Internet Engineering Task Force (IETF)
Request for Comments: 9264
Category: Standards Track
ISSN: 2070-1721

E. Wilde
Axway
H. Van de Sompel
Data Archiving and Networked Services

July 2022

Linkset: Media Types and a Link Relation Type for Link Sets

draft-ietf-httpapi-linkset-10

Abstract

This specification defines two formats and associated media types for representing sets of links as standalone documents. One format is based on JSON, and the other is aligned with the format for representing links in the HTTP "Link" header field. This specification also introduces a link relation type to support the discovery of sets of links.

Status of This Memo

This is an Internet Standards Track document.

This document is a product of the Internet Engineering Task Force (IETF). It represents the consensus of the IETF community. It has received public review and has been approved for publication by the Internet Engineering Steering Group (IESG). Further information on Internet Standards is available in Section 2 of RFC 7841[1].

Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9264[2].

Copyright Notice

Copyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info[3]) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.

ERROR: User-supplied boilerplate differs from auto-generated boilerplate (inserting auto-generated); Strings differ at position 367, 1st string ends in: [[[ection 2 of RFC 7841.Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc9264.Copyright NoticeCopyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.i

[1] https://www.rfc-editor.org/rfc/rfc7841.html#section-2
[2] https://www.rfc-editor.org/info/rfc9264
[3] https://trustee.ietf.org/license-info

etf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.]]], 2nd string ends in: [[[ection 2 of RFC 78 4111 https://www.rfc-editor.org/rfc/rfc7841.html#section-2.Information about the current status of this document, any errata, and how to provide feedback on it may be obtained at https://www.rfc-editor.org/info/rfc926422 https://www.rfc-editor.org/info/rfc9264.Copyright NoticeCopyright (c) 2022 IETF Trust and the persons identified as the document authors. All rights reserved.This document is subject to BCP 78 and the IETF Trust's Legal Provisions Relating to IETF Documents (https://trustee.ietf.org/license-info33 https://trustee.ietf.org/license-info) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Revised BSD License text as described in Section 4.e of the Trust Legal Provisions and are provided without warranty as described in the Revised BSD License.]]] (at line 6)

Table of Contents

1 Introduction....4

2 Terminology....5

3 Use Cases and Motivation....6

3.1 Third-Party Links....6

3.2 Challenges Writing to the HTTP "Link" Header Field....6

3.3 Large Number of Links....6

4 Document Formats for Sets of Links....8

4.1 HTTP Link Document Format: application/linkset....8

4.2 JSON Document Format: application/linkset+json....9

4.2.1 Set of Links....9

4.2.2 Link Context Object....9

4.2.3 Link Target Object....9

4.2.4 Link Target Attributes....11

4.2.5 JSON Extensibility....14

5 The "profile" Parameter for Media Types to Represent Sets of Links....15

6 The "linkset" Relation Type for Linking to a Set of Links....16

7 Examples....17

7.1 Set of Links Provided as "application/linkset"....17

7.2 Set of Links Provided as "application/linkset+json"....18

7.3 Discovering a Link Set via the "linkset" Link Relation Type....21

7.4 Link Set Profiles....21

7.4.1 Using a "profile" Attribute with a "linkset" Link....21

7.4.2 Using a "profile" Parameter with a Link Set Media Type....22

7.4.3 Using a Link with a "profile" Link Relation Type....22

8 IANA Considerations....24

8.1 Link Relation Type: linkset....24

8.2 Media Type: application/linkset....24

8.3 Media Type: application/linkset+json....25

9 Security Considerations....27

10 References....28

10.1 Normative References....28

10.2 Informative References....28

Appendix A JSON-LD Context....30

Acknowledgements....34

Authors' Addresses....35

1. Introduction

Resources on the Web often use typed Web Links [RFC8288], either in resource representations -- for example, using the <link> element for HTML documents or in the HTTP "Link" header field for documents of any media type. In some cases, however, providing links in this manner is impractical or impossible, and delivering a set of links as a standalone document is preferable.

Therefore, this specification defines two formats for representing sets of Web Links and their attributes as standalone documents. One serializes links in the same format as the format used in the HTTP "Link" header field, and the other serializes links in JSON. It also defines associated media types to represent sets of links, and the "linkset" relation type to support the discovery of any resource that conveys a set of links as a standalone document.

2. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This specification uses the terms "link context" and "link target" in the same manner that "Web Linking" [RFC8288] uses them.

In the examples provided in this document, links in the HTTP "Link" header field are shown on separate lines in order to improve readability. Note, however, that as per Section 5.5 of "HTTP Semantics" [RFC9110], line breaks are deprecated in values for HTTP fields; only whitespaces and tabs are supported as separators.

3. Use Cases and Motivation

The following sections describe use cases in which providing links by means of a standalone document instead of in an HTTP "Link" header field or as links embedded in the resource representation is advantageous or necessary.

For all scenarios, links could be provided by means of a standalone document that is formatted according to the JSON-based serialization, the serialization aligned with the HTTP "Link" field format, or both. The former serialization is motivated by the widespread use of JSON and related tools, which suggests that handling sets of links expressed as JSON documents should be attractive to developers. The latter serialization is provided for compatibility with the existing serialization used in the HTTP "Link" field and to allow the reuse of tools created to handle it.

It is important to keep in mind that when providing links by means of a standalone representation, other links can still be provided using other approaches, i.e., it is possible to combine various mechanisms to convey links.

3.1. Third-Party Links

In some cases, it is useful that links pertaining to a resource are provided by a server other than the one that hosts the resource. For example, this allows:

- Providing links in which the resource is involved not just as a link context but also as a link target, with a different resource being the link context.
- Providing links pertaining to the resource that the server hosting that resource is not aware of.
- External management of links pertaining to the resource in a special-purpose link management service.

In such cases, links pertaining to a resource can be provided by another, specific resource. That specific resource may be managed, by the same custodian or by another custodian, as the resource to which the links pertain. For clients intent on consuming links provided in that manner, it would be beneficial if the following conditions were met:

- Links are provided in a document that uses a well-defined media type.
- The resource to which the provided links pertain is able to link to the resource that provides these links using a well-known link relation type.

These requirements are addressed in this specification through the definition of two media types and a link relation type, respectively.

3.2. Challenges Writing to the HTTP "Link" Header Field

In some cases, it is not straightforward to write links to the HTTP "Link" header field from an application. This can, for example, be the case because not all required link information is available to the application or because the application does not have the capability to directly write HTTP fields. In such cases, providing links by means of a standalone document can be a solution. Making the resource that provides these links discoverable can be achieved by means of a typed link.

3.3. Large Number of Links

When conveying links in an HTTP "Link" header field, it is possible for the size of the HTTP response fields to become unpredictable. This can be the case when links are determined dynamically in a manner dependent on a range of contextual factors. It is possible to statically configure a web server to correctly handle large HTTP response fields by specifying an upper bound for their size. But when the number of links is unpredictable, estimating a reliable upper bound is challenging.

Section 15 of "HTTP Semantics" [RFC9110] defines error codes related to excess communication by the user agent ("413 Content Too Large" and "414 URI Too Long"), but no specific error codes are defined to indicate that response field content exceeds the upper bound that can be handled by the server and thus has been truncated. As a result, applications take countermeasures aimed at controlling the size of the HTTP

"Link" header field -- for example, by limiting the links they provide to those with select relation types, thereby limiting the value of the HTTP "Link" header field to clients. Providing links by means of a standalone document overcomes challenges related to the unpredictable (to the web server implementation) nature of the size of HTTP "Link" header fields.

4. Document Formats for Sets of Links

This section specifies two document formats to convey a set of links. Both are based on the abstract model specified in Section 2 of "Web Linking" [RFC8288], which defines a link as consisting of a "link context", a "link relation type", a "link target", and optional "target attributes":

- The format defined in Section 4.1 is nearly identical to the field value of the HTTP "Link" header field as specified in Section 3 of [RFC8288].
- The format defined in Section 4.2 is expressed in JSON [RFC8259].

Links provided in the HTTP "Link" header field are intended to be used in the context of an HTTP interaction, and contextual information that is available during an interaction is used to correctly interpret them. Links provided in link sets, however, can be reused outside of an HTTP interaction, when no such contextual information is available. As a result, implementers of link sets should strive to make them self-contained by adhering to the following recommendations.

For links provided in the HTTP "Link" header field that have no anchor or that use relative references, the URI of the resource that delivers the links provides the contextual information that is needed for their correct interpretation. In order to support use cases where link set documents are reused outside the context of an HTTP interaction, it is RECOMMENDED to make them self-contained by adhering to the following guidelines:

- For every link provided in the set of links, explicitly provide the link context using the "anchor" attribute.
- For the link context ("anchor" attribute) and link target ("href" attribute), use URI references that are not relative references (as defined in Section 4.1 of [RFC3986]).

If these recommendations are not followed, the interpretation of links in link set documents will depend on which URI is used as the context.

For a "title" attribute provided on a link in the HTTP "Link" header field, the language in which the title is expressed is provided by the "Content-Language" header field of the HTTP interaction with the resource that delivers the links. This does not apply to "title" attributes provided for links in link set documents because that would constrain all links in a link set to having a single title language and would not support determining title languages when a link set is used outside of an HTTP interaction. In order to support use cases where link set documents are reused outside the context of an HTTP interaction, it is RECOMMENDED to make them self-contained by using the "title*" attribute instead of the "title" attribute because "title*" allows expressing the title language as part of its value by means of a language tag. Note that, in this regard, language tags are matched case insensitively (see Section 2.1.1 of [RFC5646]). If this recommendation is not followed, accurately determining the language of titles provided on links in link set documents will not be possible.

Note also that Section 3.3 of [RFC8288] deprecates the "rev" construct that was provided by [RFC5988] as a means to express links with a directionality that is the inverse of direct links that use the "rel" construct. In both serializations for link sets defined here, inverse links may be represented as direct links using the "rel" construct and by switching the roles of the resources involved in the link.

4.1. HTTP Link Document Format: application/linkset

This document format is nearly identical to the field value of the HTTP "Link" header field as defined in Section 3 of [RFC8288], more specifically by its ABNF [RFC5234] production rule for "Link" and its subsequent rules. It differs from the format for field values of the HTTP "Link" header field only in that not only spaces and horizontal tabs are allowed as separators but also newline characters as a means to improve readability for humans. The use of non-ASCII characters in the field value of the HTTP "Link" header field is not allowed and as such is also not allowed in "application/linkset" link sets.

The assigned media type for this format is "application/linkset".

When converting an "application/linkset" document to a field value for the HTTP "Link" header field, newline characters MUST be removed or MUST be replaced by whitespace (SP) in order to comply with Section 5.5 of [RFC9110].

Implementers of "application/linkset" link sets should strive to make them self-contained by following the recommendations provided in Section 4 regarding their use outside the context of an HTTP interaction.

It should be noted that the "application/linkset" format specified here is different from the "application/link-format" format specified in [RFC6690] in that the former fully matches the field value of the HTTP "Link" header field as defined in Section 3 of [RFC8288], whereas the latter introduces constraints on that definition to meet requirements for Constrained RESTful Environments (CoRE).

4.2. JSON Document Format: application/linkset+json

This document format uses JSON [RFC8259] as the syntax to represent a set of links. The set of links follows the abstract model defined by Section 2 of [RFC8288].

The assigned media type for this format is "application/linkset+json".

In the interests of interoperability, "application/linkset+json" link sets MUST be encoded using UTF-8 as per Section 8.1 of [RFC8259].

Implementers of "application/linkset+json" link sets should strive to make them self-contained by following the recommendations provided in Section 4 regarding their use outside the context of an HTTP interaction.

The "application/linkset+json" serialization allows for OPTIONAL support of a JSON-LD serialization. This can be achieved by adding an appropriate context to the "application/linkset+json" serialization using the approach described in Section 6.1 of [W3C.REC-json-ld]. Communities of practice can decide which context best meets their application needs. Appendix A shows an example of a possible context that, when added to a JSON serialization, allows it to be interpreted as Resource Description Framework (RDF) data [W3C.REC-rdf11-concepts].

4.2.1. Set of Links

In the JSON representation of a set of links:

- A set of links is represented in JSON as an object that MUST contain "linkset" as its sole member.
- The value of the "linkset" member is an array in which a distinct JSON object -- the "link context object" (see Section 4.2.2) -- is used to represent links that have the same link context.
- Even if there is only one link context object, it MUST be wrapped in an array.

4.2.2. Link Context Object

In the JSON representation, one or more links that have the same link context are represented by a JSON object -- the link context object. A link context object adheres to the following rules:

- Each link context object MAY contain an "anchor" member with a value that represents the link context. If present, this value MUST be a URI reference and SHOULD NOT be a relative reference as defined in Section 4.1 of [RFC3986].
- For each distinct relation type that the link context has with link targets, a link context object MUST contain an additional member. The value of this member is an array in which a distinct JSON object -- the "link target object" (see Section 4.2.3) -- MUST be used for each link target for which the relationship with the link context (value of the encompassing "anchor" member) applies. The name of this member expresses the relation type of the link as follows:
  - For registered relation types (Section 2.1.1 of [RFC8288]), the name of this member is the registered name of the relation type.
  - For extension relation types (Section 2.1.2 of [RFC8288]), the name of this member is the URI that uniquely represents the relation type.
- Even if there is only one link target object, it MUST be wrapped in an array.

4.2.3. Link Target Object

In the JSON representation, a link target is represented by a JSON object -- the link target object. A link target object adheres to the following rules:

- Each link target object MUST contain an "href" member with a value that represents the link target. This value MUST be a URI reference and SHOULD NOT be a relative reference as defined in Section 4.1 of [RFC3986]. Cases where the "href" member is present but no value is provided for it (i.e., the resource providing the set of links is the target of the link in the link target object) MUST be handled by providing an "href" member with an empty string as its value ("href": "").
- In many cases, a link target is further qualified by target attributes. Various types of attributes exist, and they are conveyed as additional members of the link target object as detailed in Section 4.2.4.

The following example of a JSON-serialized set of links represents one link with its core components: link context, link relation type, and link target.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "next": [
        {"href": "https://example.com/foo"}
      ]
    }
  ]
}
```

Figure 1: Simple linkset example

The following example of a JSON-serialized set of links represents two links that share a link context and relation type but have different link targets.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "item": [
        {"href": "https://example.com/foo1"},
        {"href": "https://example.com/foo2"}
      ]
    }
  ]
}
```

Figure 2: Linkset with two links with the same context

The following example shows a set of links that represents two links, each with a different link context, link target, and relation type. One relation type is registered, and the other is an extension relation type.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "next": [
        {"href": "https://example.com/foo1"}
      ]
    },
    { "anchor": "https://example.net/boo",
      "https://example.com/relations/baz" : [
        {"href": "https://example.com/foo2"}
      ]
    }
  ]
}
```

Figure 3: Linkset with two links with different contexts

4.2.4. Link Target Attributes

A link may be further qualified by target attributes as defined by Section 2 of [RFC8288]. Three types of attributes exist:

- Serialization-defined attributes as described in Section 3.4.1 of [RFC8288].
- Extension attributes defined and used by communities as allowed by Section 3.4.2 of [RFC8288].
- Internationalized versions of the "title" attribute as defined by [RFC8288] and of extension attributes allowed by Section 3.4 of [RFC8288].

The handling of these different types of attributes is described in the sections below.

4.2.4.1. Target Attributes Defined by Web Linking

Section 3.4.1 of [RFC8288] defines the following target attributes that may be used to annotate links: "hreflang", "media", "title", "title*", and "type"; these target attributes follow different occurrence and value patterns. In the JSON representation, these attributes MUST be conveyed as additional members of the link target object as follows:

"hreflang": The "hreflang" target attribute, defined as optional and repeatable by [RFC8288], MUST be represented by an "hreflang" member, its value MUST be an array (even if there is only one value to be represented), and each value in that array MUST be a string -- representing one value of the "hreflang" target attribute for a link -- that follows the same model as the syntax discussed in [RFC8288].

"media": The "media" target attribute, defined as optional and not repeatable by [RFC8288], MUST be represented by a "media" member in the link target object, and its value MUST be a string that follows the same model as the syntax discussed in [RFC8288].

"title": The "title" target attribute, defined as optional and not repeatable by [RFC8288], MUST be represented by a "title" member in the link target object, and its value MUST be a JSON string.

"title*": The "title*" target attribute, defined as optional and not repeatable by [RFC8288], is motivated by character encoding and language issues and follows the model defined in [RFC8187]. The details of the JSON representation that applies to "title*" are described in Section 4.2.4.2.

"type": The "type" target attribute, defined as optional and not repeatable by [RFC8288], MUST be represented by a "type" member in the link target object, and its value MUST be a string that follows the same model as the syntax discussed in [RFC8288].

The following example illustrates how the "hreflang" (repeatable) target attribute and the "type" (not repeatable) target attribute are represented in a link target object.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "next": [
        { "href":     "https://example.com/foo",
          "type":     "text/html",
          "hreflang": [ "en" , "de" ]
        }
      ]
    }
  ]
}
```

Figure 4: Linkset with "hreflang" and "type" target attributes

4.2.4.2. Internationalized Target Attributes

In addition to the target attributes described in Section 4.2.4.1, Section 3.4 of [RFC8288] also supports attributes that follow the content model of [RFC8187]. In [RFC8288], these target attributes are recognizable by the use of a trailing asterisk in the attribute name, such as "title*". The content model of [RFC8187] uses a string-based microsyntax that represents the character encoding, an optional language tag, and the escaped attribute value encoded according to the specified character encoding.

The JSON serialization for these target attributes MUST be as follows:

- An internationalized target attribute is represented as a member of the link context object with the same name (including the "*") as the attribute.
- The character encoding information as prescribed by [RFC8187] is not preserved; instead, the content of the internationalized attribute is represented as a JSON string.
- The value of the internationalized target attribute is an array that contains one or more JSON objects. The name of one member of such JSON objects is "value", and its value is the actual content (in its unescaped version) of the internationalized target attribute, i.e., the value of the attribute from which the encoding and language information are removed. The name of another, optional member of such JSON objects is "language", and its value is the language tag [RFC5646] for the language in which the attribute content is conveyed.

The following example illustrates how the "title*" target attribute as defined by Section 3.4.1 of [RFC8288] is represented in a link target object.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "next": [
        { "href":     "https://example.com/foo",
          "type":     "text/html",
          "hreflang": [ "en" , "de" ],
          "title":    "Next chapter",
          "title*":   [ { "value": "nächstes Kapitel" ,
                          "language" : "de" } ]
        }
      ]
    }
  ]
}
```

Figure 5: Linkset with "title" and "title*" target attributes

The above example assumes that the German title contains an umlaut character (in the original syntax, it would be encoded as title*=UTF-8'de'n%c3%a4chstes%20Kapitel), which gets encoded in its unescaped form in the JSON representation. Implementations MUST properly decode/encode internationalized target attributes that follow the model of [RFC8187] when transcoding between the "application/linkset" format and the "application/linkset+json" format.

4.2.4.3. Extension Target Attributes

Extension target attributes (e.g., as listed in Section 4.2.4.1) are attributes that are not defined by Section 3.4.1 of [RFC8288] but are nevertheless used to qualify links. They can be defined by communities in any way deemed necessary, and it is up to them to make sure their usage is understood by target applications. However, lacking standardization, there is no interoperable understanding of these extension attributes. One important consequence is that their cardinality is unknown to generic applications. Therefore, in the JSON serialization, all extension target attributes are treated as repeatable.

The JSON serialization for these target attributes MUST be as follows:

- An extension target attribute is represented as a member of the link target object with the same name as the attribute, including the "*" if applicable.
- The value of an extension attribute MUST be represented by an array, even if there is only one value to be represented.
- If the extension target attribute does not have a name with a trailing asterisk, then each value in that array MUST be a JSON string that represents one value of the attribute.
- If the extension attribute has a name with a trailing asterisk (it follows the content model of [RFC8187]), then each value in that array MUST be a JSON object. The value of each such JSON object MUST be structured as described in Section 4.2.4.2.

The following example shows a link target object with three extension target attributes. The value for each extension target attribute is an array. The first two are regular extension target attributes, with the first one ("foo") having only one value and the second one ("bar") having two. The last extension target attribute ("baz*") follows the naming rule of [RFC8187] and therefore is encoded according to the serialization described in Section 4.2.4.2.

```
{ "linkset":
  [
    { "anchor": "https://example.net/bar",
      "next": [
        { "href": "https://example.com/foo",
          "type": "text/html",
          "foo":  [ "foovalue" ],
          "bar":  [ "barone", "bartwo" ],
          "baz*": [ { "value": "bazvalue" ,
                      "language" : "en" } ]
        }
      ]
    }
  ]
}
```

Figure 6: Linkset with extension target attributes

4.2.5. JSON Extensibility

The Web Linking model [RFC8288] provides for the use of extension target attributes as discussed in Section 4.2.4.3. The use of other forms of extensions is NOT RECOMMENDED. Limiting the JSON format in this way allows unambiguous round trips between links provided in the HTTP "Link" header field, sets of links serialized according to the "application/linkset" format, and sets of links serialized according to the "application/linkset+json" format.

Cases may exist in which the use of extensions other than those discussed in Section 4.2.4.3 may be useful -- for example, when a link set publisher needs to include descriptive or technical metadata for internal consumption. If such extensions are used, they MUST NOT change the semantics of the JSON members defined in this specification. Agents that consume JSON linkset documents can safely ignore such extensions.

5. The "profile" Parameter for Media Types to Represent Sets of Links

As a means to convey specific constraints or conventions (as per [RFC6906]) that apply to a link set document, the "profile" parameter MAY be used in conjunction with the media types "application/linkset" and "application/linkset+json" as detailed in 4.1 and 4.2, respectively. For example, the parameter could be used to indicate that a link set uses a specific, limited set of link relation types.

The value of the "profile" parameter MUST be a non-empty list of space-separated URIs, each of which identifies specific constraints or conventions that apply to the link set document. When providing multiple profile URIs, care should be taken that the corresponding profiles are not conflicting. Profile URIs MAY be registered in the IANA's "Profile URIs" registry in the manner specified by [RFC7284].

The presence of a "profile" parameter in conjunction with the "application/linkset" and "application/linkset +json" media types does not change the semantics of a link set. As such, clients with and without knowledge of profile URIs can use the same representation.

Section 7.4.2 shows an example of using the "profile" parameter in conjunction with the "application/linkset +json" media type.

6. The "linkset" Relation Type for Linking to a Set of Links

The target of a link with the "linkset" relation type provides a set of links, including links in which the resource that is the link context participates.

A link with the "linkset" relation type MAY be provided in the header field and/or the body of a resource's representation. It may also be discovered by other means, such as through client-side information.

A resource MAY provide more than one link with a "linkset" relation type. Multiple such links can refer to the same set of links expressed using different media types, or to different sets of links, potentially provided by different third-party services.

The set of links provided by the resource that is the target of a "linkset" link may contain links in which the resource that is the context of the "linkset" link does not participate. User agents MUST process each link in the link set independently, including processing of the link context and link target, and MAY ignore links from the link set in which the context of the "linkset" link does not participate.

A user agent that follows a "linkset" link and obtains links for which anchors and targets are expressed as relative references (as per Section 4.1 of [RFC3986]) MUST determine what the context is for these links; it SHOULD ignore links for which it is unable to unambiguously make that determination.

As a means to convey specific constraints or conventions (as per [RFC6906]) that apply to a link set document, the "profile" attribute MAY be used in conjunction with the "linkset" link relation type. For example, the attribute could be used to indicate that a link set uses a specific, limited set of link relation types. The value of the "profile" attribute MUST be a non-empty list of space-separated URIs, each of which identifies specific constraints or conventions that apply to the link set document. Profile URIs MAY be registered in the IANA's "Profile URIs" registry in the manner specified by [RFC7284]. Section 7.4.1 shows an example of using the "profile" attribute on a link with the "linkset" relation type, making both the link set and the profile(s) to which it complies discoverable.

7. Examples

7.1 and 7.2 show examples whereby a set of links is provided as "application/linkset" and "application/linkset +json" documents, respectively. Section 7.3 illustrates the use of the "linkset" link relation type to support the discovery of sets of links, and Section 7.4 shows how to convey profile information pertaining to a link set.

7.1. Set of Links Provided as "application/linkset"

Figure 7 shows a client issuing an HTTP GET request against resource <https://example.org/links/resource1>.

```
GET /links/resource1 HTTP/1.1
Host: example.org
```

Figure 7: Client HTTP GET request

Figure 8 shows the response to the GET request of Figure 7. The response contains a "Content-Type" header field specifying that the media type of the response is "application/linkset". A set of links, revealing authorship and versioning related to resource <https://example.org/resource1>, is provided in the response body. The HTTP "Link" header field indicates the availability of an alternate representation of the set of links using media type "application/linkset+json".

```
HTTP/1.1 200 OK
Date: Mon, 12 Aug 2019 10:35:51 GMT
Server: Apache-Coyote/1.1
Content-Length: 1023
Content-Type: application/linkset
Link: <https://example.org/links/resource1>
      ; rel="alternate"
      ; type="application/linkset+json"

<https://authors.example.net/johndoe>
   ; rel="author"
   ; type="application/rdf+xml"
   ; anchor="https://example.org/resource1",
<https://example.org/resource1?version=3>
   ; rel="latest-version"
   ; type="text/html"
   ; anchor="https://example.org/resource1",
<https://example.org/resource1?version=2>
   ; rel="predecessor-version"
   ; type="text/html"
   ; anchor="https://example.org/resource1?version=3",
<https://example.org/resource1?version=1>
   ; rel="predecessor-version"
   ; type="text/html"
   ; anchor="https://example.org/resource1?version=2",
<https://example.org/resource1?version=1>
   ; rel="memento"
   ; type="text/html"
   ; datetime="Thu, 13 Jun 2019 09:34:33 GMT"
   ; anchor="https://example.org/resource1",
<https://example.org/resource1?version=2>
   ; rel="memento"
   ; type="text/html"
   ; datetime="Sun, 21 Jul 2019 12:22:04 GMT"
   ; anchor="https://example.org/resource1",
<https://authors.example.net/alice>
   ; rel="author"
   ; anchor="https://example.org/resource1#comment=1"
```

Figure 8: Response to HTTP GET includes a set of links

7.2. Set of Links Provided as "application/linkset+json"

Figure 9 shows the client issuing an HTTP GET request against <https://example.org/links/resource1>. In the request, the client uses an "Accept" header field to indicate that it prefers a response in the "application/linkset+json" format.

```
GET links/resource1 HTTP/1.1
Host: example.org
Accept: application/linkset+json
```

Figure 9: Client HTTP GET request expressing preference for an "application/linkset+json" response

Figure 10 shows the response to the HTTP GET request of Figure 9. The set of links is serialized according to the media type "application/linkset+json".

```
HTTP/1.1 200 OK
Date: Mon, 12 Aug 2019 10:46:22 GMT
Server: Apache-Coyote/1.1
Content-Type: application/linkset+json
Link: <https://example.org/links/resource1>
      ; rel="alternate"
      ; type="application/linkset"
Content-Length: 1246

{ "linkset":
  [
    { "anchor": "https://example.org/resource1",
      "author": [
        { "href": "https://authors.example.net/johndoe",
          "type": "application/rdf+xml"
        }
      ],
      "memento": [
        { "href": "https://example.org/resource1?version=1",
          "type": "text/html",
          "datetime": "Thu, 13 Jun 2019 09:34:33 GMT"
        },
        { "href": "https://example.org/resource1?version=2",
          "type": "text/html",
          "datetime": "Sun, 21 Jul 2019 12:22:04 GMT"
        }
      ],
      "latest-version": [
        { "href": "https://example.org/resource1?version=3",
          "type": "text/html"
        }
      ]
    },
    { "anchor": "https://example.org/resource1?version=3",
      "predecessor-version": [
        { "href": "https://example.org/resource1?version=2",
          "type": "text/html"
        }
      ]
    },
    { "anchor": "https://example.org/resource1?version=2",
      "predecessor-version": [
        { "href": "https://example.org/resource1?version=1",
          "type": "text/html"
        }
      ]
    },
    { "anchor": "https://example.org/resource1#comment=1",
      "author": [
        { "href": "https://authors.example.net/alice"}
      ]
    }
  ]
}
```

Figure 10: Response to the client's request for the linkset

7.3. Discovering a Link Set via the "linkset" Link Relation Type

Figure 11 shows a client issuing an HTTP HEAD request against resource <https://example.org/resource1>.

```
HEAD resource1 HTTP/1.1
Host: example.org
```

Figure 11: Client HTTP HEAD request

Figure 12 shows the response to the HEAD request of Figure 11. The response contains an HTTP "Link" header field with a link that has the "linkset" relation type. It indicates that a set of links is provided by resource <https://example.org/links/resource1>, which provides a representation with media type "application/linkset+json".

```
HTTP/1.1 200 OK
Date: Mon, 12 Aug 2019 10:45:54 GMT
Server: Apache-Coyote/1.1
Link: <https://example.org/links/resource1>
      ; rel="linkset"
      ; type="application/linkset+json"
Content-Length: 236
Content-Type: text/html;charset=utf-8
```

Figure 12: Response to HTTP HEAD request

7.4. Link Set Profiles

The examples in this section illustrate the use of the "profile" attribute for a link with the "linkset" link relation type and the "profile" attribute for a link set media type. The examples are inspired by the implementation of link sets by GS1 (the standards body behind many of the world's barcodes).

7.4.1. Using a "profile" Attribute with a "linkset" Link

Figure 13 shows a client issuing an HTTP HEAD request against trade item 09506000134352 at <https://id.gs1.org/01/9506000134352>.

```
HEAD /01/9506000134352 HTTP/1.1
Host: id.gs1.org
```

Figure 13: Client HTTP HEAD request

Figure 14 shows the server's response to the request of Figure 13, including a "linkset" link with a "profile" attribute that has the profile URI <https://www.gs1.org/voc/?show=linktypes> as its value. Dereferencing that URI yields a profile document that lists all the link relation types that a client can expect when requesting the link set made discoverable by the "linkset" link. The link relation types are presented in abbreviated form, e.g., <gs1:activityIdeas>, whereas the actual link relation type URIs are available as hyperlinks on the abbreviations, e.g., <https://www.gs1.org/voc/activityIdeas>. For posterity, that profile document was saved in the Internet Archive at <https://web.archive.org/web/20210927160406/https://www.gs1.org/voc/?show=linktypes> on 27 September 2021.

```
HTTP/1.1 307 Temporary Redirect
Date:  Mon, 27 Sep 2021 16:03:07 GMT
Server: nginx
Link: <https://id.gs1.org/01/9506000134352?linkType=all>
      ; rel="linkset"
      ; type="application/linkset+json"
      ; profile="https://www.gs1.org/voc/?show=linktypes"
Location: https://example.com/risotto-rice-with-mushrooms/
```

Figure 14: Response to the client's HEAD request, including a "profile" attribute for the "linkset" link

7.4.2. Using a "profile" Parameter with a Link Set Media Type

Figure 15 shows a client issuing an HTTP HEAD request against the link set <https://id.gs1.org/01/9506000134352?linkType=all> that was discovered through the HTTP interactions shown in Section 7.4.1.

```
HEAD /01/9506000134352?linkType=all HTTP/1.1
Host: id.gs1.org
```

Figure 15: Client HTTP HEAD request

Figure 16 shows the server's response to the request of Figure 15. Note the "profile" parameter for the "application/linkset+json" media type, which has as its value the same profile URI <https://www.gs1.org/voc/?show=linktypes> as was used in Figure 14.

```
HTTP/1.1 200 OK
Date:  Mon, 27 Sep 2021 16:03:33 GMT
Server: nginx
Content-Type: application/linkset+json;
     profile="https://www.gs1.org/voc/?show=linktypes"
Content-Length: 396
```

Figure 16: Response to the client's HEAD request, including a "profile" parameter for the "application/linkset+json" media type

7.4.3. Using a Link with a "profile" Link Relation Type

Note that the response shown in Figure 16 from the link set resource is equivalent to the response shown in Figure 17, which leverages the "profile" link relation type defined in [RFC6906].

```
HTTP/1.1 200 OK
Date:  Mon, 27 Sep 2021 16:03:33 GMT
Server: nginx
Content-Type: application/linkset+json
Link: <https://www.gs1.org/voc/?show=linktypes>; rel="profile"
Content-Length: 396
```

Figure 17: Response to the client's HEAD request, including a "profile" link

A link with a "profile" link relation type as shown in Figure 17 can also be conveyed in the link set document itself. This is illustrated by Figure 18. Following the recommendation that all links in a link set document

should have an explicit anchor, such a link has the URI of the link set itself as the anchor and the profile URI as the target. Multiple profile URIs are handled by using multiple "href" members.

```
{ "linkset":
  [
    { "anchor": "https://id.gs1.org/01/9506000134352?linkType=all",
      "profile": [
            {"href": "https://www.gs1.org/voc/?show=linktypes"}
      ]
    },
     { "anchor": "https://id.gs1.org/01/9506000134352",
       "https://gs1.org/voc/whatsInTheBox": [
         {"href": "https://example.com/en/packContents/GB"}
       ]
    }
  ]
}
```

Figure 18: A linkset that declares the profile it complies with, using a "profile" link

8. IANA Considerations

8.1. Link Relation Type: linkset

The link relation type below has been registered by IANA in the "Link Relation Types" registry as per Section 4.2 of [RFC8288]:

Relation Name: linkset

Description: The link target of a link with the "linkset" relation type provides a set of links, including links in which the link context of the link participates.

Reference: RFC 9264

8.2. Media Type: application/linkset

The Internet media type "application/linkset" for a linkset encoded as described in Section 4.1 has been registered by IANA in the "Media Types" registry as per [RFC6838].

Type name: application

Subtype name: linkset

Required parameters: N/A

Optional parameters: profile

Encoding considerations: Linksets are encoded according to the definitions provided in [RFC8288]. The encoding discussed in [RFC8288] is based on the general encoding rules specified by HTTP [RFC9110] and allows specific parameters to be extended by the indication of character encoding and language as defined by [RFC8187].

Security considerations: The security considerations of RFC 9264 apply.

Interoperability considerations: N/A

Published specification: RFC 9264

Applications that use this media type: This media type is not specific to any application, as it can be used by any application that wants to interchange Web Links.

Additional information:

Magic number(s): N/A

File extension(s): This media type does not propose a specific extension.

Macintosh file type code(s): TEXT

Person & email address to contact for further information: Erik Wilde <erik.wilde@dret.net>

Intended usage: COMMON

Restrictions on usage: none

Author: Erik Wilde <erik.wilde@dret.net>

Change controller: IESG

8.3. Media Type: application/linkset+json

The Internet media type "application/linkset+json" for a linkset encoded as described in Section 4.2 has been registered by IANA in the "Media Types" registry as per [RFC6838].

Type name: application

Subtype name: linkset+json

Required parameters: N/A

Optional parameters: profile

Encoding considerations: The encoding considerations of [RFC8259] apply.

Security considerations: The security considerations of RFC 9264 apply.

Interoperability considerations: The interoperability considerations of [RFC8259] apply.

Published specification: RFC 9264

Applications that use this media type: This media type is not specific to any application, as it can be used by any application that wants to interchange Web Links.

Additional information: Magic number(s): N/A

File extension(s): JSON documents often use ".json" as the file extension, and this media type does not propose a specific extension other than this generic one.

Macintosh file type code(s):	TEXT
Person & email address to contact for further information:	Erik Wilde <erik.wilde@dret.net>
Intended usage:	COMMON
Restrictions on usage:	none
Author:	Erik Wilde <erik.wilde@dret.net>
Change controller:	IESG

9. Security Considerations

The security considerations of Section 7 of [RFC3986] apply, as well as those of Web Linking [RFC8288] as long as the latter are not specifically discussing the risks of exposing information in HTTP header fields.

In general, links may cause information leakage when they expose information (such as URIs) that can be sensitive or private. Links may expose "hidden URIs" that are not supposed to be openly shared and that may not be sufficiently protected. Ideally, none of the URIs exposed in links should be supposed to be "hidden"; instead, if these URIs are supposed to be limited to certain users, then technical measures should be put in place so that accidentally exposing them does not cause any harm.

For the specific mechanisms defined in this specification, two security considerations should be taken into account:

- The Web Linking model always has an "implicit context", which is the resource of the HTTP interaction. This original context can be lost or can change when self-contained link representations are moved. Changing the context can change the interpretation of links when they have no explicit anchor or when they use relative URIs. Applications may choose to ignore links that have no explicit anchor or that use relative URIs when these are exchanged in standalone resources.
- The model introduced in this specification supports "third-party links", where one party can provide links that have another party's resource as an anchor. Depending on the link semantics and the application context, it is important to verify that there is sufficient trust in that third party to allow it to provide these links. Applications may choose to treat third-party links differently than cases where a resource and the links for that resource are provided by the same party.

10. References

10.1. Normative References

[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>.

[RFC3986] Berners-Lee, T., Fielding, R., and L. Masinter, "Uniform Resource Identifier (URI): Generic Syntax", STD 66, RFC 3986, DOI 10.17487/RFC3986, January 2005, <https://www.rfc-editor.org/info/rfc3986>.

[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>.

[RFC5646] Phillips, A., Ed. and M. Davis, Ed., "Tags for Identifying Languages", BCP 47, RFC 5646, DOI 10.17487/RFC5646, September 2009, <https://www.rfc-editor.org/info/rfc5646>.

[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>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8187] Reschke, J., "Indicating Character Encoding and Language for HTTP Header Field Parameters", RFC 8187, DOI 10.17487/RFC8187, September 2017, <https://www.rfc-editor.org/info/rfc8187>.

[RFC8259] Bray, T., Ed., "The JavaScript Object Notation (JSON) Data Interchange Format", STD 90, RFC 8259, DOI 10.17487/RFC8259, December 2017, <https://www.rfc-editor.org/info/rfc8259>.

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

[RFC9110] Fielding, R., Ed., Nottingham, M., Ed., and J. Reschke, Ed., "HTTP Semantics", STD 97, RFC 9110, DOI 10.17487/RFC9110, June 2022, <https://www.rfc-editor.org/info/rfc9110>.

[W3C.REC-json-ld] Sporny, M., Ed., Kellogg, G., Ed., and M. Lanthaler, Ed., "JSON-LD 1.1: A JSON-based Serialization for Linked Data", W3C Recommendation REC-json-ld-20140116, July 2020, <https://www.w3.org/TR/json-ld/>.

10.2. Informative References

[DCMI-TERMS] Dublin Core Metadata Initiative, "DCMI Metadata Terms", January 2020, <https://www.dublincore.org/specifications/dublin-core/dcmi-terms/>.

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

[RFC6690] Shelby, Z., "Constrained RESTful Environments (CoRE) Link Format", RFC 6690, DOI 10.17487/RFC6690, August 2012, <https://www.rfc-editor.org/info/rfc6690>.

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

[RFC7284] Lanthaler, M., "The Profile URI Registry", RFC 7284, DOI 10.17487/RFC7284, June 2014, <https://www.rfc-editor.org/info/rfc7284>.

[W3C.REC-rdf11-concepts] Cyganiak, R., Ed., Wood, D., Ed., and M. Lanthaler, Ed., "RDF 1.1 Concepts and Abstract Syntax", W3C Consortium Recommendation REC-rdf11-concepts, February 2014, <https://www.w3.org/TR/rdf11-concepts/>.

Appendix A. JSON-LD Context

A set of links rendered according to the JSON serialization defined in Section 4.2 can be interpreted as RDF triples by adding a JSON-LD context [W3C.REC-json-ld] that maps the JSON keys to corresponding Linked Data terms. And, as per Section 6.1 of [W3C.REC-json-ld], when delivering a link set that is rendered according to the "application/linkset+json" media type to a user agent, a server can convey the availability of such a JSON-LD context by using a link with the relation type "http://www.w3.org/ns/json-ld#context" in the HTTP "Link" header field.

Figure 19 shows the response to an HTTP GET against the URI of a link set resource and illustrates this approach to support the discovery of a JSON-LD context. This example is inspired by the GS1 implementation and shows a link set that uses relation types from the GS1 vocabulary at <https://www.gs1.org/voc/> that are expressed as HTTP URIs.

```
HTTP/1.1 200 OK
Date: Mon, 11 Oct 2021 10:48:22 GMT
Server: Apache-Coyote/1.1
Content-Type: application/linkset+json
Link: <https://example.org/contexts/linkset.jsonld>
      ; rel="http://www.w3.org/ns/json-ld#context"
      ; type="application/ld+json"
Content-Length: 1532

{
  "linkset": [
    {
      "anchor": "https://id.gs1.org/01/09506000149301",
      "https://gs1.org/voc/pip": [
        {
          "href": "https://example.com/en/defaultPage",
          "hreflang": [
            "en"
          ],
          "type": "text/html",
          "title": "Product information"
        },
        {
          "href": "https://example.com/fr/defaultPage",
          "hreflang": [
            "fr"
          ],
          "title": "Information produit"
        }
      ],
      "https://gs1.org/voc/whatsInTheBox": [
        {
          "href": "https://example.com/en/packContents/GB",
          "hreflang": [
            "en"
          ],
          "title": "What's in the box?"
        },
        {
          "href": "https://example.com/fr/packContents/FR",
          "hreflang": [
            "fr"
          ],
          "title": "Qu'y a-t-il dans la boite?"
        },
        {
          "href": "https://example.com/fr/packContents/CH",
          "hreflang": [
            "fr"
          ],
          "title": "Qu'y a-t-il dans la boite?"
        }
      ],
      "https://gs1.org/voc/relatedVideo": [
        {
          "href": "https://video.example",
          "hreflang": [
            "en",
            "fr"
          ],
          "title*": [
            {
              "value": "See it in action!",
              "language": "en"
            },
            {
```

In order to obtain the JSON-LD context conveyed by the server, the user agent issues an HTTP GET against the link target of the link with the "http://www.w3.org/ns/json-ld#context" relation type. The response to this GET is shown in Figure 20. This particular JSON-LD context maps "application/linkset+json" representations of link sets to Dublin Core terms [DCMI-TERMS]. Note that the "linkset" entry in the JSON-LD context is introduced to support links with the "linkset" relation type in link sets.

```
HTTP/1.1 200 OK
Content-Type: application/ld+json
Content-Length: 658

{
  "@context": [
    {
      "@version": 1.1,
      "@vocab": "https://gs1.org/voc/",
      "anchor": "@id",
      "href": "@id",
      "linkset": {
        "@id": "@graph",
        "@context": {
          "linkset": "linkset"
        }
      },
      "title": {
        "@id": "http://purl.org/dc/terms/title"
      },
      "title*": {
        "@id": "http://purl.org/dc/terms/title"
      },
      "type": {
        "@id": "http://purl.org/dc/terms/format"
      }
    },
    {
      "language": "@language",
      "value": "@value",
      "hreflang": {
        "@id": "http://purl.org/dc/terms/language",
        "@container": "@set"
      }
    }
  ]
}
```

Figure 20: JSON-LD context mapping to Dublin Core terms

Applying the JSON-LD context of Figure 20 to the link set of Figure 19 allows transforming the "application/linkset+json" link set to an RDF link set. Figure 21 shows the latter represented by means of the "text/turtle" RDF serialization.

```
<https://example.com/en/defaultPage>
        <http://purl.org/dc/terms/format>
        "text/html" .
<https://example.com/en/defaultPage>
        <http://purl.org/dc/terms/language>
        "en" .
<https://example.com/en/defaultPage>
        <http://purl.org/dc/terms/title>
        "Product information" .
<https://example.com/en/packContents/GB>
        <http://purl.org/dc/terms/language>
        "en" .
<https://example.com/en/packContents/GB>
        <http://purl.org/dc/terms/title>
        "What's in the box?" .
<https://example.com/fr/defaultPage>
        <http://purl.org/dc/terms/language>
        "fr" .
<https://example.com/fr/defaultPage>
        <http://purl.org/dc/terms/title>
        "Information produit" .
<https://example.com/fr/packContents/CH>
        <http://purl.org/dc/terms/language>
        "fr" .
<https://example.com/fr/packContents/CH>
        <http://purl.org/dc/terms/title>
        "Qu'y a-t-il dans la boite?" .
<https://example.com/fr/packContents/FR>
        <http://purl.org/dc/terms/language>
        "fr" .
<https://example.com/fr/packContents/FR>
        <http://purl.org/dc/terms/title>
        "Qu'y a-t-il dans la boite?" .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/pip>
        <https://example.com/en/defaultPage> .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/pip>
        <https://example.com/fr/defaultPage> .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/relatedVideo>
        <https://video.example> .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/whatsInTheBox>
        <https://example.com/en/packContents/GB> .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/whatsInTheBox>
        <https://example.com/fr/packContents/CH> .
<https://id.gs1.org/01/09506000149301>
        <https://gs1.org/voc/whatsInTheBox>
        <https://example.com/fr/packContents/FR> .
<https://video.example>
        <http://purl.org/dc/terms/language>
        "en" .
<https://video.example>
        <http://purl.org/dc/terms/language>
        "fr" .
<https://video.example>
        <http://purl.org/dc/terms/title>
        "See it in action!"@en .
<https://video.example>
        <http://purl.org/dc/terms/title>
        "Voyez-le en action!"@fr .
```


Figure 21: RDF serialization of the linkset resulting from applying the JSON-LD context

Acknowledgements

Thanks for comments and suggestions provided by Phil Archer, Dominique Guinard, Mark Nottingham, Julian Reschke, Rob Sanderson, Stian Soiland-Reyes, Sarven Capadisli, and Addison Phillips.

Authors' Addresses

Erik Wilde
Axway
EMail: erik.wilde@dret.net

Herbert Van de Sompel
Data Archiving and Networked Services
EMail: herbert.van.de.sompel@dans.knaw.nl
URI: https://orcid.org/0000-0002-0715-6126