Using JSContact in Registration Data Access Protocol (RDAP) JSON Responses

Abstract

   This document describes an RDAP extension which represents entity
   contact information in JSON responses using JSContact.

Status of This Memo

Copyright Notice

Table of Contents

1. Introduction

This document specifies an extension to the Registration Data Access Protocol (RDAP) that allows RDAP servers to use JSContact [I-D.ietf-jmap-jscontact] to represent the contact information associated with entities in RDAP responses, instead of jCard [RFC7095]. It also describes the process by which an RDAP server can transition from jCard to JSContact. RDAP query and response extensions are defined to facilitate the transition process.

1.1. Rationale

According to the feedback from RDAP Pilot Working Group [RDAP-PILOT-WG], a group of RDAP server implementers representing registries and registrars of generic TLDs, the most commonly raised implementation concern, for both servers and client implementers, related to the use of jCard [RFC7095] to represent the contact information associated with entities. Working Group members reported jCard to be unintuitive, complicated to implement for both clients and servers, and incompatible with best practices for RESTful APIs.

JSContact [I-D.ietf-jmap-jscontact] provides a simpler and more efficient representation for contact information with regard to time and effort saved in processing it. In addition, similarly to jCard, it provides a means to represent internationalised and unstructured contact information. Support for internationalised contact information has been recognised being necessary to facilitate the future internationalisation of registration data directory services.

1.2. Conventions Used in This Document

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 BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

2. JSContact

The JSContact specification defines a data model and JSON representation of contact information that can be used for data storage and exchange in address book or directory applications. It aims to be an alternative to the vCard data format [RFC6350] and to be unambiguous, extendable and simple to process. In contrast with jCard, it is not a direct mapping from the vCard data model and expands semantics where appropriate.

The JSContact specification declares two main object types: "Card", which represents a single contact card, and "CardGroup" which represents a collection of Card objects. For the purpose of this

document, only Card objects are considered. To avoid confusion, in the following of this document, the term "JSCard" is used to refer to "JSContact Card".

JSCard differs from jCard in that it:

  *follows an object-oriented rather than array-oriented approach;

  *is simple to process;

  *requires no extra work in serialization/deserialization from/to a data model;

  *includes no "jagged" arrays;

  *prefers maps rather than arrays to implement collections.

[I-D.ietf-jmap-jscontact-vcard] provides informational guidance on the conversion of jCard into JSCard, and vice versa.

3. Using JSCard objects in RDAP Responses

Entity objects in RDAP responses MAY include a "jscard" property whose value is a JSCard object instead of the "vCardArray" property defined in [RFC9083].

Servers returning the "jscard" property in their response MUST include "jscard_0" in the "rdapConformance" array.

The JSCard "uid" property SHOULD contain the same value as the RDAP "handle" property.

Since most of the JSCard collections are represented as maps, map keys must be defined. To aid interoperability, RDAP providers are RECOMMENDED to use as map keys the following string values and labels defined in [RFC5733]:

  *"org" in the "organizations" map when there is a single <contact:org> element. If both internationalised and localized forms exist, the key MUST be used for the internationalised form;

  *"addr" in the "addresses" map when there is a single <contact:addr> element. If both internationalised and localized forms exist, the key MUST be used for the internationalised form;

  *"email" in the "emails" map for the <contact:email> element;

  *"voice" in the "phones" map for the <contact:voice> element;

  *"fax" in the "phones" map for the <contact:fax> element.

If present, the localized versions of name, organization and postal address MUST be inserted into the "localizations" map. The following is an elided example of an RDAP entity lookup response including a JSCard object that presents the "localizations" map (See PDF for non-ASCII character string).

```
...
"jscard": {
  "@type" : "Card",
  "uid" : "7e0636f5-e48f-4a32-ab96-b57e9c07c7aa",
  "fullName" : "Vasya Pupkin",
  "organizations" : {
    "org" : {
      "@type" : "Organization",
      "name" : "My Company"
    }
  },
  "addresses" : {
    "addr" : {
      "@type" : "Address",
      "street" : [ {
        "@type" : "StreetComponent",
        "type" : "name",
        "value" : "1 Street"
      }, {
        "@type" : "StreetComponent",
        "type" : "postOfficeBox",
        "value" : "01001"
      } ],
      "locality" : "Kyiv",
      "countryCode" : "UA"
    }
  },
  "localizations" : {
    "ua" : {
      "/jscard/addresses/addr" : {
        "@type" : "Address",
        "street" : [ {
          "@type" : "StreetComponent",
          "type" : "name",
          "value" : "1, Улица"
        }, {
          "@type" : "StreetComponent",
          "type" : "postOfficeBox",
          "value" : "01001"
        } ],
        "locality" : "Киев",
        "countryCode" : "UA"
      },
      "/jscard/fullName" : "Вася Пупкин",
      "/jscard/organizations/org" : {
        "@type" : "Organization",
        "name" : "Моя Компания"
      }
    }
```

```
    }
  }
  ...
```

Figure 1: Example of handling localizations in JSContact

Implementers MAY use different mapping schemes to define keys for additional entries of the aforementioned maps or others. For example, a mapping scheme may consist in using a trivial sequential number (e.g. "url-1", "url-2", etc.)

The following is an example of an RDAP entity including a JSCard object that has been converted from the example in section 5.1 of [RFC9083].

```json
{
    "rdapConformance": [
        "rdap_level_0",
        "jscard_0"
    ],
    "objectClassName" : "entity",
    "handle":"XXXX",
    "jscard":{
      "@type": "Card",
      "uid": "XXXX",
      "fullName": "Joe User" ,
      "name": {
        "components": [
          {
            "@type": "NameComponent",
            "type": "surname",
            "value": "User"
          },
          {
            "@type": "NameComponent",
            "type": "personal",
            "value": "Joe"
          },
          {
            "@type": "NameComponent",
            "type": "suffix",
            "value": "ing. jr"
          },
          {
            "@type": "NameComponent",
            "type": "suffix",
            "value": "M.Sc."
          }
        ]
      },
      "kind": "individual",
      "preferredContactLanguages": {
        "fr": {
          "@type": "ContactLanguage",
          "pref": 1
        },
        "en": {
          "@type": "ContactLanguage",
          "pref": 2
        }
      },
      "organizations": {
        "org": {
          "@type": "Organization",
```

```
          "name": "Example"
        }
      },
      "titles": {
        "title": {
          "@type": "Title",
          "title": "Research Scientist"
        },
        "role": {
          "@type": "Title",
          "title": "Project Lead"
        }
      },
      "addresses": {
        "addr": {
          "@type": "Address",
          "contexts": {
            "work": true
          },
          "street": [
            {
              "@type": "StreetComponent",
              "type": "name",
              "value": "4321 Rue Somewhere"
            },
            {
              "@type": "StreetComponent",
              "type": "extension",
              "value": "Suite 1234"
            }
          ],
          "locality": "Quebec",
          "region": "QC",
          "postcode": "G1V 2M2",
          "country": "Canada",
          "coordinates": "geo:46.772673,-71.282945",
          "timeZone": "Etc/GMT+5"
        },
        "home": {
          "@type": "Address",
          "contexts": {
            "private": true
          },
          "fullAddress": "123 Maple Ave\nSuite 90001\nVancouver\nBC\n1
        }
      },
      "phones": {
        "voice" : {
          "@type": "Phone",
```

```
        "contexts": {
          "work": true
        },
        "features": {
          "voice": true,
          "cell": true,
          "video": true,
          "text": true
        },
        "pref": 1,
        "phone": "tel:+1-555-555-1234;ext=102"
      }
    },
    "emails": {
      "email": {
        "@type": "EmailAddress",
        "contexts": {
          "work": true
        },
        "email": "joe.user@example.com"
      }
    },
    "online": {
      "key": {
        "@type" : "Resource",
        "contexts": {
          "work": true
        },
        "type": "uri",
        "label": "key",
        "resource": "http://www.example.com/joe.user/joe.asc"
      },
      "url": {
        "@type" : "Resource",
        "contexts": {
          "private": true
        },
        "type": "uri",
        "label": "url",
        "resource": "http://example.org"
      }
    }
  },
  "roles":[ "registrar" ],
  "publicIds":[
    {
      "type":"IANA Registrar ID",
      "identifier":"1"
    }
```

```
    ],
    "remarks":[
      {
        "description":[
          "She sells sea shells down by the sea shore.",
          "Originally written by Terry Sullivan."
        ]
      }
    ],
    "links":[
      {
        "value":"http://example.com/entity/XXXX",
        "rel":"self",
        "href":"http://example.com/entity/XXXX",
        "type" : "application/rdap+json"
      }
    ],
    "events":[
      {
        "eventAction":"registration",
        "eventDate":"1990-12-31T23:59:59Z"
      }
    ],
    "asEventActor":[
      {
        "eventAction":"last changed",
        "eventDate":"1991-12-31T23:59:59Z"
      }
    ]
}
```

Figure 2: Example of using JSContact in RDAP response

3.1. RDAP Query Parameters

Two new query parameters are defined for the purpose of this document.

The query parameters are OPTIONAL extensions of path segments defined in [RFC9082]. They are as follows:

  *"jscard": a boolean value that allows a client to request the "jscard" property in the RDAP response;

  *"jcard": a boolean value that allows a client to request the "vcardArray" property in the RDAP response.

These parameters are furtherly explained in Section 4.

4. Transition Considerations

4.1. RDAP Features Supporting a Transition Process

4.1.1. Notices and Link Relationships

RDAP allows servers to communicate service information to clients through notices. According to Section 4.3 of [RFC9083], an RDAP response may contain one or more notice objects. Each notice may include a set of link objects, which can be used to provide clients with references and documentation. These link objects may have a "rel" property which defines the relationship type, as described in [RFC8288], Section 4. The transition process outlined in this document uses two link relation types, namely "related" and "alternate", described in [RFC8288].

4.1.2. rdapConformance Property

The information about the specifications used in the construction of the response is also described by the strings which appear in the "rdapConformance" property of the RDAP response.

4.1.3. Query Parameters

Clients can ask servers to use the query parameters defined in Section 3.1 in accordance with [RFC9082].

4.2. Transition Procedure

The principles of the procedure for jCard to JSCard transition are based on the best practices in [API-DEPRECATION].

The procedure consists of four contiguous stages. During the procedure, the presence of "jscard_0" tag in the rdapConformance array indicates that JSCard is returned instead of jCard. The date and time format used to notify clients about the stages of this procedure is defined in [RFC3339].

4.2.1. Goals

The procedure described in this document aims to achieve the following goals:

  *only one contact representation would be included in the response;

  *the response would always be compliant to [RFC9083] because:

    -being the "jscard" property a response extension, its presence would be signaled by the "jscard_0" conformance tag;

    -being "vcardArray" property optional in a response, its absence would be allowed;

  *clients would be informed about the transition timeline;

  *the backward compatibility would be guaranteed throughout the transition;

  *servers and clients could execute their transitions independently.

4.2.2. Transition Stages

4.2.2.1. Stage 1: only jCard provided

This stage corresponds to providing jCard as the default contact card [RFC9083]. The RDAP server is not able to provide an alternate contact card. The rdapConformance array MUST NOT contain the "jscard_0" tag.

4.2.2.2. Stage 2: jCard sunset

During this stage, the server uses jCard by default, but the RDAP server will return JSCard if the client sets the query parameter "jscard" to 1/true/yes. The rdapConformance array MUST contain the "jscard_0" tag if JSCard is returned.

From this stage on, the RDAP server MUST include the "jscard_0" tag in the rdapConformance array of the help response to signal clients that JSCard can be returned instead of jCard.

The RDAP server SHOULD include a notice titled "jCard sunset end".
Such a notice includes a description reporting the jCard sunset end
date and time and two OPTIONAL links:

   *"related": a link to a URI-identified resource documenting the
    transition procedure;

   *"alternate": if JSCard is not requested, a link to an alternate
    result view identified by the current query string plus the
    parameter "jscard" set to 1/true/yes (Figure 3); otherwise, only
    the "related" link can be provided (Figure 4).

```
"notices": [
  {
    "title": "jCard sunset end",
    "description": ["2022-07-01T00:00:00Z"],
    "links": [{
        "value": "http://example.net/entity/XXXX",
        "rel": "related",
        "type": "text/html",
        "href": "http://www.example.com/jcard_deprecation.html"
      },
      {
        "value": "http://example.net/entity/XXXX",
        "rel": "alternate",
        "type": "application/rdap+json",
        "href": " http://example.net/entity/XXXX?jscard=1"
      }
    ]
  }
]
```

                 Figure 3: jCard sunset - JSCard not requested

```
"notices": [
  {
    "title": "jCard sunset end",
    "description": ["2022-07-01T00:00:00Z"],
    "links": [
      {
        "value": "http://example.net/entity/XXXX?jscard=1",
        "rel": "related",
        "type": "text/html",
        "href": "http://www.example.com/jcard_deprecation.html"
      }
    ]
  }
]
```

Figure 4: jCard sunset - JSCard requested

4.2.2.3. Stage 3: jCard deprecation

This stage corresponds to the provisioning of JSCard by default, but
the RDAP will return jCard if the client sets the query parameter
"jcard" to 1/true/yes. The rdapConformance array contains the
"jscard_0" tag unless jCard is returned. The "jscard" query
parameter MUST be ignored.

The RDAP server SHOULD return a notice titled "jCard deprecation
end". Such a notice includes a description reporting the jCard
deprecation end date and time and two OPTIONAL links:

  *"related": a link to a URI-identified resource documenting the
   transition procedure;

  *"alternate": if jCard is not requested, a link to an alternate
   result view identified by the current query string plus the
   parameter "jcard" set to 1/true/yes (Figure 5); otherwise, a link
   to the result view identified by the current query string without
   the parameter "jcard" (Figure 6).


```
"notices": [
  {
    "title": "jCard deprecation end",
    "description": ["2022-12-31T23:59:59Z"],
    "links": [
      {
        "value": "http://example.net/entity/XXXX",
        "rel": "related",
        "type": "text/html",
        "href": "http://www.example.com/jcard_deprecation.html"
      },
      {
        "value": "http://example.net/entity/XXXX",
        "rel": "alternate",
        "type": "application/rdap+json",
        "href": " http://example.net/entity/XXXX?jcard=1"
      }
    ]
  }
]
```

Figure 5: jCard deprecation - jCard not requested

```
"notices": [
  {
    "title": "jCard deprecation end",
    "description": ["2022-12-31T23:59:59Z"],
    "links": [
      {
        "value": "http://example.net/entity/XXXX?jcard=1",
        "rel": "related",
        "type": "text/html",
        "href": "http://www.example.com/jcard_deprecation.html"
      },
      {
        "value": "http://example.net/entity/XXXX?jcard=1",
        "rel": "alternate",
        "type": "application/rdap+json",
        "href": " http://example.net/entity/XXXX"
      }
    ]
  }
]
```

Figure 6: jCard deprecation - jCard requested

4.2.2.4. Stage 4: jCard deprecated

This stage corresponds to providing JSCard as default contact card.
The RDAP server is not able to provide an alternate contact card.
The rdapConformance array always contains "jscard_0" tag. The RDAP
server doesn't include any notice about the jCard deprecation
process. Both "jscard" and "jcard" query parameters MUST be ignored.

4.2.2.5. Length

The length of both jCard sunset and jCard deprecation periods are
not fixed by this specification. Best practices in REST API
deprecation suggest that, depending on the deprecated API's reach,
user base and service offering, a convenient time could be anywhere
between 3 - 8 months. Anyway, RDAP providers are RECOMMENDED to
monitor the server log to figure out whether declared times need to
be changed to meet client requirements.

5. Implementation Status

NOTE: Please remove this section and the reference to RFC 7942 prior
to publication as an RFC.

This section records the status of known implementations of the
protocol defined by this specification at the time of posting of
this Internet-Draft, and is based on a proposal described in RFC
7942 [RFC7942]. The description of implementations in this section

is intended to assist the IETF in its decision processes in
progressing drafts to RFCs. Please note that the listing of any
individual implementation here does not imply endorsement by the
IETF. Furthermore, no effort has been spent to verify the
information presented here that was supplied by IETF contributors.
This is not intended as, and must not be construed to be, a catalog
of available implementations or their features. Readers are advised
to note that other implementations may exist.

According to RFC 7942, "this will allow reviewers and working groups
to assign due consideration to documents that have the benefit of
running code, which may serve as evidence of valuable
experimentation and feedback that have made the implemented
protocols more mature. It is up to the individual working groups to
use this information as they see fit".

5.1. IIT-CNR/Registro.it RDAP Server

   *Responsible Organization: Institute of Informatics and Telematics
    of National Research Council (IIT-CNR)/Registro.it

   *Location: https://rdap.pubtest.nic.it/

   *Description: This implementation includes support for RDAP
    queries using data from the public test environment of .it ccTLD.

   *Level of Maturity: This is an "alpha" test implementation.

   *Coverage: This implementation includes all of the features
    described in this specification.

   *Contact Information: Mario Loffredo, mario.loffredo@iit.cnr.it

5.2. IIT-CNR/Registro.it RDAP Client

   *Responsible Organization: Institute of Informatics and Telematics
    of National Research Council (IIT-CNR)/Registro.it

   *Location: https://web-rdap.pubtest.nic.it/

   *Description: This is a Javascript web-based RDAP client. RDAP
    responses are retrieved from RDAP servers by the browser, parsed
    into an HTML representation, and displayed in a format improving
    the user experience. RDAP responses containing JSCard objects are
    handled identically to those containing jCard objects. Raw
    versions of RDAP responses including either jCard or JSCard
    objects are provided.

   *Level of Maturity: This is an "alpha" test implementation.

*Coverage: This implementation includes all of the features
   described in this specification.

  *Contact Information: Francesco Donini,
   francesco.donini@iit.cnr.it

5.3. client.rdap.org

  *Location: https://client.rdap.org/

  *Description: This is a web-based "single page" RDAP client. RDAP
   responses are retrieved from RDAP servers by the browser, and
   parsed into an HTML representation. RDAP responses containing
   JSCard objects are handled identically to those containing jCard
   objects.

  *Level of Maturity: This is an "alpha" test implementation.

  *Coverage: This implementation implements client support for
   parsing JSCard objects in RDAP responses.

  *Contact Information: Gavin Brown, feedback@rdap.org

5.4. CentralNic Registry

  *Responsible Organization: CentralNic Group PLC

  *Location: https://rdap.centralnic.com/{tld}

  *Description: This server is the product RDAP service for all top-
   level domains on the CentralNic registry platform.

  *Level of Maturity: Production quality.

  *Coverage: This implementation includes all of the features
   described in this specification.

  *Contact Information: support@centralnic.com

6. IANA Considerations

IANA is requested to register the following values in the RDAP
Extensions Registry:

  *Extension identifier: jscard_0

  *Registry operator: Any

  *Published specification: This document.

  *Contact: IETF <iesg@ietf.org>

*Intended usage: This extension represents a contact card provided
  in an RDAP response according to the JSContact specification [I-
  D.ietf-jmap-jscontact].

7. Security Considerations

   Unlike jCard, the formatted name as well as any other personally
   identifiable information is not required in JSCard. The only
   mandatory property, namely "uid", is not a sensitive information as
   it happens, instead, for the "fn" property in jCard. Therefore,
   redacted properties can be merely excluded without using placeholder
   values. This means that, with reference to what is described in [I-
   D.ietf-regext-rdap-redacted], only the "Removal" method can be used
   for redacting JSContact properties whereas the "Empty Value" is also
   used for redacting jCard.

8. Acknowledgements

   The authors would like to acknowledge the following individuals for
   their contributions to this document: Jasdip Singh and Francesco
   Donini.

9. References

9.1. Normative References

   [I-D.ietf-jmap-jscontact] Stepanek, R. and M. Loffredo, "JSContact:
             A JSON representation of contact data", Work in Progress,
             Internet-Draft, draft-ietf-jmap-jscontact-08, 20 October
             2021, <https://www.ietf.org/archive/id/draft-ietf-jmap-
             jscontact-08.txt>.

   [I-D.ietf-jmap-jscontact-vcard] Loffredo, M. and R. Stepanek,
             "JSContact: Converting from and to vCard", Work in
             Progress, Internet-Draft, draft-ietf-jmap-jscontact-
             vcard-07, 20 October 2021, <https://www.ietf.org/archive/
             id/draft-ietf-jmap-jscontact-vcard-07.txt>.

   [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>.

   [RFC3339] Klyne, G. and C. Newman, "Date and Time on the Internet:
             Timestamps", RFC 3339, DOI 10.17487/RFC3339, July 2002,
             <https://www.rfc-editor.org/info/rfc3339>.

   [RFC5733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)
             Contact Mapping", STD 69, RFC 5733, DOI 10.17487/RFC5733,
             August 2009, <https://www.rfc-editor.org/info/rfc5733>.

[RFC6350]   Perreault, S., "vCard Format Specification", RFC 6350,
             DOI 10.17487/RFC6350, August 2011, <https://www.rfc-
             editor.org/info/rfc6350>.

[RFC7095]   Kewisch, P., "jCard: The JSON Format for vCard", RFC
             7095, DOI 10.17487/RFC7095, January 2014, <https://
             www.rfc-editor.org/info/rfc7095>.

[RFC7942]   Sheffer, Y. and A. Farrel, "Improving Awareness of
             Running Code: The Implementation Status Section", BCP
             205, RFC 7942, DOI 10.17487/RFC7942, July 2016, <https://
             www.rfc-editor.org/info/rfc7942>.

[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>.

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

[RFC9082]   Hollenbeck, S. and A. Newton, "Registration Data Access
             Protocol (RDAP) Query Format", STD 95, RFC 9082, DOI
             10.17487/RFC9082, June 2021, <https://www.rfc-editor.org/
             info/rfc9082>.

[RFC9083]   Hollenbeck, S. and A. Newton, "JSON Responses for the
             Registration Data Access Protocol (RDAP)", STD 95, RFC
             9083, DOI 10.17487/RFC9083, June 2021, <https://www.rfc-
             editor.org/info/rfc9083>.

9.2. Informative References

[API-DEPRECATION] Sandoval, K., "How to Smartly Sunset and Deprecate
             APIs", August 2019, <https://web.archive.org/web/
             20200417084255/https://nordicapis.com/how-to-smartly-
             sunset-and-deprecate-apis/>.

[I-D.ietf-regext-rdap-redacted] Gould, J., Smith, D., Kolker, J.,
             and R. Carney, "Redacted Fields in the Registration Data
             Access Protocol (RDAP) Response", Work in Progress,
             Internet-Draft, draft-ietf-regext-rdap-redacted-02, 18
             November 2021, <https://www.ietf.org/archive/id/draft-
             ietf-regext-rdap-redacted-02.txt>.

[RDAP-PILOT-WG] ICANN RDAP Pilot WG, "RDAP Pilot Report", April
             2019, <https://www.icann.org/en/system/files/files/rdap-
             pilot-report-25apr19-en.pdf>.

Appendix A. Change Log

A.1. Change from 00 to 01

    1. Changed category from "Best Current Practice" to "Standards
       Track"

    2. Replaced the example of [Figure 2](#)

    3. Changed the title of the "Migration from JCard to JSCard"
       section to "Transition Considerations"

    4. Added [Section 3.1](#)

    5. Updated [Section 6](#)

    6. Updated [Section 7](#)

    7. Rearranged the description of stage 1 in [Section 4.2.2](#)

    8. Changed the names of the transition stages 1 and 2

    9. Corrected [Figure 3](#), [Figure 5](#), [Figure 6](#)

    10. Changed the rdapConformance tag "jscard_level_0" to "jscard"

    11. Removed the "Best Practices for deprecating a REST API
        features" section, but added a useful reference.

A.2. Change from 01 to 02

    1. Removed the sentence "which cannot be represented using jCard"
       in [Section 1.1](#).

A.3. Change from 02 to 03

    1. Updated section "Conventions Used in This Document".

    2. Updated the contact in "IANA Considerations" section.

    3. Changed the reference draft-loffredo-jmap-jscontact-vcard to
       draft-ietf-jmap-jscontact-vcard.

    4. Added reference to RFC8174.

    5. Other minor edits.

A.4. Change from 03 to 04

    1. Updated the reference draft-dalal-deprecation-header to draft-
       ietf-httpapi-deprecation-header.

A.5. Initial WG version

1. Ported from draft-loffredo-regext-rdap-jcard-deprecation-04
   renamed to draft-ietf-regext-rdap-jscontact-00.

A.6. Change from 00 to 01

1. Updated [Section 3](#) and [Figure 2](#).

A.7. Change from 01 to 02

1. Updated [Section 2](#) and [Figure 2](#).

A.8. Change from 02 to 03

1. Replaced references to obsolete RFC7482 and RFC7483 with
   RFC9082 and RFC9083.

2. Updated [Section 3](#) and [Figure 2](#).

A.9. Change from 03 to 04

1. Changed the references to Internet Drafts.

2. Added an example showing how localizations are treated in
   JSContact.

3. Changed the position of section "Goals" in [Section 4.2](#).

4. Added three more implementations to [Section 5](#).

5. Changed the rdapConformance tag "jscard" to "jscard_0"

6. Added clarifications addressing the feedback provided by Jasdip
   Singh about version -03.

7. Added [Section 8](#).

8. Other minor edits.

A.10. Change from 04 to 05

1. Updated [Figure 2](#) to make it compliant with draft-ietf-jmap-
   jscontact-09.

A.11. Change from 05 to 06

1. Reviewed the notices presented in [Section 4.2.2.2](#) and [Section
   4.2.2.3](#).

A.12. Change from 06 to 07

1. Corrected the JSON Pointer expressions in Figure 1.

2. Other minor edits.

A.13. Change from 07 to 08

1. Corrected a nit in Figure 1.

2. Removed the reference to draft-ietf-httpapi-deprecation-header.

3. Replaced the "deprecation" link relation type with "related".

4. Moved the reference to JSContact drafts to the "Normative References" section.

Authors' Addresses

Mario Loffredo
IIT-CNR/Registro.it
Via Moruzzi,1
56124 Pisa
Italy

Email: mario.loffredo@iit.cnr.it
URI: http://www.iit.cnr.it

Gavin Brown
CentralNic Group plc
Saddlers House, 44 Gutter Lane
London
EC2V 6BR
United Kingdom

Phone: +44 20 33 88 0600
Email: gavin.brown@centralnic.com
URI: https://www.centralnic.com