Architecture of the World Wide Web

Editor's Draft 26 September 2003

This version:
http://www.w3.org/2001/tag/2003/webarch-20030926/
Latest editor's draft:
http://www.w3.org/2001/tag/webarch/
Previous version:
http://www.w3.org/2001/tag/2003/webarch-20030918/
Latest version:
http://www.w3.org/TR/webarch/

Editor:
Ian Jacobs, W3C
Authors:
See acknowledgments.

Abstract

The World Wide Web is an information space. This space consists of information objects collectively called "resources." Uniform Resource Identifiers (URIs) are used to identify these resources; URIs have global scope within this space. The information space is a network of Web resources that are interconnected via URIs and descriptive metadata.

This information space is the basis of, and is shared by, a number of information systems. These systems include the "traditional" hyperlink Web (where users interact with resources via links) as well as emerging Semantic Web and Web Services technologies. Within each of these systems, agents (including browsers, servers, spiders, and proxies) provide, retrieve, create, analyze, or reason about resources. They do so via representations of resource state, which are constructed from data formats such as HTML, and descriptive metadata.

Web architecture encompasses both protocols that define the information space by way of identification and representation, and protocols that define the interaction of agents within an information system making use of of the space.

Web architecture is influenced by social requirements and software engineering principles, leading to design choices that constrain the behavior of systems using the Web in order to achieve desired properties: to be an efficient, scalable, shared information space that can continue to grow indefinitely across languages, cultures, and information media.

This document is organized to reflect the three dimensions of Web architecture: identification, interaction, and representation.

Status of this document

This section describes the status of this document at the time of its publication. Other documents may supersede this document. A list of current W3C publications and the latest revision of this technical report can be found in the W3C technical reports index at http://www.w3.org/TR/.

This document has been developed by W3C's Technical Architecture Group (TAG) (charter). Please send comments on this document to the public W3C TAG mailing list www-tag@w3.org (archive).

A complete list of changes since the previous Working Draft is available on the Web. This draft includes some editorial notes and also references to open TAG issues. These do not represent all open issues in the document. They are expected to disappear from future drafts.

The TAG has published a number of findings that address specific architecture issues. Parts of those findings may appear in subsequent drafts.

This document is intended to inform discussions about issues of Web architecture. Where current practice conflicts with this document, this document and related findings are intended to help the relevant parties resolve their differences. Some parts of this document may fill in gaps in published specifications or may call attention to known weaknesses in those specifications.

Publication as a Working Draft does not imply endorsement by the W3C Membership. This is a draft document and may be updated, replaced or obsoleted by other documents at any time. It is inappropriate to cite this document as other than "work in progress." The latest information regarding patent disclosures related to this document is available on the Web.

Table of Contents

Highlighted entries in this table of contents link to principles, constraints, good practice notes, and design choices emphasized in the document.

(Note 13 context.)

14. This is true as of the publication of this document, but may change if RFC3023 is updated. (Note 14 context.)

9. Acknowledgments

This document was authored by the W3C Technical Architecture Group which included the following participants: Tim Berners-Lee (co-Chair, W3C), Tim Bray (Antarctica Systems), Dan Connolly (W3C), Paul Cotton (Microsoft Corporation), Roy Fielding (Day Software), Chris Lilley (W3C), David Orchard (BEA Systems), Norman Walsh (Sun), and Stuart Williams (co-Chair, Hewlett-Packard).

The TAG thanks people for their thoughtful contributions on the TAG's public mailing list, www-tag (archive).

List of Principles and Good Practice Notes

The following principles and good practice notes explained in this document are listed here for convenience.

---

1. Introduction

The World Wide Web (WWW, or simply Web) is an information space in which the information objects, referred to collectively as ***resources***, are identified by global identifiers called URIs.

A travel scenario is used throughout this document to illustrate some typical behavior of ***Web agents*** —software acting on this information space on behalf of a person, entity, or process. Agents include servers, proxies, browsers, spiders, multimedia players, and other ***user agents*** (software acting on behalf of a person).

Story

While planning a trip to Mexico, Nadia reads "Oaxaca weather information: '`http://weather.example.com/oaxaca`'" in a glossy travel magazine. Nadia has enough experience with the Web to recognize that "`http://weather.example.com/oaxaca`" is a URI. Given the context in which the URI appears, she expects that it allows her to access relevant weather information. When Nadia enters the URI into her browser:

1. The browser performs an information retrieval action in accordance with its configured behavior for resources identified via the "http" URI scheme.
2. The authority responsible for "weather.example.com" responds to the retrieval action, providing information in a response.
3. The browser displays the retrieved information, which includes links to other information via additional URI references[1]. Nadia can follow these links to initiate new retrieval request actions.

This scenario (elaborated on throughout the document) illustrates the three architectural divisions of the Web that are discussed in this document:

XLink10
"XML Linking Language (XLink) Version 1.0", S. DeRose, E. Maler, D. Orchard, 27 June 2001. This W3C Recommendation is available at http://www.w3.org/TR/2001/REC-xlink-20010627/.

XML10
"Extensible Markup Language (XML) 1.0 (Second Edition)", T. Bray, J. Paoli, C.M. Sperberg-McQueen, E. Maler, 6 October 2000. This W3C Recommendation is available at http://www.w3.org/TR/2000/REC-xml-20001006.

XMLNS
"Namespaces in XML", T. Bray, D. Hollander, A. Layman, 14 Jan 1999. This W3C Recommendation is available at http://www.w3.org/TR/1999/REC-xml-names-19990114/.

XPTRFR
"XPointer Framework", P. Grosso, E. Maler, J. Marsh, N. Walsh, eds., 25 March 2003. This W3C Recommendation is available at http://www.w3.org/TR/2003/REC-xptr-framework-20030325/.

8. End Notes

1. Section 4.1 of [URI] defines a ***URI reference*** to be either a URI or a relative URI reference (e.g., "../file#id"). (Note 1 context.)
2. User agent configurations are usually defined by URI scheme, with exceptions defined by further substring matches within the URI. (Note 2 context.)
3. Access to the resource may require the use of more than one protocol. For instance, both DNS and HTTP are typically used to communicate with an origin server responsible for an "http" URI. Note also that other protocols than HTTP may be used to interact with a resource identified by an "http" URI. (Note 3 context.)
4. "Identity" (the characteristics of something) is distinct from "Identification" (the means used to name it). (Note 4 context.)
5. This principle dates back at least as far as Douglas Engelbart's seminal work on open hypertext systems; see section Every Object Addressable in [Eng90]. (Note 5 context.)
6. See the TAG finding *"URIs, Addressability, and the use of HTTP GET"* for some details about the interaction of this principle in HTTP application design. (Note 6 context.)
7. Some of these scheme specifications use the term "designate," which we take to mean the same thing as "identify." (Note 7 context.)
8. This is true even for HTTP PUT interactions, for example, as the authority may configure the server to accept or reject PUT data based on media type, validity constraints, or other constraints. (Note 8 context.)
9. The value of a 'Content-Type' field is an Internet Media Type. (Note 9 context.)
10. The title is somewhat misleading. It's not the URIs that change, it's what they identify. (Note 10 context.)
11. @@Text here on why SMTP part of Web@@ (Note 11 context.)
12. The term "Internet Media Type" has replaced the deprecated term "MIME type." (Note 12 context.)
13. In this document, the phrase "data optimized for machines" means that processors can make use of the data unattended by a human overseer.

http://www.w3.org/TR/2002/REC-P3P-20020416/.

RDDL
"Resource Directory Description Language (RDDL)", J. Borden, T. Bray, eds., 1 June 2003. This document is available at http://www.tbray.org/tag/rddl/rddl3.html.

RDF10
"Resource Description Framework (RDF) Model and Syntax Specification", O. Lassila, R. R. Swick, eds., 22 February 1999. This W3C Recommendation is available at http://www.w3.org/TR/1999/REC-rdf-syntax-19990222/.

REST
" Representational State Transfer (REST)", Chapter 5 of "Architectural Styles and the Design of Network-based Software Architectures", Doctoral Thesis of R. T. Fielding, 2000. Available at http://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm.

RFC2141
IETF "RFC 2141: URN Syntax", R. Moats, May 1997. Available at http://www.ietf.org/rfc/rfc2141.txt.

RFC2718
IETF "Guidelines for new URL Schemes", L. Masinter, H. Alvestrand, D. Zigmond, R. Petke, November 1999. Available at: http://www.ietf.org/rfc/rfc2718.txt.

RFC3023
IETF "RFC 3023: XML Media Types", M. Murata, S. St. Laurent, D. Kohn, January 2001. Available at: http://www.rfc-editor.org/rfc/rfc3023.txt

RFC3236
IETF "RFC 3236: The 'application/xhtml+xml' Media Type", M. Baker, P. Stark, January 2002. Available at: http://www.rfc-editor.org/rfc/rfc3236.txt

RFC3401
IETF "RFC 3401: Dynamic Delegation Discovery System (DDDS) Part One: The Comprehensive DDDS", M. Mealing, October 2002. Available at: http://www.rfc-editor.org/rfc/rfc3401.txt

SOAP12
"SOAP Version 1.2 Part 1: Messaging Framework", M. Gudgin, M. Hadley, N. Mendelsohn, J.-J. Moreau, H. Frystyk Nielsen, eds., 24 June 2003. This W3C Recommendation is available at http://www.w3.org/TR/2003/REC-soap12-part1-20030624/.

SVG10
"Scalable Vector Graphics (SVG) 1.1 Specification", J. Ferraiolo, Fujisawa Jun, D. Jackson, eds., 14 January 2003. This W3C Recommendation is available at http://www.w3.org/TR/2003/REC-SVG11-20030114/.

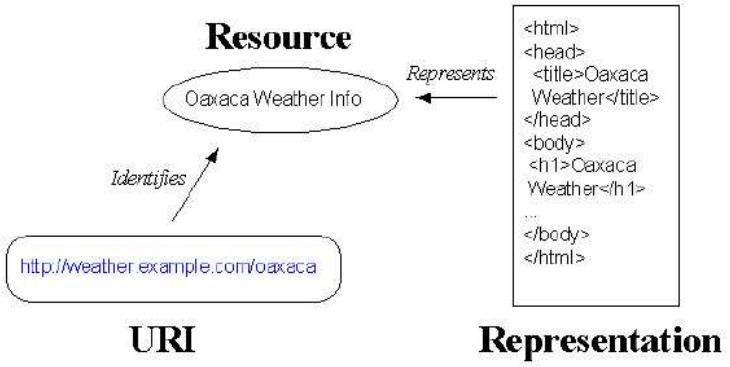
UNICODE
See the Unicode Consortium home page for information about the latest version of Unicode and character repertoires.

UniqueDNS
" IAB Technical Comment on the Unique DNS Root", B. Carpenter, 27 September 1999. Available at http://www.icann.org/correspondence/iab-tech-comment-27sept99.htm.

XHTML10
"XHTML 1.0: The Extensible HyperText Markup Language: A Reformulation of HTML 4 in XML 1.0", S. Pemberton et al., 26 January 2000, revised 1 August 2002. Available at http://www.w3.org/TR/2002/REC-xhtml1-20020801/.

1. Identification. Each resource is identified by a Uniform Resource Identifier (URI). In this travel scenario, the resource involves the weather in Oaxaca and the URI is "`http://weather.example.com/oaxaca`".
2. Interaction. Web agents exchange information via messages; these messages arise as the result of actions requested by a user or called for by a rendering engine while processing hypermedia-aware data formats. Protocols define the syntax and semantics of agent interactions, as well as the sequence of interactions expected for a given task. In the travel scenario, Nadia uses her browser to perform a retrieval action for the identified resource. The browser uses its configuration[2] to determine how to locate the identified information, which might be via a cache of prior retrieval actions, by contacting an intermediary (e.g., a proxy server), or by direct access to the server identified by the URI. In this example, the browser uses HTTP[3] to retrieve a representation from the origin server at "weather.example.com".
3. Representation. Messages carry representations of a resource. A resource communicates everything about its state through these representations, which are built from a non-exclusive set of data formats, used separately or in combination (including XHTML, CSS, PNG, XLink, RDF/XML, SVG, and SMIL animation). In this scenario, Nadia's browser receives representations in the form of an XHTML document and several SVG weather map images. The browser interprets the XHTML representation data, which in turn call for retrieval of weather maps through reference of their URIs, which results in rendering the SVG images.

The following illustration shows the simplest relationship between identifier, resource, and representation.



Editor's note: The TAG may include additional illustrations in this document to help explain important terms and their relationships.

1.1. About this Document

This document is an ongoing attempt to describe the properties we desire of the Web and the design choices that have been made to achieve them.

This document promotes re-use of existing standards when suitable, and gives some guidance on how to innovate in a manner consistent with the Web architecture.

1.1.1. Audience of this Document

The intended audience for this document includes:

1. Participants in W3C Activities; i.e., developers of Web technologies and specifications in W3C.
2. Other groups and individuals developing technologies to be integrated into the Web.
3. Implementers of W3C specifications, and those who use the resulting products.

The terms MUST, MUST NOT, SHOULD, SHOULD NOT, and MAY are used in accordance with RFC 2119 [RFC2119].

Readers will benefit from familiarity with the Requests for Comments (RFC) series from the IETF, some of which define pieces of the architecture discussed in this document.

1.1.2. Scope of this Document

This document focuses on the architecture of the Web. Other groups inside and outside W3C also address specialized aspects of Web architecture, including accessibility, internationalization, device independence, and Web Services. The section on Architectural Specifications includes some references.

This document attempts to balance the value of brevity and precision with the value of illustrative examples. TAG findings provide more background, motivation, and examples.

The architecture described in this document is primarily the result of experience. There has been some theoretical and modeling work in the area of Web architecture, notably Roy Fielding's work on "Representational State Transfer" [REST].

2. Identification and Resources

Uniform Resource Identifiers (URI), defined in "Uniform Resource Identifiers (URI): Generic Syntax" [URI], are central to Web architecture. URIs identify (i.e., name) resources.[4] Parties who wish to communicate about something agree upon a shared set of identifiers and on their meanings. This shared vocabulary has a tangible value: it reduces the cost of communication. The ability to use common

"Web Content Accessibility Guidelines 1.0," W. Chisholm, G. Vanderheiden, and I. Jacobs, eds., 5 May 1999. This W3C Recommendation is http://www.w3.org/TR/1999/WAI-WEBCONTENT-19990505/.

WSA
"Web Services Architecture," D. Booth, M. Champion, C. Ferris, F. McCabe, E. Newcomer, D. Orchard eds., 14 May 2003. This W3C Working Draft is http://www.w3.org/TR/2003/WD-ws-arch-20030514/. The latest version of this document is available at http://www.w3.org/TR/ws-arch/.

XAG
"XML Accessibility Guidelines", D. Dardailler, S. Palmer, C. McCathieNevile, 3 October 2002. This W3C Working Draft is http://www.w3.org/TR/2002/WD-xag-20021003. The latest version is available at http://www.w3.org/TR/xag.

7.3. Non-Normative References

Cool
"Cool URIs don't change" T. Berners-Lee, W3C, 1998 Available at http://www.w3.org/Provider/Style/URI.

DAMLOIL
"DAML+OIL (March 2001) Reference Description", D. Connolly, F. van Harmelen, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, 18 Dec 2001. This W3C Note is available at http://www.w3.org/TR/2001/NOTE-daml+oil-reference-20011218.

EXTLANG
"Web Architecture: Extensible Languages, T. Berners-Lee, D. Connolly, 10 February 1998. This W3C Note is available at http://www.w3.org/TR/1998/NOTE-webarch-extlang-19980210.

Eng90
"Knowledge-Domain Interoperability and an Open Hyperdocument System", D. C. Engelbart, June 1990.

Dan Connolly's list of URI schemes is a useful resource for finding out which references define various URI schemes.

IETFXML
IETF "Guidelines For The Use of XML in IETF Protocols," S. Hollenbeck, M. Rose, L. Masinter, eds., 2 November 2002. This IETF Internet Draft is available at http://www.imc.org/ietf-xml-use/xml-guidelines-07.txt. If this document is no longer available, refer to the ietf-xml-use mailing list.

IRI
IETF "Internationalized Resource Identifiers (IRIs)", M. Duerst, M. Suignard, Nov 2002. This IETF Internet Draft is available at http://www.w3.org/International/iri-edit/draft-duerst-iri.html. If this document is no longer available, refer to the home page for Editing 'Internationalized Resource Identifiers (IRIs)'.

OWL10
"Web Ontology Language (OWL) Reference Version 1.0", M. Dean, D. Connolly, F. van Harmelen, J. Hendler, I. Horrocks, D. L. McGuinness, P. F. Patel-Schneider, L. A. Stein, eds., 12 Nov 2002. This W3C Working Draft is available at http://www.w3.org/TR/2002/WD-owl-ref-20021112/.

P3P10
"The Platform for Privacy Preferences 1.0 (P3P1.0) Specification", M. Marchiori, ed., 16 April 2002. This W3C Recommendation is available at

URI
"Uniform Resource Identifiers (URI): Generic Syntax" (T. Berners-Lee, R. Fielding, L. Masinter, Eds.) is currently being revised. The IETF Internet Draft draft-fielding-uri-rfc2396bis-03 is expected to obsolete RFC 2396, which is the current URI standard. "Architecture of the World Wide Web" uses the concepts and terms defined in draft-fielding-uri-rfc2396bis-03, preferring them to those defined in RFC 2396. The TAG is tracking the evolution of draft-fielding-uri-rfc2396bis-03.

RFC2616
IETF "RFC 2616: Hypertext Transfer Protocol —HTTP/1.1 ", J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, June 1999. Available at http://www.ietf.org/rfc/rfc2616.txt.

RFC2717
IETF "Registration Procedures for URL Scheme Names", R. Petke, I. King, November 1999. Available at http://www.ietf.org/rfc/rfc2717.txt.

7.2. Architectural Specifications

ATAG10
"Authoring Tool Accessibility Guidelines 1.0," J. Treviranus, C. McCathieNevile, I. Jacobs, and J. Richards, eds., 3 February 2000. This W3C Recommendation is http://www.w3.org/TR/2000/REC-ATAG10-20000203/.

CHARMOD
"Character Model for the World Wide Web," M. Dürst and F. Yergeau, eds., 30 April 2002. This W3C Working Draft is http://www.w3.org/TR/2002/WD-charmod-20020430/. The latest version is available at http://www.w3.org/TR/charmod/.

DIPRINCIPLES
"Device Independent Principles," R. Gimson, Ed., 18 September 2001. This W3C Working Draft is http://www.w3.org/TR/2001/WD-di-princ-20010918/. The latest version is available at http://www.w3.org/TR/di-princ/.

Fielding
"Principled Design of the Modern Web Architecture", R.T. Fielding and R.N. Taylor, UC Irvine. In Proceedings of the 2000 International Conference on Software Engineering (ICSE 2000), Limerick, Ireland, June 2000, pp. 407-416. This document is available at http://www.ics.uci.edu/~fielding/pubs/webarch_icse2000.pdf.

RFC1958
IETF "RFC 1958: Architectural Principles of the Internet", B. Carpenter, June 1996. Available at http://www.ietf.org/rfc/rfc1958.txt.

QA
"QA Framework: Specification Guidelines," D. Hazaël-Massieux, L. Henderson, L. Rosenthal, D. Dimitriadis, K. Gavrylyuk, eds., 10 February 2003.This W3C Working Draft is http://www.w3.org/TR/2003/WD-qaframe-spec-20030210/. The latest version is available at http://www.w3.org/TR/qaframe-spec/.

UAAG10
"User Agent Accessibility Guidelines 1.0," I. Jacobs, J. Gunderson, E. Hansen, eds., 17 December 2002. This W3C Recommendation is http://www.w3.org/TR/2002/REC-UAAG10-20021217/.

WCAG10

identifiers across communities is what motivates global naming in Web architecture.

When a representation of one resource refers to another resource with a URI, a ***link*** is formed between the two resources. The networked information space is built of linked resources, and the large-scale effect is a shared information space. The value of the Web grows exponentially as a function of the number of linked resources (the "network effect").

A URI must be assigned to a resource in order for the resource to be named, shared, or linked to within the information space. It follows that a resource should be assigned a URI if a third party might reasonably want to link to it, make or refute assertions about it, retrieve or cache a representation of it, include all or part of it by reference into another representation, annotate it, or perform other operations on it.

> ***Principle: Use URIs***
>
> A URI SHOULD be assigned to each resource that is intended to be identified, shared, or described by reference.[5]

Web architecture does not constrain resources to be uniquely named; a resource may be assigned more than one URI.

There are many benefits to assigning a URI to a resource. Some are by design (e.g., linking, book marking, and caching), others (e.g., global search services) were not predicted.[6]

Editor's note: The TAG has not yet reached agreement about whether to distinguish "information resources" from other types of resources. An information resource is one that conveys information (via representations). See TAG issue httpRange-14. Related to the concept of "information resource" is the expression "on the Web". Roy Fielding suggested this definition of "on the Web": "A resource is considered to be "on the Web" if it can be independently referred to by at least one URI, even if access to that resource is restricted." Others have expressed that actual access should be a requirement as well.

2.1. Comparing Identifiers

The most straightforward way of establishing that two parties are referring to the same resource is to compare, as character strings, the URIs they are using. In section 6 of [URI], equivalence or difference of URIs is determined by string comparison.

> ***Good practice: Compare URI characters***
>
> If a URI has been assigned to a resource, Web agents SHOULD refer to the resource using the same URI, character for character.

The term "character" refers to URI characters as defined section 2 of [URI].

Although it is possible to determine that two URIs are equivalent, it is generally not possible to be sure that two URIs that are not equivalent identify different resources. Applications may apply rules beyond basic string comparison (e.g., for "http" URIs, the authority agent is case-insensitive) to reduce the risk of false negatives and positives. Please refer to section 6.1 of [URI] for more information about reducing the risk of false negatives and positives.

Web agents that reach conclusions based on comparisons done through means other than those licensed by relevant specifications take responsibility for any problems that result.

Agents should not assume, for example, that "`http://weather.example.com/Oaxaca`" and "`http://weather.example.com/oaxaca`" identify the same resource, since none of the specifications involved states that the path part of an "http" URI is case-insensitive.

To help parties know when they are referring to the same resource, it follows that URI producers should be conservative about the number of different URIs they produce for the same resource.

Thus, the parties responsible for weather.example.com should not use both "`http://weather.example.com/Oaxaca`" and "`http://weather.example.com/oaxaca`" to refer to the same resource; agents will not detect the equivalence relationship by following specifications.

There may be other ways to establish that two parties are identifying the same resource that are not based on string comparison; see the section on future directions for determining that two URIs identify the same resource.

2.2. URI Opacity

Although it is tempting to guess at the nature of a resource by inspection of a URI that identifies it, this is not licensed by specifications; this is called ***URI opacity***.

> ***Good practice: URI Opacity***
>
> People and software making use of URIs assigned outside of their own authority MUST NOT attempt to infer properties of the referenced resource except as licensed by relevant normative specifications or by URI assignment policies published by the relevant URI assignment authority.

The example URI used in the travel scenario ("`http://weather.example.com/oaxaca`") suggests that the identified resource has something to do with the weather in Oaxaca. A site reporting the weather in Oaxaca could just as easily be identified by the URI "`http://vjc.example.com/315`". And the URI "`http://weather.example.com/vancouver`" might identify the resource "my photo album."

- Uniform Resource Identifier (URI)
- Web agent
- access method
- backwards compatibility
- data format
- extensible format
- format specification
- forwards compatibility
- fragment identifier
- link
- message
- message metadata
- namespace document
- representation
- representation data
- representation metadata
- resource
- resource metadata
- safe interaction
- secondary resource
- unsafe interaction
- user agent
- xml-based

7. References

7.1. Normative References

Editor's note: The usage of a normative reference in this document needs clarification.

IANASchemes
IANA's online registry of URI Schemes is available at http://www.iana.org/assignments/uri-schemes.
Dan Connolly's list of URI schemes is a useful resource for finding out which references define various URI schemes.

MEDIATYPEREG
IANA's online registry of Internet Media Types is available at http://www.iana.org/assignments/media-types/index.html.

RFC2045
IETF "RFC 2045: Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies", N. Freed, N. Borenstein, November 1996. Available at http://www.ietf.org/rfc/rfc2045.txt.

RFC2046
IETF "RFC 2046: Multipurpose Internet Mail Extensions (MIME) Part Two: Media Types", N. Freed, N. Borenstein, November 1996. Available at http://www.ietf.org/rfc/rfc2046.txt.

RFC2119
IETF "RFC 2119: Key words for use in RFCs to Indicate Requirement Levels", S. Bradner, March 1997. Available at http://www.ietf.org/rfc/rfc2119.txt.

discussion in the community for a variant of XHTML optimized for the construction of namespace documents which meet the goals described in this section. Note, however, that RDDL (or a format like it) is no more universally correct than any other type of representation. Namespace developers should give careful consideration to choosing the most appropriate format for their application, keeping in mind the utility of information meant for people to read as well as information optimized for machines.

5. Glossary

Glossary not yet completed.

5.1. Principles, Constraints, etc.

Editor's note: The TAG is still experimenting with the categorization of points in this document. This list is likely to change. It has also been suggested that the categories clearly indicate their primary audience.

The important points of this document are categorized as follows:

Constraint
: An architectural constraint is a restriction in behavior or interaction within the system. Constraints may be imposed for technical, policy, or other reasons.

Design Choice
: In the design of the Web, some design choices, like the names of the <p> and <li> elements in HTML, or the choice of the colon character in URIs, are somewhat arbitrary; if <par>, <elt>, or * had been chosen instead, the large-scale result would, most likely, have been the same. Other design choices are more fundamental; these are the focus of this document.

Good practice
: Good practice —by software developers, content authors, site managers, users, and specification writers —increases the value of the Web.

Principle
: An architectural principle is a fundamental rule that applies to a large number of situations and variables. Architectural principles include "separation of concerns", "generic interface", "self-descriptive syntax," "visible semantics," "network effect" (Metcalfe's Law), and Amdahl's Law: "The speed of a system is determined by its slowest component."

Property
: Architectural properties include both the functional properties achieved by the system, such as accessibility and global scope, and non-functional properties, such as relative ease of evolution, re-usability of components, efficiency, and dynamic extensibility.

6. Index

On the other hand, the "mailto" URI scheme specification states that mail URIs identify Internet mailboxes. That normative specification authorizes people and software to infer that the identified resource is an Internet mailbox, although it doesn't guarantee that the authority who minted the URI is actually using the URI to identify a mailbox.

People and software using a URI assigned outside of their own authority should make as few inferences as possible about the identified resource based on the identifier. The more a piece of software depends on such inferences, the more fragile it becomes to change and the lower its generic utility.

For information about how agents convey information about a resource, see the section on retrieving a representation.

Editor's note: See TAG issue metadataInURI-31 and TAG finding The use of Metadata in URIs. There is a related principle that has not yet been captured: don't restrict (e.g., through specifications) the URI space allotted to resource owners. See TAG issue siteData-26: Web site metadata improving on robots.txt, w3c/p3p and favicon etc.

2.3. URI Schemes

In the URI "`http://weather.example.com/`", the "http" that appears before the colon (":") is a URI scheme name. Each URI scheme has a normative specification that explains how to assign identifiers within that scheme. The URI syntax is thus a federated and extensible naming mechanism wherein each scheme's specification may further restrict the syntax and semantics of identifiers within that scheme. Furthermore, the URI scheme specification specifies whether and how an agent can dereference the URI.

Several URI schemes allow URI addressing of resources in information systems that pre-date the Web:

- mailto URIs identify[7] Internet mailboxes:
  `mailto:nobody@example.org`
- ftp URIs identify files and directories accessible using the FTP protocol:
  `ftp://example.org/aDirectory/aFile`
- news URIs identify USENET newsgroups:
  `news:comp.infosystems.www`
- tel URIs identify terminals on the telephone network:
  `tel:+1-816-555-1212`

Other URI schemes have been introduced since the advent of the Web, including those introduced as a consequence of new protocols. Examples of such URI schemes include:

- http URIs:
  `http://www.example.org/something?with=arg1;and=arg2`
- ldap URIs:
  `ldap://ldap.example.org/c=GB?objectClass?one`
- URNs:

```
urn:oasis:SAML:1.0
```

The Internet Assigned Numbers Authority (IANA) maintains a registry [IANASchemes] of mappings between URI scheme names and scheme specifications. For instance, the IANA registry indicates that the "http" scheme is defined in [RFC2616]. The process for registering a new URI scheme is defined in [RFC2717].

Since many aspects of URI processing are scheme-dependent, and since a huge amount of deployed software already processes URIs of well-known schemes, the cost of introducing a new URI scheme is high.

Good practice: New URI schemes

Authors of specifications SHOULD NOT introduce a new URI scheme when an existing scheme provides the desired properties of identifiers and their relation to resources.

Consider our travel scenario: should the authority providing information about the weather in Oaxaca register a new URI scheme "weather" for the identification of resources related to the weather? They might then publish URIs such as "`weather://travel.example.com/oaxaca`". While the Web architecture allows the definition of new schemes, there is a cost to registration and especially deployment of new schemes. When an agent dereferences such a URI, if what really happens is that HTTP GET is invoked to retrieve an HTML representation of the resource, then an "http" URI would have sufficed. If a URI scheme exists that meets the needs of an application, designers should use it rather than invent one. Furthermore, designers should expect that it will prove useful to be able to share a URI across applications, even if that utility is not initially evident.

If the motivation behind registering a new scheme is to allow an agent to launch a particular application when retrieving a representation, such dispatching can be accomplished at lower expense by registering a new Internet Media Type instead. Deployed software is more likely to handle the introduction of a new media type than the introduction of a new URI scheme.

The use of unregistered URI schemes is discouraged for a number of reasons:

- There is no generally accepted way to locate the scheme specification.
- Someone else may be using the scheme for other purposes.
- One should not expect that general-purpose software will do anything useful with URIs of this scheme; the network effect is lost.

2.4. Fragment Identifiers

Story

When navigating within the XHTML data that Nadia receives as a representation of the resource identified by "`http://weather.example.com/oaxaca`", Nadia finds that the URI "`http://weather.example.com/oaxaca#tom`" refers to information about

To further complicate matters, DTDs establish the ID type in the Infoset whereas W3C XML Schema produces a PSVI but does not modify the original Infoset. This leaves open the possibility that a processor might only look in the Infoset and consequently would fail to recognize schema-assigned IDs.

Editor's note: W3C's XML Core Working Group is investigating the question of fragment identifier semantics.

TAG issue fragmentInXML-28: Do fragment identifiers refer to a syntactic element (at least for XML content), or can they refer to abstractions?

TAG issue xmlIDSemantics-32: How should the problem of identifying ID semantics in XML formats be addressed in the absence of a DTD? See TAG finding *"How should the problem of identifying ID semantics in XML formats be addressed in the absence of a DTD?"*.

4.9.6. Media Types for XML

RFC 3023 defines the Internet Media Types `application/xml` and `text/xml`, and describes a convention whereby XML-based data formats use Internet Media Types with a `+xml` suffix, for example `image/svg+xml`.

These Internet Media Types create two problems: First, for data identified as `text/*`, Web intermediaries are allowed to "transcode", i.e., convert one character encoding to another. Since XML documents are designed to allow them to be self-describing, and since this is a good and widely-followed practice, any such transcoding will make the self-description false and will cause the document to be not well-formed.

Second, representations whose Internet Media Types begin with `text/` are required, unless the `charset` parameter is specified, to be considered to be encoded in US-ASCII. In the case of XML, since it is self-describing, it is good practice to omit the `charset` parameter, and since XML is very often not encoded in US-ASCII, the use of "`text/`" Internet Media Types effectively precludes this good practice.

Good practice: XML and `text/*`

In general, Internet Media Types beginning with `text/` SHOULD NOT be assigned to XML representations.

4.10. Future Directions for Representations and Formats

There remain open questions regarding resource representations. The following sections identify a few areas of future work in the Web community. The TAG makes no commitment at this time to pursuing these issues.

4.10.1. Namespace Document Formats

The Resource Directory Description Language [RDDL] is a proposal under

information.

Issue: namespaceDocument-8: What should a "namespace document" look like?

Issue: abstractComponentRefs-37: Definition of abstract components with namespace names and frag ids

4.9.5. Fragment Identifiers and ID Semantics

Suppose that the URI "`http://example.com/oaxaca`" defines a resource with representations encoded in XML. What, then, is the interpretation of the URI "`http://example.org/oaxaca#weather`"?

The URI specification [URI] makes it clear that the interpretation depends on the media type of the representation data. Per the previous good practice note on fragment identifiers, designers of an XML-based format specification should define the semantics of fragment identifiers in that format. The XPointer Framework [XPTRFR] provides a interoperable starting point.

When the media type assigned to representation data is `application/xml`, there are no semantics defined for fragment identifiers,[14] and authors should not make use of fragment identifiers in such data. The same is true if the assigned media type has the suffix `+xml` (defined in "XML Media Types" [RFC3023]) as there is no normative specification of fragment semantics in this case.

Many people assume that the fragment identifier `#abc`, when referring to XML data, identifies the element in the document with the ID "abc". However, there is no normative support for this assumption and it is problematic in practice.

Unfortunately, there are a number of open issues associated with finding the element with the ID "abc" in an XML document. In XML, the quality of "being an ID" is associated with the type of the attribute, not it's name. Consider the following fragment: `<section name="foo">`. Does the `section` element have the ID "foo"? One cannot answer this question by examining the element and its attributes alone. Finding the IDs in a document requires additional processing.

1. Processing the document with a processor that recognizes DTD attribute list declarations (in the external or internal subset) might reveal a declaration that identifies the name attribute as an ID. **Note:** This processing is not necessarily part of validation. A non-validating, DTD-aware processor can perform ID assignment.
2. Processing the document with a W3C XML Schema might reveal an element declaration that identifies the name attribute as an `xs:ID`.
3. In practice, processing the document with another schema language, such as RELAX NG, might reveal the attributes of type ID. Many modern specifications begin processing XML at the Infoset level and do not specify normatively how an Infoset is constructed. For those specifications, any process that establishes the ID type in the Infoset (and/or Post Schema Validation Infoset, or PSVI) may successfully identify the attributes of type ID.

tomorrow's weather in Oaxaca. This URI includes the fragment identifier "tom" (the string after the "#").

2.4.1. Secondary Resources Identified With a Fragment Identifier

The ***fragment identifier*** of a URI allows indirect identification of a ***secondary resource*** by reference to a primary resource and additional information. More precisely:

- If URI "U" identifies primary resource "R", and
- a representation of "R" is in the data format "F", and
- the format specification for "F" specifies that fragment identifiers in instances of "F" identify secondary resources, then
- the URI for the secondary resource identified within an instance of "F" by fragment identifier "fragid" is "U#fragid".

The secondary resource may be some portion or subset of the primary resource, some view on representations of the primary resource, or some other resource.

The syntax and semantics of fragment identifiers are defined by the set of representations that might result from a retrieval action on the primary resource. Fragment identifier semantics differ among format specifications. The presence of a fragment identifier in a URI does not imply that a retrieval action will take place.

Interpretation of the fragment identifier during a retrieval action is performed solely by the agent; the fragment identifier is not passed to other systems during the process of retrieval. This means that some intermediaries in the Web architecture (e.g., proxies) have no effect on fragment identifiers and that redirection (in HTTP [RFC2616], for example) does not account for them.

2.4.2. Fragment Identifiers and Content Negotiation

Story

The authority responsible for "weather.example.com" provides a visual map of the meteorological conditions in Oaxaca as part of the representation served for "`http://weather.example.com/oaxaca`". They encode the same visual map in a number of image formats to meet different needs (e.g., they might serve PNG, SVG, and JPEG/JFIF). Nadia's browser and the server engage in HTTP content negotiation, so that Nadia receives the best image format her browser can handle or the image format she usually prefers.

The URI "`http://weather.example.com/oaxaca/map#zicatela`" refers to a portion of the weather map that shows the Zicatela Beach, where Nadia intends to go surfing.

Clients can do something useful with fragment identifiers if the format used by the

representation, e.g. SVG, defines fragment identifier semantics. On the other hand, clients cannot do anything useful with the fragment identifier and the PNG or JPEG/JFIF representations because their fragment identifier semantics (which are undefined) are inconsistent with those of SVG.

Errors can occur when the authority responsible for a resource publishes a URI with a fragment identifier and representations of the resource do not have consistent fragment identifier semantics. Thus, the authority responsible for "`http://weather.example.com/oaxaca/map#zicatela`" introduces the possibility of error by making available PNG and JPEG/JFIF representations when a fragment identifier may be present.

> ***Good practice: Content negotiation with fragments:***
>
> Authorities responsible for minting a URI with a fragment identifier and who use content negotiation to serve multiple representations of the identified resource SHOULD NOT serve representations with inconsistent fragment identifier semantics.

See related TAG issues httpRange-14 and RDFinXHTML-35 and abstractComponentRefs-37.

2.5. Using a URI to Access a Resource

To dereference a URI means to access the resource identified by the URI. Access may take many forms, including retrieving a representation (e.g., using HTTP GET or HEAD), modifying the state of the resource (e.g., using HTTP POST or PUT), and deleting the resource (e.g., using HTTP DELETE).

When accessing a resource, an agent applies relevant specifications in succession, beginning with the specification of the context in which the URI is found (e.g., a format or protocol specification, or an application). Any one of these specifications may define more than one access mechanism (e.g., the HTTP protocol defines a number of ***access methods***, including GET, HEAD, and POST). Note that the information governing the choice of access mechanism may be found in the context, not the URI itself (e.g., the choice of HTTP GET v. HTTP HEAD). The TAG finding *"URIs, Addressability, and the use of HTTP GET and POST."* discusses issues surrounding multiple access mechanisms and the relation to URI addressability.

Some URI schemes (e.g., the URN scheme [RFC 2141]) do not define dereference mechanisms.

See related TAG issue metadataInURI-31: Should metadata (e.g., versioning information) be encoded in URIs?

2.5.1. Retrieving a Representation

One of the most important actions involving a resource is to retrieve a representation of it (for example, by using HTTP GET; HTTP POST does **not**

and interoperability risks associated with them.

See the TAG finding *"Using QNames as Identifiers in Content"* for more information. See also TAG issues rdfmsQnameUriMapping-6 and qnameAsId-18.

4.9.4. Namespace Documents

> ***Story***
>
> Nadia receives a representation from "weather.example.com" in an unfamiliar data format. She knows enough about XML to recognize which XML namespace the elements belong to. Since the namespace is identified by a URI, she asks her browser to retrieve a representation of the namespace via that URI. Nadia is requesting the ***namespace document***.

Nadia gets back some useful data that allows her to learn more about the data format. Nadia's browser may also be able to use data optimized for machines[13] automatically to perform useful tasks on Nadia's behalf, such as download additional agents to process and render the format.

There are many reasons why a person or other agent might want more information about the namespace. A person might want to:

- understand its purpose,
- learn how to use the markup vocabulary in the namespace,
- find out who controls it,
- request authority to access schemas or collateral material about it, or
- report a bug or situation that could be considered an error in some collateral material.

A namespace document should also support the automatic retrieval of other Web resources that support processing the markup from this vocabulary. Useful information to processors includes:

- schemas, to use for validation,
- style sheets, to use for presentation,
- ontologies, to use for making inferences, or
- any number of other application-specific details.

> ***Good practice: Namespace documents***
>
> XML namespace designers SHOULD make available material intended for people to read and material optimized for machines in order to meet the needs of those who will use the namespace vocabulary.

In general, there is no "best" data format for encoding a namespace document. See the section on future work regarding namespace document formats for more

difficult, perhaps impractical in some cases.

Good practice: Use Namespaces

Format specifications that create new XML vocabularies SHOULD place all element names and global attribute names in a namespace.

Attributes are always scoped by the element on which they appear. In that respect they are a somewhat special case. An attribute that is "global," that is, one that might meaningfully appear on a great many elements, including elements in other namespaces, should be explicitly placed in a namespace. Local attributes, ones associated with only a particular element, need not be included in a namespace since their meaning will always be clear from the context provided by that element.

The `type` attribute from W3C XML Schema is an example of a global attribute. It can be used by authors of any vocabulary to make an assertion about the type of the element on which it appears. The `type` attribute occurs in the W3C XML Schema namespace and must always be fully qualified. The `frame` attribute on an HTML table is an example of a local attribute. There is no value in placing that attribute in a namespace since the attribute is very unlikely to be useful on an element other than an HTML table.

The most significant technical drawback to using namespaces is that they do not interact well with DTDs. DTDs perform validation based on the lexical form of the name, making prefixes semantically significant in ways that are not desirable. As other schema technologies become widely deployed, this drawback will diminish in significance.

4.9.3. Hyperlinks in XML

Sophisticated linking mechanisms have been invented for XML formats. XPointer allows links to address content that does not have an explicit, named anchor. XLink allows links to have multiple ends and to be expressed either inline or in "link bases" stored external to any or all of the resources identified by the links it contains.

For formats based on XML, format designers should examine XLink and the XPointer framework for inspiration. To define fragment identifier syntax, use at least the XPointer Framework and XPointer element() Schemes.

TAG issue: What is the scope of using XLink? xlinkScope-23.

If a future revision of RFC 3023 identifies the XPointer Framework, element(), and perhaps other ancillary schemes as the fragment identifier syntax for XML documents, authors will be able to rely on at least those schemes for all XML documents.

Good practice: QName caution

Format specification designers MAY use Qualified Names (QNames) for identifiers in representation data, but should be aware of the limitations

retrieve a representation of the identified resource). The authority responsible for assigning a URI to a resource determines which representations are used for interaction with the resource.[8] The representations communicate all or part of the state of the resource.

Good practice: Available representations

Owners of important resources SHOULD make available representations of those resources.

As an example of representation retrieval, suppose that the URI "`http://weather.example.com/oaxaca`" is used within an `a` element of an SVG document. A succession of agents carries out the retrieval by implementing the following relevant specifications:

1. The SVG 1.0 Recommendation [SVG10], which imports the link semantics defined in XLink 1.0 [XLink10]. Section 17.1 of the SVG specification suggests that interaction with an `a` link involves retrieving a representation of a resource, identified by the XLink `href` attribute: "By activating these links (by clicking with the mouse, through keyboard input, and voice commands), users may visit these resources."
2. The attribute `xlink:href` is defined in section 5.4 of the XLink 1.0 [XLink10] specification, which states that "The value of the href attribute must be a URI reference as defined in [IETF RFC 2396], or must result in a URI reference after the escaping procedure described below is applied."
3. The URI specification [URI] states that "Each URI begins with a scheme name that refers to a specification for assigning identifiers within that scheme." The URI scheme name in this example is "http".
4. To find out what specification defines the "http" scheme, we look up the mapping in [IANASchemes]; the "http" URI scheme is defined in the HTTP/1.1 specification (RFC 2616 [RFC2616], section 3.2.2).
5. The HTTP/1.1 specification defines several access mechanisms. In this SVG context, the user agent employs the GET method (defined in section 9.3 of [RFC2616]) to retrieve the representation. The HTTP/1.1 specification explains how the server constructs the response (section 6 of [RFC2616]), including the 'Content-Type' field [9]. Section 1.4 states "HTTP communication usually takes place over TCP/IP connections." Though not shown in this example, the user agent would continue this process by implementing those specifications.
6. The user agent interprets the returned representation according to the value of the 'Content-Type' field, by looking up which specification defines the Internet Media Type in the [MEDIATYPEREG] registry.

Note that, in general, one cannot determine the Internet Media Type(s) of representation(s) of a resource by inspecting a URI for that resource. For example, do not assume that all representations of "`http://example.com/page.html`" are HTML. The HTTP protocol does not constrain the Internet Media Type based on the path component of the URI; the server is free to return a representation in the PNG or any other format for that URI.

2.5.2. Safe Interaction

Nadia's retrieval of weather information qualifies as a "safe" interaction; a ***safe interaction*** is one where the agent does not commit to anything beyond the interaction and is not responsible for any consequences other than the interaction itself (e.g., a read-only query or lookup). Other Web interactions resemble orders more than queries. These ***unsafe interactions*** may cause a change to the state of a resource and the user may be held responsible for the consequences of these interactions. Unsafe interactions include subscribing to a newsletter, posting to a list, or modifying a database.

Safe interactions are important because these are interactions where users can browse with confidence and where agents (e.g., search engines and browsers that pre-cache data for the user) can follow links safely. Users (or agents acting on their behalf) do not commit themselves to anything by querying a resource or following a link.

> ***Principle: Safe retrieval***
>
> Agents do not incur obligations by retrieving a representation.

For instance, it is incorrect to publish a link (e.g., "`http://example.com/oaxaca/newsLetter`") that, when followed, subscribes a user to a mailing list. Remember that search engines may follow such links.

For more information about safe and unsafe operations using HTTP GET and POST, and handling security concerns around the use of HTTP GET, see the TAG finding *"URIs, Addressability, and the use of HTTP GET and POST."*

2.6. URI Persistence and Ambiguity

The value of a URI increases with the predictability of interactions using that URI.

> ***Good practice: URI persistence***
>
> Parties responsible for a URI SHOULD service that URI predictably and consistently.

Service breakdowns include:

- No service available (i.e., dereferencing fails).
- Inconsistent representations served. Note the difference between a resource owner changing representations predictably in light of the nature of the resource (e.g., the weather in Oaxaca changes) and the owner changing representations arbitrarily.
- Improper use of content negotiation, such as serving two images as equivalent through HTTP content negotiation, where one image represents a square and the other a circle.

govern whether or not XML ought to be used, as well as specific guidelines on how it ought to be used. While it is directed at Internet applications with specific reference to protocols, the discussion is generally applicable to Web scenarios as well.

The discussion here should be seen as ancillary to the content of [IETFXML]. Refer also to "XML Accessibility Guidelines" [XAG] for help designing XML formats that lower barriers to Web accessibility for people with disabilities.

4.9.1. When to Use an XML-Based Format

XML defines textual data formats that are naturally suited to describing data objects which are hierarchical and processed in an in-order sequence. It is widely, but not universally applicable for format specifications; an audio or video format, for example, is unlikely to be well suited to representation in XML. Design constraints that would suggest the use of XML include:

1. Explicit representation of a hierarchical structure.
2. The data's usefulness should outlive the tools currently used to process it.
3. Ability to support internationalization in a self-describing way that makes confusion over coding options unlikely.
4. Early detection of encoding errors with no requirement to "work around" such errors.
5. A high proportion of human-readable textual content.
6. Potential composition of the data format with other XML-encoded formats.

4.9.2. XML Namespaces

> ***Story***
>
> The authority responsible for "weather.example.com" realizes that it can provide more interesting representations by creating instances that consist of elements defined in different XML-based formats, such as XHTML, SVG, and MathML.

How do the application designers ensure that there are no naming conflicts when they combine elements from different formats (e.g., suppose that the "p" element is defined in two or more XML formats)? "Namespaces in XML" [XMLNS] provides a mechanism for establishing a globally unique name that can be understood in any context.

The "expanded name" of an XML element or attribute name is the combination of its namespace URI and its local name. This is represented lexically in documents by associating namespace names with (optional) prefixes and combining prefixes and local names with a colon as described in "Namespaces in XML."

Format specification designers that declare namespaces thus provide a global context for instances of the data format. Establishing this global context allows those instances (and portions thereof) to be re-used and combined in novel ways not yet imagined. Failure to provide a namespace makes such re-use more

representations which embody legally-binding transactions are an obvious example. In such cases, the use of digital signatures may be appropriate to achieve immutability.

On the other hand, where such requirements are not in play, representations that are re-usable and re-purposable are in general higher in value, particularly in the case where the information's utility may be long-lived.

See also TAG issues Issue formattingProperties-19 and contentPresentation-26.

4.8. Hyperlinks

One of the greatest strengths of HTML as a format is that it allows authors to embed hyperlink references to other resources via URIs. The simplicity of `<a href="#foo">` as a link to `foo` and `<a name="foo">` as the anchor `foo` are partly (perhaps largely) responsible for the birth of the hypertext Web as we know it today.

There are linking paradigms that are more powerful than HTML's single-ended, single-direction, inline hyperlinks. But HTML-style hyperlinks are easy to understand, and they can be authored by individuals (or other agents) that have no control or write access to the other end point.

Good practice: Link mechanisms

Format specification designers SHOULD provide mechanisms for identifying links to other resources and to portions of representations (via fragment identifiers).

Good practice: Web linking

Format specification designers SHOULD provide mechanisms that allow Web-wide linking, not just internal document linking.

Good practice: URI genericity

Format specification designers SHOULD allow authors to use URIs without constraining them to a limited set of URI schemes.

See also the section on hyperlinks in XML.

4.9. XML-Based Data Formats

Many representations contain data which is ***XML-based***, that is to say conforms to the syntax rules defined in the XML specification [XML10]. This section discusses issues that are specific to such formats. Anyone seeking guidance in this area is urged to consult the "Guidelines For the Use of XML in IETF Protocols" [IETFXML], which contains a very thorough discussion of the considerations that

There are strong social expectations that once a URI identifies a particular resource, it should continue indefinitely to refer to that resource; this is called ***URI persistence***. URI persistence is always a matter of policy and commitment on the part of authorities servicing URIs. The choice of a particular URI scheme provides no guarantee that those URIs will be persistent or that they will not be persistent.

URI ambiguity refers to the use of the same URI to refer to more than one distinct thing.

Good practice: URI ambiguity

Avoid URI ambiguity.

Ambiguous use of a URI can be costly. Suppose that one division of Example, Inc. maintains data about company Web pages, including who created them and when. Another division in the company maintains data about companies, including who created them and when. The second data set uses the URIs of the organization's home page to identify the organization itself. When the two data sets are merged, the reuse of the URI causes information about companies to be merged with that of the home pages, resulting in potentially costly nonsense.

Ambiguity is an error and should not be confused with indirect identification, which is common. One indirectly identifies things though related things, such as people by their mailboxes, and organizations by their Web pages. For example, when conference organizers ask meeting participants to register by giving their email addresses, both parties know that they are using the mailbox identifier to indirectly identify the person. In terms of Web architecture, "`mailto:joe@example.com`" still identifies a mailbox, not a person.

The statement "'`http://www.example.com/moby`' identifies 'Moby Dick'" is a source of ambiguity since one could understand the statement to refer to very distinct resources: a particular printing of this work, or the work itself in an abstract sense, or the fictional white whale, or a particular copy of the book on the shelves of a library (via the Web interface of the library's online catalog), or the record in the library's electronic catalog which contains the metadata about the work, or the Gutenberg project's online version.

HTTP [RFC2616] has been designed to help service URIs. For example, HTTP redirection (via some of the 3xx response codes) permits servers to tell a user agent that further action needs to be taken by the user agent in order to fulfill the request (e.g., the resource has been assigned a new URI). In addition, content negotiation also promotes consistency, as a site manager is not required to define new URIs when adding support for a new format specification. Protocols that do not support content negotiation (e.g., FTP) require a new identifier when a new format is introduced.

For more discussion about URI persistence, refer to [Cool].[10]

2.7. Access Control

As we have seen, identification of a resource is distinct from interaction with that resource. It is reasonable to limit access to the resource (e.g., for security reasons), but it is unreasonable to prohibit others from merely identifying the resource.

As an analogy: A building might have a policy that the public may only enter via the main front door, and only during business hours. People employed in the building and in making deliveries to it might use other doors as appropriate. Such a policy would be enforced by a combination of security personnel and mechanical devices such as locks and pass-cards. One would not enforce this policy by hiding some of the building entrances, nor by requesting legislation requiring the use of the front door and forbidding anyone to reveal the fact that there are other doors to the building.

Story

Nadia and Dirk both subscribe to the "weather.example.com" newsletter. Nadia wishes to point out an article of particular interest to Dirk, using a URI. The authority responsible for "weather.example.com" can offer Nadia and Dirk the benefits of URIs (e.g., book marking and linking) and still limit access to the newsletter to authorized parties.

The Web provides several mechanisms to control access to resources, none of which relies on hiding or suppressing URIs for those resources. For more information on identification and access control, please refer to the TAG finding "*'Deep Linking' in the World Wide Web*."

2.8. Future Directions for Identifiers

There remain open questions regarding identifiers on the Web. The following sections identify a few areas of future work in the Web community. The TAG makes no commitment at this time to pursuing these issues.

2.8.1. Internationalized Identifiers

The integration of internationalized identifiers (i.e., composed of characters beyond those allowed by [URI]) into the Web architecture is an important and open issue. See TAG issue IRIEverywhere-27 for discussion about work going on in this area.

2.8.2. Determination that Two URIs Identify the Same Resource

Emerging Semantic Web technologies, including "DAML+OIL" [DAMLOIL] and "Web Ontology Language (OWL)" [OWL10], define RDF properties such as `equivalentTo` and `FunctionalProperty` to state —or at least claim —formally that two URIs identify the same resource.

2.8.3. Work on Dynamic Authority Delegation

Near the middle of the spectrum, there are container formats such as SOAP which fully expect to be composed from multiple namespaces but which provide an overall semantic relationship of message envelope and payload.

These relationships can be mixed and nested arbitrarily. In principle, a SOAP message can contain a JPEG image that contains an RDF comment that references a vocabulary of terms for describing the image.

Composition is related to but distinct from the question of whether a data format is intended as a final form. For example, one can imagine embedding SVG in a JPEG image thus combining final-form and reusable data, whereas a SOAP envelope might provide nothing more than a container for a particular payload that has no final form at all.

TAG issue xmlProfiles-29: When, whither and how to profile W3C specifications in the XML Family?

TAG issue mixedUIXMLNamespace-33: Composability for user interface-oriented XML namespaces

TAG issue xmlFunctions-34: XML Transformation and composability (e.g., XSLT, XInclude, Encryption)

TAG issue RDFinXHTML-35: Syntax and semantics for embedding RDF in XHTML

4.7. Presentation, Content, and Interaction

Replacement text from C. Lilley expected.

In many cases, the information encoded in a data format is logically separable from the choice of ways in which it may be presented to a human, and the modes of interaction it may support.

While such separation is, where possible, often advantageous, it is clearly not always possible and in some cases not desirable either.

4.7.1. Final-form and Re-usable Data Formats

Final-form data formats are not designed to allow modification or uses other than that intended by their designers. An example would be PDF, which is designed to support the presentation of page images on either screen or paper, and is not readily used in any other way. XSL Formatting Objects (XSL-FO) share this characteristic.

XHTML, on the other hand, can be and is put to a variety of uses including direct display (with highly flexible display semantics), processing by network-sensitive Web spiders to support search and retrieval operations, and reprocessing into a variety of derivative forms.

There are many cases where final-form is an application requirement;

Forwards compatibility
Format version M is forwards compatible with respect to format version N if M is compatible with N and M predates N.
Backwards compatibility
Format version M is backwards compatible with respect to format version N if M is compatible with N and N predates M.

> ***Good practice: Format compatibility***
>
> Format designers SHOULD define extensibility models that allow forwards compatible and backwards compatible changes.

Naturally, even if M and N are compatible but different versions of a format, agents will process instances of them differently. For instance, if format version M is forwards compatible with format version N, M processors that encounter N instances might handle unknown elements by ignoring them entirely, or by ignoring element tags but continuing to process element content. Different format specifications may require different compatibility behavior.

> ***Good practice: Compatibility behavior***
>
> Format designers SHOULD define expected behavior when agents designed to process one version of a format encounter a compatible version of the format.

In some cases, format designers require that new features be supported (i.e., not ignored). In this case, the new version of the format (N) may be backwards compatible with the earlier version (M), but M is not forwards compatible with N. The SOAP 1.2 Recommendation [SOAP12], for example, defines the `mustUnderstand` attribute in section 5.2.3.

For more information on format extensibility, refer to "Web Architecture: Extensible Languages" [EXTLANG].

4.6. Composition of Data Formats

Many modern data format specifications include mechanisms for composition. These mechanisms range from relatively shallow and limited to relatively deep and sophisticated.

Toward the shallow end of the spectrum, it is possible to embed text comments in some image formats, such as JPEG/JFIF. Although these comments are embedded in the containing data, they have little or no effect on the content of the image.

Towards the deep end, it is possible to compose XML documents with elements from a variety of namespaces. How these namespaces interact and what effect an element's namespace has on its ancestors, siblings, and descendents is not always obvious.

The Dynamic Delegation Discovery System (DDDS) ([RFC3401] and related RFCs) is used to implement lazy binding of strings to data, in order to support dynamically configured delegation systems. This system is designed to allow resolution of any type of URI, in particular URNs.

2.8.4. Non-hierarchical Administration

One area of work involves the creation of globally unique identifiers in a file-sharing system without centralized or hierarchical administration.

3. Interaction

Web agents exchange information via messages that are constructed according to a non-exclusive set of messaging protocols (e.g., HTTP, FTP, NNTP, SMTP[11], etc.). A ***message*** exchanged between Web agents is an octet sequence consisting of representation data and possibly three types of metadata:

1. ***Resource metadata***: Metadata about the resource (e.g., HTTP 'Alternates' and 'Vary' headers).
2. ***Representation metadata***: Metadata about the representation data (e.g., HTTP Content-Type field, Etags). Of particular importance is the Internet Media Type[12] (defined in RFC 2046 [RFC2046]) value of the Content-Type field, which governs the authoritative interpretation of representation data. **Note:** The terms "data format" and "media type" are often used interchangeably. The phrase "media type M" is shorthand for "the data format defined by the specification(s) paired with Internet Media Type M in the IANA registry."
3. ***Message metadata***: Metadata about the message (e.g., the HTTP Transfer-encoding header). A message may even include metadata about the message metadata (e.g., for message-integrity checks).

This section describes the architectural principles and constraints regarding interactions between agents, including such topics as network protocols and interaction styles, along with interactions between the Web as a system and the people that make use of it. This will include the role of architectural styles, such as REST and SOAP, and the impact of meta-architectures, such as Web Services and the Semantic Web.

3.1. Messages in the Travel Scenario

> ***Story***
>
> Nadia retrieves a representation from the weather site "`http://weather.example.com/oaxaca`". She then follows the link labeled "satellite image" in the representation data and retrieves a representation of another resource, a satellite photo of the Oaxaca region.

The link to the satellite image is an HTML link encoded as `<a href="http://example.com/satimage/oaxaca">satellite image</a>`. Nadia's browser

analyzes the URI and determines that its scheme is "http". The browser opens a network connection to port 80 on the server at "example.com" and sends a "GET" message as specified by the HTTP protocol, requesting a representation of the resource identified by "/satimage/oaxaca".

The server sends a response message to the browser, once again according to the HTTP protocol. The message consists of several headers and a JPEG image.

The browser reads the headers, learns from the 'Content-Type' field that the Internet Media Type of the representation data is `image/jpeg`, reads the sequence of octets that comprises the representation data, and renders the image.

Story

Nadia decides to book a vacation to Oaxaca at "booking.example.com." She completes a series of HTML forms and is ultimately asked for credit card information to purchase the airline tickets. She provides this information in another HTML form. When she presses the "Purchase" button, her browser opens another network connection to the server at "booking.example.com" and sends a message conforming to the rules for an HTTP POST request.

As described by the HTML specification, the message data consists of a set of name/value pairs corresponding to the HTML form fields. Note that this is not a safe interaction; Nadia wishes to change the state of the system by exchanging money for airline tickets.

The server reads the POST request, and after performing the booking transaction returns a message to Nadia's browser that contains a representation of the results of Nadia's request. The representation data is in HTML so that it can be saved or printed out for Nadia's records. Note that neither the data transmitted with the POST nor the data received in the response necessarily correspond to any resource named by a URI.

3.2. Authoritative Representation Metadata

Successful communication between two parties using a piece of information relies on shared understanding of the meaning of the information. Thousands of independent parties can identify and communicate about a Web resource. To give these parties the confidence that they are all talking about the same thing when they refer to "the resource identified by the following URI ..." the design choice for the Web is, in general, that the owner of a resource assigns the authoritative interpretation of representations of the resource. See the TAG finding *"Client handling of MIME headers"* for related discussion. See also TAG issue rdfURIMeaning-39.

In our travel scenario, the authority responsible for "weather.example.com" has license to create representations of this resource and assign their authoritative interpretation. Which representation(s) Nadia receives depends on a number of factors, including:

more rapidly by agents in those cases where they can be loaded into memory and used with little or no conversion.

Textual formats are often more portable and interoperable, since there are fewer choices for representation of the basic units (characters), and those choices are well-understood and widely implemented.

Textual formats also have the considerable advantage that they can be directly read and understood by human beings. This can simplify the tasks of creating and maintaining processing software, and allow the direct intervention of humans in the processing chain without recourse to tools any more complex than the ubiquitous text editor. Finally, it simplifies the necessary human task of learning about new data formats (the "View Source" effect).

It is important to emphasize that intuition as to such matters as data size and processing speed are not a reliable guide in data format design; quantitative studies are essential to a correct understanding of the trade-offs.

TAG issue binaryXML-30: Effect of Mobile on architecture - size, complexity, memory constraints. Binary Infosets, storage efficiency.

4.5. Extensibility and Versioning

Editor's note: The TAG has begun work on a finding on the topic of extensibility and versioning. The text in this section is likely to change once that finding has been published.

A format is ***extensible*** if instances of the format can include terms from other vocabularies. For example, XML and XML Namespaces allow format designers to create and combine vocabularies.

Good practice: Format extensibility

Format designers should create extensible formats.

Versioning is the term for the evolution of formats and documents. Versioning is achieved through extensibility mechanisms and format redefinition. When a format definition changes (e.g., by addition or removal of element or attribute definitions), this creates a new version of the format. When an instance of a format includes elements from another data format, this creates an extension of the instance; the original format definition has not changed. An example is a SOAP message with a header block; this is called a SOAP extension, not a new version of the SOAP format.

The following terms define important relationships among different versions of a format:

Compatible formats
: Format version M is compatible with format version N if agents designed to process all instances of M can also process all instances of N. The compatibility relationship is not always symmetric.

in the face of known error conditions.

> ***Good practice: Specify error handling***
>
> Format specification designers SHOULD specify agent behavior in the face of error conditions.

See the TAG finding *"Client handling of MIME headers"* for more discussion about error reporting. See also TAG issue errorHandling-20.

4.3. Information Hiding

When designing specifications that address independent functions of a system, normative dependencies between the specifications are, in general, harmful since such dependencies impede the independent evolution of the specifications.

For example, it is a strength of XML that XPath cannot query the HTTP header. It is a strength of HTTP that it does not refer to details of the underlying TCP to the extent that it cannot be run over a different transport service. Similarly, the RDF data graph has a significance that is independent of the actual serialization.

Sometimes it is necessary (and even good for given application) to cross specification boundaries. For example, it is good for an HTTP agent to be aware of TCP speeds and round trip times to different mirror servers in order to optimize the choice of server.

> ***Good practice: Identify features that peek***
>
> Format specification designers SHOULD clearly identify the features of a format specification that peek across specification boundaries.

4.4. Binary and Textual Data Formats

A textual data format is one in which the data is specified as a sequence of characters. HTML, Internet e-mail, and all XML-based formats are textual. In modern textual data formats, the characters are usually taken from the Unicode repertoire [UNICODE].

Binary data formats are those in which portions of the data are encoded for direct use by computer processors, for example thirty-two bit little-endian two's-complement and sixty-four bit IEEE double-precision floating-point. The portions of data so represented include numeric values, pointers, and compressed data of all sorts.

In principle, all data can be represented using textual formats.

The trade-offs between binary and textual data formats are complex and application-dependent. Binary formats can be substantially more compact, particularly for complex pointer-rich data structures. Also, they can be consumed

1. Whether the authority responsible for "weather.example.com" responds to requests at all;
2. Whether the authority responsible for "weather.example.com" makes available one or more representations for the resource identified by "`http://weather.example.com/oaxaca`";
3. Whether Nadia has access privileges to such a representation;
4. If the authority responsible for "weather.example.com" has provided more than one representation (in different formats such as HTML, PNG, or RDF, in different languages such as English and Spanish, etc.), the resulting representation may depend on negotiation between the user agent and server that occurs as part of the HTTP transaction.
5. When Nadia made the request. Since the weather in Oaxaca changes, Nadia should expect that representations will change over time.

3.2.1. Inconsistencies between Metadata and Representation Data

Inconsistencies between the format of representation data and assigned representation metadata do occur. Examples that have been observed in practice include:

- The actual character encoding of a representation is inconsistent with the charset parameter in the representation metadata.
- The namespace of the root element of the representation data is inconsistent with the value of the 'Content-Type' field in HTTP headers.

User agents should detect such inconsistencies but should not resolve them without involving the user (for example, due to security issues).

> ***Principle: Authoritative server metadata***
>
> User agents MUST NOT silently ignore authoritative server metadata.

Thus, for example, if the parties responsible for "weather.example.com" mistakenly label the satellite photo of Oaxaca as "image/gif" instead of "image/jpeg", and if Nadia's browser detects a problem, Nadia's browser must not silently ignore the problem and render the JPEG image. Nadia's browser can notify Nadia of the problem, notify Nadia and take corrective action, etc. Of course, user agent designers should not ignore usability issues when handling this type of error; notification may be discreet, and handling may be tuned to meet the user's preferences.

See the TAG finding *"Client handling of MIME headers"* for more in-depth discussion and examples.

Furthermore, server managers can help reduce the risk of error through careful assignment of representation metadata.

Principle: Don't guess metadata

Server managers MUST ensure that representation metadata is appropriate for each representation.

3.3. The Representational State Transfer (REST) Model

Good practice: Understand REST

Designers of protocols SHOULD invest time in understanding the REST model and consider the role to which of its principles could guide their design:

- statelessness
- clear assignment of roles to parties
- uniform address space
- limited, uniform set of verbs

3.4. Moved here from other sections temporarily

- Introduce notions of client and server? Relation of client to agent and user agent. Relation of server to resource owner.

4. Representations and Formats

The ***representation*** of the state of a resource is an octet sequence consisting of:

1. ***Representation data***: Electronic data expressed in one or more ***data formats*** used separately or in combination. A data format (e.g., XHTML, CSS, PNG, XLink, RDF/XML, and SMIL animation) is defined by a ***format specification***. A format specification governs the handling of fragment identifiers.
2. Representation metadata

Web agents use representations to modify as well as retrieve resource state.

The first data format used to build representations was HTML. Since then, data formats for the Web have flourished. The Web architecture does not constrain which data formats can be used to build representations. This flexibility is important because there is constant evolution in applications. This evolution results in new data formats and refinements of existing formats.

Some characteristics of a data format make it easier to integrate into the Web architecture. We examine some of those characteristics below. This document does not address generally beneficial characteristics of a specification such as readability, simplicity, attention to programmer goals, attention to user needs, accessibility, internationalization, etc. The section on architectural specifications includes references to additional format specification guidelines.

4.1. Interoperability and the Use of Standard Format Specifications

For a data format to be usefully interoperable between two parties, the parties must have a shared understanding of its syntax and semantics. This is *not* to imply that a sender of data can count on constraining its treatment by a receiver; simply that making good use of electronic data usually requires knowledge of its designers' intentions.

Good practice: Format specification availability

To promote the interoperability of a Web data format, there SHOULD be a stable, normative specification for it that is a widely available Web resource.

Good practice: Media type registration

A format specification SHOULD have an officially registered Internet Media Type (see the [MEDIATYPEREG] registry).

See TAG finding "*Internet Media Type registration, consistency of use*" for more information.

Good practice: Specified fragment identifier semantics

A format specification SHOULD define the syntax and semantics of fragment identifiers.

Although the Web architecture allows for the deployment of new data formats, the creation and deployment of new formats (and agents able to handle them) is very expensive. Thus, before inventing a new data format, designers should carefully consider re-using one that is already available. For example, if a format is required to contain human-readable text with embedded hyperlinks, it is almost certainly better to use HTML for this purpose than to invent a new format.

4.2. Error Handling

Errors occur in the deployment of software and data. The degree to which errors are tolerated varies depends on application context. User agents act on behalf of the user and therefore are expected to help the user understand the nature of errors, and possibly overcome them. User agents that correct errors without the consent of the user are not acting on the user's behalf.

Principle: Error recovery

Silent recovery from error is harmful.

To promote interoperability, specifications should set expectations about behavior