Torry Harris

soa

Web Service Extensions

1. Abstract:

Service-oriented architecture (SOA) is a best-practice architecture pattern for the systematic design of request/reply applications.

Web Service extensions (WS-*) are conventions that allow systems to access the services through networks, in a neutral manner. The major WS specifications are WS-Core Messaging, WS-Metadata, WS-Security, WS-Discovery, WS-Reliable Messaging, WS-Transactions, WS-Enumeration Transfer and Eventing and WS-Management.

The purpose of this paper is to enlighten the reader about the various Web Services conventions, which are re-inventing technology today.

2. Web Services and SOA:

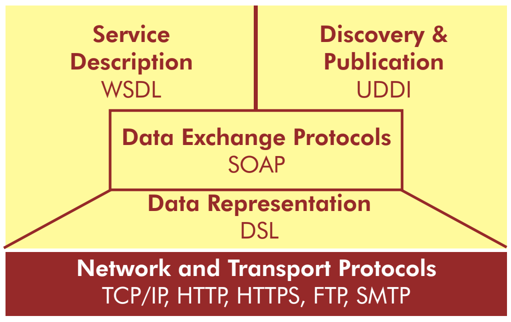
The primary intentions of SOA are: business-level software modularity and rapid, non-intrusive reuse of business software in new runtime contexts. We need to understand the essence of SOA, as well as its strengths and limitations, to identify its role in the overall architecture of today's modern enterprise IT.

Web Service consumers are software applications that embed a service interface proxy. Simply put, any software that uses the following standards is a web-service:

- Web Services Description Language (WSDL)
- Simple Object Access Protocol (SOAP)
- Universal Description, Discovery and Integration (UDDI)

A brief description of the standards is given below. These standards have been referred to throughout the document.

- SOAP: It is an XML based message format having bindings to underlying protocols (Ex: HTTP, SMTP etc).
- WSDL: It is an XML based language for describing Web Services and how to access them.
- UDDI: It is a protocol for publishing metadata about Web Services, to enable applications to find Web Services either at design time or run time.



Web Services Architecture diagram

As evidenced by the definitions, Web services are about technology specifications, whereas SOA is a software architectural principle/methodology. Web services are based on Internet standards, are platform agonistic, are widely available, have complete vendor support and are a key enabler of SOA. The remainder of the sections describes the basic WS Extensions in detail.

3. WS-Core Messaging:

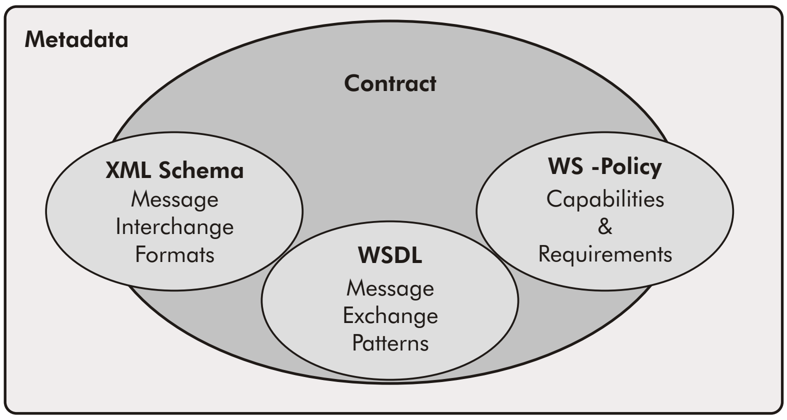
Core Messaging is one of the WS-Specifications used to express messages in a definite form in the web services architecture. Messaging in Web Services is achieved using SOAP, which is an XML document information item containing an envelope (<Envelope>), header (<Header>) and a body (<Body>), collectively also called as the SOAP elements.

WS-Addressing is a common addressing mechanism required to route the messages and address the services in the multitransport messaging environment. The Addressing specification contains the To/From/ReplyTo/FaultTo header blocks and is used to address the agents that process the message and its replies. These headers rely on a WS-Addressing defined structure called an 'endpoint reference' that combines the information needed to properly address a SOAP message.

4. WS-Metadata:

Web Services use metadata to describe what other endpoints (identified by URI's) need to know, to interact with them.

The Web Service Metadata provides information such as WSDL (specifies message exchange patterns), XML Schema (specifies the message interchange formats) and WS Policy (specifies the capabilities and requirements of web services). These are known as Metadata Units. The collection of units is represented by a Metadata element. In brief, Metadata tells a user: whom the information is from, who it is intended for and also the details about the information content. Metadata tags (XML tags) would not interfere with the normal working of the application but would allow a preprocessor to interpret the metadata tags and generate the appropriate code for Web Services.



Web Services Metadata Architecture

Web Services metadata specification defines three request-response message interactions:

- One retrieves the WS-Policy associated with the receiving end point (requester) or target namespace.
- The second retrieves the XML Schema associated with the receiving end point (requester) or target namespace.
- The third retrieves the WSDL associated with the receiving end point or target namespace.

5. WS-Security:

WS-Security defines SOAP extensions in order to implement client authentication, message integrity and message confidentiality at the message level. The advantage of WS-Security is that it works in conjunction with other Web Service extensions.

SOAP is regarded as the standard for communication. Hence, a secure transport for SOAP messages like Secure Sockets Layer (SSL) or Transport Layer Security (TLS) etc. should be available between the communicating Web Services. This also requires provision for end-to-end security to maximize the reach of Web Services. Authentication involves identification of the sender, service or other resources. Once authenticated, the user needs to be authorized to use a specific resource. Message confidentiality keeps an unauthorized third party from reading a message during transmission.

WS-Trust uses the mechanisms of WS-Security for the issuance, exchange and validation of security tokens. In order to ensure that the communication between 2 parties is secure, security credentials should be exchanged between the 2 parties i.e. one party needs to determine if the other party can be trusted. This mechanism is achieved by issuing security tokens that establish the trust relationship.

WS-Secure Conversation is built on top of WS-Security to enable secure conversation between two services.

WS-Security Policy complements WS-Security by specifying a set of rules to accommodate a wide variety of security models.

6. WS-Discovery:

This specification defines a protocol to locate services i.e. a developer of a client application can learn about the existence of a Web Service, its capabilities and method of interaction, given the URL to a common directory containing Web Service information. UDDI specifies a protocol for querying and updating this common directory. A UDDI entry contains 3 parts: service provider, Web Services offered and bindings to the implementations. Each of these parts provides progressively more detailed information about the Web service.

The specifications intend to meet the following requirements:

- Enable smooth transitions between ad hoc and managed networks.
- Enable discovery of resource-limited service implementations.
- Support bootstrapping to other Web service protocols as well as other transports.
- Enable discovery of services by type and within scope.

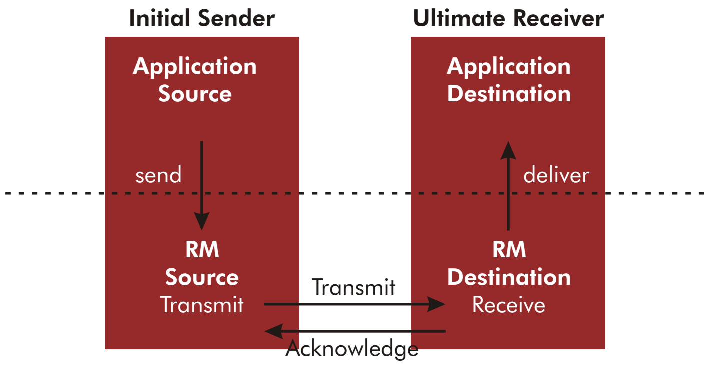
7. WS-Reliable Messaging:

WS-Reliable Messaging describes a protocol that allows messages to be delivered reliably between distributed applications in the presence of a software component failure, system failure, or network failures. At the most basic level, reliable messaging refers to the ability of a sender to deliver a message "once and only once" to its intended receiver.

Many conditions may interrupt an exchange of messages between two services. This is especially an issue when unreliable transport protocols such as HTTP 1.0 and SMTP [SMTP] are used for transmission or when a message exchange spans multiple transport-layer connections. WS-Reliable Messaging is a protocol that enables the end-to-end reliable delivery of messages based on specific delivery assurance characteristics. The specification defines four different message delivery assurances:

- At-Least-Once Delivery: Each message is delivered at least one time.
- At-Most-Once Delivery: Duplicate messages will not be delivered.
- Exactly-Once Delivery: This is the logical "and" of the two prior delivery assurances.
- In-Order Delivery: Messages are delivered in the same order they were sent.

The diagram below illustrates the entities and events involved in a simple reliable message exchange. The Application Source sends a message for reliable delivery. The Reliable Messaging (RM) Source accepts the message and transmits it one or more times to the RM destination. Once the RM Destination receives the message, it acknowledges the delivery. Finally, the RM Destination delivers the message to the Application Destination.



A simple reliable message exchange

Protocol Pre-conditions:

- The RM Source must have an endpoint reference that uniquely identifies the RM destination endpoint.
- The source must have knowledge of the destination's policies, if any, and the RM Source must be capable of formulating messages that adhere to these policies.

If a secure exchange of messages is required, then the RM source and RM destination must have a security policy in place.

8. WS-Transactions:

Transactions are an important building block in all of the business applications. A transaction can be defined as a unit of work that involves one or more resources and is either completed in its entirety or is not done at all. Participating resources are locked for the duration of the transaction. Ws-Transactions describe an extensive framework for short duration and long duration business transactions.

WS-Transaction defines two distinct transactional models:

- WS-Atomic Transaction: These correspond to the traditional transactions with ACID properties (Atomicity, Consistency, Isolation and Durability). This model supports transactions that are short-lived.
- WS-Business Activity: This specifies the protocols for long running transactions. The model allows participants to take part in the transactions without the resources being locked. This model has a mechanism to inform the participants to take necessary actions if the transaction outcome is different from the expected outcome of the participants.

Co-ordination is the act of one agent (the coordinator) disseminating a message to all the other participants, to make sure that all the participants receive the same specific message. WS Co-ordination specifies the basic transaction infrastructure support. This specification describes a framework for providing protocols that co-ordinate the actions of distributed applications. It includes:

- Creation of a coordinator for a specific coordination protocol.
- Registration of the participants with the coordinator.
- Propagation of context: Here the coordinator takes the initiative of passing the transaction related information to the participants.

9. WS-Enumeration Transfer and Eventing:

This section briefs about the specifications that provide enumeration of service resources, their state management, which involves sessions between the source and requestor and event notification in the web services architecture. They are based on WS-Enumeration [WS-Enum], WS-Transfer [WS-Transfer], and WS-Eventing [WS-Eventing].

WS-Enumeration specification defines simple rules for enumeration that allows the data source to provide a session abstraction, called an enumeration context, to a consumer through a sequence of data items. Many scenarios require data exchange using more than just a single request/response message pair. Types of applications that require these longer data exchanges include database queries, data streaming etc. Enumeration, in particular, is achieved though establishing a session between the data source and the requestor. In its simplest form, WS-Enumeration defines a single operation, 'Pull', which allows a data source, to produce a sequence of XML elements in the body of a message. In addition to enumerating the data entities present in a Web service, it is convenient to be able to perform several basic operations on them. These operations are introduced in the WS-Transfer specification.

WS-Transfer is a protocol for manipulating resources and their representations. A requestor can create, modify, delete resources, as well as retrieve XML representations of those resources. Specifically WS-Transfer defines two operations for sending and receiving the XML representation of a given resource and two operations for creating and deleting a resource and its corresponding XML representation. This specification needs to meet the following requirements:

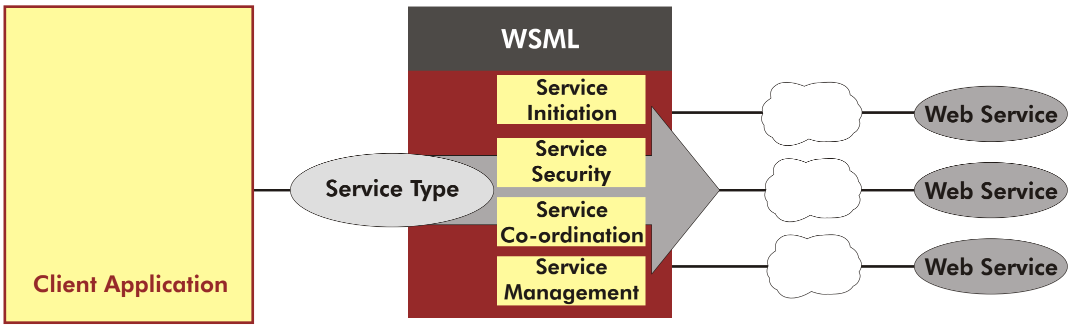
- Provide a protocol for managing resources and their representations.
- Minimize additional mechanisms beyond the current Web Service architecture.

Web services often want to receive messages when events occur in other services and applications. This specification defines a protocol for one Web service (called an "event sink") to register interest in another Web service (called an "event source") in receiving messages about events (called "notifications"). This registration process is called Subscription. To improve robustness, the subscription is leased by an event source to an event sink, and the subscription expires over time. An event source may allow an event sink to renew the subscription. When an event source determines that an event has occurred, it will provide that information to all of the matching subscriptions. This is similar to publishing topics or subjects in a traditional publish/subscribe event notification system. There are many mechanisms by which event sources may deliver events to event sinks.

Subscriptions lease resources that must be eventually recovered. The primary mechanism used to reclaim resources is an expiration time for each subscription. The other mechanisms include querying the status of the subscription, renewing the subscription or unsubscribing the subscription.

10. WS-Management:

WS-Management is a reusable middleware platform that enables communication between the client and the server application. It builds on several components of the architecture, providing a common set of operations that are required by all systems management solutions. The philosophy of Web Service Management Layer (WSML) is to extract all the web service related code present on the client and place it inside the WSML. Whenever a client makes a request to the server for some functionality, the request will actually be made to the service stub (shown in the figure). The WSML will translate this request to an available Web Service and send the result back to the client.



Web Services Management Architecture

11. Conclusion:

What has been presented in this paper, is an easy to understand overview of the various web service extensions that are currently at different points of standardization. The effective and wide scale use of web services within enterprises require robust implementations of these extensions to be in place and used along with the other necessary infrastructure required for maintenance and management of the services.

12. References:

- Web Services Architecture and Its Specifications: Essentials for Understanding WS-* Mar.2005. By Luis Felipe Cabrera, Chris Kurt
- SOA and Web Services – The road to EAI http://java.sun.com/developer/technicalArticles/WebServices/soa/
- Securing the Web Services and JAVA WSDP 1.5 XWS Security Framework http://java.sun.com/developer/technicalArticles/WebServices/security/
- Web Services Policy Framework http://specs.xmlsoap.org/ws/2004/09/policy/ws-policy.pdf
- Web Services Dynamic Discovery http://msdn.microsoft.com/ws/2005/04/ws-discovery/
- From Open Grid Services Infrastructure to WS- Resource Framework http://www.globus.org/wsrf/specs/ogsi_to_wsrf_1.0.pdf
- W3C member submission http://www.w3.org/Submission/

Torry Harris

soa

Torry Harris