W3C®

Architecture of the World Wide Web, First Edition

Editor's Draft 8 June 2004

This version:
http://www.w3.org/2001/tag/2004/webarch-20040608/
Latest editor's draft:
http://www.w3.org/2001/tag/webarch/
Previous version:
http://www.w3.org/2001/tag/2004/webarch-20040510/
Latest version:
http://www.w3.org/TR/webarch/

Editor:
Ian Jacobs, W3C
Authors:
See acknowledgments .

Abstract

The World Wide Web is a network-spanning information space of resources interconnected by links. This information space is the basis of, and is shared by, a number of information systems. Within each of these systems, agents (people and software) retrieve, create, display, analyze, and reason about resources.

Web architecture includes the definition of the information space in terms of identification and representation of its contents, and of the protocols that support the interaction of agents in an information system making use of the space. Web architecture is influenced by social requirements and software engineering **principles** . These lead to design choices and **constraints** on the behavior of systems that use the Web in order to achieve desired properties of the shared information space: efficiency, scalability, and the potential for indefinite growth across languages, cultures, and media. **Good practice** by agents in the system is also important to the success of the

Comment [skw1]: This phrase continues to give problems. Link is the problem word here because I believe it is intended to in the sense of a reference or association or named relationship made between resources. Whereas some have taken in it in the sense of a (physical) communications link between resources.

Comment [skw2]: Identification in the sense of naming(Pat Hayes D sense)… or identification in the sense of an operational method for retrieving representations (C sense)? I read it in the sense of naming objects in the information space but I don't know whether that is strongly (D) or (C).

system. This document reflects the three bases 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 is the 8 June 2004 Editor's Draft of "Architecture of the World Wide Web, First Edition." This draft takes into account TAG decisions at the 12-14 May 2004 face-to-face meeting and the 7 June 2004 teleconference , as well as reviewer comments on the 10 May 2004 draft. Please comments about this document to the TAG mailing list public-webarch-comments@w3.org (public archive).

This document has been developed by W3C's Technical Architecture Group (TAG) (charter). A complete list of changes to this document since the first public Working Draft is available on the Web.

The TAG charter describes a process for issue resolution by the TAG. In accordance with those provisions, the TAG maintains a running issues list . The First Edition of "Architecture of the World Wide Web" does not address every issue that the TAG has accepted since it began work in January 2002. The TAG has selected a subset of issues that the First Edition does address to the satisfaction of the TAG; those issues are identified in the TAG's issues list. The TAG intends to address the remaining (and future) issues after publication of the First Edition as a Recommendation.

This document uses the concepts and terms regarding URIs as defined in draft-fielding-uri-rfc2396bis-0x, preferring them to those defined in RFC 2396. The IETF Internet Draft draft-fieldi ng-uri-rfc2396bis-0x is expected to obsolete RFC 2396 , which is the current URI standard. The TAG is tracking the evolution of draft-fielding-uri-rfc2396bis-0x.

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

List of Principles, Constraints, and Good Practice Notes

The following principles, constraints, and good practice notes are discussed in this document and listed here for convenience. There is also a free-standing summary .

General Architecture Principles

- Error recovery (principle, 1.2.3)

Identification

- Global Identifiers (principle, 2)
- Identify with URIs (practice, 2.1)
- Avoiding URI aliases (practice, 2.3.1)
- Consistent URI usage (practice, 2.3.1)
- Avoiding URI Overloading (practice, 2.4)
- New URI schemes (practice, 2.6)
- URI opacity (practice, 2.7)

1. Introduction

World Wide Web (***WWW*** , or simply ***Web***) is an information space in which the items of interest, referred to as ***resources*** , are identified by global identifiers called Uniform Resource Identifiers (***URI***).

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

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 weather information. When Nadia enters the URI into her browser:

Comment [skw3]: Up front I'd like to add the "…anything can be a resource" statement - if that is a concensus. I am minde by Pat Hayes comments "So you might want to say that the real definition of "resource" ought to be anything that can be uniquely *described* using URIs. Then just being directly identified by a URI is like the ground case of a potentially very large recursion.
"

Comment [skw4]: Well… the in line examples go outside the bounds of the travel scenario. I don't think it is used throughout.

Comment [skw5]: Nadia seems to be inferring information a bout a resource from its name!

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" provides information in a response to the retrieval request.
3. The browser displays the retrieved information, which includes hypertext links to other information. Nadia can follow these hypertext links to retrieve additional information.

This scenario illustrates the three architectural bases of the Web that are discussed in this document:

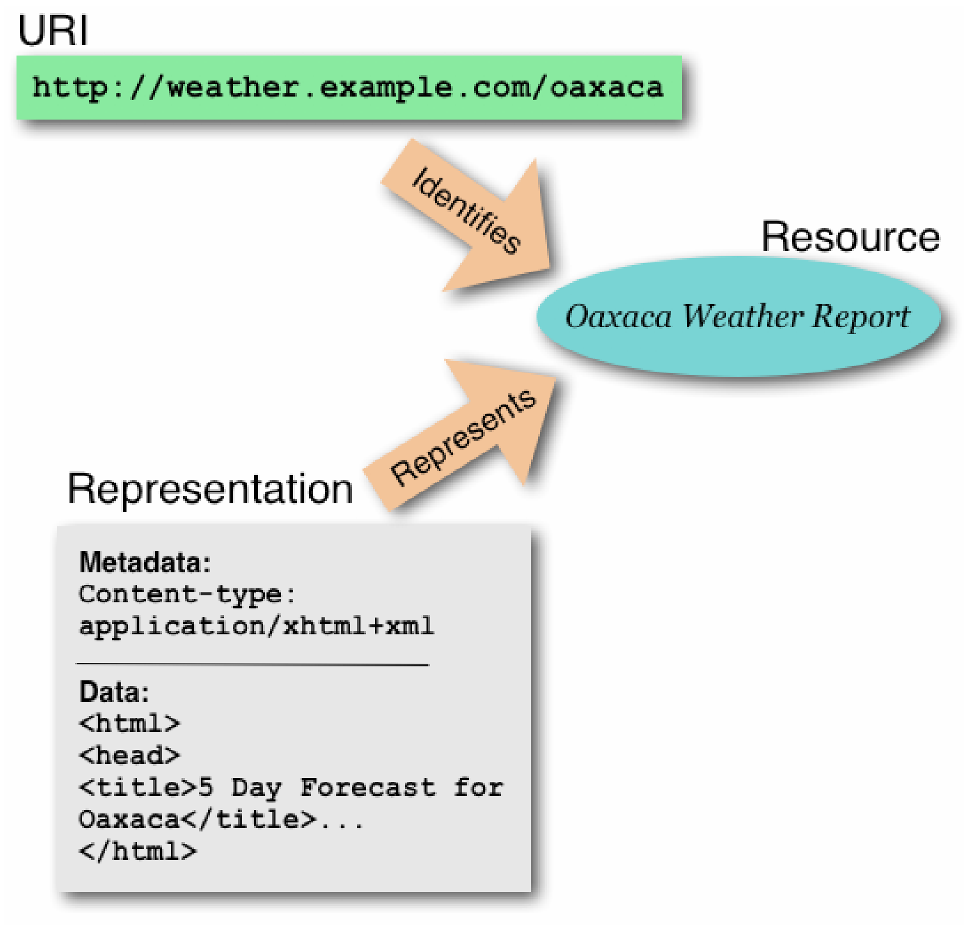
1. Identification . Each resource is identified by a URI. In this travel scenario, the resource is a periodically updated report on the weather in Oaxaca, and the URI is "http://weather.example.com/oaxaca".
2. Interaction . Protocols define the syntax and semantics of messages exchanged by agents over a network. Web agents communicate the information state of a resource through the exchange of representations . In the travel scenario, Nadia (by clicking on a hypertext link) tells her browser to request a representation of the resource identified by the URI in the hypertext link. The browser sends an HTTP GET request to the server at "weather.example.com". The server responds with a representation that includes XHTML data and the Internet media type "application/xhtml+xml".
3. Formats . Representations 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, the representation data format is XHTML. While interpreting the XHTML representation data, the browser retrieves and displays weather maps identified by URIs within the XHTML.

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

Comment [skw6]: (C) or (D) sense? I can read both.

Comment [skw7]: (C) sense

Comment [skw8]: Representations of what? Resources? Resource State? (to some fidelity)



In the remainder of this document, we highlight important architectural points regarding Web identifiers, protocols, and formats.

1.1. About this Document

This document describes the properties we desire of the Web and the design choices that have been made to achieve them. It promotes re-use of existing standards when suitable, and gives guidance on how to innovate in a manner consistent with the Web architecture.

The terms MUST, MUST NOT, SHOULD, SHOULD NOT, and MAY are used in the principles, constraints, and good practice notes in accordance with RFC 2119 [RFC2119]. However, this document does not include conformance provisions for these reasons:

- Conforming software is expected to be so diverse that it would not be useful to be able to refer to the class of conforming software agents.
- Some of the good practice notes concern people; specifications generally define conformance for software, not people.
- The addition of a conformance section is not likely to increase the utility of the document.

1.1.1. Audience of this Document

This document is intended to inform discussions about issues of Web architecture. The intended audience for this document includes:

1. Participants in W3C Activities
2. Other groups and individuals designing technologies to be integrated into the Web
3. Implementers of W3C specifications
4. Web content authors and publishers

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.

Note: This document does not distinguish in any formal way the terms "language" and "format." Context determines which term is used. The phrase "specification designer" encompasses language, format, and protocol designers.

1.1.2. Scope of this Document

This document presents the general 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 references.

This document strikes a balance between brevity and precision while including illustrative examples. TAG findings are informational documents that complement the current document by providing more detail about selected topics. This document includes some excerpts from the findings. Since the findings evolve independently, this document also includes references to approved TAG findings. For other TAG issues covered by this document but without an approved finding, references are to entries in the TAG issues list .

Comment [skw9]: Hmmm… this does invite the response… "Oh no it doesn't!" on both counts ☺

Many of the examples in this document that involve human activity suppose the familiar Web interaction model where a person follows a link via a user agent, the user agent retrieves and presents data, the user follows another link, etc. This document does not discuss in any detail other interaction models such as voice browsing (see, for example, [VOICEXML2]). For instance, when a graphical user agent running on a laptop computer or hand-held device encounters an error, the user agent can report errors directly to the user through visual and audio cues, and present the user with options for resolving the errors. On the other hand, when someone is browsing the Web through voice input and audio-only output, stopping the dialog to wait for user input may reduce usability since it is so easy to "lose one's place" when browsing with only audio-output. This document does not discuss how the principles, constraints, and good practices identified here apply in all interaction contexts.

1.1.3. Principles, Constraints, and Good Practice Notes

The important points of this document are categorized as follows:

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 limited by its slowest component."

Constraint
: 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 `paragraph` had been chosen instead of `p` or asterisk (*) instead of colon, the large-scale result would, most likely, have been the same. Other design choices are more fundamental; these are the focus of this document. Design choices can lead to constraints, i.e., restrictions in behavior or interaction within the system. Constraints may be imposed for technical, policy, or other reasons to achieve certain properties of 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.

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

1.2. General Architecture Principles

A number of general architecture principles apply to all three bases of Web architecture.

1.2.1. Orthogonal Specifications

Identification, interaction, and representation are orthogonal concepts, meaning that technologies used for identification, interaction, and representation may evolve independently. For instance:

- One identifies a resource with a URI. One may assign and then use a URI without building any representations of the resource or determining whether any representations are available.
- A generic URI syntax allows agents to function in many cases without knowing specifics of URI schemes.
- In many cases one may change the representation of a resource without disrupting references to the resource.

Comment [skw10]: (C) and (D) sense problem again. In the (C) sense identification entails a scheme definition that give an operationalised account of how dereference the resource. In the (D) not such account is necessary,although it is useful if the URI c-identifies the thing it denotes (d-identifies).

Comment [skw11]: Give an example of such a change.

When two specifications are orthogonal, one may change one without requiring changes to the other, even if one has dependencies on the other. For example, although the HTTP specification depends on the URI specification, the two may evolve independently. This orthogonality increases the flexibility and robustness of the Web. For example, one may refer by URI to an image without knowing anything about the format chosen to represent the image. This has facilitated the introduction of image formats such as PNG and SVG without disrupting existing references to image resources.

Orthogonal abstractions benefit from orthogonal specifications. Specifications should clearly indicate those features that simultaneously access information from otherwise orthogonal abstractions. For example a specification should draw attention to a feature that requires information from both the header and the body of a message.

Comment [skw12]: I don't understand what this is trying to say.

Although the HTTP, HTML, and URI specifications are orthogonal for the most part, they are not entirely so. Experience demonstrates that where they are not, problems have arisen:

- The HTML specification — a data format specification — includes a protocol extension of sorts: it specifies how a user agent sends HTML form data to a server (as a URI query string). The design works reasonably well, although there are limitations related to internationalization (see the TAG finding *" URIs, Addressability, and the use of HTTP GET and POST "*) and the query string design impinges on the server design. Software developers (for example, of [CGI] applications) might have an easier time finding the specification if it were published separately and then cited from the HTTP, URI, and HTML specifications.
- The HTML specification allows content providers to instruct HTTP servers to build response headers from `META` element instances. This is an abstraction violation; the software developer community would benefit from being able to find all HTTP headers from the HTTP specification (including any associated extension registries and specification updates per IETF process). Perhaps as a result, this feature of the HTML specification is not widely deployed. Furthermore, this design has led to confusion in user agent development. The HTML specification states that `META` in conjunction with `http-equiv` is intended for HTTP servers, but many HTML user agents interpret `http-equiv='refresh'` as a client-side instruction.
- Some content authors use the `META` / `http-equiv` approach to declare the character encoding scheme of an HTML document. By design, this is a hint that an HTTP server should emit a corresponding "Content-Type" header field. In practice, the use of the hint in servers is not widely deployed. Furthermore, many user agents use this information to override the "Content-Type" header sent by the server, violating protocol semantics.

1.2.2. Extensibility

The information in the Web and the technologies used to represent that information change over time. Extensibility describes the property of a technology that promotes both evolution and interoperability. Some examples of successful technologies designed to allow change while minimizing disruption include:

- the fact that URI schemes are orthogonally specified;
- the use of an open set of Internet media types in mail and HTTP to specify document interpretation;
- the separation of the generic XML grammar and the open set of XML namespaces for element and attribute names;
- extensibility models in Cascading Style Sheets (CSS), XSLT 1.0, and SOAP;
- user agent plug-ins.

An example of an unsuccessful extension mechanism is HTTP mandatory extensions. The community has sought mechanisms to extend HTTP, but apparently the costs of the mandatory extension proposal (notably in complexity) outweighed the benefits and thus hampered adoption.

Comment [skw13]: Informative reference to relevant informational RFC.

Below we discuss the property of "extensibility," exhibited by URIs, some data formats, and some protocols (through the incorporation of new messages).

Subset language : one language is a subset (or, "profile") of a second language if any document in the first language is also a valid document in the second language and has the same interpretation in the second language.

Extended language : If one language is a subset of another, the latter superset is called an extended language; the difference between the languages is called the extension. Clearly, extending a language is better for interoperability than creating an incompatible language.

Comment [skw14]: Do we have any tame computational linguist (eg. Henry) who would endorse these terms or offer us terms that are generally used in the computational linguistic community?

Ideally, many instances of a superset language can be safely and usefully processed as though they were in the language subset. Languages that exhibit this property are said to be "extensible." Language designers can facilitate extensibility by defining how implementations must handle unknown extensions — for example, that they be ignored (in some way) or should be considered errors.

For example, from early on in the Web, HTML agents followed the convention of ignoring unknown elements. This choice left room for innovation (i.e., non-standard elements) and encouraged the deployment of HTML. However, interoperability problems arose as well. In this type of environment, there is an inevitable tension between interoperability in the short term and the desire for extensibility. Experience shows that designs that strike the right balance between allowing change and preserving interoperability are more likely to thrive and are less likely to disrupt the Web community. Orthogonal specifications help reduce the risk of disruption.

For further discussion, see the section on versioning and extensibility . See also TAG issue xmlProfiles-29 .

1.2.3. Error Handling

Errors occur in networked information systems. An error condition can be well-specified (e.g., well-formedness errors in XML or 4xx client errors in HTTP) or arise unpredictably. ***Error correction*** means that an agent repairs an condition so that within the system, it is as though the error never occurred. One example of error correction involves data retransmission in response to a temporary hardware failure. ***Error recovery*** means that an agent does not repair an error condition but continues processing.

Comment [skw15]: I don't think that I quite take this definition as is. Error Recovery involves that use of some strategy that is explicitly aware that an error has occurred. Ie. the continued processing in some sense is a contingency for a possible error condition that has been anticipated. It is not that the error is simply ignored and the agent continues. The continuation (in a recovery situation) is dealing with the fact that the error occurred.

Agents frequently *correct* errors without user awareness, sparing users the details of complex network communications. On the other hand, it is important that agents *recover* from error in a way that is transparent to users, since the agents are acting on their behalf.

Comment [skw16]: Suggest replace "transparent" with "visible".

Principle: Error recovery

Agents that recover from error by making a choice without the user's consent are not acting on the user's behalf.

An agent is not required to interrupt the user (e.g., by popping up a confirmation box) to obtain consent. The user may indicate consent through pre-selected configuration options, modes, or selectable user interface toggles, with appropriate reporting to the user when the agent detects an error. Agent developers should not ignore usability issues when designing error recovery behavior.

To promote interoperability, specification designers should identify predictable error conditions. Experience has led to the following observations about error-handling approaches.

Comment [skw17]: ..with URI? (joke)

- Protocol designers should provide enough information about an error condition so that an agent can address the error condition. For instance, an HTTP 404 message (not found) is useful because it allows user agents to present relevant information to users, enabling them to contact the representation provider in case of problems.
- Experience with the cost of building a user agent to handle the diverse forms of ill-formed HTML content convinced the designers of the XML specification to require that agents fail upon encountering ill-formed content. Because users are unlikely to tolerate such failures, this design choice has pressured all parties into respecting XML's constraints, to the benefit of all.
- An agent that encounters unrecognized content may handle it in a number of ways, including by considering it an error; see also the section on extensibility and versioning .
- Error behavior that is appropriate for a person may not be appropriate for software. People are capable of exercising judgement in ways that software applications generally cannot. An informal error response may suffice for a person but not for a processor.

Comment [skw18]: s/message /status code/

See the TAG issues contentTypeOverride-24 and errorHandling-20 .

1.2.4. Protocol-based Interoperability

The Web follows Internet tradition in that its important interfaces are defined in terms of protocols, by specifying the syntax, semantics, and sequence of the messages interchanged. Protocols designed to be resilient in the face of widely varying environments have helped the Web scale and have facilitated communication across multiple trust boundaries. Traditional application programming interfaces (APIs) do not always take these constraints into account, nor should they be required to. One effect of protocol-based design is that the technology shared among agents often lasts longer than the agents themselves.

Comment [skw19]: sequencing constraints

Comment [skw20]: I think I'd make the point that protocols promote interoperability while API's promote application/implementation portability. Both are important. The two become entangled when message content contains scripts or behaviours that the recipient is expected to execute. To be interoperable the script writer has to assume the existence of a run-time environment and may have to probe for the existence of API features in order to be fully interoperable.

It is common for programmers working with the Web to write code that generates and parses these messages directly. It is less common, but not unusual, for end users to have direct exposure to these messages. It is often desirable to provide users with access to format and protocol details: allowing them to " view source ," whereby they may gain expertise in the workings of the underlying system.

2. Identification

In order to communicate internally, a community agrees (to a reasonable extent) on a set of terms and their meanings. One design goal for the Web, from its inception, has been to create a global community in which any party can share information with any other party. To achieve this goal, the Web makes use of a single global identification mechanism. The global scope promotes large-scale "network effects": the value of an identifier increases the more it is used (e.g., the more it is used in hypertext links).

Comment [skw21]: I'd delete this sentence entirely. It opens up big philosophical questions. "reasonable extent" is also very fuzzy.

Comment [skw22]: If this is a reference to URIs then it is a "single global system of identifiers". It is not clear to me that we have a "single global identification mechanism".

Comment [skw23]: I'd amend this to "the more it is used consistently.

Principle: Global Identifiers

Global naming leads to global network effects.

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].

2.1. Benefits of URIs

The choice of syntax for global identifiers is somewhat arbitrary; what is important is their global scope. The ***Uniform Resource Identifier*** ([URI], currently being revised) mechanism has been successfully deployed since the creation of the Web. There are substantial benefits to participating in the existing network of URIs, including linking, bookmarking, caching, and indexing by search engines. A resource should be assigned a URI if another 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.

Comment [skw24]: Avoid the 'mechanism' word.

Comment [skw25]: Is the sense of 'link to' here the same as 'refer to'? Again possibly taking 'link' in a physical rather than a referring sense.

Software developers should expect that it will prove useful to be able to share a URI across applications, even if that utility is not initially evident. The TAG finding " *URIs, Addressability, and the use of HTTP GET and POST* " discusses additional benefits and considerations of URI addressability.

Good practice: Identify with URIs

To benefit from and increase the value of the World Wide Web, agents should provide URIs as identifiers for resources.

Comment [skw26]: I feel uncomfortable with this phrasing. URI are the identifiers of the Web. A la Eng 90 it should be possible to refer to any (web) resource by URI. I don't understand how "agents provide". Somehow as association arises between an identifier and a thing. That may be through deployment or it may be denotational.

Other mechanisms for identifying resources (see the section on future directions for identifiers) may expand the Web as we know it today. However, there are substantial costs to creating a new identification mechanism that has the same properties as URIs.

Comment [skw27]: Expunge 'mechanisms'. Suggest "Other systems of identifiers…"

2.2. URI/Resource Relationships

To keep communication costs down, by design a URI identifies one resource. Since the scope of a URI is global, the resource identified by a URI does not depend on the context in which the URI appears (see also the section about indirect identification).

Comment [skw28]: This makes sense only for Pat Hayes (C) sense of identifies ie. where the URI is used for resource retrieval/access.
Personally, I am also not sold on the context-freeness of URI usage.

Just as one might wish to refer to a person by different names (by full name, first name only, sports nickname, romantic nickname, and so forth), Web architecture allows the assignment of more than one URI to a resource. URIs that identify the same resource are called ***URI aliases*** The section on URI aliases discusses some of the potential costs of creating multiple URIs for the same resource.

Comment [skw29]: I'm wondering if we can speak of an association between an identifier and a resource and avoid the tar-pit of assignment and authority?

The following sections address other questions about the relationship between URIs and resources, including:

- How much can I tell about a resource by inspection of a URI that identifies it? See in particular the sections on URI schemes and Information Resources .
- Who determines what resource a URI identifies? See the section on URI ownership .
- Can the resource identified by a URI change over time? See in particular the sections on URI persistence and representation management .
- Since more than one URI can identify the same resource, how do I know which URIs identify the same resource? See in particular the sections on URI comparison and assertions that two URIs identify the same resource .
- Are there resources that are not identified by any URI? In a system where the only resource identification mechanism is the URI, the question is only of philosophical interest (similarly, if a tree falls in the forest and nobody is around to hear it, does it make a sound?). The

Comment [skw30]: This is flippant and dismissive - I'd delete it. I can speak of my paternal grandmother without naming her - but she is identified wrt to a named object (me).

advent of other resource identification mechanisms may change the nature of this question and answer.

2.3. URI Comparisons

The most straightforward way of establishing that two parties are referring to the same resource is to compare, character-by-character, the URIs ~~they are using~~used in making the reference. Two URIs that are identical (character for character) refer to the same resource. Because Web architecture allows the assignment of more than one URI to a resource, two URIs that are not character for character identical can still refer to the same resource (i.e., they do not necessarily refer to different resources). There is generally a higher computational cost to determine that two different URIs refer to the same resource.

To reduce the risk of a false negative (i.e., an incorrect conclusion that two URIs do not refer to the same resource) or a false positive (i.e., an incorrect conclusion that two URIs do refer to the same resource), certain specifications ~~license applications to apply~~specify equivalence tests in addition to character-by-character comparison. For example, for "http" URIs, the authority component (the part after "//" and before the next "/") is defined to be case-insensitive. Thus, the "http" URI specification ~~licenses applications~~allows agents to conclude that authority components in two "http" URIs are equivalent when those strings are character-by-character equivalent or differ only by case. Agents that reach conclusions based on comparisons that are not ~~licensed~~ specified by relevant specifications take responsibility for any problems that result. Section 6 of [URI] provides more information about comparing URIs and reducing the risk of false negatives and positives.

Comment [skw31]: I find the notion of specificiations licensing… grates. I've lightly edited such that specifications specify.

See the section below on approaches other than string comparison that allow different ~~parties~~ agents to ~~assert that two URIs identify the same resource~~conclude that two URIs identify the same resource .

2.3.1. URI aliases

Although there are benefits (such as naming flexibility) to URI aliases, there are also costs. For example, the ~~assignment~~ association of more than one URI ~~for~~ with a resource undermines the network effect. URI aliases can also raise the cost or may even make it impossible for software to determine by following specifications that the URIs identify the same resource. URI producers should thus be conservative about the number of different URIs they ~~produce~~ associate with ~~for~~ the same resource.

Good practice: Avoiding URI aliases

A URI owner SHOULD NOT ~~create~~ associate arbitrarily different URIs ~~for~~ with the same resource.

Comment [skw32]: I find myself wanting to say simply "Agents SHOULD not associate arbitrarily different URI with the same resource". Resources/URI 'owners' form associations through deployment. However, other URI users may form different associations. There is a (C)/(D) difference here.

URI consumers also have a role in ensuring URI consistency. For instance, when transcribing a URI, agents should not gratuitously percent-encode characters. The term "character" refers to URI characters as defined in section 2 of [URI]; percent-encoding is discussed in section 2.1 of that specification.

Comment [skw33]: Need to check with 2396bis too.

Good practice: Consistent URI usage

If a URI has been assigned to a resource, agents SHOULD refer to the resource using the same URI, character for character.

Comment [skw34]: This is basically that agents that exchange URI should use URI they are given as given, character for character. Assignment is irrelevant to this GPN.

When a URI alias does become common currency, the URI owner should use protocol techniques such as server-side redirects to ~~connect~~ associate the two resources. The community benefits when the URI owner supports both the "official" URI and the alias.

2.4. URI Overloading

As discussed above, a URI identifies one resource. At times, agents may intentionally or unintentionally *use* a URI to identify different resources. ***URI overloading*** refers to the use of one URI to refer directly to more than one resource. Overloading often imposes a cost in communication due to the effort required to resolve ambiguities.

Comment [skw35]: (C) or (D) sense? Is it the case that a given URI both c-identifies and d-identifies the same one thing? The possibility of URI overloading acknowledged here suggests that this is already not the case.

Suppose, for example, that one organization makes use of a URI to refer to the movie "The Sting", and another organization uses the same URI to refer to a discussion forum about "The Sting." This overloading can create confusion about what the URI identifies, undermining the value of the URI. If one wanted to talk about the creation date of the resource identified by the URI, for instance, it would not be clear whether this meant "when the movie created" or "when the discussion forum about the movie was created."

Good practice: Avoiding URI Overloading

Agents SHOULD find out what resource a URI identifies before using that URI.

Comment [skw36]: How?!!! Where is an agent to get such a definitive account? And how is it to get such a thing without using the URI in question?

The section below on URI ownership examines approaches for establishing the authoritative source of information about what resource a URI identifies.

2.4.1. Indirect Identification

Listening to a news broadcast, one might hear a report on Britain that begins, "Today, 10 Downing Street announced a series of new economic measures." Generally, "10 Downing Street" identifies the official residence of Britain's Prime Minister. In this context, the news reporter is using it (as English rhetoric allows) to indirectly identify the British government. Similarly, URIs identify resources, but they can also be used in many constructs to indirectly identify arbitrary entities. Certain properties of URIs make them appealing as general-purpose identifiers. Local policy establishes what they indirectly identify.

For example, the URI "mailto:nadia@example.com" identifies an Internet mailbox (as ~~licensed~~ specified by the "mailto" URI scheme). Suppose this particular URI identifies Nadia's Internet mailbox. The organizers of a conference attended by Nadia might use "mailto:nadia@example.com" to refer indirectly to her (e.g., using the URI as a database key in their database of conference participants).

2.5. URI Ownership

To avoid URI overloading , it is important to reduce the risk that different agents will unintentionally (or intentionally) create the same URI (i.e., sequence of characters). URI scheme specifications can help reduce this risk, and commonly do so through the hierarchical delegation of authority. This approach, exemplified by the "http" and "mailto" schemes, allows the assignment of a part of URI space to one party, who may, in turn, delegate management of pieces of that space to other parties.

It is thus useful for a URI scheme to establish a unique relationship between a social entity and a URI; this is the case for the "http", "mailto", "mid", and "cid" schemes, for example. This relationship is called ***URI ownership*** . In this document, the phrase "authority responsible for domain X" indicates that the same entity owns those URIs where the authority component is domain X. This document does not address how the benefits and responsibilities of URI ownership may be delegated to other parties, such as to a server manager or to someone who has been delegated part of the URI space on a given Web server.

The approach taken for the "http" URI scheme follows the pattern whereby the Internet community delegates authority, via the IANA URI scheme registry [IANASchemes] and the DNS, over a set of URIs with a common prefix to one particular owner. One consequence of this approach is the Web's heavy reliance on the central DNS registry.

A URI owner may, upon request, provide representations of the resource identified by the URI. For example, when a URI owner uses the HTTP protocol to provide those representations, the HTTP origin server (defined in [RFC2616]) is the software agent acting on behalf of the URI owner to provide the authoritative representations for the resource identified by that URI. The owner is also responsible for accepting or rejecting requests to modify the resource identified by that URI, for example, by configuring a server to accept or reject HTTP PUT data based on Internet media type, validity constraints, or other constraints.

Recall that the Web architecture allows different URI owners to create URI aliases . This means that multiple parties may provide representations of the

Comment [skw37]: What properties? Substantiate the claim!

Comment [skw38]: How local? Loacl to what? Can there be global policies. Eg. a mailto: URI used as the object of foaf:mbox statement (inverseFunctional).

Comment [skw39]: Ownership confers rights and obligations. I think we should address URI and Resource ownership from that POV. We may then find that we have a problem with concepts of ownership.

Comment [skw40]: I think that 'create' here is a problem word. How do agents create URI. The use them. They associate them with things… but how do they 'create' them?

Comment [skw41]: I don't think any of the relevant specs. confer authority on anyone. There are RFCs about how register things with IANA, and some authority (somewhere) is conferred on IANA, presumably to make registrations. RFC2396 doesn't, the scheme specs don't… There is an element of social policy here that is orthogonal to the technology specifications.

Comment [skw42]: What is a unique relationship? 1-1, 1-N, N-1?

Comment [skw43]: What is a social entity?

Comment [skw44]: You appear to be using ownership and authority nearly synonymously. I think we need to say that authority may be vested in a specification, and organisational entity or an individual person or agent.

same resource, depending on which URI is used for interaction. A URI owner's rights extend only to the representations served for requests given that URI.

There are social expectations for responsible representation management by URI owners, discussed below. Additional social implications of URI ownership are not discussed here. However, the success or failure of these different approaches depends on the extent to which there is consensus in the Internet community on abiding by the defining specifications.

Comment [skw45]: I don't understand what this is trying to say.

2.6. URI Schemes

In the URI "http://weather.example.com/", the "http" that appears before the colon (":") names a URI scheme. Each URI scheme has a specification that explains how identifiers are assigned 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.

Examples of URIs from various schemes include:

- mailto:joe@example.org
- ftp://example.org/aDirectory/aFile
- news:comp.infosystems.www
- tel:+1-816-555-1212
- ldap://ldap.example.org/c=GB?objectClass?one
- urn:oasis:names:tc:entity:xmlns:xml:catalog

While the Web architecture allows the definition of new schemes, introducing a new scheme is costly. Many aspects of URI processing are scheme-dependent, and a significant amount of deployed software already processes URIs of well-known schemes. Introducing a new URI scheme requires the development and deployment not only of client software to handle the scheme, but also of ancillary agents such as gateways, proxies, and caches. See [RFC2718] for other considerations and costs related to URI scheme design.

Because of these costs, if a URI scheme exists that meets the needs of an application, designers should use it rather than invent one.

Good practice: New URI schemes

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

Comment [skw46]: "New URI schemes SHOULD NOT be introduced when and existing scheme provides the desired properties…." Or stated more positively "Use extisting URI schemes whenever possible."

Consider our travel scenario : should the agent 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". When a software agent dereferences such a URI, if what really happens is that HTTP GET is invoked to retrieve a representation of the resource, then an "http" URI would have sufficed.

If the motivation behind registering a new scheme is to allow a software agent to launch a particular application when retrieving a representation, such dispatching can be accomplished at lower expense via Internet media types. When designing a new data format, the appropriate mechanism to promote its deployment on the Web is the Internet media type. Media types also provide a means for building new information space applications , described below.

Note that even if an agent cannot process representation data in an unknown format, it can at least retrieve it. The data may contain enough information to allow a user or user agent to make some use of it. When an agent does not handle a new URI scheme, it cannot retrieve a representation.

2.6.1. URI Scheme Registration

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].

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 beyond URI comparison; the network effect is lost.

Note: Some URI scheme specifications (such as the "ftp" URI scheme specification) use the term "designate" where the current document uses "identify."

TAG issue siteData-36 is about expropriation of naming authority.

Comment [skw47]: Is this the right place for this comment. It seems more general to URI than to URI schemes.

2.7. URI Opacity

It is tempting to guess the nature of a resource by inspection of a URI that identifies it. However, the Web is designed so that agents communicate resource information state through representations , not identifiers. In general, one cannot determine the Internet media type of representations of a resource by inspecting a URI for that resource. For example, the ".html" at the end of "http://example.com/page.html" provides no guarantee that representations of the identified resource will be served with the Internet media type "text/html". The HTTP protocol does not constrain the Internet media type based on the path component of the URI; the URI owner is free to configure the server to return a representation using PNG or any other data format.

Resource state may evolve over time. Requiring a URI owner to publish a new URI for each change in resource state would lead to a significant number of broken links. For robustness, Web architecture promotes independence between an identifier and the identified resource.

Good practice: URI opacity

Agents making use of URIs MUST NOT attempt to infer properties of the referenced resource except as ~~licensed~~ allowed by relevant specifications.

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."

On the other hand, the URI "mailto:joe@example.com" indicates that the URI refers to a mailbox. The "mailto" URI scheme specification authorizes agents to infer that URIs of this form identify Internet mailboxes.

Comment [skw48]: I know this is picky and I keep returning to it. The language in the spec says that "mailto URI are used to designated internet mailboxes." It is not a closed defn, though it may have meant to have been. However it is not restrictive - its says that they are used to designate internet mailboxes. It does not say that they are not/may not be used to designate other things.

In some cases, relevant technical specifications license URI assignment authorities to publish assignment policies. For more information about URI opacity, see TAG issue metaDataInURI-31 .

2.8. 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 the part of the representation that conveys information about tomorrow's weather in Oaxaca. This URI includes the fragment identifier "tom" (the string after the "#").

The ***fragment identifier*** component of a URI allows indirect identification of a ***secondary resource*** by reference to a primary resource and additional identifying information. 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 defined or described by those representations. The terms "primary resource" and "secondary resource" are defined in section 3.5 of [URI].

The interpretation of fragment identifiers is discussed in the section on media types and fragment identifier semantics .

See TAG issues abstractComponentRefs-37 and DerivedResources-43 .

2.9. 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.

2.9.1. 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.

2.9.2. Assertion that Two URIs Identify the Same Resource

Emerging Semantic Web technologies, including the "Web Ontology Language (OWL)" [OWL10], define RDF properties such as `sameAs` to assert that two URIs identify the same resource or `functionalProperty` to imply it.

Comment [skw49]: (D) sense of identify.

3. Interaction

Communication between agents over a network about resources involves URIs, messages, and data. The Web's protocols (including HTTP, FTP, SOAP, NNTP, and SMTP) are based on the exchange of messages. A ***message*** may include data as well as metadata about the resource (such as the "Alternates" and "Vary" HTTP headers), the message data, and the message itself (such as the "Transfer-encoding" HTTP header). A message may even include metadata about the message metadata (for message-integrity checks, for instance). Two important classes of message are those that request a representation of an Information Resource , and those that return the result of such a request.

Story

Nadia follows a hypertext link labeled "satellite image" expecting to retrieve a satellite photo of the Oaxaca region. The link to the satellite image is an XHTML 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 configuration determines how it locates the identified information, which might be via a cache of