W3C Working Draft

W3C®

Web Services Architecture Requirements

W3C Working Draft 29 April 2002

This version:
http://www.w3.org/TR/2002/WD-wsa-reqs-20020429
Latest version:
http://www.w3.org/TR/wsa-reqs

Editors:
Daniel Austin, W. W. Grainger, Inc. <austin.d@ic.grainger.com>
Abbie Barbir, Nortel Networks, Inc. <abbieb@nortelnetworks.com>
Sharad Garg, The Intel Corporation <sharad.garg@intel.com>

Abstract

The use of Web Services on the World Wide Web is expanding rapidly as the need for application-to-application communication and interoperability grows. These services provide a standard means of communication among different software applications involved in presenting dynamic context-driven information to the user. In order to promote interoperability and extensibility among these applications, as well as to allow them to be combined in order to perform more complex operations, a standard reference architecture is needed. The Web Services Architecture Working Group at W3C is tasked with producing this reference architecture.

This document describes a set of requirements for a standard reference architecture for Web Services developed by the Web Services Architecture Working Group. These requirements are intended to guide the development of the reference architecture and provide a set of measurable constraints on Web Services implementations by which conformance can be determined.

Status of this Document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. The latest status of this document series is maintained at the W3C.

Exhibit 2004
ServiceNow v. HP
IPR2015-00707

This is the first W3C Working Draft of the Web Services Architecture Requirements document. It is a chartered deliverable of the Web Services Architecture Working Group, which is part of the Web Services Activity. Although the Working Group agreed to request publication of this document, this document does not represent consensus within the Working Group about Web services architecture requirements.

This first version of the requirements document is an early snapshot: it may contain conflicting and incomplete requirements and goals. The next version that the Working Group will publish will be more complete and polished.

Comments on this document should be sent to www-wsa-comments@w3.org (public archive). It is inappropriate to send discussion emails to this address.

Discussion of this document takes place on the public www-ws-arch@w3.org mailing list (public archive) per the email communication rules in the Web Services Architecture Working Group charter.

Patent disclosures relevant to this specification may be found on the Working Group's patent disclosure page.

This is a public W3C Working Draft for review by W3C members and other interested parties. It is a draft document and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use W3C Working Drafts as reference material or to cite them as other than "work in progress". A list of all W3C technical reports can be found at http://www.w3.org/TR/.

Table of Contents

1 Introduction

The use of Web Services on the World Wide Web is expanding rapidly as the need for application-to-application communication and interoperability grows. These services provide a standard means of communication among different software applications involved in presenting dynamic context-driven information to the user. In order to promote interoperability and extensibility among these applications, as well as to allow them to be combined in order to perform more complex operations, a standard reference architecture is needed. The Web Services Architecture Working Group at W3C is tasked with producing this reference architecture.

This document describes a set of requirements for a standard reference architecture for Web Services developed by the Web Services Architecture Working Group. These requirements are intended to guide the development of the reference architecture and provide a set of measurable constraints on Web Services implementations by which conformance can be determined.

1.1 What is a Web service?

The Working Group has jointly come to agreement on the following working definition:

Web service

[Definition: A Web service is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by XML artifacts and supports direct interactions with other software applications using XML based messages via internet-based protocols]

1.2 Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119.

> **Note:**
>
> A few words on the naming convention used here and throughout this document: all goals, critical success factors and requirements are labeled according to the following convention:
>
> [D-]A(G|F|R|UC)nnn.n.n
>
> [D-] indicates that the item is in a draft state
>
> A indicates that this is an architectural item.
>
> [G|F|R|UC] is one of Goal|Critical Success Factor|Requirement|Use Case.
>
> nnn.n.n indicates the sequence number of the item.

2 Requirements Analysis Method

Many methods of analyzing requirements for software systems are available. While each of them has strengths and weaknesses, the Web Services Architecture Working Group has decided to make use of two methods concurrently, in the hope that together each of these methods will produce a well-defined set of requirements for Web Services Architecture. The two methods chosen are the Critical Success Factor Analysis method, which will be supplemented through the use of gathering Usage Scenarios. Both of these methods are useful but represent different approaches to the problem of gathering requirements.

The Working Groups intends to use these methods together and to cross-reference the results of each approach to ensure consistency of the overall architectural direction. By ensuring that the requirements each serve to meet the goals of the Working Group through the CSF analysis, and also ensuring that the architecture is consistent with the envisioned Usage Scenarios of the Working Groups in the Web Services activity, we can develop a set of architectural requirements that will provide an architectural model that meets the needs of all of those involved.

Note that in the case of Usage Scenarios, the vast majority of these are taken from the work of other W3C Working Groups in the Web Services Activity domain. Few individual Usage Scenarios will be developed by the Web Services Architecture Working Group directly, and those only in response to perceived gaps or omissions in the work of other Working Groups. Usage scenarios will be published separately.

2.1 Understanding Critical Success Factors Analysis

The Critical Success Factors Analysis methodology for determining requirements is a top-down means of determining requirements based on the needs of the organization. For this reason it is well-suited for requirements analysis for large systems with many stakeholders and an audience with multiple and sometimes conflicting interests. The CSF analysis method begins with a mission statement and then begins to divide the mission statement into a set of very high-level goals. These high-level goals are then further divided into Critical Success Factors, which themselves are then further broken down into multiple levels of a hierarchy, becoming more concrete. At the lowest level, each CSF becomes a requirement for the system; a single, well-defined task that must be accomplished in order to be successful. Along the way, problems to be solved and assumptions made are recorded.

Once the CSF hierarchy is established and a set of requirements has been derived, these can then be arranged into a matrix for comparison with the problems identified. In order to be considered complete, each problem must be fully addressed by one or more requirements.

By analyzing the steps necessary to achieve success, and cross-referencing them against problems to be solved, a complete set of requirements can be determined that can then be correlated with specific user scenarios. Each of the requirements should apply to at least one user scenario, and generally more than one.

This methodology allows requirements to be determined that satisfy the needs of the organization and those of the user. Since architectural frameworks are built and maintained by organizations, this method allows us to create a well-defined and reasonably complete set of requirements.

3 The Analysis Hierarchy

3.1 Mission Statement

3.1.1 Mission

The mission of the Web Services Architecture Working Group is to develop and maintain a standard reference architecture for Web Services.

3.1.2 Users of Web Services Architecture

The W3C Web Services Reference Architecture is intended primarily for the W3C Web Services Architecture Working Group to analyze, prioritize and characterize the technologies that are needed to fully realize an interoperable and extensible realization of the promise of Web Services. It is also intended for the use of other working groups specifying the technologies identified and described in the architecture. A secondary target audience for the architecture are the developers implementing the specified technologies, and the wider IT community that uses these technologies to deploy Web Services.

3.2 Goals

3.2.1 Top-level Goals

The Working Group has determined that at the highest level, its goals can be divided into 6 categories. Each of these is associated with the CSFs and requirements listed in section 3.2.2

Top-level Goals for the Web Services Architecture

- *D-AG001 Interoperability*

  The Web Services Architecture should provide a reference platform for the development of interoperable Web Services across a wide array of environments.

  Critical Success Factors for this goal:

  - D-AC001
  - D-AC004
  - D-AC016

- *D-AG002 Reliability*

  The Web Services Architecture must be reliable and stable over time.

Critical Success Factors for this goal:

- D-AC007
- D-AC018
- D-AC019

- *D-AG003 Web-friendly*

  The Web Services Architecture must be consistent with the current and future evolution of the World Wide Web.

  Critical Success Factors for this goal:

  - D-AC009
  - D-AC010
  - D-AC011

- *D-AG004 Security*

  The Web Services Architecture must provide a secure environment for online processes.

  Critical Success Factors for this goal:

  - D-AC006
  - D-AC020

- *D-AG005: Scalability and Extensibility*

  The Web Services Architecture must be scalable and extensible.

  Critical Success Factors for this goal:

  - D-AC002
  - D-AC003
  - D-AC005
  - D-AC017

- *D-AG006 Team Goals*

  The Web Services Architecture Working Group will work to ensure that the Architecture will meet the needs of the user community.

  Critical Success Factors for this goal:

  - D-AC008
  - D-AC012
  - D-AC013 (and subsumed D-AC014)
  - D-AC015

3.2.2 Critical Success Factors and Requirements

Proposed lower-level goals and CSFs for the Web Services Architecture Working Group

(Each of the following goals is stated as a predicate to the following statement.)

To develop a standard reference architecture for Web Services that:

D-AC001

provides a complete reference framework that encourages the development of interoperable software products from multiple vendors and provides a defensible basis for conformance and interoperability test suites

- D-AC001.1 - Encourage the development of interoperable software products.
  - D-AC001.1.1 - Ensure that no individual implementor is favored over others.
  - D-AC001.1.2 - Identify all interfaces and messaging protocols within the architecture in a standardized way.
- D-AC001.2 -Ensures that the development of standards-based technologies identify conformance in such a way that testing software can be constructed..
- D-AC001.3 - Clearly define and publish a standard reference architecture document for implementors.
  - D-AC001.3.1 - Clearly define specific factors that determine conformance, while leaving sufficient slack in the system for vendors to add value.

D-AC002

provides modularity of Web Services components, allowing for a level of granularity sufficient to meet business goals

- D-AC002.1 - Provide conceptual clarity to allow developers to share ideas and code
  - D-AC002.1.1 - Reduce complexity by decomposition of the component's functionality and its position within the architecture
  - D-AC002.1.2 - Ease development, testing, and maintenance by providing a logical, easy to understand, and consistent organization
    - D-AC002.1.2.1 - Decrease debugging time by localizing errors due to design changes

    - D-AC002.1.3 - Allow the creation of generic rules, methods, and procedures to aid in consistent development practices

- D-AC002.2 - Support object-oriented design principles by encouraging encapsulation and information hiding by components of the architecture
    - D-AC002.2.1 - Encourage reuse by creating well-defined modules that perform a particular task
    - D-AC002.2.2 - Allow the creation and deployment of configurable objects that the end user can tailor for different purposes in a standard way.
- D-AC002.3 - Provide for Increased flexibility and maintainability because single components can be upgraded or replaced independently of others
    - D-AC002.3.1 - Support a variety of end-user interface and deployment environments by allowing standardized subsets and supersets

D-AC003

is sufficiently extensible to allow for future evolution of technology and of business goals

- D-AR003.1 separates the transport of data or means of access to Web Services from the Web Services themselves
- D-AR003.2 description of Web Services be clearly separated into abstract descriptions ("what") from their concrete realizations ("how"), or put another way, separate design time aspects from run-time aspects
- D-AR003.3 technologies following this architecture should not impede the development of complex interaction scenarios likely for future business interactions
- D-AR003.4 modules that are orthogonal must be allowed to evolve independently of each other and still work within the architecture
- D-AR003.5 modularity must support common business functions such as reliability, security, transactions, etc.
- D-AR003.6 specs that are created in conformance with the architecture do not have to go through a formal process to be considered conformant

D-AC004

ensures platform and device independence of Web Services in a way that does not preclude any programming model nor assume any particular mode of communication between the individual components

- D-AC004.1 Focus on using platform independent development tools and languages.
- D-AC004.2 Interfaces to web resources must be properly defined and designed.
- D-AC004.3 Focus on defining the architecture in terms of components and the relationships between them. Components are defined in terms of interfaces, that define their inputs and outputs and also the form and constraints on those inputs and outputs. The relationships between components are described in terms of messages and the protocols by means of which these messages are transmitted among the interfaces of the components that make up the architecture.

The Web Services Architecture should:

- D-AR004.1 provide consistent definition of web resources
- D-AR004.2 provide well-defined interfaces for Web Services
- D-AR004.3 use XML based techniques for defining messages/protocols for invoking web resources

D-AC005

applies the principle of simplicity and is defined such that it does not impose high barriers to entry for its intended audience

The reference architecture should be easily understandable by the target audience.

- D-AC005.1 does it avoid specialized jargon not familiar to ordinary software designers?
- D-AC005.2 is it stated in simple declarative sentences?
- D-AC005.3 is it organized in a way that allows important points to be located?
- D-AC005.4 does it use illustrations to visually describe key components and relationships?

The reference architecture should be as minimal as possible

- D-AC005.5 How many components does it describe?
- D-AC005.6 How many relationships among the components does it describe?
- D-AC005.7 How do these figures compare to those of notable exemplars

of good reference architectures?
- D-AC005.8 Could any components or relationships be removed without significantly limiting the value of the architecture?

The reference architecture should simplify the task of a programmer writing interoperable implementations of specifications of components described by the architecture.

- D-AC005.9 is the role played by each component in the overall architecture stated clearly?
- D-AC005.10 are the interdependencies among components noted explicitly?
- D-AC005.11 are existing specs that fulfill the role of a given component referenced?
- D-AC005.12 are the resulting implementations actually interoperable?

The reference architecture should simplify the task of an application programmer using the specifications it describes.

- D-AC005.13 does the reference architecture not force a programmer to use exotic constructions?
- D-AC005.14 Can the architecture be implemented without large amounts of code?
- D-AC005.15 Does it allow simple invocations as well as elaborations with more functionality when building Web Services or applications that employ web services?

AC006

addresses the security of Web Services across distributed domains and platforms

- AC006.1 The construction of a Web Services Threat Model based on thorough analysis of existing and foreseeable threats to Web service endpoints and their communication.
- AC006.2 The establishment of a set of Web Services Security Policies to counter and mitigate the security hazards identified in the threat model.
- AC006.3 The construction of a Web Services Security Model that captures the security policies (to be executed by security mechanisms).
- AC006.3 The construction of a Web Services Security Model that captures the security policies (to be executed by security mechanisms).

- AC006.4 The realization of the security model in the form of a Web Services Security Framework that is an integral part of the Web Services Architecture (which is the ultimate deliverable of this working group).

Requirements

- D-AR006.10 The description of a Web service SHOULD include security policy.
- D-AR006.11 The architecture must provide an interface for Web Services to directly communicate with their underlying infrastructure.

  The interface is for negotiating services that an infrastructure may provide to, or perform on behalf of, a requesting Web Services. Such value-added services may include: security, content delivery, QoS, etc. For instance, a Web service may instruct (via the interface) the security agents of its infrastructure to defend against DOS/DDOS attacks on its behalf.

Editorial note	
The WG has not yet reached consensus on the merits of D-AR006.11 and there is still considerable debate	

- There are six aspects in the security framework for Web Services architecture: Accessibility, Authentication, Authorization, Confidentiality, Integrity, and Non-repudiation. Together they form the foundation for secure Web Services.
- D-AR006.1 Accessibility to a Web service can be impaired by DOS/DDOS attacks. It is understood that there's little a Web service residing well above the transport layer of a network stack can effectively detect such transgression, let alone deploy countermeasures. Therefore, the security framework must provide recourse for Web Services to mitigate the hazard.

Editorial note	
The WG has not yet reached consensus on the merits of D-AR006.1 and there is still considerable debate	

There are six aspects in the security framework for Web Services architecture: Accessibility, Authentication, Authorization, Confidentiality, Integrity, and Non-repudiation. Together they form the foundation for secure Web Services.

- D-AR006.2.1 The security framework must include Authentication for the identities of communicating parties.
- D-AR0062.2 The security framework must include Authentication for data (sent and received by communicating parties).
- D-AR006.3 The security framework must include Authorization, with allowance for the coexistence of dissimilar authorization models.
- D-AR006.4 The security framework must include Confidentiality.

- D-AR006.5 The security framework must include (data) Integrity.
- D-AR006.6 The security framework must include Non-repudiation between transacting parties.

  Note that there is a close relationship among D-AR006.2.1, D-AR006.2.2, D-AR006.5, and D-AR006.6, a la digital signature.
- D-AR006.7 The security framework must include Key Management, pertaining to Public Key Encryption (PKE) and Key Distribution Center (KDC).
- D-AR006.8 The security framework document SHOULD provide some guidelines for securing private keys, though the methods for securing private keys is outside the scope of the architecture.
- D-AR006.9 The security framework document SHOULD recommend a baseline for trust models.

D-AC007

is reliable, and stable, and whose evolution is predictable over time

D-AC007.1 Reliability of Architecture

D-AC007.2 Stability of Architecture

D-AC007.2.1 when a standard changes, then the change will be clear and consistent with the rest of the reference architecture components.

- D-AC007.2.2 A new version of a standard should clearly describe its backward compatibility status with its earlier version in text.
- D-AC007.2 .3 A new version of a standard must not conflict with other standards in the reference architecture that do not conflict with the old version of the standard.
- D-AC007.2.4 The evolution of identified technologies should be considered especially with reference to other industry standards.

D-AC007.3 Predictable Evolution of Architecture

- D-AC007.3.1 The reference architecture must define a framework for growth of the architecture.
- D-AC007.3.2 Non-normative extension guidelines to be specified for each standard

D-AC008

is consistent and coherent. This applies to both the reference architecture itself and the document that contains its definition.

- D-AC008.1 Simple visualization of architecture in the form of a two-dimensional diagram
- D-AC008.2 Architecture supports the concepts used in commonly accepted design patterns.

- D-AC008.3 Architectural components work together to form a logical whole.
- D-AC008.4 Architecture does not do the same or similar things in mutually incompatible ways; it is not self-contradictory.
- D-AC008.5 There shall not be wildly different means to achieve the same ends in the architecture.

D-AC009

is aligned with the semantic web initiative at W3C.

- D-AR009.1 Any meta data about any aspect of the Web Services reference architecture should be expressible with an RDF based language (such as RDF itself, RDF Schema, DAML+OIL)
- D-AR009.2 All recommendations produced by the working group include a normative mapping between all XML technologies and RDF/XML.
- D-AR009.3 All conceptual elements should be addressable directly via a URI reference.

D-AC010

uses W3C XML technologies in the development of the Web Services architecture to the extent that this is compatible with the overall goals listed here.

- D-AC010.1 Each new architectural area is representable in a syntactic schema language like XML Schema.

D-AC011

is consistent with the existing web.

- D-AC011.1 The Web Services reference architecture complies with the architectural principles and design goals of the existing web.

  Derived sub-goals:
  - D-AC011.1.1 universal identifiers (resources identifiable by URI)
  - D-AC011.1.2 simplicity
  - D-AC011.1.3 opaqueness
  - D-AC011.1.4 decentralization
  - D-AC011.1.5 statelessness
  - D-AC011.1.6 scalability of component interactions
  - D-AC011.1.7 generality of interfaces
  - D-AC011.1.8 immediate deployment of components
  - D-AC011.1.9 intermediary components to reduce interaction latency
  - D-AC011.1.10 enforces security
  - D-AC011.1.12 encapsulate legacy systems
  - D-AC011.1.13 caching semantics

  - D-AC011.1.14 platform independence
  - D-AC011.1.15 Leverage existing and emerging Web standards.
  - D-AC011.1.16 Complies with Web architecture principles and design of existing and emerging Web.
  - D-AC011.1.17 User can use browser to interact.
  - D-AC011.1.18 Program developer can use well- defined description of interface for Web service.
  - D-AC011.1.19 Decentralized discovery.
  - D-AC011.1.20 XML description.
  - D-AC011.1.21 Uses XML.
  - D-AC011.1.22 RDF models for technologies produced.
- D-AC011.2 The Web Services reference architecture recommends the use of existing web technologies which adhere to the above principles and which provide clear functional coverage of the responsibilities and constraints for a component identified in the reference architecture.

  Derived critical success factors:
  - D-AC011.2.1 Use of a standard identifier technology (URI)
  - D-AC011.2.2 Use of a standard transport technology (HTTP/S over TCP/UDP/IP)
  - D-AC011.2.3 Use of a standard data encoding technology (XML)

Editorial note	
Entries 11.1.15-22 on this list are still provisional.	

In addition, the Working Group will also act to:

D-AC012

identify or create user scenarios and use cases that support and illustrate the requirements and web services architecture

- D-AR012.1 - terms must be well defined and used consistently
- D-AR012.2 - use cases organized around usage scenarios, usage scenarios should reflect common usage patterns for architecture
- D-AR012.3 - target audience for architectural deliverables must be defined
- D-AR012.4 - usage scenarios and use cases must be referencable via URI(reference)
- D-AR012.5 - architecture should support use cases at all levels of WS activity
- D-AR012.6 - usage scenarios and use cases shall be used as justification for recommending the formation of new WSA WGs

D-AC013 (subsumes D-AC014)

co-ordinate with other W3C Working Groups, the Technical Architecture Groups and other groups doing Web Services related work in order to maintain a coherent architecture for Web Services

- D-AR013.1 Go through the W3C review process, and satisfy dependencies as listed in the charter.
- D-AR013.2 The documents produced are used as input to charter new Web Services Working Groups.
- D-AR013.3 Maintain liaisons with relevant external groups, such as the ones listed in the charter and possibly others.

D-AC015

organize its efforts in such a way as to address vital time-to-market issues for its products, including iterating over successive refinements of the overall requirements for the standard reference architecture.

- D-AC015.1 Is the Web Services Activity a center for Web Services standards specification, that is is the community able to start new working groups in a manner that is usable by the community?
- D-AC015.2 Is the WSA perceived as a reliable forum for architectural guidance? Do other working groups ask for advice from the WSA, or do they not bother?
- D-AC015.3 Is the WSA document perceived as usable and referenceable in time for products? New/revised products would be able to reference this WSArch doc if it was delivered in time for their products.
- D-AC015.4 Does the WSA demonstrate a reasonable number of re-use decisions rather than re-inventing?
- D-AC015.5 Is the architecture document regularly revised?
- D-AC015.6 Is the architecture document regularly referenced by other specifications, including but not limited to W3C specifications?
- D-AC015.7 Is there a lack of press/developer commentary that refers to time-to-market problems with WSA? To paraphrase, no press is good press on this issue.

D-AC016

examine architectural and technology issues that might prevent interoperability, and recommend existing standards and technologies where available. Also to recommend the formation of new Working Groups to develop areas of the Web Services Architecture where the need for standardization and specification has been identified.

The Web Services Architecture WG should:

- D-AR016.1 Identify what constitutes interoperability
  - D-AR016.1.1 in architectural realm.

  - D-AR016.1.2 in technological realm.
- D-AR016.2. Identify existing
  - D-AR016.2.1 architecture that supports interoperability
  - D-AR016.2.2 technologies that support interoperability
- D-AR016.3. Identify gaps
  - D-AR016.3.1 in architectural realm.
  - D-AR016.3.2 in technological realm.
- D-AR016.4 Formation of WGs to address gaps
  - D-AR016.4.1 in architectural realm.
  - D-AR016.4.2 in technological realm.

D-AC017

provides guidance for the development of the Web Services infrastructure needed to implement common business functions in a standards-based environment

- D-AR017.1 The Web Services Architecture must support common business functions, to the extent that those functions are defined in similar methodologies such as EDI.
- D-AR017.2 The Web Services Architecture must support reliable messaging and routing.
- D-AR017.3 The Web Services Architecture must support unique message IDs and message sequencing.
- D-AR017.4 The Web Services Architecture must support reliable transaction processing.

D-AC018

provide a standard set of metrics for measuring aspects of Web Services such as reliability, quality of service, and performance, and to define a standard means of measurement of these metrics and instrumentation for management of Web Services.

- D-AC018.1 Develop a standard convention of measuring Web Services metrics so different service providers, implementors and consumers can reach service level agreements.
  - D-AC018.1.1 The standard should include definitions of metrics such as Quality of Service, Reliability of Service and other metrics.
  - D-AC018.1.2 The reference architecture should provide guidelines on measuring those metrics.
  - D-AC018.1.3 Metrics can be independently verified.
- D-AC018.2 Define standard management instrumentations to Web Services.
  - D-AC018.2.1 The standard should define but not limited to instrumentations such as starting, suspending, and retiring services.

- D-AC018.2.2 The instrumentations should confirm to other goals of this working group.
- D-AC018.2.3 The definition of management framework is out of scope. There are a number of such technologies available: www.dmtf.org.
- D-AC018.2.4 The instrumentations may be exposed as Web Services.

- D-AC018.3. Clearly define and publish reference architecture for implementors.
    - D-AC018.3.1 Clearly define and publish reference Web Services management model.
    - D-AC018.4 security policies, handling various QoS aspects, negotiation of service level agreements (SLAs) must be facilitated by technologies conforming to this architecture.

D-AC019

ensure reliable, stable, and predictably evolvable Web Services.

- D-AC019.1 Web Services created using WSA can be reliably discovered, accessed, and executed.
- D-AC019.2 Web Services created using WSA can be implemented such that they are stable with respect to their definitions.
- D-AC019.3 Web Services created using WSA may be evolved/extended while maintaining their reliability and stability.

D-AC020

To develop a standard reference architecture for Web Services that enables privacy protection for the consumer of a Web service across multiple domains and services.

- D-AC020.1 Is it possible for a service consumer to know the privacy policies of the service provider(s) that it is going to deal with? (eg. hooks for P3P)
- D-AC020.1 Private data provision during a Web service transaction SHOULD NOT exceed the consumer's consent, where the consumer must be provided with reasonable means for opt-out.
- D-AR020.1 It must be possible to advertise privacy policies for Web Services

3.3 Analysis Matrix: Problems vs. CSFs

Editorial note	
TBD	

3.4 Analysis Matrix: User Scenarios vs. CSFs

Editorial note	
TBD	

4 Glossary

Editorial note	
The Working Group intends to publish a separate glossary in a future document, using agreed-upon definitions for common terms throughout the W3C Web Services activity.	

Architecture

The software architecture of a program or computing system is the structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them." [BASS98]

Binding

An association between an Interface, a concrete protocol and a data format. A Binding specifies the protocol and data format to be used in transmitting messages defined by the associated Interface.

Interface

A logical grouping of operations. An Interface represents an abstract Service type, independent of transmission protocol and data format.

Message

The basic unit of communication between a Web service and a Client: data to be communicated to or from a Web service as a single logical transmission.

Operation

A set of messages related to a single Web service action.

Port

An association between a Binding and a network address, specified by a URI, that may be used to communicate with an instance of a Service. A Port indicates a specific location for accessing a Service using a specific protocol and data format.

Reference Architecture

A reference architecture is the generalized architecture of several end systems that share one or more common domains. The reference architecture defines the infrastructure common to the end systems and the interfaces of components that will be included in the end systems. The reference architecture is then instantiated to create a software architecture of a specific system. The definition of the reference architecture facilitates deriving and extending new software architectures for classes of systems. A reference architecture, therefore, plays a dual role with regard to specific target software architectures. First, it generalizes and extracts common functions and configurations. Second, it provides a base for instantiating target systems that use that common base more reliably and cost effectively.[Gallagher2000]

Web service

A Web service is a software application identified by a URI, whose interfaces and binding are capable of being defined, described and discovered by XML artifacts and supports direct interactions with other software applications using XML based messages via internet-based protocols

5 Acknowledgments

The editors would like to thank the following Working Group members for their contributions to this document: Mark Baker, Doug Bunting, Mike Champion, Roger Cutler, Suresh Damodaran, Paul Denning, Chris Ferris, Hugo Haas, Hao He, Dave Hollander, Joe Hui, Yin-Leng Husband, Mike Mahan, Nilo Mitra, Dave Orchard

This document is a product of the Web Services Architecture Working Group whose members are:

Apple	Mike Brumbelow
Artesia Technologies	Dipto Chakravarty
AT&T	Mark Jones
AT&T	Ayse Dilber
BEA Systems	David Orchard
Boeing Company	Gerald Edgar
Carnegie Mellon University	Katia Sycara
ChevronTexaco	Roger Cutler
Cisco Systems Inc	Sandeep Kumar
Cisco Systems Inc	Krishna Sankar
Compaq	Yin-Leng Husband
Compaq	Kevin Perkins
Computer Associates	Igor Sedukhin
Contivo	Dave Hollander
CrossWeave, Inc.	Timothy Jones
DaimlerChrysler Research and Technology	Mario Jeckle
DaimlerChrysler Research and Technology	Hans-Peter Steiert

DISA	Marcel Jemio
Documentum	Don Robertson
EDS	Mike Ballantyne
EDS	Waqar Sadiq
Ericsson	Nilo Mitra
Exodus/Digital Island	Joseph Hui
France Telecom	Shishir Garg
Hewlett-Packard Company	Zulah Eckert
IBM	Heather Kreger
IBM	Jim Knutson
Intalio Inc	Bob Lojek
Intel Corporation	Sharad Garg
Intel Corporation	Joel Munter
IONA	Steve Vinoski
IONA	Eric Newcomer
Ipedo	Srinivas Pandrangi
Ipedo	Alex Cheng
Macromedia	Glen Daniels
Macromedia	Tom Jordahl
MartSoft Corp.	Jin Yu
MartSoft Corp.	Jun Chen
Microsoft Corporation	Allen Brown
Microsoft Corporation	Henrik Nielsen
MITRE Corporation	James Davenport
MITRE Corporation	Paul Denning
Nokia	Michael Mahan
Nortel Networks	Abbie Barbir
Oracle Corporation	Jeff Mischkinsky
Planetfred, Inc.	Mark Baker
Rogue Wave Software	Patrick Thompson
Rogue Wave Software	David Noor
SAP	Sinisa Zimek
SeeBeyond Technology Corp	Alan Davies
Software AG	Michael Champion
Software AG	Nigel Hutchison
Sterling Commerce(SBC)	Suresh Damodaran
Sun Microsystems, Inc.	Chris Ferris
Sun Microsystems, Inc.	Doug Bunting
Sun Microsystems, Inc.	Mark Hapner
Sybase, Inc.	Himagiri Mukkamala
Systinet	Anne Thomas Manes

The Thomson Corporation	Hao He
TIBCO Software, Inc.	Scott Vorthmann
T-Nova Deutsche Telekom Innovationsgesellschaft	Jens Meinkoehn
VeriSign, Inc.	Michael Mealling
W. W. Grainger, Inc.	Tom Carroll
W. W. Grainger, Inc.	Daniel Austin
W3C	Hugo Haas
W3C	David Booth
Waveset Technologies	Darran Rolls
webMethods, Inc.	Prasad Yendluri
XQRL Inc.	Tom Bradford
XQRL Inc.	Daniela Florescu

6 References

6.1 Normative References

BASS98
Bass, L., Clements, P., and Kazman, R. Software Architecture in Practice. Reading, Mass.: Addison Wesley, 1998.

Gallagher2000
Gallagher, Brian P. Using the Architecture Tradeoff Analysis Method to Evaluate a Reference Architecture: A Case Study CMU/SEI-2000-TN-007 June 2000 (See http://www.sei.cmu.edu/publications/documents/00.reports/00tn007/00tn007.html.)

6.2 Informative References

CSF-Primer
Bullen, C. and J. Rockart -- A Primer on Critical Success Factors, MIT Sloan School of Management Working Paper 1220-81

7 Change Log

Date	Editor	Description
20020426	HH	Fixed typos
20020426	CBF	Removed duplicate entry for D-AC006.2, 6.3 and 6.11, fixed typos
20020425	HH	Added D-AR006.10, fixed typo
20020425	CBF	fixed links that link checker complained about
20020425	HH	Accessibility tweaks, removed prevloc, specified status, pubrules compliance

20020425	CBF	tweaked curr, latest uri for candidate WD
20020424	DBA	made changes based on suggestions to the list, also changed wording of 16
20020423	DBA	tried to clean up each goal, modified top-level goal text, general document repair, relettering, status section, publishing details.
20020422	DBA	integrated many changes, modified document structure
20020422	abbie	changed goal 11 and renumbered the sub gaols
20020418	CBF	Relocated RFC2119 section. Incorporated abstract into introduction. Revised vision, fixed a few typos, assigned numbering scheme to CSFs and Requirements, updated D-AG0003 - D-AG0008. Releveled things a bit. Removed Usage Scenarios (now separate document). References tweaked. Incorporated Hugo's revised Status section.