Internet Engineering Task Force
Internet-Draft
Intended status: Informational
Expires: September 10, 2021

L. Svensson
R. Verborgh
Ghent University - imec
H. Van de Sompel
Data Archiving and Networked Services
March 9, 2021

Indicating, Discovering, Negotiating, and Writing Profiled Representations
draft-svensson-profiled-representations-01

Abstract

This document details approaches for enriching HTTP interactions with information pertaining to the profiles to which resource representations conform. It surveys approaches that were recently introduced to indicate the profile of a resource representation, and to make representations that conform to a profile discoverable. It introduces a generally applicable approach to negotiate for a resource representation that conforms to a profile preferred by a user agent. That approach leverages the existing content negotiation mechanism but applies it to the profile dimension to which it was previously not applied. The document also shows how a server can convey which profiled representations it is able to accept from a user agent.

Table of Contents

1. Introduction

Any given web resource can typically be represented in a variety of ways. For example, the same information could be rendered according to different media types, say, as XML or JSON. But in many cases, variations in representation other than those inherent to a given media type are also possible. For example, the same structured data could be rendered in XML according to different XML Schema [W3C.REC-xmlschema11-1-20120405]. Or the same RDF graph could be expressed on the basis of different vocabularies.

This dimension of variability regarding representations that goes beyond media types has been acknowledged for quite some time. For example, in 2000, the Dublin Core Metadata Initiative (DCMI) introduced "Application Profiles" to explicitly acknowledge that the same metadata can be represented in various ways and defined them as "schemas which consist of data elements drawn from one or more namespaces, combined together by implementers, and optimised for a particular local application" [HeeryAndPatel].

1.1. Terminology

Inspired by the term used in [RFC6906] to refer to this extra dimension of variability, this document uses the term "profile" to mean a description of structural and/or semantic constraints on representations of resources that apply in addition to the constraints inherently indicated by their MIME type:

- Profiles can be dependent on a media type. For example, this is the case for XML, with constraints being expressed using an XML Schema. This is also the case for JSON that offers a wide range of options regarding the use of tree structure, keys, and value types. In these cases, the meaning of the term "profile" intended by this document coincides with the use of the same term in [RFC6906].

- Profiles can be independent of media type. For example, this is the case for RDF graphs that can be rendered according to various media types, while constraints can be expressed in a manner that is independent of media type, among others, using SHACL [W3C.REC-shacl-20170720]. In these cases, the meaning of the term "profile" intended by this document is a slight extension of the one intended by [RFC6906].

1.2. Purpose

When it comes to HTTP interactions, profiles have received little attention despite their de facto existence and the added-value they can bring for building rich applications. Such applications benefit from knowledge regarding the nature of a representation that a client obtains from a server, that a client sends to a server, and that a server is willing to accept from a client, beyond what is conveyed by the representation's MIME type. These applications are also be helped by an ability to discover representations rendered according to a profile they can handle, or, optimally, an ability to explicitly request a rendering according to a preferred profile.

A common approach to handle profiles is to register them as a media type, dedicated to the combination of an actual media type and a

profile of it. Media types that illustrate this approach include "application/activity+json", "application/calendar+json", and "application/calendar+xml. This approach allows conveying all necessary profile information in HTTP interactions, e.g. using the "Accept" and "Content-Encoding" HTTP headers and the "type" attribute for web links. As such it supports indicating, discovering, and content negotiating (in the media type dimension) for profiled representations. This registration-based approach may be feasible for profiles that are expected to be very widely used but is not practical in case support for many different profiles is required. Also, the "calendar" examples illustrate that the registration-based approach is not ideal when a profile applies to multiple media types. And, the "activity" example illustrates that the approach supports indicating what the major ingredient of a profiled representation is (i.e. the ActivityStreams Vocabulary) but becomes problematic when indications are also needed regarding additional vocabularies used in representations.

Another approach to handle profiles leverages the ability provided by [RFC6838] to register parameters when registering a media type. Some media types have used this capability to register an attribute dedicated to conveying profiles of the media type. For example, for "application/ld+json" the "profile" parameter has been registered for this purpose. The approach provides flexibility for handling many profiles, including ones that are not yet known when registering the media type. It also supports indicating, discovering, and content negotiating (in the media type dimension) for profiled representations using common approaches. But the approach remains problematic because it ties profile information to a media type, depends on registering a parameter to convey profile information when registering a new media type, and, realistically, on the registration of the same parameter name (i.e. "profile" as suggested in [RFC6906]) for all media types for which registrants deem that conveying profile information is important. Additionaly, [RFC6838] discourages registering parameters for previously registered media types, making it highly questionable that a uniform attribute to convey profile information across all media types could retroactively be defined.

Recognizing the importance of profiles and the problems with the aforementioned approaches to handle them, specifications have started to introduce alternative approaches to express information about resource representation profiles in HTTP interactions:

- [RFC6906] introduces the "profile" link relation type that is generally applicable for indicating the profile of a resource representation that is sent by a client to a server or by a server to a client.

- [I-D.nottingham-link-hint] introduces a capability to make profiled representations discoverable via web links by using the "formats" attribute to express the profile of a linked resource.

Section 2 and Section 3 provide a concise overview of the approaches introduced by [RFC6906] and [I-D.nottingham-link-hint], to respectively indicate and discover the profile of resource representations. Section 4 specifies a generally applicable approach to negotiate for representations that conform to a profile preferred by a user agent. Section 5 shows how servers can convey which profiled representations they are able to accept from user agents.

1.3. Notational Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

This specification uses the terms "link context" and "link target" as defined in [RFC8288]. These terms respectively correspond with "Context IRI" and "Target IRI" as used in [RFC5988]. Although defined as IRIs, in common scenarios they are also URIs.

In the examples provided in this document, links in the HTTP "Link" header are shown on separate lines in order to improve readability. Note, however, that as per Section 3.2 of [RFC7230], line breaks are not allowed in values for HTTP headers; only whitespaces and tabs are supported as seperators.

2. Indicating Profiled Representations

As per [RFC6906], a web link with a "profile" link relation type can be used to indicate the profile of a representation that is exchanged in HTTP interactions. Figure 1 shows a client requesting a representation from a server and Figure 2 shows the server responding. The response includes a "Link" header ([RFC8288]) that contains a link with the "profile" relation type. The link target <http://purl.org/dc/terms/> indicates the profile of the response body (not shown).

Figure 3 shows a client submitting a representation to a server, using the same approach to express the profile to which that representation complies. Section 5 describes how a server can convey the profiles it supports for representations that are submitted by a user agent.

```
GET /some/resource HTTP/1.1
Host: example.org
Connection: close
```

Figure 1: Client requests a representation

```
HTTP/1.1 200 OK
Content-Type: application/xml ; charset=utf-8
Link: <http://purl.org/dc/terms/>; rel="profile"
Content-Length: 23364
Connection: close

...
```

Figure 2: Server indicates the profile of the returned representation

```
PUT /some/resource HTTP/1.1
Host: example.org
Content-Type: application/xml ; charset=utf-8
Link: <http://purl.org/dc/terms/>; rel="profile"
Content-Length: 23364
Connection: close

...
```

Figure 3: Client indicates the profile of a representation submitted to a server

In some cases organisations use separate servers to perform content negotiation and to deliver resources, e. g. when static content is served through content delivery networks. The servers that perform the content negotiation interact with the client requesting the resource and then typically refer to the correct representation using a 303 redirect. In those cases the servers that deliver the representations are not profile aware and thus cannot add the appropriate "Link" headers to the response. Instead the server performing the negotiation will have to supply that information. This is done by adding an "anchor" attribute pointing to the representation the link header refers to. Figure 4 shows the response to a Figure 1 where the server performing the negotiation redirects the client to the resource specified in the "Location" header and by using the same URI in the "anchor" attribute of the "Link" header indicates that the information in the "Link" header does not apply to this response but to the resource redirected to.

```
HTTP/1.1 300 See other
Location: https://static.example.org/other/resource
Link: <http://purl.org/dc/terms/>; rel="profile"
       ; anchor="https://static.example.org/other/resource"

...
```

Figure 4: Server indicates the profile of the representation referred to in the 'Location' header

3. Discovering Profiled Representations

The link hints capability introduced in [I-D.nottingham-link-hint], can be leveraged by a server to make profiled representations discoverable by including a "formats" attribute on web links. Figure 5 shows a response to the client request of Figure 1 in which the server uses this technique. The "Link" header indicates the profile of the returned representation but also points at two alternative representations, each of which conforms to another profile.

```
HTTP/1.1 200 OK
Content-Type: application/xml
Link: <http://purl.org/dc/terms/> ; rel="profile" ,
      </some/other_resource_1>
         ; rel="alternate" ; type="application/xml"
         ; formats="http://example.org/our_internal_xml_profile" ,
      </some/other_resource_2>
         ; rel="alternate"
         ; formats="http://example.org/our_community_profile"
Content-Length: 23364
Connection: close

...
```

Figure 5: Server makes profiled representations discoverable

4. Negotiating for Profiled Representations

Section 1.2 describes two approaches for conveying profile information in HTTP interactions that make it possible to negotiate for profiled representations by applying content negotiation in the media type dimension. It also indicates the restricted applicability of these approaches, in both cases a result of their direct dependence on the media type registration process.

This section describes an approach that applies content negotiation in a dimension that it was previously not applied to. The profile negotiation approach introduced here is generally applicable for resource representations, irrespective of media type. These are its core aspects:

- In order to allow a user agent to inform a server about its preferences regarding profiles for resource representations, the "Accept-Profile" HTTP header introduced here is used. A user agent can specify several profiles and use quality indicators (q-values) to indicate preferences.

- In order to allow a server to express its support for profile negotiation the "Vary" HTTP header is used, in this case, with the "Accept-Profile" value.

- In order to allow a server to convey the profile of a representation delivered to a user agent, rather than introducing a server-side counterpart to the client-side "Accept-Profile" header, the existing "profile" link approach introduced by [RFC6906] is used. If a representation conforms to multiple profiles, a distinct "profile" link is used per profile; the order in which these links are provided has no relevance.

- In order to allow a server to convey the profiles it supports, web links with the "formats" link hint introduced in [I-D.nottingham-link-hint] are used to convey the profiles of the link target resources. This approach uniformely applies to reponses to HTTP HEAD/GET/PUT/POST/PATCH.

- Throughout the profile negotiation approach, a profile MUST be referred to by a URI. This is the case for the content of the "Accept-Profile" HTTP header, for the target of web links with the "profile" link relation, and for the content of the "formats" link hint for web links.

4.1. Profile Negotiation Details

Profile negotiation uses the content negotiation processes described in Section 3.4 of RFC 7231 [RFC7231] but applies them to the profile dimension. Both proactive negotiation and reactive negotiation for profiles can be supported by servers. Both are described in more detail in the remainder of this section.

In profile negotiation, a profile MUST be referred to by a URI that, from here onwards, is named a profile URI. If the profile URI is dereferencable it SHOULD lead to a document that details the profile. If the profile URI is not dereferencable (e.g. a URN [RFC8141] or an

info-URI [RFC4452]) facilities SHOULD be available to allow user agents and servers to understand their meaning, e.g. community registries of profiles.

4.1.1. Proactive Profile Negotiation

In proactive profile negotiation, the user agent uses the "Accept-Profile" HTTP header to inform the server about the agent's preference regarding profiles to be used for representing a resource in the server's response. In case a user agent wants to express a preference for a single profile, the value of the header is that profile's URI. In case a user agent wants to express a preference for multiple profiles, the value of the header is a list containing each profile's URI, seperated by commas. Alternatively, multiple "Accept-Profile" HTTP headers can be used, each conveying a single profile URI. Quality indicators (q-values) MAY be used to rank profile preferences. The order in which profile URIs are conveyed or the duplicate mentioning of a same profile URI MUST NOT be interpreted as significant.

A server that supports proactive profile negotiation for the resource that a user agent interacts with:

- MUST include a "Vary" HTTP header containing the value "accept-profile" in its response to the user agent.
- MUST include a "Link" HTTP header containing a link with the "profile" relation type that has as link target the profile URI of the resource representation returned to the user agent. In case the representation conforms to additional profiles known to the server, such a "profile" link SHOULD be included for each.
- SHOULD convey the availability of alternate profiled representations of the resource by using the link hint approach described in Section 4.1.2.

These requirements for servers that support proactive profile negotiation also apply when:

- The user agent expressed a profile preference in its request by using an "Accept-Profile" header but the server cannot return a representation that conforms to a preferred profile.
- The user agent did not express a profile preference using an "Accept-Profile" header in its request.

A user agent SHOULD interpret the abscence of a "Vary" HTTP header with an "accept-profile" value in a response from a server as the

lack of support for profile negotiation for the resource the user agent interacts with.

A server SHOULD consider representations that do not conform to any of the profiles listed by a user agent in an "Accept-Profile" header as non-interpretable by the agent. As such, honoring the user agent preferences in the profile dimension SHOULD take precedence over honoring content negotiation in other dimensions.

In Figure 6 a user agent requests an RDF serialization from a server and expresses preference for two media types using the "Accept" header and two profiles using the "Accept-Profile" header. It uses q-values to express a preference for the profile with profile URI <http://example.org/shapes/shape-1> over the one with profile URI <http://example.org/shapes/shape-2>.

```
GET /document HTTP/1.1
Host: example.org
Accept: text/turtle, application/rdf+xml
Accept-Profile: "http://example.org/shapes/shape-1" ; q=0.8 ,
                "http://example.org/shapes/shape-2" ; q=0.5
Connection: close
```

Figure 6: Client expresses a preference for two profiles

Figure 7 shows the server's response to the request of Figure 6. By means of the "Vary" header, the server expresses support for negotiation in both the media type and profile dimensions. The "profile" link with link target <http://example.org/shapes/shape-2> indicates that the server was able to honor the user agent's second profile preference. Another "profile" link shows that the delivered representation also conforms to a profile with profile URI <http://example.org/shapes/shape-3>. Furthermore, using an "alternate" link, the server indicates support for another profile with <http://example.org/shapes/shape-4> as profile URI. Note that, even if the user agent does not express profile preferences using the "Accept-Profile" header and the server's "Vary" header would be the same, the "Link" header would still include a "profile" link to indicate the profile of the representation returned by the server.

```
HTTP/1.1 200 OK
Content-Type: text/turtle
Vary: Accept, Accept-Profile
Link: <http://example.org/shapes/shape-2>
      ; rel="profile" ,
      <http://example.org/shapes/shape-3>
      ; rel="profile" ,
      <http://example.org/document>
      ; rel="alternate"
      ; type="text/turtle"
      ; formats="http://example.org/shapes/shape-4"
Content-Length: 8724
Connection: close

...
```

Figure 7: Response honors a user agent's preference

Figure 8 shows the response to the request of Figure 6 in case the server supports profile negotiation for the resource at hand but can not return a representation that conforms to a profile preferred by the user agent. The server has chosen to nevertheless return a representation that conforms to profile <http://example.org/shapes/shape-4>, which is not among the ones preferred by the user agent. The server also reveals the existence of a representation that conforms to profile <http://example.org/shapes/shape-5>. The server could also choose not to return a default representation in which case it would return a "406 Not Acceptable" HTTP response code and no response body. It would not provide any "profile" links but might use "alternate" links with a "formats" attribute to indicate the existence of supported profiles.

```
HTTP/1.1 200 OK
Content-Type: text/turtle
Vary: Accept, Accept-Profile
Link: <http://example.org/shapes/shape-4>
      ; rel="profile" ,
      <http://example.org/document>
      ; rel="alternate"
      ; type="text/turtle"
      ; formats="http://example.org/shapes/shape-5"
Content-Length: 6333
Connection: close

...
```

Figure 8: Response does not honor a user agent's preference but includes default representation

Figure 9 shows the response to the request of Figure 6 in case the server does not support profile negotiation for the resource <http://example.org/document>. It does support negotiation in the media type dimension and has honoured one of the user agent's preferences with that regard, as can be seen by the "Vary" and "Content-Type" headers.

```
HTTP/1.1 200 OK
Content-Type: text/turtle
Vary: Accept
Content-Length: 8724
Connection: close

...
```

Figure 9: Response indicates lack of support for proactive profile negotiation

4.1.2. Reactive Profile Negotiation

In reactive profile negotiation, the user agent selects the profiled representation that best meets its preferences on the basis of a list of possible representations it obtains from the server. A server that supports reactive profile negotiation MUST provide such a list of supported profiled representation as a set of links in the "Link" header. Each of these links:

- SHOULD have the "alternate" relation type.

- MUST use the "formats" link hint to convey the profile URI of the profile to which the resource that is the link target conforms.

- SHOULD use the "allow" link hint to convey the HTTP methods that are supported by the resource that is the link target.

Figure 10 shows a user agent issuing a HTTP HEAD on a resource in order to determine whether profiled representations are available for it. Figure 11 shows the response of a server that supports reactive profile negotiation. By means of "alternate" links in the "Link" header, the server indicates support for two profiled representations for the resource at hand, and, for each, indicates the URI at which they can be accessed, as well as their respective profile URIs, media types, and supported HTTP methods. On the basis of this response, the client can decide whether any of the linked resources conform to a preferred profile, and, if so, access the respective link target. Figure 12 shows the response to an HTTP HEAD issued on the link target <http://example.org/bibrecord/1/DC> of the first "alternate" link.

```
HEAD /bibrecord/1 HTTP/1.1
Host: example.org
Accept: application/xml
Connection: close
```

Figure 10: Client determines support for profiles

```
HTTP/1.1 200 OK
Content-Type: text/plain
Link: <http://example.org/bibrecord/1/DC>
      ; rel="alternate"
      ; type="application/xml"
      ; formats="http://purl.org/dc/terms/"
      ; allow="HEAD,GET,PATCH" ,
      <http://example.org/bibrecord/1/BIBFRAME>
      ; rel="alternate"
      ; formats="http://id.loc.gov/ontologies/bibframe/"
      ; allow="HEAD,GET"
Content-Length: 200
Connection: close
```

Figure 11: Server supports two profiles

```
HTTP/1.1 200 OK
Content-Type: application/xml
Link: <http://purl.org/dc/terms/>
      ; rel="profile" ,
      <http://example.org/bibrecord/1/BIBFRAME>
      ; rel="alternate"
      ; formats="http://id.loc.gov/ontologies/bibframe/"
      ; allow="HEAD,GET"
Allow: HEAD, GET, PATCH
Accept-Patch: application/xml-patch+xml
Content-Length: 458
Connection: close
```

Figure 12: Response to a client accessesing a profiled representation

4.2. Accept-Profile HTTP Header Syntax

Figure 13 describes the syntax of the "Accept-Profile" HTTP header, using the grammar defined in RFC 5234 [RFC5234] and the rules defined in Section 3.2 of RFC 7230 [RFC7230]. The definitions of "URI-reference" and "weight" are imported from RFC 7230 [RFC7230] and RFC 7231 [RFC7231], respectively.

```
Accept-Profile = "Accept-Profile" ":"
OWS (accept-value) *(OWS "," OWS accept-value) OWS
accept-value = "<" URI-reference ">" [weight] | accept-value-ext
```

Figure 13: ABNF for the "Accept-Profile" HTTP header

5. Writing Profiled Representations

A user agent that wants to submit a profiled representation to a server can use the reactive negotiation approach to determine the nature of a server's support with this regard.

A server that allows user agents to submit profiled representations SHOULD follow the directions for reactive negotiation described in Section 4.1.2.

A client that submits a representation that complies to a profile that was not advertised by the server by means of the reactive negotiation approach, SHOULD assume that the server is not able to process it.

A server that fails a submission request due to receiving a payload with a profile that it does not support MUST respond with a "422

Unprocessable Entity" HTTP status code and SHOULD use the approach described in Section 4.1.2 to convey profiles that are supported.

Continuing from Figure 12, Figure 14 shows a user agent issuing a HTTP PATCH against resource <http://example.org/bibrecord/1/DC> in order to update it. It uses the "Content-Encoding" header to convey the XML Patch media type of its message body and a link with a "profile" relation type in the "Link" header to indicate the profile to which it conforms. Figure 15 shows the server's response, indicating that the patch was applied successfully; the server returns the updated representation as message body.

```
PATCH /bibrecord/1/DC HTTP/1.1
Host: example.org
Content-Encoding: application/xml-patch+xml
Link: <http://purl.org/dc/terms/>
      ; rel="profile"
Accept: application/xml
Content-Length: 321
Connection: close

...
```

Figure 14: Client submits profiled representation

```
HTTP/1.1 200 OK
Content-Type: application/xml
Content-Location: http://example.org/bibrecord/1/DC
Link: <http://example.org/bibrecord/1/DC>
      ; rel=profile" ,
      <http://example.org/bibrecord/1/BIBFRAME>
      ; rel="alternate"
      ; formats="http://id.loc.gov/ontologies/bibframe/"
      ; allow="HEAD,GET"
Allow: HEAD, GET, PATCH
Accept-Patch: application/xml-patch+xml
Content-Length: 592
Connection: close

...
```

Figure 15: Server indicates successful submission of profiled representation by client

If, in Figure 14, the user agent would have issued an HTTP PATCH on resource <http://example.org/bibrecord/1/DC> indicating that the

profile URI of the patch was <http://id.loc.gov/ontologies/bibframe/>, the response would have a 422 HTTP status code to express that the profile is not supported by the resource at hand. Such a response is shown in Figure 16.

```
HTTP/1.1 422 Unprocessable Entity
Content-Type: text/plain
Link: <http://example.org/bibrecord/1/DC>
      ; rel="alternate"
      ; type="application/xml"
      ; formats="http://purl.org/dc/terms/"
      ; allow="HEAD,GET,PATCH"
Content-Length: 110
Connection: close

...
```

Figure 16: Server indicates unsuccessful submission of profiled representation by client

6. IANA Considerations

This memo requires IANA to register the Accept-Profile HTTP header defined in Section 4.2 in the appropriate IANA registry:

- Header Field Name: Accept-Profile
- Applicable Protocol: Hypertext Transfer Protocol (HTTP)
- Status: Informational
- Author/Change controller: IETF
- Specification document(s): this document

7. Security Considerations

8. Acknowledgments

Many thanks to our colleagues for feedback: Enno Meijers, Michael L. Nelson

9. References

9.1. Normative References

[I-D.nottingham-link-hint]
Nottingham, M., "HTTP Link Hints", draft-nottingham-link-hint-02 (work in progress), March 2020.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC5234] Crocker, D., Ed. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", STD 68, RFC 5234, DOI 10.17487/RFC5234, January 2008, <https://www.rfc-editor.org/info/rfc5234>.

[RFC6838] Freed, N., Klensin, J., and T. Hansen, "Media Type Specifications and Registration Procedures", BCP 13, RFC 6838, DOI 10.17487/RFC6838, January 2013, <https://www.rfc-editor.org/info/rfc6838>.

[RFC6906] Wilde, E., "The 'profile' Link Relation Type", RFC 6906, DOI 10.17487/RFC6906, March 2013, <https://www.rfc-editor.org/info/rfc6906>.

[RFC7230] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing", RFC 7230, DOI 10.17487/RFC7230, June 2014, <https://www.rfc-editor.org/info/rfc7230>.

[RFC7231] Fielding, R., Ed. and J. Reschke, Ed., "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content", RFC 7231, DOI 10.17487/RFC7231, June 2014, <https://www.rfc-editor.org/info/rfc7231>.

9.2. Informative References

[HeeryAndPatel]
Heery, R. and M. Patel, "Application Profiles: Mixing and Matching Metadata Schemas", 2000, <http://www.ariadne.ac.uk/issue25/app-profiles>.

[RFC4452] Van de Sompel, H., Hammond, T., Neylon, E., and S. Weibel, "The "info" URI Scheme for Information Assets with Identifiers in Public Namespaces", RFC 4452, DOI 10.17487/RFC4452, April 2006, <https://www.rfc-editor.org/info/rfc4452>.

[RFC5988] Nottingham, M., "Web Linking", RFC 5988, DOI 10.17487/RFC5988, October 2010, <https://www.rfc-editor.org/info/rfc5988>.

[RFC8141] Saint-Andre, P. and J. Klensin, "Uniform Resource Names (URNs)", RFC 8141, DOI 10.17487/RFC8141, April 2017, <https://www.rfc-editor.org/info/rfc8141>.

[RFC8288] Nottingham, M., "Web Linking", RFC 8288, DOI 10.17487/RFC8288, October 2017, <https://www.rfc-editor.org/info/rfc8288>.

[W3C.REC-shacl-20170720] Knublauch, H. and D. Kontokostas, "Shapes Constraint Language (SHACL)", World Wide Web Consortium Recommendation REC-shacl-20170720, July 2017, <https://www.w3.org/TR/2017/REC-shacl-20170720>.

[W3C.REC-xmlschema11-1-20120405] Gao, S., Sperberg-McQueen, M., Thompson, H., Mendelsohn, N., Beech, D., and M. Maloney, "W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures", World Wide Web Consortium Recommendation REC-xmlschema11-1-20120405, April 2012, <https://www.w3.org/TR/2012/REC-xmlschema11-1-20120405>.

Authors' Addresses

Lars G. Svensson

Email: lars.svensson@web.de
URI: https://orcid.org/0000-0002-8714-9718

Ruben Verborgh
Ghent University - imec
Sint-Pietersnieuwstraat 41
Ghent 9000
Belgium

Phone: +32 9 331 49 10
Email: ruben.verborgh@ugent.be
URI: https://ruben.verborgh.org/

Herbert Van de Sompel
Data Archiving and Networked Services
Anna van Saksenlaan 51
The Hague 2593 HW
Netherlands

Email: herbert.van.de.sompel@dans.knaw.nl
URI: https://orcid.org/0000-0002-0715-6126