# WS-\* Standards

Abstract

"WS-\*" is a prefix used to indicate specifications associated with web services and there exist many WS-\* standards including WS-Addressing, WS-Policy, WS-Security, etc. This paper introduces and discusses many of the specifications that might be considered a part of "WS-\*".

1. Introduction

Before this rise of XML standards, enterprise integration products could only implement proprietary messaging technologies. These technologies were used to provide features such as security, a messaging format, and reliable messaging. With the introduction of SOAP as a standard messaging format, this has started to change dramatically. SOAP is now supported in every integration product, and it provides a common ground for other XML specifications.

When SOAP was introduced, it was tightly related to web services. But, more and more, it is seen as a standard message definition. The specifications built on SOAP are often referred to as the WS-\* specifications because they all start with WS-. Some of these specifications are already supported by many integration products and are seen as standards in the industry, i.e., these specifications are the basic web services framework established by first-generation standards represented by SOAP, WSDL, and UDDI. Specifications may complement, overlap, and compete with each other.

Section 2 describes first-generation standards. Section 3 discuss some current WS-\* Standards. Finally, Section 4 presents our conclusion.

2 First-Generation Standards

**SOAP**

SOAP (abbreviation for Simple Object Access Protocol) is a messaging [protocol](https://en.wikipedia.org/wiki/Protocol_(computing)) specification for exchanging structured information in the implementation of [web services](https://en.wikipedia.org/wiki/Web_service) in [computer networks](https://en.wikipedia.org/wiki/Computer_network). Its purpose is to provide [extensibility](https://en.wikipedia.org/wiki/Extensibility), [neutrality](https://en.wikipedia.org/wiki/Neutrality_(philosophy)) and independence. It uses [XML Information Set](https://en.wikipedia.org/wiki/XML_Information_Set) for its [message format](https://en.wikipedia.org/wiki/Message_format), and relies on [application layer](https://en.wikipedia.org/wiki/Application_layer) protocols, most often [Hypertext Transfer Protocol](https://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol) (HTTP) or [Simple Mail Transfer Protocol](https://en.wikipedia.org/wiki/Simple_Mail_Transfer_Protocol) (SMTP), for message negotiation and transmission.

SOAP allows processes running on disparate operating systems (such as [Windows](https://en.wikipedia.org/wiki/Microsoft_Windows) and [Linux](https://en.wikipedia.org/wiki/Linux)) to communicate using [Extensible Markup Language](https://en.wikipedia.org/wiki/XML) (XML). Since Web protocols like HTTP are installed and running on all operating systems, SOAP allows clients to invoke web services and receive responses independent of language and platforms.

Although SOAP can be used in a variety of messaging systems and can be delivered via a variety of transport protocols, the initial focus of SOAP is remote procedure calls transported via HTTP. Now it has many features. SOAP is a communication protocol designed to communicate via Internet.It can extend HTTP for XML messaging, providing data transport for Web services. SOAP can exchange complete documents or call a remote procedure, and also be used for broadcasting a message. SOAP enables client applications to easily connect to remote services and invoke remote methods. Other frameworks including CORBA, DCOM, and Java RMI provide similar functionality to SOAP, but SOAP messages are written entirely in XML and are therefore uniquely platform- and language-independent.

**WSDL**

The Web Services Description Language (WSDL [/ˈwɪz dəl/](https://en.wikipedia.org/wiki/Help:IPA/English)) is an [XML](https://en.wikipedia.org/wiki/XML)-based [interface description language](https://en.wikipedia.org/wiki/Interface_description_language) that is used for describing the functionality offered by a [web service](https://en.wikipedia.org/wiki/Web_service). The acronym is also used for any specific WSDL description of a web service (also referred to as a WSDL file), which provides a machine-readable description of how the service can be called, what parameters it expects, and what data structures it returns. Therefore, its purpose is roughly similar to that of a [type signature](https://en.wikipedia.org/wiki/Type_signature) in a programming language.

The WSDL describes services as collections of network endpoints, or ports. The WSDL specification provides an [XML](https://en.wikipedia.org/wiki/XML) [format](https://en.wikipedia.org/wiki/File_format) for documents for this purpose. The abstract definitions of ports and messages are separated from their concrete use or instance, allowing the reuse of these definitions. A port is defined by associating a [network address](https://en.wikipedia.org/wiki/Network_address)with a reusable binding, and a collection of ports defines a service. Messages are abstract descriptions of the data being exchanged, and port types are abstract collections of supported operations. The concrete protocol and data format specifications for a particular port type constitutes a reusable binding, where the operations and messages are then bound to a concrete network protocol and message format. In this way, WSDL describes the public interface to the Web service.

WSDL is often used in combination with [SOAP](https://en.wikipedia.org/wiki/SOAP) and an [XML Schema](https://en.wikipedia.org/wiki/XML_Schema_(W3C)) to provide Web services over the [Internet](https://en.wikipedia.org/wiki/Internet). A client program connecting to a Web service can read the WSDL file to determine what operations are available on the server. Any special [datatypes](https://en.wikipedia.org/wiki/Datatypes) used are embedded in the WSDL file in the form of XML Schema. The client can then use SOAP to actually call one of the operations listed in the WSDL file using for example XML over HTTP.

**UDDI**

Universal Description, Discovery and Integration is a [platform-independent](https://en.wikipedia.org/wiki/Platform-independent), [Extensible Markup Language](https://en.wikipedia.org/wiki/Extensible_Markup_Language) protocol that includes a (XML-based) registry by which businesses worldwide can list themselves on the [Internet](https://en.wikipedia.org/wiki/Internet), and a mechanism to register and locate [web service](https://en.wikipedia.org/wiki/Web_service) applications. UDDI is an open industry initiative, sponsored by the Organization for the Advancement of Structured Information Standards ([OASIS](https://en.wikipedia.org/wiki/OASIS_(organization))), for enabling businesses to publish service listings and discover each other, and to define how the services or software applications interact over the Internet.

UDDI was originally proposed as a core [Web service](https://en.wikipedia.org/wiki/Web_service) standard. It is designed to be interrogated by [SOAP](https://en.wikipedia.org/wiki/SOAP_(protocol)) messages and to provide access to [Web Services Description Language](https://en.wikipedia.org/wiki/Web_Services_Description_Language)(WSDL) documents describing the protocol bindings and message formats required to interact with the web services listed in its directory.

3 WS-\* Specifications

**XML**

Extensible Markup Language (XML) is a [markup language](https://en.wikipedia.org/wiki/Markup_language) that defines a set of rules for encoding [documents](https://en.wikipedia.org/wiki/Electronic_document) in a [format](https://en.wikipedia.org/wiki/File_format) that is both [human-readable](https://en.wikipedia.org/wiki/Human-readable_medium) and [machine-readable](https://en.wikipedia.org/wiki/Machine-readable_data).

The design goals of XML emphasize simplicity, generality, and usability across the [Internet](https://en.wikipedia.org/wiki/Internet). It is a textual data format with strong support via [Unicode](https://en.wikipedia.org/wiki/Unicode) for different [human languages](https://en.wikipedia.org/wiki/Language). Although the design of XML focuses on documents, the language is widely used for the representation of arbitrary [data structures](https://en.wikipedia.org/wiki/Data_structure) such as those used in [web services](https://en.wikipedia.org/wiki/Web_service).

Several [schema systems](https://en.wikipedia.org/wiki/XML_schema) exist to aid in the definition of XML-based languages, while programmers have developed many [application programming interfaces](https://en.wikipedia.org/wiki/Application_programming_interface) (APIs) to aid the processing of XML data.

**Messaging Specification**

***WS-Addressing***

SOAP uses XML Information Set for its message format, and relies on application layer protocols, most often Hypertext Transfer Protocol (HTTP) or Simple Mail Transfer Protocol (SMTP), for message negotiation and transmission. WS-Addressing is a standardized way of including message routing data within SOAP headers. Instead of relying on network-level transport to convey routing information, a message utilizing WS-Addressing may contain its own dispatch metadata in a standardized SOAP header. The network-level transport is only responsible for delivering that message to a dispatcher capable of reading the WS-Addressing metadata. Once that message arrives at the dispatcher specified in the URI, the job of the network-level transport is done.

Web Services Addressing defines two interoperable constructs that convey information that is typically provided by transport protocols and messaging systems. It is a standardized way of including routing data with SOAP headers. An endpoint references is an XML structure encapsulating information for addressing a message to a web service. It contains destination address of the message, additional parameters and optional metadata. The two constructs are endpoint references and message information headers. A web service endpoint is an entity where web service messages can be targeted. Endpoint references is used for convey the information to access a web service endpoint, but are also used to provide addresses for individual messages sent to and from web services. The message information headers convey end-to end message characteristics including addressing for source, destination endpoints and message identity. The network-level transport is responsible for delivering the message to a dispatcher and the job of network-level transport is done as long as the message arrives at the dispatcher. There are several properties of message addressing including message destination, source endpoint, reply endpoint, fault endpoint, action, unique message ID and relationship to previous messages. Source endpoint is the end point of the service that dispatched the message. Reply end point is the endpoint to which reply messages should be dispatched. Fault endpoint is the endpoint to which fault messages should be dispatched. Action is the value indicating the semantics of the message, mechanisms for key exchange.

**Metadata Exchange Specification**

***JSON-WSP***

JSON-WSP (JavaScript Object Notation Web-Service Protocol) is a web-service protocol that uses JSON for service description, requests and responses. It is inspired from JSON-RPC, but the lack of a service description specification with documentation in JSON-RPC sparked the design of JSON-WSP.

***WS-Policy***

WS-Policy provides flexible grammar for expressing the capabilities, requirements and general characteristics of entities in an XML Web services-based system. It allows web services to use XML to advertise their polities and for web service consumers to specify policy requirements. It defines a policy to be a collection of policy alternatives. Some policy assertions specify traditional requirement. Other policy assertions have no wire manifestation yet to proper service selection and usage. It does not specify how policies are discovered or attached to a web service. The goal of WS-policy is to provide the mechanisms to enable Web services applications to specify policy information such as an XML infoset called policy expression that contains domain-specific, Web Service policy information and a core set of constructs to indicate how choices of domain-specific policy assertions apply in a Web series environment. An effective policy will be computed containing intersection of both policies if both provider and consumer specify a policy. The assertions in the new policy must not contradict each other except synonymous assertions, which is considered not compatible by a policy intersection. WS-policy defines a flexible and powerful structure for expressing constrains of any form but the implementations of WS-policy processing used by web series stacks don’t implement much of the flexibility. Many useful features of WS-policy cannot be sued for web services to interoperate with a full range of web service stacks because of the lack of implementation support.

**Security Specification**

***WS Security***

The OASIS WS-Security specification is the open standard for Web services security. Its goal is to let applications secure SOAP message exchanges by providing encryption, integrity, and authentication support. Let's look at how it provides authentication support for SOAP messaging.

WS-Security offers a general-purpose mechanism for associating security tokens with message content. The specification defines three approved token types:

*UsernameToken Profile*

*X.509 Certificate Token Profile*

*SAML (Security Assertion Markup Language) Token Profile*

Each of these profiles defines how to use its token type within the WS-Security specification. For example, the UsernameToken Profile describes how a Web service client can supply a UsernameToken as a way to identify the requestor by a username and optionally by supplying a password.

To authenticate using WS-Security, a SOAP header needs to be added to the SOAP envelope. This header would contain the WS-Security information.

***XML Signature***

XML signatures can be used to sign a resource of any [type](https://en.wikipedia.org/wiki/MIME#Content-Type), typically XML documents, but anything that is accessible via a [URL](https://en.wikipedia.org/wiki/Uniform_Resource_Locator) can be signed. An XML signature used to sign a resource outside its containing XML document is called a [detached signature](https://en.wikipedia.org/wiki/Detached_signature); if it is used to sign some part of its containing document, it is called an enveloped signature; if it contains the signed data within itself it is called an enveloping signature.

XML Signature is more flexible than other forms of digital signatures such as [Pretty Good Privacy](https://en.wikipedia.org/wiki/Pretty_Good_Privacy) and [Cryptographic Message Syntax](https://en.wikipedia.org/wiki/Cryptographic_Message_Syntax), because it does not operate on [binary data](https://en.wikipedia.org/wiki/Binary_data), but on the [XML Infoset](https://en.wikipedia.org/wiki/XML_Infoset), allowing to work on subsets of the data, having various ways to bind the signature and signed information, and perform transformations. Another core concept is canonicalization, that is to sign only the "essence", eliminating meaningless differences like whitespace and line endings.

***WS-trust***

WS-trust emerged in the SOAP world to allow web services to share user identities by incorporating standard security tokens into SOAP headers. It is short for Web Services Trust Language and it is and specification which provides extensions to WS-security. It defines a number of new elements and concepts including Security Token Service, the formats of the messages used to request security tokens and the responses to those messages. Security Token Service is a web service that issues security tokens as defined in the WS-Security specification. It converts locally issued tokens into a format shared with web services providers and also for converting tokens into a format that can be used by local applications. WS-Trust standard specifies that Security Token Service can be used by web service clients and providers. The WS-Trust specification was authored by representatives of a number of companies. It can engage in secure communication designed work within the web services framework. WS-Trust deals with managing software security tokens include SAML tokens and Username Tokens. It defines protocols to issue, cancel, renew WS-Security tokens and enable secure message through Web Services. STS converts whatever security token being used locally into a standard SAML security token containing the user’s identity on the web service client side. STS validates incoming security tokens and generate new local token for consumption by other applications on the web service provider side.

**Resource Specification**

Resources, which are entities addressable by an endpoint reference that provide an XML representation. These standards define a mechanism for acquiring XML-based representations of entities using the Web service infrastructure. Specifically, it defines two operations for sending and receiving the representation of a given resource and two operations for creating and deleting a resource and its corresponding representation.

**Transaction Specification**

a Web Services specification developed by BEA Systems, IBM, and Microsoft. The WS-Transaction specification describes coordination types that are used with the extensible coordination framework described in the WS-Coordination specification. It defines two coordination types: Atomic Transaction (AT) for individual operations, and Business Activity (BA) for long running transactions. Developers can use either or both of these coordination types when building applications that require consistent agreement on the outcome of distributed activities.

**Other Specifications**

The Web Services Business Process Execution Language (WS-BPEL), commonly known as BPEL (Business Process Execution Language), is an OASIS standard executable language for specifying actions within business processes with web services. Processes in BPEL export and import information by using web service interfaces exclusively.

WS-Management (Web Services-Management) is a DMTF open standard defining a SOAP-based protocol for the management of servers, devices, applications and various Web services. WS-Management provides a common way for systems to access and exchange management information across the IT infrastructure.

4 Conclusion

SOAP was long the standard approach to web service interfaces, although it’s been dominated by REST in recent years, with REST now representing more than 70% of public APIs according to [Stormpath](https://stormpath.com/blog/rest-vs-soap" \t "_blank). Still, SOAP remains the preferred protocol for certain use cases. The general consensus among experts these days is that REST is the typically preferred protocol unless there’s a compelling reason to use SOAP (and there are some cases in which SOAP is preferred). For instance, if you need more robust security, SOAP’s support for WS-Security can come in handy. It offers some additional assurances for data privacy and integrity. It also provides support for identity verification through intermediaries rather than just point-to-point, as provided by SSL (which is supported by both SOAP and REST). Another advantage of SOAP is that it offers built-in retry logic to compensate for failed communications. REST, on the other hand, doesn’t have a built-in messaging system. If a communication fails, the client has to deal with it by retrying. There’s also no standard set of rules for REST. This means that both parties (the service and the consumer) need to understand both content and context.

At the end of the day, the best protocol is the one that makes the most sense for the organization, the types of clients that you need to support, and what you need in terms of flexibility. Most new APIs are built using REST and JSON, simply because it typically consumes less bandwidth and is easier to understand both for developers implementing initial APIs as well as other developers who may write other services against it. Because it’s more easily consumed by most of today’s web browsers, REST+JSON has become the defacto technology for the majority of public APIs. However, SOAP remains a valuable protocol in some circumstances. Plus, you don’t have to look far to find die-hard fans advocating for SOAP for certain use cases.

References

<https://stackoverflow.com/questions/840653/wsdl-vs-rest-pros-and-cons>

<https://www.ibm.com/developerworks/webservices/tutorials/ws-understand-web-services1/ws-understand-web-services1.html?ca=drs->

<https://en.wikipedia.org/wiki/Web_service>

<https://documentation.progress.com/output/ua/OpenEdge_latest/index.html#page/bpm-web/web-services-standards.html>

<https://en.wikipedia.org/wiki/List_of_web_service_specifications>

<https://doveltech.com/innovation/the-pros-and-cons-of-web-services/>