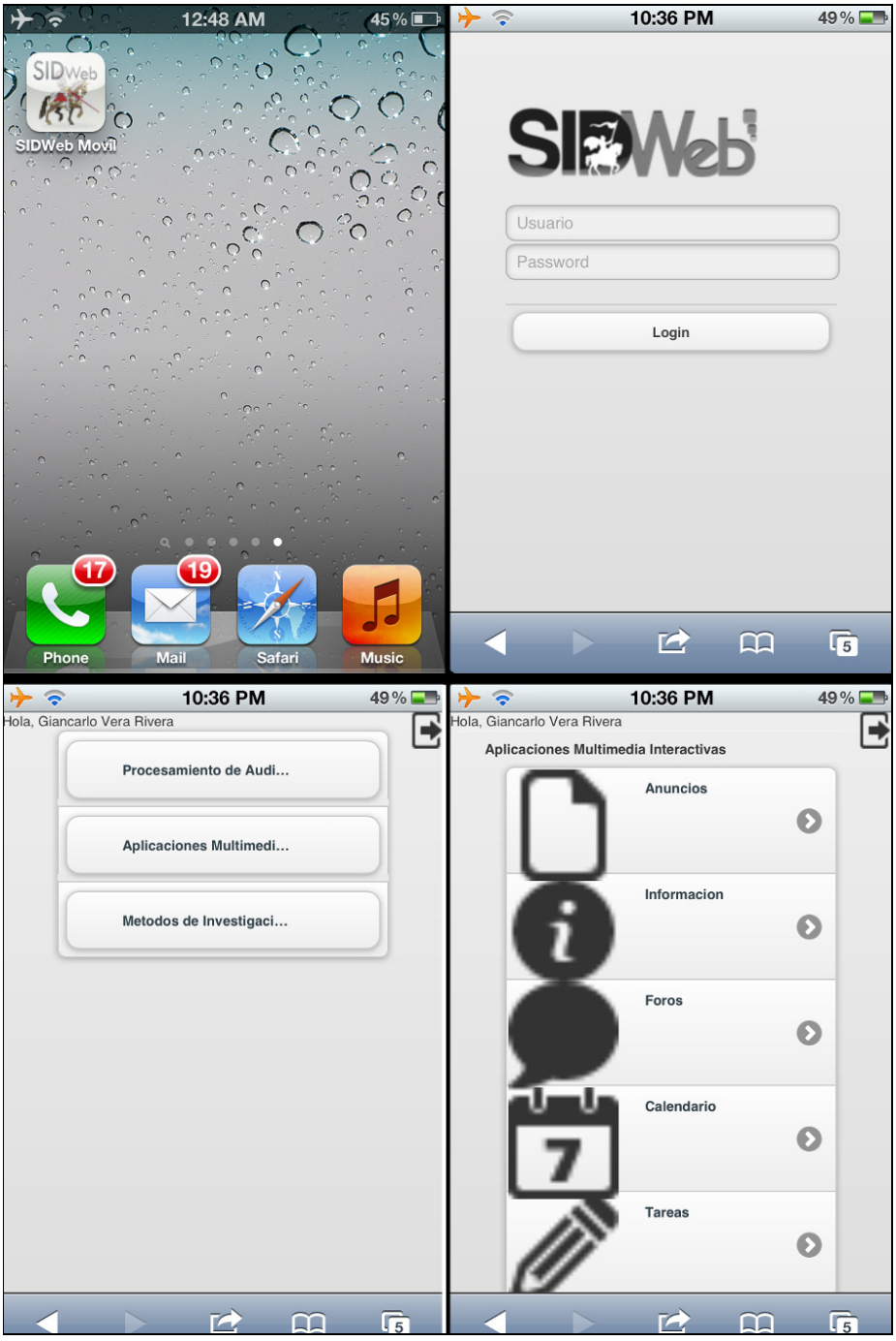

Figura 5.9: SIDWeb Móvil sobre iPhone 4.

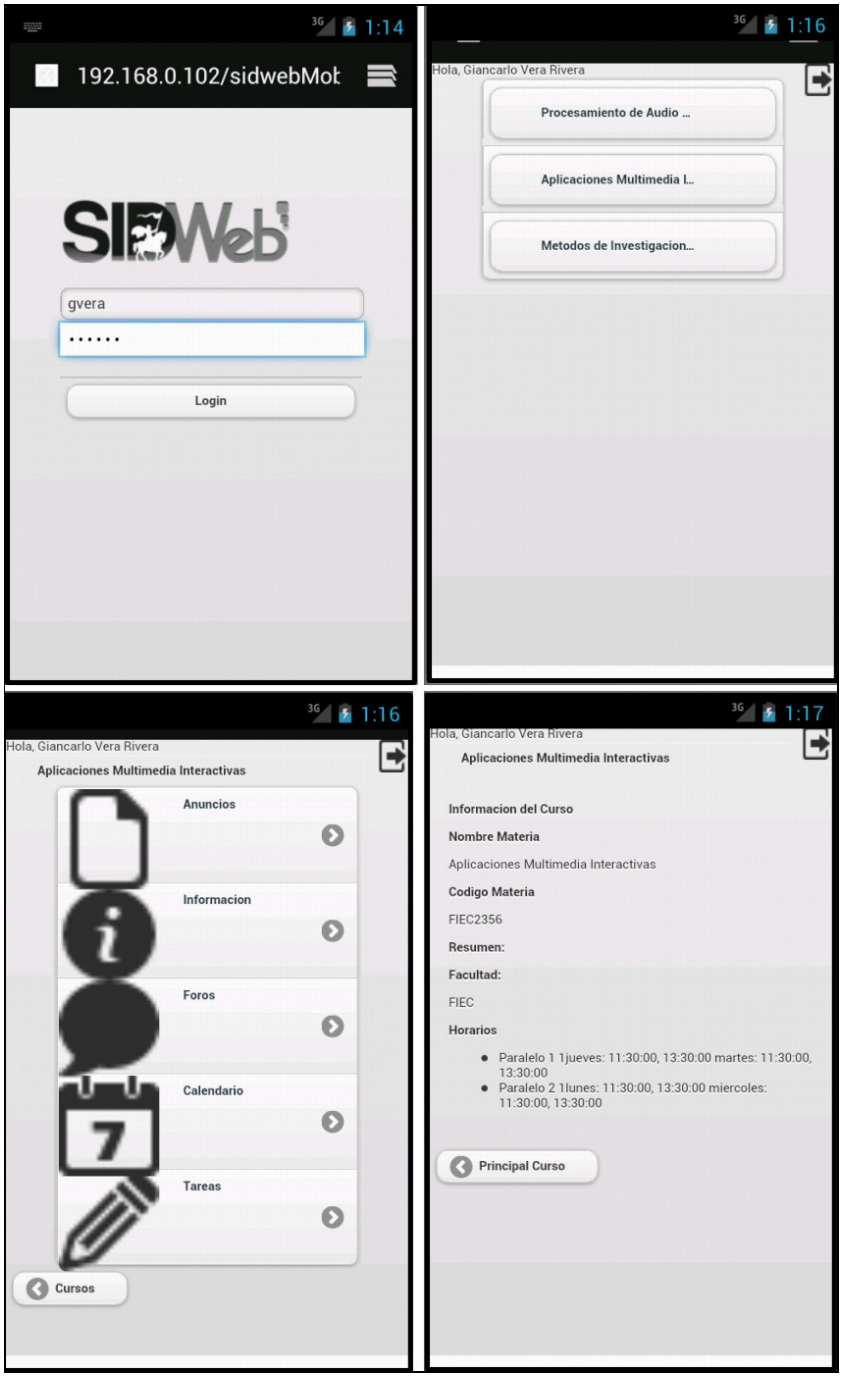

Figura 5.10: SIDWeb Móvil sobre Android

5.3. Pruebas con Usuarios

El objetivo de las pruebas con usuarios era constatar que se recibieran las respuestas adecuadas a las requisiciones realizadas a SIDWeb API mediante el uso de la aplicación SIDWeb Móvil. Se utilizaron las funcionalidades que ofrece el aplicativo móvil desarrollado sobre un dispositivo iPhone 4 con sistema operativo iOS 5, a cada usuario se le nombró las tareas que debía realizar, se constato que haya realizado la tarea y se midió el tiempo que le tomo realizarla. La tabla X muestra los resultados de la prueba.

Las pruebas se realizaron con cinco individuos los que fueron identificados con el código “Usr[x]” (donde [x] es un número entre 1 y 5), los perfiles de los usuarios eran variados, todos los usuarios estaban habituados al uso de aplicaciones Web, el usuario 1 y 5 eran estudiantes que usan el software SIDWeb, el usuario 2 y 3 tenían familiaridad con la administración del software SIDWeb, mientras que el usuario 4 no era estudiante ni tenía familiaridad con la administración del software. Para los ensayos se creó un curso de prueba que contenía datos en las secciones de anuncios, foros, calendario y tareas.

Tareas	Usr 1		Usr 2		Usr 3		Usr 4		Usr 5	
	Resultado	Tiempo (s)	Resultado	Tiempo (s)	Resultado	Tiempo (s)	Resultado	Tiempo (s)	Resultado	Tiempo (s)
Hacer Login	OK	15	OK	17	OK	16	OK	15	OK	16
Ingresar a Curso	OK	5	OK	5	OK	4	OK	4	OK	5
Ver Anuncios	OK	6	OK	8	OK	4	OK	7	OK	6
Ver Foros	OK	7	OK	6	OK	7	OK	6	OK	6
Responder Foro	OK	13	OK	14	OK	20	OK	15	OK	15
Ver Tareas	OK	5	OK	6	OK	7	OK	5	OK	6
Ver Calendario	OK	7	OK	8	OK	6	OK	6	OK	7
Hacer Logout	OK	3	OK	3	OK	4	OK	5	OK	4

Tabla IV: Resultados de las pruebas con usuarios

Como se observa en la tabla IV todos los usuarios realizaron satisfactoriamente las pruebas, los tiempos más extensos se observan en las tareas de “login” y “responder foro” dado que estas tareas implicaban ingreso de datos, mientras las tareas que solo implicaban navegación en la aplicación poseen tiempos mucho menores.

Se realizó una prueba sobre un dispositivo iPhone 3 bajo el sistema operativo iOS 3.1, esta prueba falló dado que el aplicativo SIDWeb Mobile esta construido sobre jQuery Mobile y este tiene soporte desde la versión 3.2 de iOS.

[1] Soporte de plataformas en jQuery Mobile
http://jquerymobile.com/gbs/

5.4. Discusión de los resultados

Los resultados esperados por este trabajo era construir una interfaz que habilite la comunicación del software SIDWeb 4 con otras aplicaciones las cuales se basen o integren sus servicios con SIDWeb 4, la interfaz desarrollada: SIDWeb API permite la comunicación de SIDWeb 4 con otras aplicaciones tal como se aplicó con la comunicación del aplicativo SIDWeb Móvil y la consola de pruebas, al haber sido desarrollado utilizando un formato de respuesta sencillo y liviano como JSON los tiempos de carga y actualización de información mejoran.

Durante las pruebas realizadas con los usuarios se recibieron recomendaciones acerca de la apariencia de los botones, la ubicación de los mismos y formatos de fechas, sin embargo se considero satisfactorios los resultados puesto que el objetivo era constatar que SIDWeb API estaba respondiendo adecuadamente a todas las requisiciones realizadas por el aplicativo móvil.

Mediante el aplicativo SIDWeb Móvil se reduce la cantidad de desplazamientos verticales para visualizar contenidos y se elimina la necesidad de realizar acercamientos para interactuar con los contenidos. El aplicativo brinda un acceso multiplataforma al haber sido probado en dos emuladores de dispositivos móviles y en dispositivos habilitados con el sistema operativo iOS v3.2-5 (iPhone, iPad).

CONCLUSIONES Y RECOMENDACIONES

Conclusiones

1. Mediante la construcción de SIDWeb API se constató la posibilidad de interactuar con aplicaciones externas, como es el caso del aplicativo SIDWeb Móvil que hace uso de los servicios que provee SIDWeb a través del API.
2. El uso de canales seguros para comunicación de datos en SIDWeb API es imprescindible dado que la encriptación que estos ayudan a prevenir el robo o acceso no autorizado a información sensible.
3. La utilización del Framework Symfony facilitó en gran medida el desarrollo del API, puesto que, aunque hay un tiempo involucrado para el aprendizaje del mismo, esto se ve recompensado al momento de la utilización de características del Framework que disminuyen el tiempo de desarrollo.
4. El uso de un formato de transmisión con poca sobrecarga como JSON es importante en interfaces como SIDWeb API dado que el generar contenidos más livianos lo hace más ventajoso considerando factores

críticos en aplicaciones móviles como el ancho de banda y la velocidad de procesamiento limitado.

Recomendaciones

1. El uso de jQuery Mobile el cual fue empleado en el desarrollo de SIDWeb Móvil, dio la posibilidad de realizar pruebas en distintas plataformas realizando el desarrollo en una sola aplicación, por lo que es recomendable considerar el uso de tecnologías multi-plataforma al momento del desarrollo de aplicativos que tienen por objeto ser utilizados en más de una plataforma.
2. SIDWeb API intercambia información sensible y protege estos datos mediante el uso de canales seguros, por lo que cualquier aplicación que gestione información sensible debe utilizar mecanismos para proteger tal información.
3. La consulta de documentación y de foros relacionados a las tecnologías empleadas en el desarrollo del SIDWeb API y SIDWeb Móvil redujo el tiempo de solución de problemas encontrados, por lo que se recomienda hacer uso de estos recursos al momento de solucionar problemas que se presentan en el desarrollo de sistemas.

BIBLIOGRAFÍA

1. VIRPI ROTO – ANTTI OULASVIRTA, Need for Non-Visual Feedback with Long Response Times in Mobile HCI, http:// research.nokia.com/files/MobileFeedback.pdf, fecha de consulta 28 Diciembre 2012, p. 1.
2. JUN GONG – PETER TARASEWICH, Guidelines for Handheld Mobile Device Interface Design, http://www.mendeley.com/research/guidelines-for-handheld-mobile-device-interface-design/, fecha de consulta 28 Diciembre 2012, p. 4.
3. MATT JONES – GARY MARSDEN – NORLIZA MOHD – KEVIN BOONE, Improving Web Interaction on Small Displays, http://www.cs.waikato.ac.nz/oldcontent/mattj/web8.pdf, fecha de consulta 8 Enero 2012, p. 8.
4. RAYMOND YEE, Pro Web 2.0 Mashups, Apress, 2008, p. xxvii.
5. GABRIEL SVENNERBERG, Beginning Google Maps API 3, Apress, 2010, p. 2.
6. DENISE GOSNELL, Professional Web APIs: Google, eBay, Amazon.com, MapPoint, FedEx, Wiley Publishing Inc., 2005, p.19.
7. JOHN PAUL MUELLER, Mining Google Web Services: Building Applications with the Google API, Sybex, 2006 p. 7.

8. SUBBU ALLAMARAJU, RESTful Web Services Cookbook, O'Reilly, 2010.
9. DANIEL JACOBSON – GREG BRAIL – DAN WOODS, API's: A Strategy Guide, O'Reilly, 2011, p.32.
10. KEVIN MAKICE, Twitter API Up and Running, O'Reilly, 2009, p. 2.
11. KEVIN MAKICE, Twitter API Up and Running, O'Reilly, 2009, p. 46.
12. DANIEL JACOBSON – GREG BRAIL – DAN WOODS, API's: A Strategy Guide, O'Reilly, 2011, p.32.
13. KEVIN MAKICE, Twitter API Up and Running, O'Reilly, 2009, p. 134 – 137.
14. FABIEN POTENCIER, Practical Symfony Create professional web applications with PHP and Symfony 1.3 & 1.4, Doctrine 1.2, Sensio S.A., 2011, p. 87.
15. FABIEN POTENCIER, Practical Symfony Create professional web applications with PHP and Symfony 1.3 & 1.4, Doctrine 1.2, Sensio S.A., 2011, p. 34.
16. LEONARD RICHARDSON, RESTful Webservices, O'Reilly, 2007, p.79.
17. NURZHAN NURSEITOV – MICHAEL PAULSON – RANDALL REYNOLDS – CLEMENTE IZURIETA, Comparison of JSON and XML Data Interchange Formats: A Case Study,

http://www.cs.montana.edu/izurieta/pubs/caine2009.pdf, fecha de consulta 5 Febrero 2012.

18. LEONARD RICHARDSON, RESTful Webservices, O'Reilly, 2007, p.44.
19. JON REID, jQuery Mobile, O'Reilly, 2011, p. 1.