                            CoRE Link Format
                      draft-ietf-core-link-format-12

Abstract

   This document defines Web Linking using a link format for use by
   constrained web servers to describe hosted resources, their
   attributes and other relationships between links. Based on the HTTP
   Link Header field defined in RFC5988, the CoRE Link Format is carried
   as a payload and is assigned an Internet media type. A well-known
   URI is defined as a default entry-point for requesting the links
   hosted by a server.

Table of Contents

1. Introduction

   The Constrained RESTful Environments (CoRE) working group aims at
   realizing the Representational State Transfer (REST) architecture
   [REST] in a suitable form for the most constrained nodes (e.g. 8-bit
   microcontrollers with limited memory) and networks (e.g. 6LoWPAN
   [RFC4944]). CoRE is aimed at Machine-to-Machine (M2M) applications
   such as smart energy and building automation.

   The discovery of resources hosted by a constrained server is very
   important in machine-to-machine applications where there are no
   humans in the loop and static interfaces result in fragility. The
   discovery of resources provided by an HTTP [RFC2616] Web Server is
   typically called Web Discovery and the description of relations
   between resources is called Web Linking [RFC5988]. In the present
   document we refer to the discovery of resources hosted by a
   constrained web server, their attributes and other resource relations
   as CoRE Resource Discovery.

   The main function of such a discovery mechanism is to provide
   Universal Resource Identifiers (URIs, called links) for the resources
   hosted by the server, complemented by attributes about those
   resources and possible further link relations. In CoRE this
   collection of links is carried as a resource of its own (as opposed
   to HTTP headers delivered with a specific resource). This document
   specifies a link format for use in CoRE Resource Discovery by
   extending the HTTP Link Header format [RFC5988] to describe these
   link descriptions. The CoRE Link Format is carried as a payload and
   is assigned an Internet media type. A well-known relative URI
   "/.well-known/core" is defined as a default entry-point for
   requesting the list of links about resources hosted by a server, and
   thus performing CoRE Resource Discovery. This specification is
   applicable for use with CoAP [I-D.ietf-core-coap], HTTP or any other
   suitable web transfer protocol. The link format can also be saved in
   file format.

1.1. Web Linking in CoRE

   Technically the CoRE Link Format is a serialization of a typed link
   as specified in [RFC5988], used to describe relationships between
   resources, so-called "Web Linking". In this specification Web
   Linking is extended with specific constrained M2M attributes, links
   are carried as a message payload rather than in an HTTP Link Header
   field, and a default interface is defined to discover resources
   hosted by a server. This specification also defines a new relation
   type "hosts" (from the verb "to host"), which indicates that the
   resource is hosted by the server from which the link document was
   requested.

In HTTP, the Link Header can be used to carry link information about
a resource along with an HTTP response. This works well for the
typical use case for a web server and browser, where further
information about a particular resource is useful after accessing it.
In CoRE the main use case for Web Linking is the discovery of which
resources a server hosts in the first place. Although some resources
may have further links associated with them, this is expected to be
an exception. For that reason the CoRE Link Format serialization is
carried as a resource representation of a well-known URI. The CoRE
Link Format does re-use the format of the HTTP Link Header
serialization defined in [RFC5988].

1.2. Use Cases

Typical use cases for Web Linking on today's web include e.g.
describing the author of a web page or describing relations between
web pages (next chapter, previous chapter etc.). Web Linking can
also be applied to M2M applications, where typed links are used to
assist a machine client in finding and understanding how to use
resources on a server. In this section a few use cases are described
for how the CoRE Link Format could be used in M2M applications. For
further technical examples see Section 5. As there are a large range
of M2M applications, these use cases are purposely generic. This
document assumes that different deployments or application domains
will define the appropriate REST Interface Descriptions along with
Resource Types to make discovery meaningful.

1.2.1. Discovery

In M2M applications, for example home or building automation, there
is a need for local clients and servers to find and interact with
each other without human intervention. The CoRE Link Format can be
used by servers in such environments to enable Resource Discovery of
the resources hosted by the server.

Resource Discovery can be performed either unicast or multicast.
When a server's IP address is already known, either a priori or
resolved via the Domain Name System (DNS) [RFC1034][RFC1035], unicast
discovery is performed in order to locate the entry point to the
resource of interest. In this specification, this is performed using
a GET to "/.well-known/core" on the server, which returns a payload
in the CoRE Link Format. A client would then match the appropriate
Resource Type, Interface Description and possible Media type
[RFC2045] for its application. These attributes may also be included
in the query string in order to filter the number of links returned
in a response.

Multicast resource discovery is useful when a client needs to locate

a resource within a limited scope, and that scope supports IP multicast. A GET request to the appropriate multicast address is made for "/.well-known/core". In order to limit the number and size or responses, a query string is recommended with the known attributes. Typically a resource would be discovered based on its Resource Type and/or Interface Description, along with possible application specific attributes.

1.2.2. Resource Collections

RESTful designs of M2M interfaces often make use of collections of resources. For example an index of temperature sensors on a data collection node or a list of alarms on a home security controller. The CoRE Link Format can be used to make it possible to find the entry point to a collection and traverse its members. The entry point of a collection would always be included in "/.well-known/core" to enable its discovery. The members of the collection can be defined either through the Interface Description of the resource along with a parameter resource for the size of the collection, or by using the link format to describe each resource in the collection. These links could be located under "/.well-known/core" or hosted for example in the root resource of the collection.

1.2.3. Resource Directory

In many deployment scenarios, for example constrained networks with sleeping servers, or large M2M deployments with bandwidth limited access networks, it makes sense to deploy resource directory entities which store links to resources stored on other servers. Think of this as a limited search engine for constrained M2M resources.

The CoRE Link Format can be used by a server to register resources with a resource directory, or to allow a resource directory to poll for resources. Resource registration can be achieved by having each server POST their resources to "/.well-known/core" on the resource directory. This in turn adds links to the resource directory under an appropriate resource. These links can then be discovered by any client by making a request to a resource directory lookup interface.

1.3. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119].

This specification requires readers to be familiar with all the terms and concepts that are discussed in [RFC5988] and [RFC6454]. In addition, this specification makes use of the following terminology:

Web Linking
   A framework for indicating the relationships between web
   resources.

Link
   Also called "typed links" in RFC5988. A link is a typed
   connection between two resources identified by URIs. Made up of a
   context URI, a link relation type, a target URI, and optional
   target attributes.

Link Format
   A particular serialization of typed links.

CoRE Link Format
   A particular serialization of typed links based on the HTTP Link
   Header field serialization defined in Section 5 of RFC5988, but
   carried as a resource representation with a media type.

Attribute
   Properly called "Target Attribute" in RFC5988. A key/value pair
   that describes the link or its target.

CoRE Resource Discovery
   When a client discovers the list of resources hosted by a server,
   their attributes and other link relations by accessing "/.well-
   known/core".

2. Link Format

The CoRE Link Format extends the HTTP Link Header field specified in
[RFC5988]. The format does not require special XML or binary
parsing, is fairly compact, and is extensible - all important
characteristics for CoRE. It should be noted that this link format
is just one serialization of typed links defined in [RFC5988], others
include HTML link, Atom feed links [RFC4287] or HTTP Link Header
fields. It is expected that resources discovered in the CoRE Link
Format may also be made available in alternative formats on the
greater Internet. The CoRE Link Format is only expected to be
supported in constrained networks and M2M systems.

Section 5 of [RFC5988] did not require an Internet media type for the
defined link format, as it was defined to be carried in an HTTP
header. This specification thus defines the Internet media type
"application/link-format" for the CoRE Link Format (see Section 7.3).
Whereas the HTTP Link Header field depends on [RFC2616] for its
encoding, the CoRE Link Format is encoded as UTF-8 [RFC3629]. A
decoder of the format is not expected to (but not prohibited from)

validate UTF-8 encoding and doesn't need to perform any UTF-8
normalization. UTF-8 data can be compared bit-wise, which allows
values to contain UTF-8 data without any added complexity for
constrained nodes.

The CoRE link format is equivalent to the [RFC5988] link format,
however the ABNF in the present document is repeated with
improvements to be compliant with [RFC5234] and includes new link
parameters. As in [RFC5988], multiple link descriptions are
separated by commas. Note that commas can also occur in quoted
strings and URIs but do not end a description. In order to convert
an HTTP Link Header field to this link format, first the "Link:" HTTP
header is removed, any LWS is removed, the header value is converted
to UTF-8 and any percent-encodings decoded.

```
Link             = link-value-list
link-value-list = [ link-value *[ "," link-value ]]
link-value      = "<" URI-Reference ">" *( ";" link-param )
link-param      = ( ( "rel" "=" relation-types )
                / ( "anchor" "=" <"> URI-Reference <"> )
                / ( "rev" "=" relation-types )
                / ( "hreflang" "=" Language-Tag )
                / ( "media" "=" ( MediaDesc / ( <"> MediaDesc <"> ) ) )
                / ( "title" "=" quoted-string )
                / ( "title*" "=" ext-value )
                / ( "type" "=" ( media-type / quoted-mt ) )
                / ( "rt" "=" relation-types )
                / ( "if" "=" relation-types )
                / ( "sz" "=" cardinal )
                / ( link-extension ) )
link-extension = ( parmname [ "=" ( ptoken / quoted-string ) ] )
                / ( ext-name-star "=" ext-value )
ext-name-star  = parmname "*" ; reserved for RFC2231-profiled
                               ; extensions.  Whitespace NOT
                               ; allowed in between.
ptoken          = 1*ptokenchar
ptokenchar      = "!" / "#" / "$" / "%" / "&" / "'" / "("
                / ")" / "*" / "+" / "-" / "." / "/" / DIGIT
                / ":" / "<" / "=" / ">" / "?" / "@" / ALPHA
                / "[" / "]" / "^" / "_" / "`" / "{" / "|"
                / "}" / "~"
media-type      = type-name "/" subtype-name
quoted-mt       = <"> media-type <">
relation-types = relation-type
                / <"> relation-type *( 1*SP relation-type ) <">
relation-type  = reg-rel-type / ext-rel-type
reg-rel-type   = LOALPHA *( LOALPHA / DIGIT / "." / "-" )
ext-rel-type   = URI
cardinal        = "0" / ( %x31-39 *DIGIT )
LOALPHA         = <defined in RFC2616>
quoted-string   = <defined in RFC2616>
URI             = <defined in RFC3986>
URI-Reference   = <defined in RFC3986>
type-name       = <defined in RFC4288>
subtype-name    = <defined in RFC4288>
MediaDesc       = <defined in W3C.REC-html401-19991224>
Language-Tag    = <defined in RFC5646>
ext-value       = <defined in RFC5987>
parmname        = <defined in RFC5987>
```

2.1. Target and context URIs

Each link conveys one target URI as a URI-reference inside angle
brackets ("<>"). The context URI of a link (also called base URI in
[RFC3986]) is determined by the following rules in this
specification:

(a) The context URI is set to the anchor parameter, when specified,
     or

(b) Origin of the target URI, when specified

(c) Origin of the link format document's base URI.

2.2. Link relations

Since links in the CoRE Link Format are typically used to describe
resources hosted by a server, and thus in the absence of the relation
parameter the new relation type "hosts" is assumed (see Section 7.2).
The "hosts" relation type (from the verb "to host") indicates that
the target URI is a resource hosted by the server (i.e. server hosts
resource) indicated by the context URI. The target URI MUST be a
relative URI of the context URI for this relation type.

To express other relations, links can make use of any registered
relation by including the relation parameter. The context of a
relation can be defined using the anchor parameter. In this way,
relations between resources hosted on a server, or between hosted
resources and external resources can be expressed.

2.3. Use of anchors

As per Section 5.2 of [RFC5988] a link description MAY include an
"anchor" attribute, in which case the context is the URI included in
that attribute. This is used to describe a relationship between two
resources. A consuming implementation can however choose to ignore
such links. It is not expected that all implementations will be able
to derive useful information from explicitly anchored links.

3. CoRE link attributes

The following CoRE specific target attributes are defined in addition
to those already defined in [RFC5988]. These attributes describe
information useful in accessing the target link of the relation, and
in some cases can use the syntactical form of a URI. Such a URI MAY
be dereferenced (for instance to obtain a description of the link
relation), but that this is not part of the protocol and MUST NOT be

done automatically on link evaluation. When attributes values are
compared, they MUST be compared as strings.

3.1. Resource type 'rt' attribute

The resource type "rt" attribute is an opaque string used to assign
an application specific semantic type to a resource. One can think
of this as a noun describing the resource. In the case of a
temperature resource this could be e.g. an application-specific
semantic type like "OutdoorTemperature" or a URI referencing a
specific concept in an ontology like
"http://sweet.jpl.nasa.gov/2.0/phys.owl#Temperature". Multiple
resource types MAY be included in the value of this parameter, each
separated by a space, similar to the relation attribute. The
registry for Resource Type values is defined in Section 7.4.

The resource type attribute is not meant to used to assign a human
readable name to a resource. The "title" attribute defined in
[RFC5988] is meant for that purpose. The resource type attribute
MUST NOT appear more than once in a link.

3.2. Interface description 'if' attribute

The Interface Description "if" attribute is an opaque string used to
provide a name or URI indicating a specific interface definition used
to interact with the target resource. One can think of this as
describing verbs usable on a resource. The Interface Description
attribute is meant to describe the generic REST interface to interact
with a resource or a set of resources. It is expected that an
Interface Description will be re-used by different resource types.
For example the resource types "OutdoorTemperature", "DewPoint" and
"RelHumidity" could all be accessible using the interface description
"http://www.example.org/myapp.wadl#sensor". Multiple interface
descriptions MAY be included in the value of this parameter, each
separated by a space, similar to the relation attribute. The
registry for Interface Description values is defined in Section 7.4.

The Interface Description could be for example the URI of a Web
Application Description Language (WADL) [WADL] definition of the
target resource "http://www.example.org/myapp.wadl#sensor", a URN
indicating the type of interface to the resource "urn:myapp:sensor",
or an application-specific name "Sensor". The Interface Description
attribute MUST NOT appear more than once in a link.

3.3. Maximum size estimate 'sz' attribute

The maximum size estimate attribute "sz" gives an indication of the
maximum size of the resource representation returned by performing a

GET on the target URI. For links to CoAP resources this attribute is
not expected to be included for small resources that can comfortably
be carried in a single Maximum Transmission Unit (MTU), but SHOULD be
included for resources larger than that. The maximum size estimate
attribute MUST NOT appear more than once in a link.

Note that there is no defined upper limit to the value of the sz
attributes. Implementations MUST be prepared to accept large values.
One implementation strategy is to convert any value larger than a
reasonable size limit for this implementation to a special value
"Big", which in further processing would indicate that a size value
was given that was so big that it cannot be processed by this
implementation.

4. Well-known Interface

Resource discovery in CoRE is accomplished through the use of a well-
known resource URI which returns a list of links about resources
hosted by that server and other link relations. Well-known resources
have a path component that begins with "/.well-known/" as specified
in [RFC5785]. This document defines a new well-known resource for
CoRE Resource Discovery "/.well-known/core".

A server implementing this specification MUST support this resource
on the default port appropriate for the protocol for the purpose of
resource discovery. It is however up to the application which links
are included and how they are organized. The resource "/.well-known/
core" is meant to be used to return links to the entry points of
resource interfaces on a server. More sophisticated link
organization can be achieved by including links to CoRE Link Format
resources located elsewhere on the server, for example to achieve an
index. In the absence of any links, a zero-length payload is
returned. The resource representation of this resource MUST be the
CoRE Link Format described in Section 2.

The CoRE resource discovery interface supports the following
interactions:

o Performing a GET on "/.well-known/core" to the default port
   returns a set of links available from the server (if any) in the
   CoRE Link Format. These links might describe resources hosted on
   that server, on other servers, or express other kinds of link
   relations as described in Section 2.

o Filtering may be performed on any of the link format attributes
   using a query string as specified in Section 4.1. For example
   [GET /.well-known/core?rt=TemperatureC] would request resources

with the resource type TemperatureC. A server is not however
required to support filtering.

o More capable servers such as proxies could support a resource
   directory by requesting the resource descriptions of other end-
   points or allowing servers to POST requests to "/.well-known/
   core". The details of such resource directory functionality is
   however out of scope for this document, and is expected to be
   specified separately.

4.1. Query Filtering

A server implementing this document MAY recognize the query part of a
resource discovery URI as a filter on the resources to be returned.
The query part should conform to the following syntax. Note that
this only defines querying for a single parameter at a time.

```
filter-query   = resource-param "=" query-pattern
resource-param = "href" / parmname
query-pattern  = search-token [ "*" ]
search-token   = *search-char
search-char    = unreserved / pct-encoded
                 / ":" / "@"   ; from pchar
                 / "/" / "?"   ; from query
                 / "!" / "$" / "'" / "(" / ")"
                 / "+" / "," / ";" / "="  ; from sub-delims
parmname       = <defined in RFC5987>
pct-encoding   = <defined in RFC3986>
unreserved     = <defined in RFC3986>
```

The resource-param "href" refers to the URI-reference between the "<"
and ">" characters of a link. Other resource-param values refer to
the link attribute they name. Filtering is performed by comparing
the normalized query-pattern (decode percent-encoding and convert to
UTF8) against the value of the attribute identified by the resource-
param for each link-value in the collection of resources identified
by the URI path.

If the decoded query-pattern does not end with "*", a link value
matches the query only if the value of the attribute or URI-reference
denoted by the resource-param is byte-wise identical to the
normalized query-pattern. If the decoded query-pattern ends with
"*", it is sufficient that the remainder of the query-pattern be a
prefix of the value denoted by the resource-param. A query-pattern
of "*" matches to an empty string value as well as to any other non-
empty string. It is not expected that very constrained nodes support

filtering. Implementations not supporting filtering MUST simply
ignore the query string and return the whole resource for unicast
requests.

When using a transfer protocol like the Constrained Application
Protocol (CoAP) that supports multicast requests, special care needs
to be taken. A multicast request with a query string SHOULD NOT be
responded to if filtering is not supported or if the filter does not
match (to avoid a needless response storm). The exception is in
cases where the IP stack interface is not able to indicate that the
destination address was multicast.

5. Examples

A few examples of typical link descriptions in this format follows.
Multiple resource descriptions in a representation are separated by
commas. Linefeeds are also included in these examples for
readability. Although the following examples use CoAP response
codes, the examples are applicable to HTTP as well (the corresponding
response code would be 200 OK).

This example includes links to two different sensors sharing the same
Interface Description.

REQ: GET /.well-known/core

RES: 2.05 Content
</sensors/temp>;if="sensor",
</sensors/light>;if="sensor"

Without the linefeeds inserted here for readability, the format
actually looks as follows.

</sensors/temp>;if="sensor",</sensors/light>;if="sensor"

This example arranges link descriptions hierarchically, with the
entry point including a link to a sub-resource containing links about
the sensors.

```
REQ: GET /.well-known/core

RES: 2.05 Content
</sensors>;ct=40

REQ: GET /sensors

RES: 2.05 "Content"
</sensors/temp>;rt="TemperatureC";if="sensor",
</sensors/light>;rt="LightLux";if="sensor"
```

An example query filter may look like:

```
REQ: GET /.well-known/core?rt=LightLux

RES: 2.05 "Content"
</sensors/light>;rt="LightLux";if="sensor"
```

This example shows the use of an anchor attribute to relate the
temperature sensor resource to an external description and to an
alternative URI.

```
REQ: GET /.well-known/core

RES: 2.05 "Content"
</sensors>;ct=40;title="Sensor Index",
</sensors/temp>;rt="TemperatureC";if="sensor",
</sensors/light>;rt="LightLux";if="sensor",
<http://www.example.com/sensors/t123>;anchor="/sensors/temp"
;rel="describedby",
</t>;anchor="/sensors/temp";rel="alternate"
```

If a client is interested to find relations about a particular
resource, it can perform a query on the anchor parameter:

```
REQ: GET /.well-known/core?anchor=/sensors/temp

RES: 2.05 "Content"
<http://www.example.com/sensors/temp123>;anchor="/sensors/temp"
;rel="describedby",
</t>;anchor="/sensors/temp";rel="alternate"
```

The following example shows a large firmware resource with a size
attribute. The consumer of this link would use the sz attribute to
determine if the resource representation is too large and if block

transfer would be required to request it. In this case a client with
only a 64 KiB flash might only support a 16-bit integer for storing
the sz attribute. Thus a special flag or value should be used to
indicate "Big" (larger than 64 KiB).

REQ: GET /.well-known/core?rt=firmware

RES: 2.05 "Content"
</firmware/v2.1>;rt="firmware";sz=262144

6. Security Considerations

This document has the same security considerations as described in
Section 7 of [RFC5988]. The "/.well-known/core" resource MAY be
protected e.g. using DTLS when hosted on a CoAP server as per
[I-D.ietf-core-coap] Section 10.2.

Some servers might provide resource discovery services to a mix of
clients that are trusted to different levels. For example, a
lighting control system might allow any client to read state
variables, but only certain clients to write state (turn lights on or
off). Servers that have authentication and authorization features
SHOULD support authentication features of the underlying transport
protocols (HTTP or DTLS/TLS) and allow servers to return different
lists of links based on a client's identity and authorization. While
such servers might not return all links to all requesters, not
providing the link does not, by itsef, control access to the relevant
resource - a bad actor could know or guess the right URIs. Servers
can also lie about the resources available. If it is important for a
client to only get information from a known source, then that source
needs to be authenticated.

Multicast requests using CoAP for the well-known link-format
resources could be used to perform denial of service on a constrained
network. A multicast request SHOULD only be accepted if the request
is sufficiently authenticated and secured using e.g. IPsec or an
appropriate object security mechanism.

CoRE link format parsers should be aware that a link description may
be cyclical, i.e., contain a link to itself. These cyclical links
could be direct or indirect (i.e., through referenced link
resources). Care should be taken when parsing link descriptions and
accessing cyclical links.

7. IANA Considerations

7.1. Well-known 'core' URI

This memo registers the "core" well-known URI in the Well-Known URI
Registry as defined by [RFC5785].

URI suffix: core

Change controller: IETF

Specification document(s): [[this document]]

Related information: None

7.2. New 'hosts' relation type

This memo registers the new "hosts" Web Linking relation type as per
[RFC5988].

Relation Name: hosts

Description: Refers to a resource hosted by the server indicated by
the link context.

Reference: [[this document]]

Notes: This relation is used in CoRE where links are retrieved as a
"/.well-known/core" resource representation.

Application Data: None

7.3. New link-format Internet media type

This memo registers the a new Internet media type for the CoRE link
format, application/link-format.

Type name: application

Subtype name: link-format

Required parameters: None

Optional parameters: None

Encoding considerations: Binary data (UTF-8)

Security considerations:

   Multicast requests using CoAP for the well-known link-format
   resources could be used to perform denial of service on a constrained
   network. A multicast request SHOULD only be accepted if the request
   is sufficiently authenticated and secured using e.g. IPsec or an
   appropriate object security mechanism.

   CoRE link format parsers should be aware that a link description may
   be cyclical, i.e., contain a link to itself. These cyclical links
   could be direct or indirect (i.e., through referenced link
   resources). Care should be taken when parsing link descriptions and
   accessing cyclical links.

   Interoperability considerations:

   Published specification: [[this document]]

   Applications that use this media type: CoAP server and client
   implementations for resource discovery and HTTP applications that use
   the link-format as a payload.

   Additional information:

   Magic number(s):

   File extension(s): *.wlnk

   Macintosh file type code(s):

   Intended usage: COMMON

   Restrictions on usage: None

   Author: CoRE WG

   Change controller: IETF

7.4. Registry for Resource Type and Interface Description Values

   This specification establishes two new registries, one for Resource
   Type (rt=) and the other for Interface Description (if=) link target
   attribute values. This registry is similar to the Link Relation
   Registry defined in [RFC5988]. No initial entries are defined by
   this specification for either registry.

   These registries have the following requirements on values:

   o Registration values MUST be related to the intended purpose of
      these attributes as described in Section 3.

o Registered values MUST conform to the ABNF reg-rel-type definition
   of Section 2, meaning the value MUST start with a lower case
   alphabet character, followed by a sequence of lower case alphabet,
   numeric, "." or "-" characters. The value MUST NOT contain white
   space.

o It is recommended that the period "." character is used for
   dividing name segments, and that the dash "-" character is used
   for making a segment more readable. Example Interface Description
   values might be "core.batch" and "core.link-batch".

o URIs are reserved for free use as extension values for these
   attributes, and MUST NOT be registered.

Values starting with the characters "core" are reserved, and can only
be requested for registration when defined in an IETF working group
document.

Relation types are registered on the advice of a Designated Expert
(appointed by the IESG or their delegate), with a Specification
Required (using terminology from [RFC5226]).

Registration requests consist of the completed registration template
below, typically published in an RFC or Open Standard (in the sense
described by [RFC2026], Section 7). However, to allow for the
allocation of values prior to publication, the Designated Expert may
approve registration once they are satisfied that a specification
will be published.

Note that relation types can be registered by third parties, if the
Designated Expert determines that an unregistered relation type is
widely deployed and not likely to be registered in a timely manner.

The registration template for both registries is:

o Attribute Value:

o Description:

o Reference:

o Notes: [optional]

Registration requests should be sent to the (TBD)@ietf.org mailing
list, marked clearly in the subject line (e.g., "NEW RESOURCE TYPE -
example" to register an "example" relation type, or "NEW INTERFACE
DESCRIPTION - example" to register an "example" interface
description).

Within at most 14 days of the request, the Designated Expert(s) will
either approve or deny the registration request, communicating this
decision to the review list and IANA. Denials should include an
explanation and, if applicable, suggestions as to how to make the
request successful.

Decisions (or lack thereof) made by the Designated Expert can be
first appealed to Application Area Directors (contactable using
app-ads@tools.ietf.org email address or directly by looking up their
email addresses on http://www.iesg.org/ website) and, if the
appellant is not satisfied with the response, to the full IESG (using
the iesg@iesg.org mailing list).

IANA should only accept registry updates from the Designated
Expert(s), and should direct all requests for registration to the
review mailing list.

8. Acknowledgments

Special thanks to Peter Bigot, who has made a considerable number
reviews and text contributions that greatly improved the document.
In particular, Peter is responsible for improving the ABNF
descriptions and the idea for a new "hosts" relation type.

Thanks to Mark Nottingham and Eran Hammer-Lahav for the discussions
and ideas that led to this draft, and to Carsten Bormann, Martin
Thomson, Alexey Melnikov, Julian Reschke, Joel Halpern, Richard
Barnes and Peter Saint-Andre for extensive comments and contributions
that improved the text.

Thanks to Michael Stuber, Richard Kelsey, Cullen Jennings, Guido
Moritz, Peter Van Der Stok, Adriano Pezzuto, Lisa Dussealt, Alexey
Melnikov, Gilbert Clark, Salvatore Loreto, Petri Mutka, Szymon Sasin,
Robert Quattlebaum, Robert Cragie, Angelo Castellani, Tom Herbst, Ed
Beroset, Gilman Tolle, Robby Simpson, Colin O'Flynn and David Ryan
for helpful comments and discussions that have shaped the document.

9. Changelog

Changes from ietf-11 to ietf-12:

   o Changed "uri" to "href" in the filter query (#200)

   o Upgraded all ABNF to RFC5234 (#197)

o Put multiple rt= and if= values in a single attribute (as in rel=) (#199)

o Use the Origin definition (#191)

o Clarified URI fetching rules (#196)

o Added access control and other security consideration improvements (#189)

o Fixed normalization for query pattern matching (#192)

o Added an anchor restriction for hosts (#193)

o New rules for determining link context (#194)

o Described how to convert from HTTP Link Header (#190)

o Created a registry for rt= and if= values (#195)

o Integration of all other IETF LC and IESG comments.

Changes from ietf-10 to ietf-11:

o Fixed editorial nits.

Changes from ietf-09 to ietf-10:

o Changed to SHOULD NOT for multiple relation types (#178).

o Changed to SHOULD NOT for multicast response repression (#179).

o Updated ABNF for queries (#179).

o Editorial improvements from WGLC comments.

Changes from ietf-08 to ietf-09:

o Corrected ABNF and editorial nits.

o Elided empty responses to multicast request.

Changes from ietf-07 to ietf-08:

o IESG submission nits.

Changes from ietf-06 to ietf-07:

o Moved the Content-type attribute (ct=) to the base CoAP
specification.

Changes from ietf-05 to ietf-06:

o Added improved text about the encoding of the format as UTF-8,
but treating it as binary data without normalization.

Changes from ietf-04 to ietf-05:

o Removed mention of UTF-8 as this is already defined by RFC5988
(#158)

o Changed encoding considerations to "Binary data" (#157)

o Updated ABNF to disallow leading zeros in integers (#159)

o Updated examples and reference for coap-06 (#152)

o Removed the application/link-format CoAP code registration, now
included in the CoAP specification directly (#160)

Changes from ietf-03 to ietf-04:

o Removed the attribute registry (#145).

o Requested a CoAP media type for application/link-format (#144).

o Editorial and reference improvements from AD review (#146).

o Added a range limitation for ct attribute.

o Added security considerations and file extension for
application/link-format registration.

Changes from ietf-02 to ietf-03:

o Removed 'obs' attribute definition, now defined in the CoAP
Observation spec (#99).

o Changed Resource name (n=) to Resource type (rt=) and d= to if=
(#121).

o Hierarchical organization of links under /.well-known/core
removed (#95).

o Bug in Section 3.1 on byte-wise query matching fixed (#91).

o Explanatory text added about alternative Web link formats (#92).

o Fixed a bug in Section 2.2.4 (#93).

o Added use case examples (#89).

o Clarified how the CoRE link format is used and how it differs from RFC5988 (#90, #98).

o Changed the Interface definition format to quoted-string to match the resource type.

o Added an IANA registry for CoRE Link Format attributes (#100).

Changes from ietf-01 to ietf-02:

o Added references to RFC5988 (#41).

o Removed sh and id link-extensions (#42).

o Defined the use of UTF-8 (#84).

o Changed query filter definition for any parameter (#70).

o Added more example, now as a separate section (#43).

o Mentioned cyclical links in the security section (#57).

o Removed the sh and id attributes, added obs and sz attributes (#42).

o Improved the context and relation description wrt RFC5988 and requested a new "hosts" default relation type (#85).

Changes from ietf-00 to ietf-01:

o Editorial changes to correct references.

o Formal definition for filter query string.

o Removed URI-reference option from "n" and "id".

o Added security text about multicast requests.

Changes from shelby-00 to ietf-00:

o Fixed the ABNF link-extension definitions (quotes around URIs, integer definition).

   o Clarified that filtering is optional, and the query string is to
   be ignored if not supported (and the URI path processed as
   normally).

   o Required support of wildcard * processing if filtering is
   supported.

   o Removed the assumption of a default content-type.

10. References

10.1. Normative References

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
              Requirement Levels", BCP 14, RFC 2119, March 1997.

   [RFC3629] Yergeau, F., "UTF-8, a transformation format of ISO
              10646", STD 63, RFC 3629, November 2003.

   [RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform
              Resource Identifier (URI): Generic Syntax", STD 66,
              RFC 3986, January 2005.

   [RFC4288] Freed, N. and J. Klensin, "Media Type Specifications and
              Registration Procedures", BCP 13, RFC 4288, December 2005.

   [RFC5234] Crocker, D. and P. Overell, "Augmented BNF for Syntax
              Specifications: ABNF", STD 68, RFC 5234, January 2008.

   [RFC5646] Phillips, A. and M. Davis, "Tags for Identifying
              Languages", BCP 47, RFC 5646, September 2009.

   [RFC5987] Reschke, J., "Character Set and Language Encoding for
              Hypertext Transfer Protocol (HTTP) Header Field
              Parameters", RFC 5987, August 2010.

   [RFC5988] Nottingham, M., "Web Linking", RFC 5988, October 2010.

10.2. Informative References

   [I-D.ietf-core-coap]
              Shelby, Z., Hartke, K., Bormann, C., and B. Frank,
              "Constrained Application Protocol (CoAP)",
              draft-ietf-core-coap-09 (work in progress), March 2012.

   [REST]   Fielding, R., "Architectural Styles and the Design of
              Network-based Software Architectures", 2000, <http://

          www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.

   [RFC1034] Mockapetris, P., "Domain names - concepts and facilities",
              STD 13, RFC 1034, November 1987.

   [RFC1035] Mockapetris, P., "Domain names - implementation and
              specification", STD 13, RFC 1035, November 1987.

   [RFC2045] Freed, N. and N. Borenstein, "Multipurpose Internet Mail
              Extensions (MIME) Part One: Format of Internet Message
              Bodies", RFC 2045, November 1996.

   [RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H.,
              Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext
              Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

   [RFC4287] Nottingham, M., Ed. and R. Sayre, Ed., "The Atom
              Syndication Format", RFC 4287, December 2005.

   [RFC4944] Montenegro, G., Kushalnagar, N., Hui, J., and D. Culler,
              "Transmission of IPv6 Packets over IEEE 802.15.4
              Networks", RFC 4944, September 2007.

   [RFC5785] Nottingham, M. and E. Hammer-Lahav, "Defining Well-Known
              Uniform Resource Identifiers (URIs)", RFC 5785,
              April 2010.

   [RFC6454] Barth, A., "The Web Origin Concept", RFC 6454,
              December 2011.

   [WADL]   Hadley, M., "Web Application Description Language (WADL)",
              2009, <http://java.net/projects/wadl/sources/svn/content/
              trunk/www/wadl20090202.pdf>.

Author's Address

   Zach Shelby
   Sensinode
   Kidekuja 2
   Vuokatti 88600
   FINLAND

   Phone: +358407796297
   Email: zach@sensinode.com