DEFINICIÓN DE ESTÁNDARES Y RECURSOS COMO BUENAS PRÁCTICAS PARA LA ADOPCIÓN DE SERVICIOS WEB

Título	Instructivo para la definición de estándares y recursos como buenas prácticas para la adopción de servicios web.
Versión	1.0
Autor	Unidad Administrativa Especial de Catastro Distrital – Angel Lubiel Simbaqueba Ruiz.
Identificador	IPIG-01-1
Fecha de creación	2016-02-25
Descripción	El presente documento presenta y define un conjunto de pautas y prácticas que deben respetar los servicios web para lograr una interoperabilidad óptima.
Publicador	Unidad Administrativa Especial de Catastro Distrital – UAECD.
Colaboradores	No aplica.
Tipo	Texto.
Formato	Microsoft Word (.doc)
Fuente	Lineamiento 8 sobre la implementación de servicios web en el marco de la definición de la política de gestión de información geoespacial para el Distrito Capital.
Idioma	Español.
Cobertura	Bogotá Distrito Capital
Derechos	Copyright.
Palabras claves	Servicios Web, WS-I, interoperabilidad, SOA, SOAP, XML, WSDL, UDDI, Axis2, REST, QA.

CONTROL DE VERSIONES

Fecha	Autor/ Modificado por	Versión	Cambio efectuado
2016-02-25	Angel Lubiel Simbaqueba Ruiz	1.0	Primera versión del documento. No hay cambios para registrar.

CONTENIDO

1. OBJETIVO Y ALCANCE

1.1 OBJETIVO

El presente instructivo tiene por objeto presentar y definir un conjunto de pautas y prácticas que deben respetar los servicios web para lograr una interoperabilidad óptima, de forma tal que se asegure que los servicios web estén diseñados para funcionar bien para la mayor cantidad de usuarios potenciales, adoptando las mejores prácticas que se pueden aplicar para evitar estos problemas de interoperabilidad que pueden llegar a marcar la diferencia entre un servicio web que es fácil de manejar para los usuarios y uno que no lo está.

1.2 ALCANCE

El presente instructivo pretende abarcar a todas las entidades del orden Distrital que forman parte de IDECA, y como complemento al desarrollo de las mismas actividades en la materia dentro de IDE de Bogotá, buscando aprender sobre la Interoperabilidad de los servicios web, o WS-Interoperability, y aprender sobre Perfil Básico WS-I, que es un conjunto de pautas que deben respetar los servicios web para lograr una interoperabilidad óptima.

2. DEFINICIONES, SIGLAS Y ABREVIATURAS

A	
Apache Axis2	Apache Axis2 es un motor nuclear para servicios web. Es un rediseño total y una re implementación completa de la ampliamente difundida pila SOAP "Apache Axis"[i].
C	
Clase java	Las clases en Java son plantillas para la creación de objetos, en lo que se conoce como programación orientada a objetos, la cual es una de los principales paradigmas de desarrollo de software en la actualidad[ii].
CORBA	Common Object Request Broker Architecture (CORBA) es un estándar definido por Object Management Group (OMG) que permite que diversos componentes de software escritos en múltiples lenguajes de programación y que corren en diferentes computadoras, puedan trabajar juntos; es decir, facilita el desarrollo de aplicaciones distribuidas en entornos heterogéneos[iii].
E	
Eclipse	Es una plataforma de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores[iv].
H	
HTML	HTML, sigla en inglés de HyperText Markup Language (lenguaje de marcas de hipertexto), hace referencia al lenguaje de marcado para la elaboración de páginas web. Es un estándar que sirve de referencia del software que conecta con la elaboración de páginas web en sus diferentes versiones, define una estructura básica y un código (denominado código HTML) para la definición de contenido de una página web, como texto, imágenes, videos, juegos, entre otros[v].
HTTP	Hypertext Transfer Protocol - Protocolo de transferencia de hipertexto, es el protocolo de comunicación que permite las transferencias de información en la World Wide Web[vi].
I	

IDE	Un ambiente de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software[vii].
Interoperabilidad	Se define interoperabilidad como la habilidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada[viii].
Instructivo	Documento que detalla la forma de llevar a cabo una generalidad o una actividad de un proceso o un procedimiento[1].
J	
J2EE	Java Platform, Enterprise Edition o Java EE (anteriormente conocido como Java 2 Platform, Enterprise Edition o J2EE hasta la versión 1.4; traducido informalmente como Java Empresarial), es una plataforma de programación—parte de la Plataforma Java—para desarrollar y ejecutar software de aplicaciones en el lenguaje de programación Java[ix].
JAVA	Es un lenguaje de programación de propósito general, concurrente, orientado a objetos que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible[x].
JDK	Java Development Kit o (JDK), es un software que provee herramientas de desarrollo para la creación de programas en Java[xi].
JMS	La API Java Message Service (en español servicio de mensajes Java), también conocida por sus siglas JMS, es la solución creada por Sun Microsystems para el uso de colas de mensajes. Este es un estándar de mensajería que permite a los componentes de aplicaciones basados en la plataforma Java2 crear, enviar, recibir y leer mensajes. También hace posible la comunicación confiable de manera síncrona y asíncrona[xii].
JSON	JSON, acrónimo de JavaScript Object Notation, es un formato de texto ligero para el intercambio de datos. JSON es un subconjunto de la notación literal de objetos de JavaScript aunque hoy, debido a su amplia adopción como alternativa a XML, se considera un formato de lenguaje independiente[xiii].
M	
Middleware	Middleware o lógica de intercambio de información entre aplicaciones ("interlogical") es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones, o paquetes de programas, redes, hardware y/o sistemas operativos[xiv].
P	

[1] 03-01-PR-01 Procedimiento Administración Documental – UAECD.

Perfil Básico WS-I	Un conjunto de definiciones/especificaciones comúnmente aceptadas por la industria y a partir del apoyo a estándares basados en XML, junto con un conjunto de indicaciones que recomiendan cómo se deben usar las especificaciones para desarrollar servicios web interoperables entre sí[xv].
PHP	PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico[xvi].
Protocolo	En redes, un protocolo de comunicaciones o protocolo de red es la especificación de una serie de reglas para un tipo particular de comunicación. La red Internet se basa en el modelo de referencia TCP/IP (Transmission Control Protocol/Internet Protocol) que toma su nombre de los dos principales protocolos que regulan la comunicación a través de esta red[xvii].
Python	Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma[xviii].
R	
REST	Representational State Transfer - Arquitectura que, haciendo uso del protocolo HTTP, proporciona una API que utiliza cada uno de sus métodos (GET, POST, PUT, DELETE) para poder realizar diferentes operaciones entre la aplicación que ofrece el servicio web y el cliente.
S	
Servicio Web	Conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web. Estas aplicaciones o tecnologías intercambian datos entre sí con el objetivo de ofrecer unos servicios. Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web[xix].
SMTP	Simple Mail Transfer Protocol, protocolo para transferencia simple de correo", es un protocolo de red utilizado para el intercambio de mensajes de correo electrónico entre computadoras u otros dispositivos[xx].
SOA	Arquitectura Orientada a Servicios - SOA - Service Oriented Architecture. Paradigma de arquitectura para diseñar y desarrollar sistemas distribuidos.
SOAP	Protocolo Simple de Acceso a Objetos, el cual es un protocolo basado en XML, que permite la interacción entre varios dispositivos y que tiene la capacidad de transmitir información compleja.

SSL	Transport Layer Security (TLS; en español "seguridad de la capa de transporte") y su antecesor Secure Sockets Layer (SSL; en español "capa de conexión segura") son protocolos criptográficos que proporcionan comunicaciones seguras por una red, comúnmente Internet[xxi].
T	
TCP/IP	La familia de protocolos de Internet es un conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre computadoras. En ocasiones se le denomina conjunto de protocolos TCP/IP, en referencia a los dos protocolos más importantes que la componen, que fueron de los primeros en definirse, y que son los dos más utilizados de la familia: TCP (Transmission Control Protocol), Protocolo de Control de Transmisión, e, IP (Internet Protocol), Protocolo de Internet[xxii].
U	
UDDI	UDDI (Universal Description, Discovery and Integration), protocolo para publicar la información de los servicios Web. Permite comprobar qué servicios web están disponibles.
URL	Se entiende por localizador de recursos uniforme (conocido por la sigla URL, del inglés Uniform Resource Locator) es un identificador de recursos uniforme (Uniform Resource Identifier, URI) cuyos recursos referidos pueden cambiar, esto es, la dirección puede apuntar a recursos variables en el tiempo. Están formados por una secuencia de caracteres, de acuerdo a un formato modélico y estándar, que designa recursos en una red, como Internet[xxiii].
W	
WSDL	Lenguaje de Descripción de Servicios Web , Web Services Description Language, es el lenguaje de la interfaz pública para los servicios Web. Es una descripción basada en XML de los requisitos funcionales necesarios para establecer una comunicación con los servicios Web.
WS-Security	Web Service Security,WS-Security (Seguridad en Servicios Web) es un protocolo de comunicaciones que suministra un medio para aplicar seguridad a los Servicios Web[xxiv].
X	
XML	XML (Extensible Markup Language), es el formato estándar para los datos que se vayan a intercambiar.
XSD	Viene de XML Schema, es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML de una forma muy precisa, más allá de las normas sintácticas impuestas por el propio lenguaje XML[xxv].

XSLT	XSLT o Transformaciones XSL es un estándar de la organización W3C que presenta una forma de transformar documentos XML en otros e incluso a formatos que no son XML[xxvi].

3. GENERALIDADES

Partiendo del objetivo que los servicios web es permitir la comunicación entre distintos sistemas de software y hardware, donde se pueden encontrar sistemas que difieren típicamente en sus configuraciones, aún así se pueden presentar problemáticas en ello que inducen a definir protocolos estándar de comunicación, como los utilizados en la construcción de servicios web. Sin embargo, surgen temas de incompatibilidad incluso cuando se utilizan estos protocolos estándar, los cuales pueden conducir a grandes problemas de interoperabilidad de los servicios web. Con el desarrollo de este capítulo de generalidades se busca aprender, entre otros aspectos, sobre el Perfil Básico WS-I, que es un conjunto de pautas que deben respetar los servicios web para lograr una interoperabilidad óptima.

Para comprender estas mejores prácticas que se deben aplicar para evitar problemas en la interoperabilidad que afectan a los servicios web, también se debe tener una comprensión básica del Protocolo simple de acceso a objetos (SOAP) y las tecnologías relacionadas, como por ejemplo WSDL, Java y el kit de herramientas SOAP Apache Axis2 SOAP, como se abordó en el instructivo guía para la creación de servicios web bajo los principios de interoperabilidad web segura[xxvii].

3. 1 ¿POR QUÉ DE LA NECESIDAD DE LOS SERVICIOS WEB?

Por varias generaciones, los desarrolladores de software han estado creando programas de software que permitieron comunicar sistemas de información entre sí, aprovechando los avances en los lenguajes de programación, que en su momento, venían con bibliotecas de software que hacían que la programación de redes fuese posible, fácil y predecible.

Esta comunicación entre sistemas de información surgió con el uso de protocolos de alto nivel como HTTP, HTML, TCP/IP, que aún brindan un conjunto de reglas comunes que hacen posible que dos programas informáticos se comuniquen.

Pero lo anterior, sumado a la capacidad para responder rápidamente ante los cambios y optimizar los procesos de negocio, ha creado en ello un factor clave para la competitividad y el crecimiento de las organizaciones. Sin embargo, la agilidad de estas respuestas puede verse cuestionada si se apoya en entornos de IT que no pueden responder de forma flexible a los cambios que afectan a la actividad de negocio.

Para ello, surge la Arquitectura Orientada a Servicios (SOA, Service Oriented Architecture) como filosofía de diseño que permite un mejor alineamiento de las Tecnologías de Información (IT) con las necesidades de negocio, permitiendo a empleados, clientes y socios comerciales responder de forma más rápida y adaptarse adecuadamente a las presiones del mercado.

Se empezaron a desarrollar y disponer de herramientas, tecnologías, marcos de trabajo y guías necesarias para crear y mantener soluciones basadas en SOA, permitiendo la integración de aplicaciones independientes de manera que desde la red pueda accederse a sus funcionalidades, las cuales se ofrecen como servicios; y la forma más habitual de implementarla es mediante Servicios Web, que es una tecnología basada en aplicaciones que utilizan estándares para el transporte, codificación y protocolo de intercambio de información, permitiendo la intercomunicación entre sistemas de cualquier plataforma, para ser utilizada en una gran variedad de escenarios de integración.

Hoy día, los servicios Web se basan en un conjunto de estándares de comunicación, como son XML para la representación de datos, SOAP (Simple Object Access Protocol) para el intercambio de datos, y el lenguaje WSDL (Web Services Description Language) para describir las funcionalidades de un servicio Web.

Ahora, con esta estrategia de orientación a servicios se permite la creación de servicios y aplicaciones compuestas que pueden existir con independencia de las tecnologías subyacentes, como también con este modelo de servicios se facilita el acceso y consumo de los recursos de IT a través de la red, en lugar de exigir que todos los datos y lógica de negocio residan en un mismo servidor.

Así pues, esta estrategia de orientación a servicios mediante el modelo de servicios web libera el potencial que poseen las aplicaciones y recursos de IT, y lo hace disponible de forma general a toda la organización, facilitando la optimización de procesos y mejorando la agilidad empresarial.

3.2 ¿CUÁL ES LA COMPLEJIDAD Y DESAFÍO DE LOS SERVICIOS WEB?

Como se ha venido mencionando, los servicios web se basan en un conjunto de estándares de comunicación que permiten la intercomunicación entre sistemas de cualquier plataforma, pero aún así se presentan grandes diferencias en cómo dos programas de servicios web interpretan los mensajes SOAP que intercambian entre ellos, diferencias que se dan en las

especificaciones que definen las diversas y complejas tecnologías de servicios web, como SOAP, WSDL, UDDI, XML Schema y HTTP.

A pesar de la simplicidad que inspiran los servicios web mediante el uso de diversas tecnologías, la construcción actual de ellos exige *know-how*. Por ello, el construir servicios web en la actualidad, debe exigir un programador experimentado que conozca al menos un lenguaje y entorno de programación; debe conocer las redes informáticas y una familia de tecnologías estándar utilizadas en los servicios web actuales, como XML, SOAP, WSDL o XML Schema, además de comprender una cantidad de técnicas prácticas necesarias para utilizar estas tecnologías.

3.3 ¿POR QUÉ RESPETAR LAS MEJORES PRÁCTICAS Y DISCIPLINA?

En aras de ayudar a catalogar y solucionar estos problemas de interoperabilidad que se han venido mencionando con los servicios web, se ha creado la organización Web Services Interoperability Organization (WS-I) quien ha producido un conjunto de recomendaciones para desarrolladores de servicios web, que pueden ayudar a producir servicios web y consumir servicios web, y que estos funcionen bien juntos, es decir, exista un entendimiento para el intercambio de información; a este conjunto de recomendaciones se le conoce como Perfil Básico WS-I.

El hecho de comprender los problemas de interoperabilidad de servicios web y diseñar servicios proactivamente para evitarlos se ha convertido en una responsabilidad importante hoy día para los desarrolladores de servicios web.

Aunque no hay una solución definitiva para eliminar todos los problemas de interoperabilidad, más aún cuando en la computación distribuida existe un entorno heterogéneo y de actividad compleja, la mayoría de estos problemas de interoperabilidad se pueden evitar, con disciplina y las mejores prácticas descritas en el Perfil Básico WS-I.

3.4 ¿CUÁLES CON LOS ACTORES PRINCIPALES SOBRE SERVICIOS WEB?

En el ámbito de los servicios web, se sugiere describir los actores en términos de proveedores y consumidores, para ello se tienen las siguientes definiciones:

- **Consumidor:** es aquel actor consumidor responsable de hacer solicitudes a un servicio que implementa un actor proveedor.

- **Proveedor**: es aquel actor proveedor que es responsable de escuchar y procesar las solicitudes de servicio del actor consumidor.

3.5 ¿QUÉ SON LOS SERVICIOS WEB SOAP?

Partiendo de la definición de la sigla SOAP (Simple Object Access Protocol) se entiende como un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML[xxviii].

Básicamente SOAP es un paradigma de mensajería de una dirección sin estado, que puede ser utilizado para formar protocolos más complejos y completos según las necesidades de las aplicaciones que lo implementan.

Este protocolo SOAP está basado en XML y está compuesto de tres partes:

- **Sobre (envelope):** el cual define qué hay en el mensaje y cómo procesarlo.
- **Conjunto de reglas de codificación:** para expresar instancias de tipos de datos.
- **La Convención:** para representar llamadas a procedimientos y respuestas.

Además, el protocolo SOAP tiene tres características principales:

- **Extensibilidad:** donde la Seguridad y Web Security-routing son extensiones aplicadas en el desarrollo.
- **Neutralidad:** que el SOAP puede ser utilizado sobre cualquier protocolo de transporte como HTTP, SMTP, TCP o JMS.
- **Independencia:** que el SOAP permite cualquier modelo de programación.

Un mensaje SOAP es un documento XML con una estructura definida en la especificación del protocolo, donde dicha estructura la conforman las siguientes partes:

- **Envelope (obligatoria):** se considera la raíz de la estructura, es la parte que identifica al mensaje SOAP como tal.
- **Header:** esta parte es un mecanismo de extensión el cual permite enviar información relativa a como debe ser procesado el mensaje. El elemento "Header" se compone a su vez de "Header Blocks" que delimitan las unidades de información necesarias para el header.
- **Body (obligatoria):** contiene la información relativa a la llamada y la respuesta.

- **Fault:** bloque que contiene información relativa a errores que se hayan producido durante el procesado del mensaje y el envío desde el "SOAP Sender" hasta el "Ultimate SOAP Receiver".

Ventajas en el uso de SOAP:

- Debido al uso de XML el SOAP permite invocar procedimientos remotos de muchos lenguajes, por lo tanto, presenta una gran interoperabilidad.
- Al utilizar una comunicación vía HTTP es fácilmente escalable, además de ser casi siempre permitido por los cortafuegos o firewalls.
- Puede ser implementado utilizando cualquier lenguaje y ejecutado en cualquier plataforma.
- Es posible utilizarlo mediante usuario anónimo y mediante autentificación.
- Es posible transmitirlo mediante cualquier protocolo de transporte capaz de transmitir texto, típicamente HTTP o SMTP.

Desventajas en el uso de SOAP:

- Debido al uso de XML para el paso de mensajes, SOAP es considerablemente más lento que otros middleware como CORBA ya que los datos binarios se codifican como texto. Para contrarrestar este punto débil en el caso de XML con código binario incrustado se desarrolló un método optimizado de transmisión de mensajes.
- Depende del WSDL (Web Services Description Language).
- Al contrario que Java, PHP o Python, ciertos lenguajes no ofrecen un apoyo adecuado para su uso ya sea a nivel de integración o de soporte IDE.

SOAP es considerado muy potente; pero no es simple, dado que es una tecnología complicada cuya especificación para desarrollo contiene ejemplos y lenguaje confuso para que cualquier desarrollador lo entienda, esto por ende condujo a varios problemas de interoperabilidad evidenciado en las implementaciones de software que diferían en los formatos de mensaje que se generaban y esperaban.

Ante estos problemas con SOAP, en la actualidad hay otros tipos de servicios, entre ellos los más notables son los servicios web inspirados por el concepto de Transferencia de Estado Representacional (REST). El REST se considera más un estilo arquitectónico que una tecnología particular o un estándar formal. Así, los servicios construidos con RESTful (también reciben este nombre) no comparten un formato del mensaje común. El formato queda totalmente a discreción del diseñador del servicio. Por ende, los servicios RESTful por diseños sufren los mismos problemas de interoperabilidad que los servicios SOAP. Sin embargo, muchos usuarios encuentran a los servicios RESTful más fáciles de construir y utilizar.

Otras diferencias que podemos encontrar entre las especificaciones SOAP y REST son:

Servicios REST:

- Arquitectura sencilla que se ejecuta sobre HTTP
- Como se mencionó, es un "estilo de arquitectura" que explora de una forma básica la tecnología existente y los protocolos de la Web, incluyendo XML y HTTP.
- Al utilizar HTTP, es mucho más sencillo el desarrollar APIs, crear clientes y la documentación es más fácil de entender.
- Permite varios formatos de datos, dando al desarrollador la posibilidad de utilizar JSON que normalmente es más rápido y como permite la utilización de JSON, permite también un mejor soporte a los clientes del explorador.
- Tiene mejor escalabilidad y rendimiento, donde las lecturas del REST se pueden cachear.
- Soporta el esquema de seguridad SSL
- No tiene un sistema de mensajería estándar y no puede lidiar con la comunicación de fallos.
- Ideal para desarrollar soluciones en formato sencillo y práctico.

Servicios SOAP:

- Es una especificación de protocolo para intercambiar información estructurada en la implementación de Servicios Web
- Solamente permite XML.
- Las lecturas basadas en SOAP no se pueden cachear.
- En cuanto a seguridad, SOAP soporta el esquema de seguridad SSL, pero también soporta WS-Security lo que añade características de seguridad Enterprise.
- Proporciona una implementación estándar de integridad de datos y privacidad de datos.
- Proporciona fiabilidad en el sentido de lidiar con la comunicación de fallos y tiene un sistema de mensajería estándar, incluso a través de intermediaros SOAP.
- Cuando la razón de una solución es la seguridad, SOAP puede ser una opción útil e importante.

3.6 ¿Qué significa el Perfil Básico WS-I?

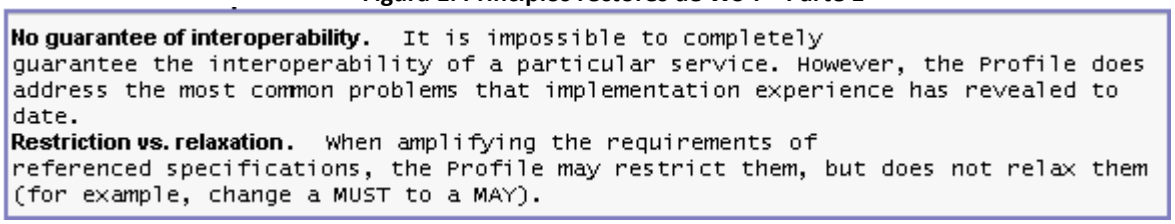
El "Perfil Básico WS-I", es una especificación técnica con un conjunto de pautas de implementación de servicios web destinadas a evitar los problemas de interoperabilidad, consideradas también como un conjunto invaluable de mejores prácticas.

A este Perfil Básico WS-I, se le asocia con la organización Web Services Interoperability Organization (WS-I), una organización de la industria TI con miembros que incluyen a IBM, Microsoft, Intel, Oracle, SAP, Hitachi y muchas otras compañías líderes. El objetivo de la WS-I

es lograr la interoperabilidad de servicios web “a través de plataformas, sistemas operativos y lenguajes de programación”. Como ente u organización, dispone de un grupo de expertos que brindan asesoramiento a desarrolladores de servicios web para ayudarlos a producir software de servicios web inter operativos.

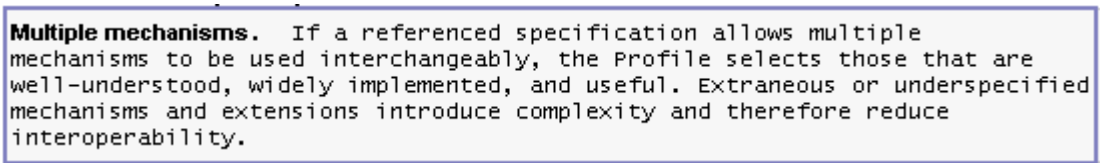
Principios rectores del Perfil Básico WS-I[xxix]

Figura 1. Principios rectores de WS-I – Parte 1



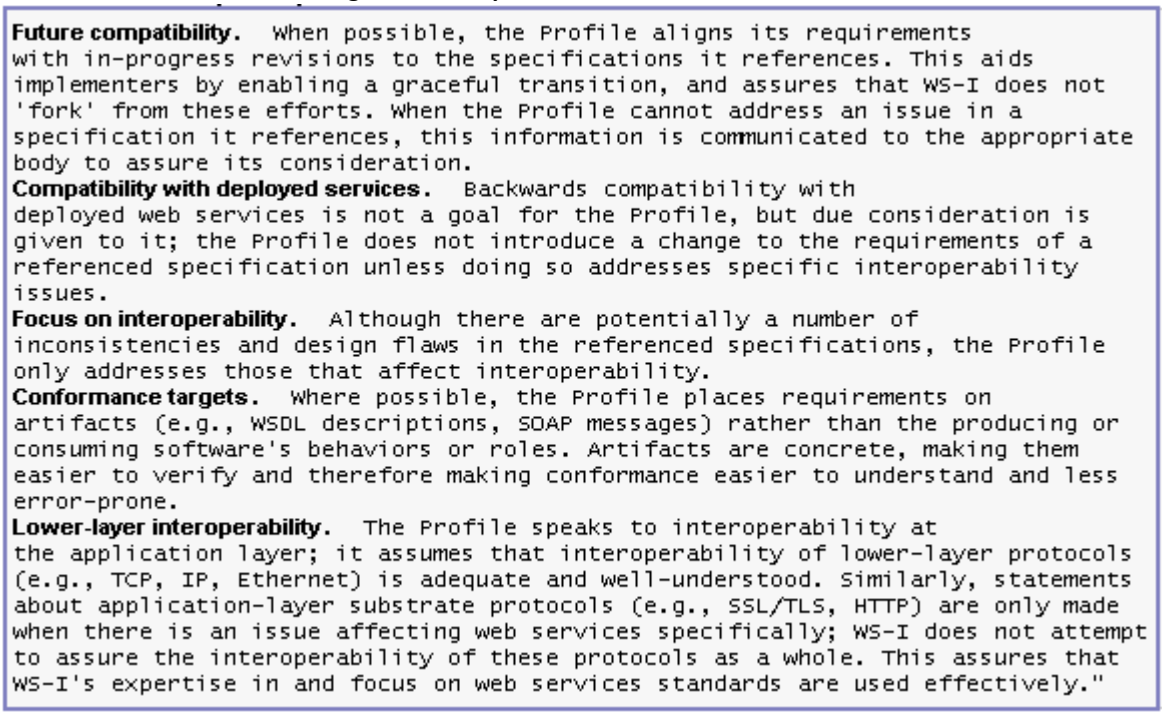
No guarantee of interoperability. It is impossible to completely guarantee the interoperability of a particular service. However, the Profile does address the most common problems that implementation experience has revealed to date.
Restriction vs. relaxation. When amplifying the requirements of referenced specifications, the Profile may restrict them, but does not relax them (for example, change a MUST to a MAY).

Figura 2. Principios rectores de WS-I – Parte 2



Multiple mechanisms. If a referenced specification allows multiple mechanisms to be used interchangeably, the Profile selects those that are well-understood, widely implemented, and useful. Extraneous or underspecified mechanisms and extensions introduce complexity and therefore reduce interoperability.

Figura 3. Principios rectores de WS-I – Parte 3



Future compatibility. When possible, the Profile aligns its requirements with in-progress revisions to the specifications it references. This aids implementers by enabling a graceful transition, and assures that WS-I does not 'fork' from these efforts. When the Profile cannot address an issue in a specification it references, this information is communicated to the appropriate body to assure its consideration.
Compatibility with deployed services. Backwards compatibility with deployed web services is not a goal for the Profile, but due consideration is given to it; the Profile does not introduce a change to the requirements of a referenced specification unless doing so addresses specific interoperability issues.
Focus on interoperability. Although there are potentially a number of inconsistencies and design flaws in the referenced specifications, the Profile only addresses those that affect interoperability.
Conformance targets. Where possible, the Profile places requirements on artifacts (e.g., WSDL descriptions, SOAP messages) rather than the producing or consuming software's behaviors or roles. Artifacts are concrete, making them easier to verify and therefore making conformance easier to understand and less error-prone.
Lower-layer interoperability. The Profile speaks to interoperability at the application layer; it assumes that interoperability of lower-layer protocols (e.g., TCP, IP, Ethernet) is adequate and well-understood. Similarly, statements about application-layer substrate protocols (e.g., SSL/TLS, HTTP) are only made when there is an issue affecting web services specifically; WS-I does not attempt to assure the interoperability of these protocols as a whole. This assures that WS-I's expertise in and focus on web services standards are used effectively."

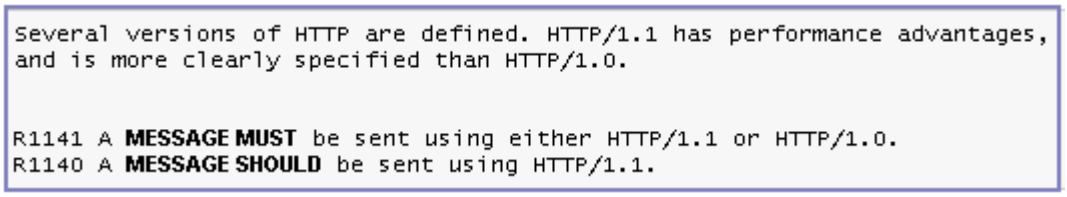
Taxonomía del Servicio WS-I

Dado que el Perfil Básico divide un servicio en "Objetivos de cumplimiento" o artefactos sobre los cuales hace recomendaciones, a continuación se presenta una lista de los artefactos que cubre dicha taxonomía del servicio WS-I:

- **MESSAGE:** son los elementos de protocolo que transportan el "envelope - Sobre", como los mensajes SOAP o HTTP.
- **ENVELOPE:** corresponde a la serialización del elemento soap:Envelope y su contenido.
- **DESCRIPTION:** son descripciones de tipo, mensajes, y los puntos de acceso de red están relacionados con los servicios web, como por ejemplo descripciones WSDL.
- **INSTANCE:** software que implementa unwsdl:porto unuddi:bindingTemplate.
- **CONSUMER:** software que invoca una INSTANCE.
- **SENDER:** software que genera un mensaje conforme a los protocoles relacionados con él.
- **RECEIVER:** software que consume un mensaje conforme a los protocolos relacionados con él, como por ejemplo procesadores SOAP.
- **REGDATA:** son elementos del registro que están involucrados en el registro y el descubrimiento de servicios web, como por ejemplo tModels UDDI.

El siguiente es un ejemplo de una recomendación de muestra extraída del Perfil Básico WS-I[xxx]:

Figura 4. Una recomendación WS-I de muestra



En la anterior muestra se puede observar que cada uno de los requisitos anteriores tiene un identificador único, que puede ser útil para citarlo en conversaciones o durante pruebas.

A continuación se ilustra como el Perfil Básico WS-I formaliza los términos que utiliza en sus recomendaciones:

Figura 5. Terminología del Perfil Básico WS-I

MUST. This word, or the terms "REQUIRED" or "SHALL," means that the definition is an absolute requirement of the specification.

MUST NOT. This phrase, or the phrase "SHALL NOT," means that the definition is an absolute prohibition of the specification.

SHOULD. This word, or the adjective "RECOMMENDED," means that there may exist valid reasons in particular circumstances to ignore a particular item, but the full implications must be understood and carefully weighed before choosing a different course.

SHOULD NOT. This phrase, or the phrase "NOT RECOMMENDED," means that there may exist valid reasons in particular circumstances when the particular behavior is acceptable or even useful, but the full implications should be understood and the case carefully weighed before implementing any behavior described with this label.

MAY. This word, or the adjective "OPTIONAL," means that an item is truly optional. One vendor may choose to include the item because a particular marketplace requires it or because the vendor feels that it enhances the product while another vendor may omit the same item. An implementation which does not include a particular option MUST be prepared to interoperate with another implementation that does include the option, though perhaps with reduced functionality. In the same vein, an implementation that does include a particular option MUST be prepared to interoperate with another implementation that does not include the option (except, of course, for the feature the option provides.)

Como nota importante con la definición de estos términos resulta relevante identificarlos puesto que en toda la recomendación del Perfil Básico se hace referencia a ella, más aún porque muchos problemas de interoperabilidad surgían de la ambigüedad de otras especificaciones.

4. INSTRUCCIÓN

En el proceso de adopción de mejores prácticas y lecciones aprendidas en una arquitectura orientada a servicios (SOA) e implementada por servicios web, es preciso y necesario identificar y seguir una serie de pasos que permitan solucionar problemas de incompatibilidad, los cuales pueden conducir a problemas de interoperabilidad entre sistemas de información a través de la comunicación con servicios web.

Para ello a continuación se dan los pasos en la definición de estándares y recursos como buenas prácticas para la adopción de servicios web, que se deben respetar para lograr una interoperabilidad óptima, y garantizar la calidad (QA) con que se deben abordar las pruebas dentro de una SOA; todo esto para que finalmente generen numerosos beneficios para las empresas, se aumente la alineación y la agilidad de las soluciones de software que implementen servicios web.

PASO 1. IDENTIFIQUE ESTOS PRERREQUISITOS

Para ilustrar la aplicación de las buenas prácticas para la adopción de servicios web, se mostraran ejemplos de código estándar que se pueden aplicar a cualquier lenguaje o entorno de programación. Para la ejecución de estos ejemplos que se mostraran en los próximos pasos de este instructivo, es necesario disponer del siguiente software instalado:

- Java 2 Standard Edition versión 1.4.2 o superior

- Apache Axis2 versión 1.0: Axis2 es un kit de herramientas SOAP de funciones completas que brindan implementaciones de varias API de servicio web que incluyen SOAP y WSDL; este kit de herramientas como Axis2 es relevante para el desarrollo de servicios web.

- Apache Geronimo u otro servidor de aplicación, se considera este servidor de aplicación Apache Geronimo J2EE (que es la base del servidor IBM WebSphere® Community Edition) porque es simple, liviano y gratuito, y se presta para instalar y ejecutar rápidamente.

PASO 2. COMBINE EL PERFIL BÁSICO WS-I Y SOAP

A continuación se describe algunas de las recomendaciones del Perfil Básico que podrían existir con los sobres SOAP, errores y el uso de HTTP.

Modo de utilizar paquetes de mensajes XML que se envían como "sobres" SOAP:

Dado que SOAP especifica un protocolo basado en mensajes XML que viajan en "*sobres",* la recomendación del Perfil Básico requiere que los servicios web utilicen este esquema tal cual como se define en la especificación SOAP, agregando varias restricciones a su uso. Dichas restricciones a considerar son:

- El Perfil requiere que los "*sobres*" se adapten a la estructura especificada en SOAP 1.1, aclarando que la mensajería SOAP 1.1 es la única estructura de mensajes que soporta el Perfil.

- Los "*sobres*" no deben contener una Document Type Declaration (DTD) o Instrucciones de Procesamiento, dado que éstas se citan por el Perfil como fuentes de vulnerabilidades de seguridad, sobrecarga de procesamiento y ambigüedad semántica.

- Los "*sobres*" no deben contener declaraciones de espacio de nombres XML xmlns:xml="http://www.w3.org/XML/1998/namespace y xmlns:xml="http://www.w3.org/XML/1998/namespace.

- En un "*sobre*", no debe haber elementos hermanos al elemento del cuerpo. El perfil dice que la interpretación de dichos elementos hermanos no es clara, así que el Perfil los prohíbe. Esto formaliza la noción de que un mensaje SOAP es un "sobre", con un único cuerpo, y contiene una carga útil.

- Dado que muchos problemas de interoperabilidad se pueden rastrear al uso de los mensajes XML codificados, el Perfil elimina los mensajes SOAP codificados prohibiendo el uso del atributo soap:encodingStyle.

Modo de utilizar errores SOAP:

Las siguientes son recomendaciones del Perfil Básico respecto a los sobres SOAP con error:

- Primero define que un error es un *sobre* que tiene un elemento individual secundario de soap:Body y de soap:Fault. Ninguna otra cosa debe interpretarse como error.

- Los códigos de estado HTTP no deben utilizarse para determinar la presencia de un error. Interprete el mensaje SOAP para determinar la presencia de un error.

- Cuando un sobre es un error, solo se permite tener elementos secundarios para faultcode, faultstring, faultactor y detalles. Estos elementos no deben estar calificados por espacio de nombres.

- Por extensibilidad, el Perfil Básico dice que los errores SOAP pueden tener cualquier cantidad de elementos secundarios que aparecen en el elemento detalle.

La aplicación de estas recomendaciones es relevante para los autores de servicios web en caso de encontrarse con problemas de interoperabilidad.

Modo de operar SOAP y HTTP:

Dado que HTTP es el protocolo de transporte más común y popular para los mensajes SOAP, y además que el Perfil Básico lo selecciona como el enlace a utilizar para los servicios web, se recomienda aplicar las siguientes buenas prácticas:

- Los mensajes se deben enviar con HTTP 1.1 y se debe utilizar siempre que sea posible.

- Los mensajes que solicitan HTTP deben utilizar el método HTTP POST y no deben utilizar HTTP Extension Framework, que permite SOAP 1.1.

- Se debe utilizar un código de estado HTTP 2XX para indicar el resultado exitoso de una solicitud HTTP. Cuando el mensaje de respuesta contiene un sobre que no es un error, se debe utilizar el código de estado 200. El código 200 o 202 debe indicar un resultado exitoso cuando no surge un sobre SOAP de la solicitud HTTP.

El valor del campo de acción SOAP en el encabezado HTTP utilizado con un mensaje de solicitud debe ser una cadena citada, esto es relevante al usar WSDL, como por ejemplo:

- Lo siguiente da como resultado encabezado HTTP que dice SOAPAction="someAction" :
<soapbind:operation soapAction="someAction" />

- Lo siguiente da como resultado un encabezado HTTP que dice SOAPAction="":
<soapbind:operation soapAction="" />

- Lo siguiente da como resultado un encabezado HTTP que dice SOAPAction="":
<soapbind:operation />

PASO 3. APLIQUE ESTAS RECOMENDACIONES SOBRE EL PERFIL BÁSICO WS-I Y WSDL

Un aspecto importante de las recomendaciones, para descripciones de servicios del Perfil Básico, consiste en establecer como requisito que esté disponible el WSDL de un proveedor de servicios para un consumidor de servicios web al solicitarlo. Esto resalta la importancia del WSDL como el contracto de servicios web que ayuda a promover un entorno de servicios que conduce a un uso "*ad hoc*" de dichos servicios web, de forma tal que siempre se podrá encontrar un servicio en un registro, solicitar su WSDL, interpretar WSDL y utilizar el servicio.

Entre las recomendaciones más relevantes se tienen:

- El Perfil Básico requiere que las descripciones del servicio se escriban en una versión XML 1.0 válida. Las especificaciones de WSDL o XML Schema no exigieron una versión particular.

- El Perfil Básico coloca restricciones en el uso de la instrucción de importación WSDL, restringiendo su uso a casos donde el documento que se importa es el mismo documento WSDL.

- Cuando importa un documento WSDL a otro documento WSDL, el Perfil Básico requiere que los dos documentos WSDL estén en el mismo espacio de nombre (el atributo targetNamespace en el elemento wsdl:definitions del WSDL que se importa deben tener el mismo valor que el atributo espacio de nombres en el elemento wsdl:import en el WSDL que realiza la importación.)

- El Perfil Básico recomienda que los elementos de un documento WSDL aparezcan en un orden específico. Los elementos de importación WSDL deben preceder a todos los demás elementos del espacio de nombres WSDL excepto wsdl:documentation. Los elementos de tipo WSDL deben preceder a todos los demás elementos del espacio de nombres WSDL exceptowsdl:documentation y wsdl:import. El elemento de documentación WSDL puede aparecer como el primer elemento secundario de wsdl:import, wsdl:part, y wsdl:definition además de los elementos citados en WSDL 1.1.

- El perfil advierte contra el uso de extensiones WSDL, porque fácilmente pueden conducir a problemas de interoperabilidad.

Respecto a introducir tipos de datos, el Perfil Básico elimina el tipo de array WSDL, como lo son: Wsdl:arrayType y soapenc:ArrayType. En cambio, los arrays se deben declarar utilizando una secuencia como la que se muestra a continuación:

Figura 6. Definición básica de array que cumple con el Perfil Básico

```
<xsd:element name="MyArray1" type="tns:MyArray1Type"/>
<xsd:complexType name="MyArray1Type">
  <xsd:sequence>
   <xsd:element name="x" type="xsd:string"
    minOccurs="0" maxOccurs="unbounded"/>
  </xsd:sequence>
</xsd:complexType>
```

Respecto al uso de los tipos de puertos, el Perfil Básico prohíbe la sobrecarga de nombres en un wsdl:portType. Esto significa que cada operación debe tener un valor definido para su atributo nombre.

Respecto al uso de enlaces, un wsdl:binding en una descripción debe ser un enlace rpc-literal o un enlace documento-literal, aspecto importante sobre el formato del mensaje.

Adicionalmente, un wsdl:binding en una descripción debe utilizar el valor de literal para que el atributo use en todos los elementos soapbind:body, soapbind:fault, soapbind:header y soapbind:headerfault.

PASO 4. RECONOZCA LOS ESCENARIOS DE USO DEL PERFIL BÁSICO WS-I

El WS-I define un conjunto de tres "Escenarios de uso" que describen tres patrones comunes para acceder a los servicios web y cómo se pueden implementar los artefactos que se encuentran en los servicios web, como por ejemplo WSDL.

Una de las funciones importantes de los siguientes escenarios de uso es que se pueden "componer", lo que significa que se pueden utilizar como bloques de construcción para armar escenarios de uso más complejos.

Primer escenario: escenario de uso unidireccional

El escenario de uso unidireccional es aplicable en situaciones en las cuales un consumidor de servicios web debe enviar un mensaje a un proveedor de servicios y el consumidor no

requiere una garantía de que el mensaje se entregará o procesará. La desventaja es que en caso de que algo salga mal, no hay un canal disponible para informar al consumidor sobre el fallo debido a las funciones de este escenario de uso:

- No se genera ni se espera un mensaje de respuesta SOAP.
- No se requiere al transporte subyacente que garantice la entrega del mensaje al proveedor.

Las ventajas de usar este escenario es que el escenario unidireccional es una buena elección porque es liviano. Puede ayudar a la escalabilidad de un sistema porque el consumidor no debe esperar que se procese o prepare la solicitud para ninguna clase de respuesta de parte del proveedor.

El documento escenarios de uso en el Perfil Básico WS-I detalla cómo se utiliza el escenario de uso unidireccional en WSDL. Este escenario unidireccional es importante cuando se requiere que se envíe un único mensaje de entrada al proveedor, y no se especifica ningún mensaje de salida.

Segundo escenario: escenario de uso solicitud/respuesta sincrónica
La solicitud/respuesta sincrónica es el tipo de patrón de uso más común para los servicios web y la computación distribuida en general, dado que permite describir con exactitud muchas situaciones que se experimentan en el mundo real. En este escenario es importante esperar a que el destinatario procese la información que le ha proporcionado y formule una respuesta que usted pueda utilizar.

En este escenario solicitud/respuesta sincrónica, el servicio recibe un mensaje de solicitud SOAP y produce un mensaje de respuesta SOAP que el cliente espera con ansiedad.

Tercer escenario: el escenario de uso básico de devolución de llamada
El escenario básico de devolución de llamada es para situaciones donde al consumidor le interesa recibir un resultado de parte del proveedor, pero el consumidor no desea quedarse a esperar el resultado.

La devolución de llamada básica se logra al componer dos escenarios de uso de devolución de llamada asincrónicos, una solicitud inicial del consumidor al proveedor, que en último lugar da como resultado una respuesta final del proveedor al consumidor, como se ilustra en la siguiente figura.

Figura 7. Escenario de devolución de llamada básico

```
1. Consumer initiates the service by sending a SOAP message bound to an HTTP
request to the Provider (the "initial request")

2. Provider acknowledges receipt using a SOAP message bound to an HTTP response to
the Consumer (the "initial response")

3. Provider completes the exchange by sending a SOAP message bound to an HTTP
request to the Consumer with the results of the initial request (the "final
request" or "callback")

4. Consumer acknowledges receipt of the callback message with a SOAP message
bound to an HTTP response (the "final response")
```

Importante aclarar que estos escenarios de uso no están especificados por ninguno de los estándares de servicios web; solo son patrones de comunicación comunes utilizados por los servicios web y las aplicaciones que los consumen, sin embargo es también importante catalogar estos escenarios, como lo ha hecho WS-I, porque los escenarios les dan a los desarrolladores asesoramiento sobre cómo implementarlos para que sean inter operativos.

PASO 5. UTILICE LAS HERRAMIENTAS DE PRUEBA WS-I

La WS-I ofrece un conjunto de herramientas de prueba que pueden utilizar los desarrolladores para verificar diversos artefactos de servicios web para el cumplimiento del Perfil Básico, de tal forma que facilita a los desarrolladores de servicios web comprobar que sus servicios cumplen los requisitos, y por ende no tienen la menor cantidad posible de problemas de interoperabilidad.

Para ello, se puede descargar una suite de herramientas de prueba directamente de WS-I. Esta suite incluye una herramienta llamada **Monitor** para interceptar mensajes que van hacia o provienen de un servicio web y una herramienta llamada **Analyzer** para inspeccionar diversos artefactos de servicios web, inclusive los mensajes interceptados utilizando Monitor.

Modo de uso de la herramienta Monitor del WS-I

Las herramientas de prueba WS-I incluyen una aplicación de monitoreo que tiene la capacidad de escuchar las conversaciones que ocurren entre un proveedor de servicios web y un consumidor, a manera de intermediario donde el consumidor habla con la aplicación de monitoreo, que reenvía solicitudes al proveedor sin modificarlas. Luego, el monitor reenvía toda respuesta del proveedor al consumidor, nuevamente sin alterarla de manera alguna,

llevando un registro de todo lo que ve. Puede ver mensajes SOAP y trafico HTTP: dos componentes importantes de los servicios web que se pueden analizar luego en busca de problemas de interoperabilidad.

Las pruebas WS-I incluyen un archivo de configuración de monitor de muestra que, para la versión Java de la suite, se puede encontrar en $WSI_HOME/java/samples/monitorConfig.xml.

A manera de ejemplo, se puede copiar este archivo de muestra a un espacio de trabajo local y se edita para especificar una ubicación para el archivo del registro que llevará.

El archivo de configuración de monitor, que también es un archivo de configuración XML, especifica los puertos donde debe esperar solicitudes de consumidores y los puertos de proveedores a los que debe reenviarlos. Los puertos de proveedores son bastante estándar. Estos puertos de escucha predeterminados deben ser seguros para la mayoría de los sistemas.

Una vez definidos estos números de puerto, se debe asegurar que los servicios de prueba se estén ejecutando, para luego iniciar el monitor utilizando un archivo de lote como (Monitor.bat) y el archivo de configuración que se haya creado para el servicio de prueba.

Figura 8. Resultado que surge del inicio de la herramienta WS-I Monitor

```
$ $WSI_HOME/java/bin/Monitor.bat -config ClassifiedSvcMonitorConfig.xml
Conformance Monitor Tool, Version: 1.0.0, Release Date: 2004-11-19
Copyright (C) 2002-2003 by The Web Services-Interoperability Organization and
Certain of its Members. All Rights Reserved. Use of this Material is governed by
WS-I licenses included within the documentation.

  comment ..................... This configuration file is used to test the WS-I
sample applications running on a single system.
  logURI ...................... log.xml
  replaceLog .................. true
  logDuration ................. 600
  timeout ..................... 3
  addStyleSheet ............... <?xml-stylesheet
href="../../wsi-test-tools/common/xsl/log.xsl" type="text/xsl" ?>
  man-in-the-middle comment ... null
  redirect [1]
    comment ................... This is a redirect example for local system, port
8080.
    listenPort ................ 4040
    host ...................... http://localhost:8080
    maxConnections ............ 1000
    readTimeoutSeconds ........ 15
  redirect [2]
    comment ................... This is a redirect example for local system, port
80.
    listenPort ................ 4041
    host ...................... http://localhost:80
    maxConnections ............ 1000
    readTimeoutSeconds ........ 15
  redirect [3]
    comment ................... This is a redirect example for local system, port
9080.
    listenPort ................ 8001
    host ...................... http://localhost:9080
    maxConnections ............ 1000
    readTimeoutSeconds ........ 15

The Monitor tool is ready to intercept and log web service messages.
Type "exit" to stop the Monitor.
```

A continuación, se debe cambiar los casos de prueba para utilizar URLs de servicio alternativo para que el Monitor tenga la oportunidad de interceptar mensajes de cada consumidor y reenviarlos a instancias de servicio en ejecución. Esto se realiza modificando cada prueba para qué cada una arme la instancia de stub de prueba con una URL de servicio explícito que anule la URL incluida en el WSDL y utilizado para crear el caso de prueba.

Para la prueba de servicio que se haya configurado, significa crear el stub como se muestra en la siguiente figura, esto para utilizar el puerto 4040 en vez de comunicarse directamente con el proveedor de servicios, que es el puerto 8080.

Figura 9. Construcción de ClassifiedServiceStub para la prueba

```
ClassifiedServiceStub stub =
new ClassifiedServiceStub(null,
"http://localhost:4040/axis2/services/ClassifiedService");
```

A continuación, se ejecuta cada caso de prueba para el servicio configurado, como se indica en la siguiente figura:

Figura 10. Ejecución de ClassifiedServiceTest

```
$ ant run.test
Buildfile: build.xml

init:

pre.compile.test:
     [echo] Stax Availability= true
     [echo] Axis2 Availability= true

compile.src:

compile.test:

run.test:
    [junit] Running com.bm.classifiedclient.ClassifiedServiceTest
    [junit] Tests run: 2, Failures: 0, Errors: 0, Time elapsed: 2.516 sec

BUILD SUCCESSFUL
Total time: 5 seconds
```

A medida que se ejecutan estas pruebas, se puede observar que el registro de la aplicación monitor salta cuando intercepta las solicitudes y respuestas del consumidor y el proveedor. Un ejemplo de las entradas del registro generado se muestran en la siguiente otra figura:

Figura 11. Entradas en el registro monitor

```
Log message entry [ID, Type, Sender]: 1, request, 127.0.0.1:4582
Log message entry [ID, Type, Sender]: 2, response, localhost:8080
```

Modo de uso de la herramienta Analyzer del WS-I

La herramienta de prueba analizador también utiliza un archivo de configuración XML. Para un ejemplo del uso, se debe empezar por copiar un archivo de muestra de la suite de prueba WS-I. Para la suite Java, el archivo de muestra se puede encontrar en $WSI_HOME/java/samples/analyzerConfig.xml. Se recomienda hacer dos copias: una copia para el servicio de prueba 1 y otra copia para el servicio de prueba 2.

La ventaja del programa analizador es la capacidad de procesar archivos del registro creado por el monitor para buscar mensajes a analizar, también puede analizar documentos WSDL y otros artefactos.

Luego se procede a editar el archivo de configuración de muestra para este servicio, agregando una referencia al servicio WSDL. El archivo de configuración resultante se muestra en la siguiente figura:

Figura 12. Referencia WSDL actualizada en el archivo de configuración del analizador

```
<wsi-analyzerConfig:wsdlReference>
  <wsi-analyzerConfig:wsdlElement type="port"
      parentElementName="BankService"
      namespace="http://userguide.axis2.apache.org/BankService">
    BankPort
  </wsi-analyzerConfig:wsdlElement>
  <wsi-analyzerConfig:wsdlURI>
    ./BankService.wsdl
  </wsi-analyzerConfig:wsdlURI>

</wsi-analyzerConfig:wsdlReference>
```

Luego, se debe asegurar que la configuración del analizador conozca la ubicación correcta del archivo de registro generado por nuestra sesión de monitoreo. Para ello se elige log.xml en el directorio actual. Esta modificación se ilustra en la siguiente figura:

Figura 13. Ubicación actualizada del archivo del registro en el archivo de configuración del analizador

```
<wsi-analyzerConfig:logFile correlationType="endpoint">
  ./log.xml
</wsi-analyzerConfig:logFile>
```

Seguidamente, se debe ejecutar el analizador, como se muestra en la siguiente otra figura:

Figura 14. Ejecución de la herramienta analizador

```
$ Analyzer.bat -config BankSvcAnalyzerConfig.xml
Conformance Analyzer Tool, Version: 1.0.0, Release Date: 2004-11-19
Copyright (C) 2002-2003 by The Web Services-Interoperability Organization and
Certain of its Members. All Rights Reserved. Use of this Material is governed by
WS-I licenses included within the documentation.

  verbose ..................... false
  Assertion Results:
    type ..................... all
    messageEntry ............. true
    assertionDescription ..... false
    failureMessage ........... true
    failureDetail ............ true
  Report File:
    replace .................. true
    location ................. report.xml
    Style Sheet:
      href ................... ../wsi-test-tools/common/xsl/report.xsl
      type ................... text/xsl
  testAssertionsFile .........
../wsi-test-tools/common/profiles/SSBP10_BP11_TAD.xml
  Message Log File:
    location ................. ./log.xml
    correlationType .......... endpoint
  WSDL Reference:
    WSDL Element:
      type ................... port
      namespace .............. http://userguide.axis2.apache.org/BankService
      name ................... BankPort
      parentElementName ...... BankService
    wsdlURI .................. bankservice/BankService.wsdl
  description ................. This file contains a sample of the configuration
file for the Basic Profile Analyzer, which can be used with the other sample
files.

Please wait while the specified artifacts are analyzed...
Conformance report has been created.
```

El resultado considerado el más interesante del analizador es el informe que produce. El informe es un archivo XML que se puede visualizar en cualquier navegador que entienda XSLT, como por ejemplo Internet Explorer o Firefox.

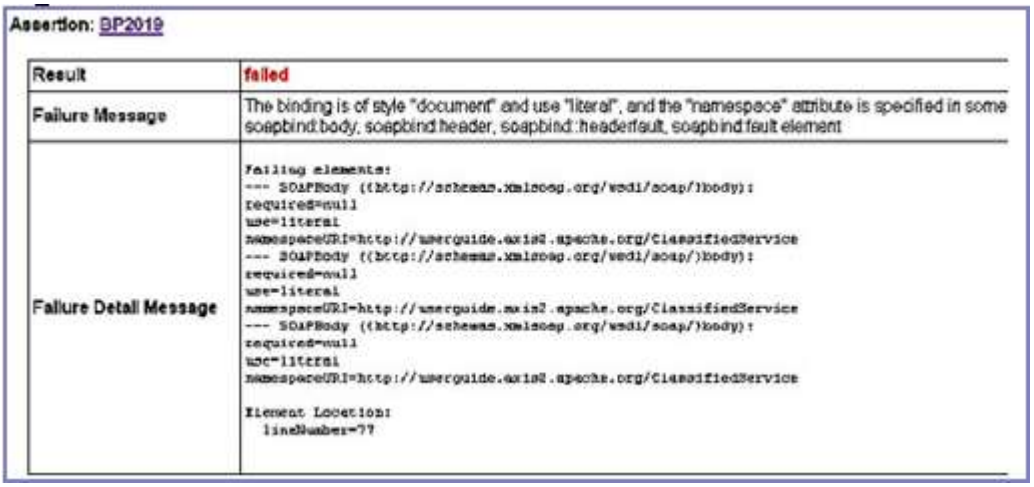
El informe del analizador brinda una evaluación general de aprobación o desaprobación de

cada servicio web que se analiza. También brinda una lista detallada de los requisitos del Perfil Básico con sus propias evaluaciones de aprobación o desaprobación. A manera de ejemplo, se puede cargar el informe del analizador para alguno de los servicios y se puede observar que ha obtenido una calificación de desaprobación, si lo tuviese, como se muestra en la siguiente figura:

Figura 15. Informe de la herramienta analizador que muestra una calificación de desaprobación

Un resultado como el anterior, evidencia que este servicio no cumple uno o más de los requisitos del Perfil Básico. El analizador le otorgó una evaluación de desaprobación. Al analizar los detalles, se puede descubrir que la descripción WSDL para el servicio creado no ha cumplido un requisito. El informe puede brindar bastantes detalles, como se muestra en la siguiente figura:

Figura 16. Informe detallado del analizador



Después de estudiar los errores y mirar el WSDL del servicio creado, se puede encontrar con que el problema debe ser el espacio de nombres que ha incluido en un elemento soap:body como se muestra en la siguiente figura:

Figura 17. Ingresar definición con espacio de nombres especificado en soap:body

```
<input name="BookReqMsg">
  <soap:body namespace="http://userguide.axis2.apache.org/ClassifiedService"
                    use="literal"/>
</input>
```

Se procede a eliminar cada definición de espacio de nombres en cada entrada en este archivo WSDL para que se vea como en la siguiente otra figura, esperando que permita que la herramienta analizador ahora si apruebe este servicio.

Figura 18. El soap:body sin espacio de nombres

```
<input name="BookReqMsg">
<soap:body
   use="literal"/>
</input>
```

Luego, se ejecuta la herramienta analizador, y se vuelve a cargar el informe en el navegador, y observa que el servicio ahora cumple completamente el Perfil Básico WS-I.

También se puede dar cuenta de que ha cambiado la descripción WSDL para el servicio dado. Ahora se debe asegurar de que WSDL siga funcionando de manera adecuada con el kit de herramientas Axis2. Para hacerlo, Se sugiere que vuelva a generar la estructura del servicio y las clases de stub de cliente utilizando el comando WSDL2Java, se compila todo y se vuelve a realizar la prueba.

También se pueden utilizar herramientas alternas a las antes mencionadas, por ejemplo si se utiliza la aplicación Eclipse, también tiene la opción de utilizar un conjunto de herramientas alternativas para probar el cumplimiento del Perfil Básico. La aplicación **Eclipse Web Tools Project** incluye una variedad de mejoras útiles para Eclipse, inclusive editores especiales para editar documentos WSDL y XML Schema, resaltado de sintaxis para muchos formatos de archivos de programación Web, y un conjunto de herramientas de prueba que pueden analizar un documento WSDL para ver si cumple con el Perfil Básico. Para algunos desarrolladores las herramientas de prueba de WS-I son impresionantes en su calidad y alcance, mientras que para otros descubren que las herramientas del Eclipse Web Tools Project se adaptan mucho mejor a su flujo de trabajo.

Disponer de un conjunto de herramientas que le permita validar e integrar estas pruebas a su entorno de desarrollo puede hacer que este proceso sea aún más eficiente. La siguiente

figura muestra un ejemplo del complemento Eclipse Web Tools.

Figura 19. El complementó Eclipse Web Tools destaca errores

Por último, para realizar la inclusión de reclamos de cumplimiento, y en caso de requerir anunciar el hecho de dar a los usuarios potenciales de la confianza de que se han realizado todos los esfuerzos para evitar problemas de interoperabilidad con sus servicios web, la misma especificación técnica del Perfil Básico describe cómo puede hacer dichos reclamos.

Para ello, se deben seguir los ejemplos para agregar un elemento wsi:Claim a las definiciones de puerto en sus documentos WSDL como se muestra en la siguiente figura:

Figura 20. Muestra de un reclamo de cumplimiento del Perfil Básico en un puerto WSDL

```
<wsdl:port name="MyPort" binding="tns:MyBinding" >
      <wsdl:documentation>
        <wsi:Claim
         conformsTo="http://ws-i.org/profiles/basic/1.1" />
      </wsdl:documentation>
```

5. REFERENCIAS

[i] < https://es.wikipedia.org/wiki/Apache_Axis2>
[ii] < http://panamahitek.com/que-son-las-clases-en-java/>
[iii] < https://es.wikipedia.org/wiki/CORBA>
[iv] < https://es.wikipedia.org/wiki/Eclipse_(software)>
[v] < https://es.wikipedia.org/wiki/HTML>
[vi] < https://es.wikipedia.org/wiki/Hypertext_Transfer_Protocol>
[vii] < https://es.wikipedia.org/wiki/Ambiente_de_desarrollo_integrado>
[viii] Institute of Electrical and Electronics Engineers. IEEE Standard Computer Dictionary: A Compilation of IEEE Standard Computer Glossaries. New York, NY: 1990.
[ix] < https://es.wikipedia.org/wiki/Java_EE>
[x] < https://es.wikipedia.org/wiki/Java_(lenguaje_de_programaci%C3%B3n)>
[xi] < https://es.wikipedia.org/wiki/Java_Development_Kit>
[xii] < https://es.wikipedia.org/wiki/Java_Message_Service>
[xiii] < https://es.wikipedia.org/wiki/JSON>
[xiv] < https://es.wikipedia.org/wiki/Middleware>
[xv] < https://es.wikipedia.org/wiki/WS-I>
[xvi] < https://es.wikipedia.org/wiki/PHP>
[xvii] < https://es.wikibooks.org/wiki/Tecnolog%C3%ADas_de_Internet/Protocolos>
[xviii] < https://es.wikipedia.org/wiki/Python>
[xix] < http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb>
[xx] < https://es.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol>
[xxi] < https://es.wikipedia.org/wiki/Transport_Layer_Security>
[xxii] < https://es.wikipedia.org/wiki/Familia_de_protocolos_de_Internet>
[xxiii] < https://es.wikipedia.org/wiki/Localizador_de_recursos_uniforme>
[xxiv] < https://es.wikipedia.org/wiki/WS-Security>
[xxv] < https://es.wikipedia.org/wiki/XML_Schema>
[xxvi] < https://es.wikipedia.org/wiki/Extensible_Stylesheet_Language_Transformations>
[xxvii] Simbaqueba, Angel. (2016). Guía práctica para la creación de servicios web bajo los principios de interoperabilidad web segura. Unidad Administrativa Especial de Catastro Distrital – UAECD, IDECA
[xxviii] < https://es.wikipedia.org/wiki/Simple_Object_Access_Protocol>
[xxix] <http://www.ws-i.org/Profiles/BasicProfile-1.1.html>
[xxx] <http://www.ibm.com/developerworks/ssa/webservices/tutorials/ws-understand-web-services6/>