Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: 3 September 2026

P. Kowalik
DENIC
M. Wullink
SIDN Labs

2 March 2026

RESTful Provisioning Protocol (RPP) Data Objects
draft-kowalik-rpp-data-objects-03

Abstract

This document defines data objects for the RESTful Provisioning Protocol (RPP) and sets up IANA RPP Data Object Registry to describe and catalogue them. Specifically, it details the logical structure, constraints, and protocol operations (including their inputs, outputs and business logic) for foundational resources: domain names, contacts, and hosts. In accordance with the RPP architecture [I-D.kowalik-rpp-architecture], these definitions focus entirely on the semantics, remaining independent of any specific data representation or media type (e.g., JSON or XML).

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at https://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on 3 September 2026.

Copyright Notice

Copyright (c) 2026 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) 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.

Table of Contents

1. Introduction

The RESTful Provisioning Protocol (RPP) defines a set of data objects used to represent and manage foundational registry resources, including domain names, contacts, and hosts. This initial list is not exhaustive; additional resource and component objects MAY be defined in future revisions or introduced via IANA registration to support new features and operational needs.

In accordance with the RPP architecture [I-D.kowalik-rpp-architecture], a core architectural principle is the clear distinction between the abstract data model and its concrete data representation. The data model defines the logical structure, relationships, and constraints of the objects, independent of formatting. The data representation defines how these abstract concepts are expressed in specific formats (e.g., JSON, XML, or YAML).

This document focuses on the data model of RPP objects and operations on them, including the data model of operation inputs, outputs as well as the necessary business logic of state transitions. This separation of concerns ensures the protocol maintains a stable semantic foundation that can be consistently implemented across different media types and easily adapted to new representation formats. For instance, the model defines a contact's name as a required string type, but it remains agnostic as to whether that string is ultimately encoded as a JSON property or an XML element.

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

The following terms related to the relationship between host objects and domain objects are used as defined in Section 1.1 of [RFC5732]:

* Subordinate host: A host name object that has a subordinate relationship to a superordinate domain name object. For example, host name "ns1.example.com" has a subordinate relationship to domain name "example.com".
* Superordinate domain: A domain name object to which a host name object is subordinate.
* Internal host: A host whose name belongs to the namespace of the repository in which the host is being used for delegation purposes.
* External host: A host whose name does not belong to the namespace of the repository in which the host is being used for delegation purposes.

The following terms are defined and used in this document

Object Authorisation Authorisation data related to the object beyond the default client-level authorisation accompanying the request, in order to authorise operations on the object. Typically it is a value related to or derived from an Authorisation Information Object provisioned with the object.

Data Object A top-level provisioned resource object that has an independent lifecycle and identity in the registry. Data Objects carry data elements describing the resource and define the set of operations that can be performed on them.

Component Object A reusable data structure that carries data only, with no operations of its own. Component Objects are embedded within Data Objects or other objects to avoid repetition of common data patterns.

Process Object An object that represents a long-running or multi-step operation initiated on a Data Object. Process Objects carry operation-related state and data, and may define their own operations to interact with the process. They have no independent existence - their lifecycle is bound to the owning Data Object.

Owner Data Object A Data Object which a process (represented as Process Object) was initiated upon and which owns this Process Object

2. Resource Definition Principles

2.1. Primitive Data Types

RPP data elements use strict typing, meaning that each element must conform exactly to its declared primitive data type, and type violations MUST be treated as errors by implementations. The exact specifications for these types, including allowed ranges, encoding, and formatting, are determined by the representation format used (e.g., JSON, XML, CBOR). New RPP non-primitive data types based on existing primitive data types MAY be defined to support additional features.

2.1.1. String

A String is a sequence of Unicode characters. Usages MAY impose additional constraints on string values, such as maximum length or allowed character sets, based on specific data element definitions. An example of a string is "host.example".

2.1.2. Integer

An Integer is a whole number, positive or negative. Usages MAY impose additional constraints on integer values, such as minimum and maximum allowable values, based on specific data element definitions. An example of an integer is 42.

2.1.3. Boolean

A Boolean represents a logical true or false value. An example of a boolean is true or false, but it MAY be different depending on the native encoding of boolean values in the representation.

2.1.4. Decimal

Decimal is a number providing an exact, base-10 representation of fractional values within a defined precision. Usage of this type MUST impose additional constraints on decimal values, such as precision or range, based on specific data element definitions. An example of a decimal is 3.14159.

2.1.5. Date

A Date is a full-date calendar date as described in [RFC3339], an example of a date is 2025-10-27.

2.1.6. Timestamp

Timestamp (Date and time attribute) values MUST be represented in Universal Coordinated Time (UTC) using the Gregorian calendar using date-time form as defined in [RFC3339]. In EPP Compatibility Profile upper case "T" and "Z" characters MUST be used. An example of a timestamp is 2025-10-27T09:42:51Z.

2.1.7. URL

A Uniform Resource Locator (URL) as defined in [RFC1738]. An example of a URL is "https://host.example".

2.1.8. Binary

Raw binary data, implementations MAY choose how to encode the binary data, for example as base64 or hexadecimal string. An example of binary data encoded as base64 is "UlBQIFNheXMgSGk=".

2.1.9. Dictionary

Notation: Dictionary[Value Type]

A Dictionary is a collection of key-value pairs where keys are unique Strings and values are of the specified primitive data type. Usages MUST define the constraints on allowed keys and values. A Dictionary differs from an Aggregation Dictionary in that it maps String keys to primitive values rather than to Component or Resource Objects.

2.1.10. Object

An Object is a composite structure containing named properties. The set of allowed property names, their data types, and their constraints are determined by the data element definition that uses this type. Usages MUST specify the expected structure, including any required or optional properties. An Object differs from a Component Object in that it is defined inline as part of a data element rather than being a standalone reusable definition registered separately.

2.2. Data Element Abstraction

Each data object is composed of logical data elements. A data element is a logical unit of information identified by a stable name, independent of its representation in any given media type. The definition for each element specifies its logical name, purpose, cardinality, data type, and constraints. The data type of a data element may also be a reference to another data object, using the target object’s stable name.

2.3. Extensibility

The RPP data model is designed to be extensible. Extensions MAY introduce new data objects, add new data elements or associations to existing objects, and define new operations or extend the inputs and outputs of existing operations with additional transient data elements.

2.3.1. Standardised and Private Extensions

RPP distinguishes between standardised and private extensions:

* Standardised extensions MUST be registered with IANA, as described in the IANA Considerations section of this document and in [I-D.ietf-rpp-architecture]. Registration ensures global uniqueness of extension identifiers and promotes interoperability across implementations.

* Private (non-standardised) extensions MAY be defined for use within specific implementations or organisations. Private extensions are not required to be registered with IANA, but MUST still use unique identifiers that are unlikely to conflict with standardised extensions or other private extensions. The use of reverse domain notation as a prefix (e.g., org.example.rpp.myElement) is RECOMMENDED for private extension identifiers to avoid naming collisions.

2.3.2. Extension Points

The following aspects of the data model are extensible:

* New data objects: Additional resource or component objects MAY be defined by extensions and registered with IANA.
* New data elements: Extensions MAY add new data elements to existing data objects. Such elements follow the same Data Element Semantics as core data elements.
* New associations: Extensions MAY introduce new associations between existing or new objects.
* New operations: Extensions MAY define entirely new operations on existing or new data objects.
* Extended operation inputs and outputs: Extensions MAY add new transient data elements to the inputs or outputs of existing operations.

When an extension adds data elements or transient parameters to a core object or operation, these additions MUST NOT alter the semantics or constraints of existing core data elements.

2.4. Data Element Semantics

The definition of each data element within an object consists of the following attributes:

* Name: A human-readable name for the data element.
* Identifier: A machine-readable, unique identifier for the element, using camelCase notation.
* Cardinality: Specifies the number of times an element may appear. The notation is as follows:
  - 1 for exactly one
  - 0-1 for zero or one
  - 0+ for zero or more
  - and 1+ for one or more
* Data Type: Defines the element’s data structure, which can be a primitive type, a Common Data Type or a reference to another component object.
* Description: Explains the purpose of the data element and any other relevant information.
* Constraints: Provides specific validation rules or limitations on top of the data type itself, such as value ranges.
* Mutability: Defines the lifecycle of the data element’s value. It MUST be one of the following:
  - create-only: The element’s value is provided during the object’s creation and cannot be modified thereafter.
  - read-only: The element’s value is managed by the server. It cannot be set or modified directly by the client, though it may change as a result of server-side operations.
  - read-write: The element’s value can be set and modified by the client.

2.5. Operations

For each data object a set of possible operations is defined together with their respective input and output data.

2.5.1. Authorisation

For each operation authorisation requirements and operation behaviour is specified.

Wherever "Object Authorisation" is mentioned, it means that an operation MAY accept or require additional authorisation data related to the object beyond default client-level authorisation, or that an operation MAY result in different processing or response if such authorisation is provided.

2.5.2. Uniform interface

For the typical set of Create, Read, Update and Delete operations the following set of input and output data model is specified on top of additional transient input data, unless an operation for the specific object tells otherwise.

2.5.2.1. Create

* Input: Object (create-only and read-write properties)
* Output: Object (read-write and read-only properties)

2.5.2.2. Read

* Input: Object identifier
* Output: Object (read-write and read-only properties)

The output Object MAY vary depending on the identity of the querying client, use of Object Authorisation information, and server policy towards unauthorised clients.

If the querying client is the sponsoring client, all available information MUST be returned.

If the querying client is not the sponsoring client but the client provides valid Object Authorisation information, all available information SHOULD be returned, however some optional elements MAY be reserved to the sponsoring client only.

If the querying client is not the sponsoring client and the client does not provide valid Object Authorisation information, server policy determines which OPTIONAL elements are returned, if any, or whether the entire request is rejected.

2.5.2.3. Update

* Input: Object identifier, Object changes (read-write properties)
* Output: Object (read-write and read-only properties)

2.5.2.4. Delete

* Input: Object identifier
* Output: Object (read-write and read-only properties) or nothing

2.5.3. Operations beyond uniform interface

For all other operations both input and output have to be fully specified.

2.5.4. Transfer Operations

Transfer operations manage the change of sponsorship of a provisioned object from one client (the sponsoring client) to another (the gaining client). They are specified once in this section as the transfer model is common across all transferable resource objects. Individual object definitions reference this section and specify any object-specific extensions to the common pattern.

RPP supports two types of transfer:

* Pull Transfer: Initiated by the gaining client. The gaining client MUST provide valid Object Authorisation to initiate the request.
* Push Transfer: Initiated by the sponsoring client, who designates a gaining client in the request.

The transfer process MAY be immediate or follow a multi-step workflow depending on server policy. If the server implements immediate transfers, the Approve and Reject operations need not be supported; the server completes the transfer upon receipt of the Create operation.

The server MAY implement local policies to prevent transfers from stalling and implement a form of automated transfer escalation, approval or cancellation when such a stalled process is recognised.

All transfer operations act on or return the Transfer Process Object and are executed in the context of Owner Data Object the operation is created upon.

> TODO: The server MUST notify the current sponsoring client of a pending transfer request. The notification mechanism is not defined in this document.
>
> TODO: Transfer-specific error conditions (object not eligible for transfer, object pending transfer, object not pending transfer) are not defined in this document.

2.5.5. Restore Operations

Restore operations manage the recovery of an object that has entered the Redemption Grace Period (RGP). They are OPTIONAL and are only available when the RGP feature is supported by the server and MAY be supported only for a subset of Data Object types.

The RGP process MAY involve two distinct steps:

* Restore Request: (REQUIRED) Initiates the recovery of an object in the redemptionPeriod state, signalling to the registry the intent to restore. This corresponds to the Create operation on the Restore Process Object. On success, the object transitions to pendingRestore.
* Restore Report: (OPTIONAL) Submits a report documenting the circumstances of the deletion and restoration, as required by registry policy. This corresponds to the Report operation on the Restore Process Object. On success, the object is returned to its pre-deletion status and all RGP status labels are removed.

Whether a restore report is required after a restore request is a matter of server policy. If the server does not require a restore report, the object returns to its pre-deletion status immediately upon a successful Create operation, bypassing the pendingRestore state.

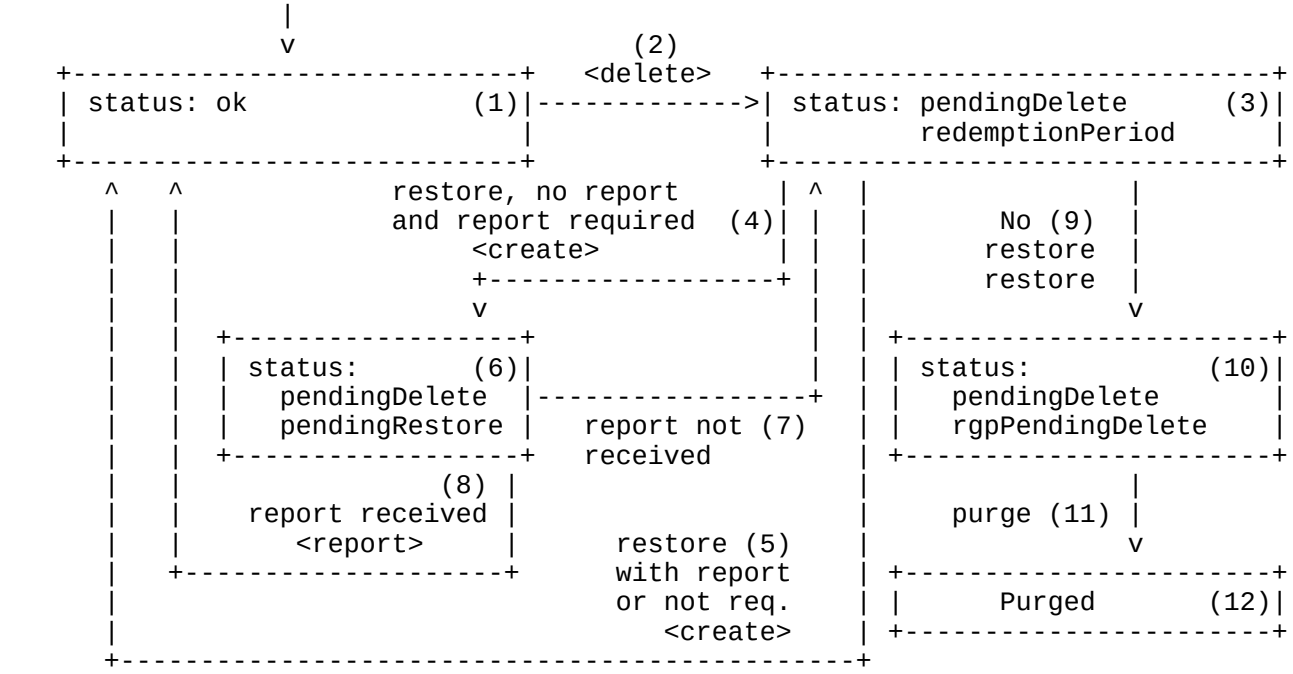
The Create operation MAY include the restore report inline to complete both steps atomically in a single operation.

All restore operations act on or return the Restore Process Object and are executed in the context of Owner Data Object the operation is created upon.

2.5.5.1. Redemption Grace Period State Diagram

The following state diagram describes the object lifecycle when the Redemption Grace Period (RGP) feature is supported. It adapts the diagram from Section 2 of [RFC3915] to the RPP data model, using RPP status labels and operations instead of EPP command names.

In the diagram below, RPP status labels are shown in the status field of the object. Standard EPP-origin status labels (e.g., ok, pendingDelete) are used alongside the RGP-specific labels defined in this document. The <create> and <report> labels refer to the Create and Report operations on the Restore Process Object, replacing the EPP extended <update> command with op=request and op=report attributes.



State descriptions:

1. The object is in normal operation (ok or other status allowing a delete operation).
2. A delete operation is received and processed.
3. RGP begins. The object enters pendingDelete + redemptionPeriod state. The object remains here until a restore operation is requested or the redemption period elapses.
4. A restore Create operation is submitted. The registry accepts the request. Go to step 8 if the redemption period elapses before a restore is received (4a).
5. If the server does not require a restore report, the object returns to its pre-deletion status (1) immediately upon a successful Create operation. If the server requires a report but the client includes it inline in the Create operation, the server processes both atomically and the object transitions directly from redemptionPeriod to its pre-deletion status (1), bypassing the pendingRestore state.
6. The object enters pendingDelete + pendingRestore state. The registry awaits a restore report from the sponsoring client.
7. If no restore report is received within the registry-defined time, the object returns to redemptionPeriod state (step 3).

8. If a restore report is received and accepted, the object returns to its pre-deletion status and all RGP status labels are removed.
9. The redemption period elapses without a restore request being received.
10. The object enters pendingDelete + rgpPendingDelete state and awaits final purge processing.
11. The pending delete period elapses and the object is purged.
12. The object is purged and available for re-registration.

2.6. EPP Compatibility Profile

RPP is designed to coexist with the Extensible Provisioning Protocol (EPP), often operating in parallel against a common backend provisioning system. While RPP is not inherently constrained by all of EPP's requirements, a specific set of rules is necessary to ensure seamless interoperability in such mixed environments.

To address this, this document defines an "EPP Compatibility Profile". This profile specifies a set of additional constraints on RPP data objects and operations that a server MUST adhere to when supporting both RPP and EPP concurrently.

Throughout this document, all constraints that are part of this profile are explicitly marked with a reference to "EPP Compatibility Profile". Implementers of systems in a mixed EPP/RPP environment MUST follow these specific constraints in addition to the base RPP requirements.

3. Common Data Types

This section defines new shared data types and structures that are re-used across multiple data object definitions and are based on the existing Primitive Data Types.

3.1. Identifier

Identifiers are character strings with a specified minimum length, a specified maximum length, and a specified format outlined in Section 2.8 of [RFC5730]. Identifiers for certain object types MAY have additional constraints imposed either by server policy, object specific specifications or both.

3.2. Client Identifier

Client identifiers are character strings with a specified minimum length, a specified maximum length, and a specified format. Contact identifiers use the "clIDType" client identifier syntax described in [RFC5730].

> | TBC: do we need this or is it a relation with an entity/RFC8543 organisation? If registrars modeled are as first class objects (organisations), then clID is nothing else but a reference to this organisation, so maybe no need to define syntax separately on identifier level (or in other words it would be defined on this object). R8.1 in the form of -02 RPP requirements includes RFC8543.

3.3. Phone Number

Telephone number syntax is derived from structures defined in [ITU.E164.2005]. Telephone numbers described in this specification are character strings that MUST begin with a plus sign ("+", ASCII value 0x002B), followed by a country code defined in [ITU.E164.2005], followed by a dot (".", ASCII value 0x002E), followed by a sequence of digits representing the telephone number. An optional "x" (ASCII value 0x0078) separator with additional digits representing extension information can be appended to the end of the value.

4. Associations

RPP allows for different types of associations (relationship) between the objects. The association may be added between 2 objects with own independent lifecycle (UML aggregation) or in the relation when one object’s existence and lifecycle is bound to the other parent/owner object (UML composition). In both cases, especially if the relation allows for cardinality higher than one on either side, the association may be assigned additional attributes, not being part of an object on either side of relation. In many cases such relation would be attributed with a single text string label, describing a role or a type of relation. Depending on the context this value might be unique, which allows using such label as a key in a dictionary.

The following generic Association Types are defined for RPP:

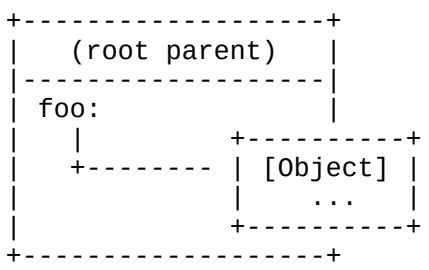
4.1. Aggregation

Notation: Aggregation[Type]

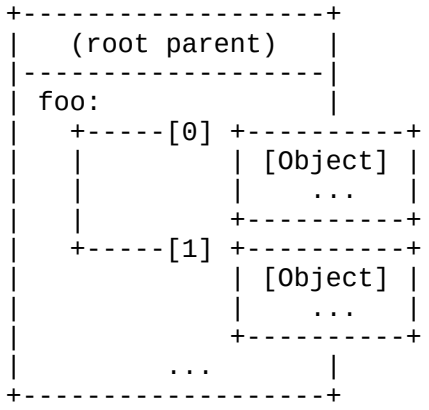
A relation between two independent objects.

If the cardinality of target object is more than 1, this represents an ordered array. It MUST be assured that the same unchanged data is always inserted in the same order in order to allow stable reference by position to data elements. In case of data insertions, deletions or updates the remaining of the data SHALL preserve its order.

Example aggregation having cardinality 1:



Example aggregation having cardinality >1:



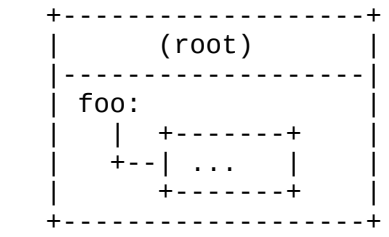
4.2. Composition

Notation: Composition[Type] or Type

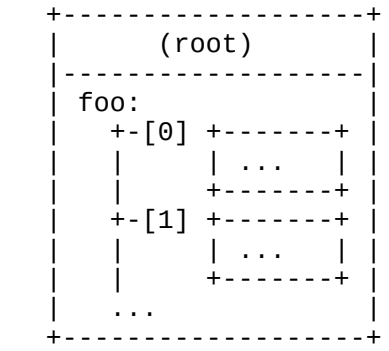
A relation between an independent parent object and 1 or more dependent child object(s).

If the cardinality of target object is more than 1, this represents an ordered array. It MUST be assured that the same unchanged data is always inserted in the same order in order to allow stable reference by position to data elements. In case of data insertions, deletions or updates the remaining of the data SHALL preserve its order.

Example composition having cardinality 1:



Example composition having cardinality >1:



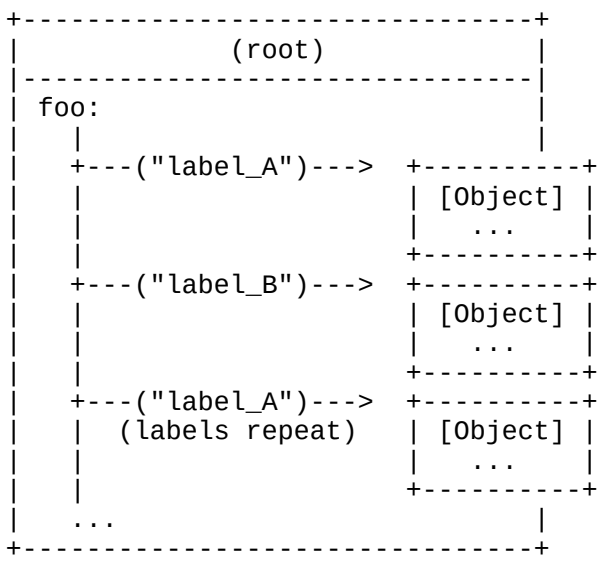
4.3. Labelled Aggregation

Notation: LabelledAggregation[Type]

A relation between two independent object with single text string attribute. Multiple associations with the same label are allowed and represent an unordered array.

A type defining such association MUST define Label Description with semantics of the label and Label Constraints with constraints related to the label.

Example labelled aggregation:



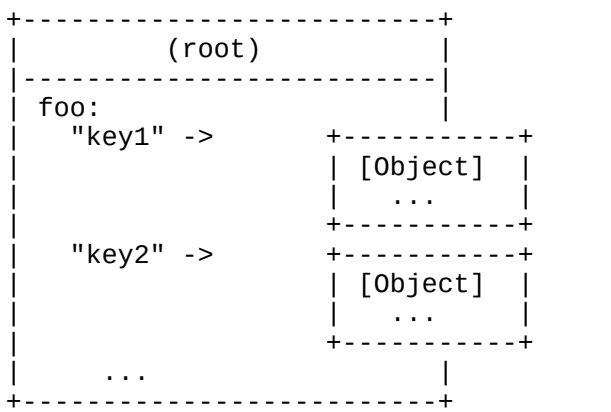
4.4. Dictionary Aggregation

Notation: DictionaryAggregation[Type]

A relation between two independent object with single text string attribute. Association labels MUST be unique allowing it to be used as dictionary key.

A type defining such association MUST define Label Description with semantics of the label and Label Constraints with constraints related to the label.

Example Dictionary Aggregation:



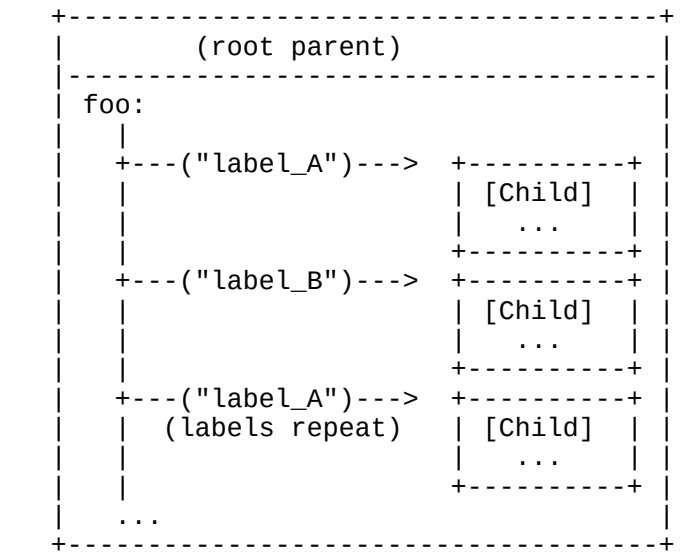
4.5. Labelled Composition

Notation: LabelledComposition[Type]

A relation between an independent parent object and a dependent child object with single text string attribute. Multiple associations with the same label are allowed.

A type defining such association MUST define Label Description with semantics of the label and Label Constraints with constraints related to the label.

Example Labelled Composition:



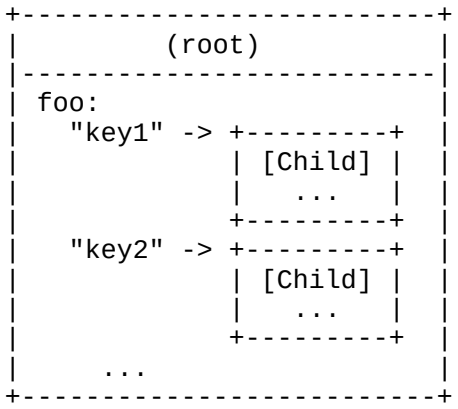
4.6. Dictionary Composition

Notation: DictionaryComposition[Type]

A relation between an independent parent object and a dependent child object with single text string attribute. Only single association with the same label is allowed allowing it to be used as dictionary key.

A type defining such association MUST define Label Description with semantics of the label and Label Constraints with constraints related to the label.

Example Dictionary Composition:



5. Component Objects

This section defines the Component Objects used in this document.

5.1. Period Object

* Name: Period Object
* Identifier: period
* Description: Represents a duration of time.
* Data Elements:
  - Value
    - Identifier: value
    - Cardinality: 1
    - Mutability: read-write
    - Data Type: Integer
    - Description: The numeric value of the period.
    - Constraints: The value MUST be from 1 to 99, inclusive.
  - Unit
    - Identifier: unit
    - Cardinality: 1
    - Mutability: read-write
    - Data Type: String
    - Description: The unit of the period.
    - Constraints: The value MUST be one of: "y" (years) or "m" (months).

5.2. Provisioning Metadata Object

* Name: Provisioning Metadata Object
* Identifier: provMetadata
* Description: Contains standard metadata about the lifecycle and ownership of a provisioned object. This metadata is common across all resource objects in the registry system.

* Data Elements:
  - Repository ID
    - Identifier: repositoryId
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Identifier
    - Description: A server-assigned unique identifier for the object.
    - Constraints: In EPP Compatibility Profile this data element MUST be provided.
  - Sponsoring Client ID
    - Identifier: spClientId
    - Cardinality: 1
    - Mutability: read-only
    - Data Type: Client Identifier
    - Description: The identifier of the client that is the current sponsor of the object.
    - Constraints: (None)
  - Creating Client ID
    - Identifier: crClientId
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Client Identifier
    - Description: The identifier of the client that created the object.
    - Constraints: (None)
  - Creation Date
    - Identifier: crDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp
    - Description: The date and time of object creation.
    - Constraints: The value is set by the server and cannot be specified by the client.
  - Updating Client ID
    - Identifier: upClientId
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Client Identifier
    - Description: The identifier of the client that last updated the object.
    - Constraints: This element MUST NOT be present if the object has never been modified.
  - Update Date
    - Identifier: upDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp

    - Description: The date and time of the most recent object modification.
    - Constraints: This element MUST NOT be present if the object has never been modified.
  - Transfer Date
    - Identifier: trDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp
    - Description: The date and time of the most recent successful object transfer.
    - Constraints: This element MUST NOT be provided if the object has never been transferred.

5.3. Status Object

* Name: Status Object
* Identifier: status
* Description: Represents one of the status values associated with a provisioning object
* Data Elements:
  - Label
    - Identifier: label
    - Cardinality: 1
    - Mutability: create-only
    - Data Type: String
    - Description: machine-readable enum label of a status
    - Constraints:
      + Exact list of allowed status labels depends on the provisioning object type. This enumeration can be expanded by extensions.
      + The status labels MUST use camel case notation and use only ASCII alphabetic characters.
      + Statuses MAY be of three categories:
      1. those explicitly set by a server. Those MUST have "server" prefix
      2. those explicitly set by a client. Those MUST have "client" prefix
      3. those indirectly controlled by provisioning object lifecycle or business logic. Those MUST NOT use either "client" or "server" prefix. They MAY use another prefix or no prefix at all
      + The following additional status labels are defined for use with the Redemption Grace Period (RGP) feature. When the RGP feature is supported, these labels MAY be specified:

        + addPeriod: The object is within the add grace period following initial registration. If the object is deleted during this period, the registry MAY provide a credit to the sponsoring client.
        + autoRenewPeriod: The object is within the auto-renew grace period following automatic renewal by the registry. If the object is deleted during this period, the registry MAY provide a credit to the sponsoring client.
        + renewPeriod: The object is within the renew grace period following an explicit renewal. If the object is deleted during this period, the registry MAY provide a credit to the sponsoring client.
        + transferPeriod: The object is within the transfer grace period following a successful transfer. If the object is deleted by the new sponsoring client during this period, the registry MAY provide a credit.
        + redemptionPeriod: A delete operation has been received and processed for the object, but the object has not yet been purged. A restore operation MAY be requested to abort the deletion. This status value MUST only appear alongside the standard pendingDelete status.
        + pendingRestore: A restore request has been accepted and the registry is waiting for a restore report from the sponsoring client. This status value MUST only appear alongside the standard pendingDelete status.
        + rgpPendingDelete: The redemption period has elapsed without a successful restore. The object has entered the purge processing state. This status value MUST only appear alongside the standard pendingDelete status. This label is used to distinguish the RGP-specific pending-delete sub-state from the broader EPP pendingDelete status.

    | TODO: find a better home for this list (own section + IANA registry). Add standard domain statuses here as well (and solve the issue of statuses not applicable to other object types like client/serverHold).

* Reason
  - Identifier: reason
  - Cardinality: 0-1
  - Mutability: create-only
  - Data Type: String
  - Description: a human-readable text that describes the rationale for the status applied to the object.
  - Constraints: None
* Due
  - Identifier: due

- Cardinality: 0-1
- Mutability: read-write
- Data Type: Timestamp
- Description: a timestamp, when this status is going to be removed automatically, or changed to other status. This field can be used to express lifecycle related information.
- Constraints: servers MAY restrict possibility to set or update this value by the client.

| TBD: Idea - model status object as Labelled Composition using
| "Label"? Con: Generic Constraints for Label will be repeated.

5.4. DNS Resource Record Object

* Name: DNS Resource Record Object
* Identifier: dnsRecord
* Description: Represents a single DNS resource record. The structure follows the top-level format of a DNS resource record as described in Section 3.2.1 of [RFC1035], adapted for use in provisioning. The RDATA field is represented as a structured object whose fields depend on the record type, following the RDATA presentation format described by the corresponding RFC defining the record type. This approach is aligned with [I-D.simmen-rpp-dns-data].
* Data Elements:
  - Name
    - Identifier: name
    - Cardinality: 1
    - Mutability: read-write
    - Data Type: String
    - Description: The owner name of the DNS entry. This MAY be the domain itself or a subordinate host name.
    - Constraints:
      + The value MUST be a syntactically valid DNS host name.
      + Absolute FQDNs (with trailing dot) and relative host names are allowed, as well as the "@" symbol representing the domain name itself.
  - Class
    - Identifier: class
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: The DNS resource record class.
    - Constraints:
      + If present, the value MUST be chosen from Section 3.2.4 (CLASS values) of [RFC1035].

        + A client SHOULD omit this element. The server MUST assume "IN" as the default class and MAY decline other values.
    - Type
        o Identifier: type
        o Cardinality: 1
        o Mutability: read-write
        o Data Type: String
        o Description: The DNS resource record type, indicating the format of the RDATA field.
        o Constraints:
            + The value MUST be a valid string representation of a resource record type as defined in [RFC1035] or other RFC describing the record type.
            + Allowed values MAY be constrained by server policies. For domain provisioning, the type would typically be constrained to the allowed parent-side entries.
            + Values MUST be converted to lower case.
            + In EPP Compatibility Profile ([RFC5732]), the following record types MUST be supported: ns, a, and aaaa.
            + In EPP Compatibility Profile with DNSSEC Extension [RFC5910], the following record types MUST additionally be supported: ds and dnskey.
    - RDATA
        o Identifier: rdata
        o Cardinality: 1
        o Mutability: read-write
        o Data Type: Object
        o Description: The actual payload data of the DNS record. The structure of this object depends on the record type and MUST follow the RDATA presentation format described by the corresponding RFC. Property names MUST be written in camelCase. All property values MUST be represented as Strings encoding the presentation format of the value.
        o Constraints:
            + The fields within RDATA MUST match the expected structure for the given record type.
            + See Section 5.4.1 for required structures in the EPP Compatibility Profile.

5.4.1. RDATA Structures in EPP Profile

This section defines the RDATA field structures required for interoperability in the EPP Compatibility Profile.

In EPP Compatibility Profile ([RFC5732]), the following RDATA structures MUST be supported:

* For NS records ([RFC1035], Section 3.3.11): nsdname (the fully qualified domain name of the name server).
* For A records ([RFC1035], Section 3.4.1): address (the IPv4 address in dotted-decimal notation).
* For AAAA records ([RFC3596], Section 2.2): address (the IPv6 address in text representation as defined in [RFC5952]).

In EPP Compatibility Profile with DNSSEC Extension ([RFC5910]), the following RDATA structures MUST additionally be supported:

* For DS records ([RFC4034], Section 5): keyTag (key tag value), algorithm (algorithm number), digestType (digest algorithm type), and digest (digest value).
* For DNSKEY records ([RFC4034], Section 2): flags (flags field value), protocol (protocol field value), algorithm (algorithm number), and publicKey (encoded public key value).

| TBC: Optional keyData inside dsData (RFC 5910 Section 4.1): In the DS Data Interface, a DS record MAY optionally contain a nested keyData element used for server-side validation of the DS hash. The draft doesn't describe this pattern - a client submitting a DS record with accompanying DNSKEY for validation.

5.5. DNS Operational Controls Object

* Name: DNS Operational Controls Object
* Identifier: dnsControls
* Description: Contains operational control parameters that a client MAY use to influence server-side DNS behaviour for a set of DNS records. A server MAY ignore these values, e.g. for policy reasons. This structure is aligned with [I-D.simmen-rpp-dns-data].
* Data Elements:
  - TTL
    - Identifier: ttl
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: Dictionary[Integer]
    - Description: Controls the caching behaviour of DNS resource records. The dictionary is keyed by lower-case record type name, with values representing the TTL in seconds for that record type.
    - Constraints:
      + The keys MUST be valid DNS resource record type names in lower case.
      + The values MUST be positive Integers.
      + The allowed value ranges MAY be constrained by server policy.

        + A server MAY ignore client-specified TTL values and apply default or policy-defined values.
    - Maximum Signature Lifetime
        o Identifier: maxSigLifetime
        o Cardinality: 0-1
        o Mutability: read-write
        o Data Type: Dictionary[Integer]
        o Description: Specifies a child's preference for the maximum number of seconds after signature generation when the parent's signature on signed DNS information should expire. The dictionary is keyed by lower-case record type name, with values representing the maximum signature lifetime in seconds for that record type. The value applies to the RRSIG resource record over the signed DNS RRset. See Section 3 of [RFC4034] for information on the RRSIG resource record. This corresponds to the maxSigLife concept from [RFC5910].
        o Constraints:
            + The keys MUST be valid DNS resource record type names in lower case.
            + The values MUST be positive Integers (minimum value of 1).
            + A server MAY ignore client-specified values and apply its own default signature expiration policy.

5.6. DNS Data Object

* Name: DNS Data Object
* Identifier: dnsData
* Description: A container for DNS resource records and associated operational controls for a provisioned object. This structure groups DNS records together with control parameters that influence server-side DNS behaviour. The structure is aligned with [I-D.simmen-rpp-dns-data].
* Data Elements:
    - Records
        o Identifier: records
        o Cardinality: 0+
        o Mutability: read-write
        o Data Type: Composition[DNS Resource Record Object]
        o Description: An array of DNS resource records associated with the provisioned object.
        o Constraints:
            + Allowed record types MAY be constrained by server policy.
            + In EPP Compatibility Profile with DNSSEC Extension [RFC5910], records of type DS and DNSKEY MUST be supported in addition to NS, A, and AAAA.

        + A server MUST support either DS or DNSKEY or both record types for DNSSEC provisioning. If provided with only DNSKEY, a server MUST calculate the DS record. If both record types are provided, a server MAY use the DNSKEY to validate the DS record.
    - Controls
      o Identifier: controls
      o Cardinality: 0-1
      o Mutability: read-write
      o Data Type: DNS Operational Controls Object
      o Description: Operational control parameters for the DNS records.
      o Constraints: (None)

5.7. Authorisation Information Object

* Name: Authorisation Information
* Identifier: authInfo
* Description: Contains information used to authorise operations on a data object. It may hold different kind of authorisation information.
* Data Elements:
  - Method
    o Identifier: method
    o Cardinality: 1
    o Mutability: create-only
    o Data Type: String
    o Description: The identifier of the RPP authorisation method.
    o Constraints:
      + The value MUST be one of the values registered at IANA as defined in [I-D.draft-wullink-rpp-core].
      + In EPP Compatibility Profile this value MUST be set to authinfo if standard password base authorisation is used
  - Authorisation Information
    o Identifier: authdata
    o Cardinality: 1
    o Mutability: create-only
    o Data Type: String
    o Description: The value of the authorisation information. It might be as simple as password string, but also more complex values like public key certificates or tokens encoded as string are possible.
    o Constraints:
      + Authorisation Information object is immutable. If the information changes (for example password is updated) a new instance MUST be created.

        + Depending on the method and server policy Authorisation Information MAY not be available for read or any other operation responding with this data element.

5.8. Postal Address Object

* Name: Postal Address Object
* Identifier: postalData
* Description: Contains the components of a postal address.
* Data Elements:
  - Street
    - Identifier: street
    - Cardinality: 0+
    - Mutability: read-write
    - Data Type: String
    - Description: The contact’s street address.
    - Constraints: Implementations MAY limit the maximum length of entries or character set.
  - City
    - Identifier: city
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: The contact’s city.
    - Constraints:
      + Implementations MAY limit the maximum length of entries or character set.
      + In EPP Compatibility Profile this data element MUST be provided.
  - State/Province
    - Identifier: sp
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: The contact’s state or province.
    - Constraints: Implementations MAY limit the maximum length of entries or character set.
  - Postal Code
    - Identifier: pc
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: The contact’s postal code.
    - Constraints:
      + Implementation MAY limit the maximum length of entries or character set.
      + The limitations MAY differ depending on Country Code (cc) data element.

- Country Code
  - Identifier: cc
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: String
  - Description: The contact’s country code.
  - Constraints:
    - The value MUST be a two-character identifier from [ISO3166-1].
    - In EPP Compatibility Profile this data element MUST be provided.

5.9. Postal Info Object

* Name: Postal Info Object
* Identifier: postalInfo
* Description: Contains postal-address information in either internationalised or localised forms.
* Data Elements:

| TBC: Contact Type is not localised (shall be the same for
| PERSON and ORG). Moving it level up would however detach it
| from related/dependant fields Name/Organisation

* Contact Type
  - Identifier: type
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: String
  - Description: Specifies whether the contact is and individual or an organisation.
  - Constraints: The value MUST be one of: "PERSON" (individual) or "ORG" (organisation).
* Name
  - Identifier: name
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: String
  - Description: The name of the individual or role.
  - Constraints:
    - Implementations MAY limit the maximum length of entries or character set.
    - In EPP Compatibility Profile this data element MUST be provided.
    - The implementations MAY require this field if Contact Type (type) is set to "PERSON".
* Organisation
  - Identifier: org

- Cardinality: 0-1
- Mutability: read-write
- Data Type: String
- Description: The name of the organisation.
- Constraints:
  - Implementations MAY limit the maximum length of entries or character set.
  - The implementations MAY require this field if Contact Type (type) is set to "ORG".

* Address
  - Identifier: addr
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: Postal Address Object
  - Description: The detailed postal address.
  - Constraints: In EPP Compatibility Profile this data element MUST be provided.

5.10. Disclose Object

| TODO: Model Disclose in universal (extendible) way

* Name: Disclose
* Identifier: disclose
* Description: TBD

5.11. Restore Report Object

* Name: Restore Report Object
* Identifier: restoreReport
* Description: Contains the redemption grace period restore report submitted by the sponsoring client as required by the RGP process ([RFC3915]). A restore report documents the state of the object before and after deletion, provides the reason for restoration, and includes mandatory client statements. This object is OPTIONAL and is only required when the RGP feature is supported and a restore report is being submitted.
* Data Elements:
  - Pre-Delete Data
    - Identifier: preData
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: A copy of the registration data that existed for the object prior to the object being deleted.
    - Constraints: None.
  - Post-Restore Data
    - Identifier: postData

    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: A copy of the registration data that exists for the object at the time the restore report is submitted.
    - Constraints: None.
- Delete Time
    - Identifier: deleteTime
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: Timestamp
    - Description: The date and time when the object delete request was sent to the server.
    - Constraints: None.
- Restore Time
    - Identifier: restoreTime
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: Timestamp
    - Description: The date and time when the original restore request operation was sent to the server.
    - Constraints:
        + This element MAY be omitted when the restore report is submitted inline within the restore request in a single-step process.
        + In EPP Compatibility Profile this element MUST be present as defined in [RFC3915].
- Restore Reason
    - Identifier: restoreReason
    - Cardinality: 0-1
    - Mutability: read-write
    - Data Type: String
    - Description: A brief explanation of the reason for restoring the object.
    - Constraints: None.
- Statements
    - Identifier: statements
    - Cardinality: 0+
    - Mutability: read-write
    - Data Type: String
    - Description: Client statements required by the RGP process.
    - Constraints: At least one and at most two statements MUST be provided. In EPP Compatibility Profile exactly two statements MUST be present as defined in [RFC3915].
- Other
    - Identifier: other
    - Cardinality: 0-1
    - Mutability: read-write

    - Data Type: String
    - Description: Any additional information needed to support the statements provided by the client.
    - Constraints: None.

6. Process Objects

This section defines the Process Objects used in this document.

6.1. Transfer Process Object

* Name: Transfer Process Object
* Identifier: transferProcess
* Description: Represents a transfer request for a provisioned object. Creating this object initiates a transfer. The object supports approve and reject as additional operations, and delete as the cancel operation. Reading the object returns the current transfer status.
* Data Elements:
  - Transfer Direction
    - Identifier: transferDir
    - Cardinality: 0-1
    - Mutability: create-only
    - Data Type: String
    - Description: Indicates whether the transfer is a "pull" or "push" transfer. Per policy, servers usually support only one model. If omitted, server policy determines the default.
    - Constraints: The value MUST be one of: "pull" (initiated by the gaining client) or "push" (initiated by the sponsoring client).
  - Gaining Client ID
    - Identifier: gainingClientId
    - Cardinality: 0-1
    - Mutability: create-only
    - Data Type: Client Identifier
    - Description: The identifier of the designated gaining client. This element is REQUIRED for push transfers and MUST NOT be provided for pull transfers.
    - Constraints: (None)
  - Transfer Status
    - Identifier: trStatus
    - Cardinality: 1
    - Mutability: read-only
    - Data Type: String
    - Description: The state of the transfer request.

  - Constraints: The value MUST be one of: "pending", "clientApproved", "clientCancelled", "clientRejected", "serverApproved", "serverCancelled".
- Requesting Client ID
  - Identifier: reqClientId
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Client Identifier
  - Description: The identifier of the client that initiated the transfer request.
  - Constraints: (None)
- Request Date
  - Identifier: requestDate
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Timestamp
  - Description: The date and time that the transfer was requested.
  - Constraints: (None)
- Acting Client ID
  - Identifier: actClientId
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Client Identifier
  - Description: For a pending pull transfer, the identifier of the sponsoring client that SHOULD act upon the request. For a pending push transfer, the identifier of the designated gaining client that SHOULD act upon the request. For all other statuses, the identifier of the client that took the indicated action.
  - Constraints: (None)
- Action Date
  - Identifier: actionDate
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Timestamp
  - Description: For a pending request, the date and time by which a response is required before an automated response action SHOULD be taken by the server. For all other statuses, the date and time when the request was completed.
  - Constraints: (None)

6.1.1. Operations

6.1.1.1. Create (Transfer Request)

* Identifier: create

The Create operation initiates a transfer by creating a Transfer Process Object associated with the provisioned object. The transfer direction and gaining client (for push transfers) are provided as create-only data elements of the Transfer Process Object.

* Authorisation:
  - Pull transfer:
    o The requesting client (gaining client) MUST provide valid Object Authorisation.
  - Push transfer:
    o Only the sponsoring client is authorised to initiate a push transfer.
* Input:
  - Owner Data Object reference
  - Transfer Process Object (create-only and read-write elements)
* Output: Transfer Process Object

6.1.1.2. Read (Transfer Query)

* Identifier: read

The Read operation allows a client to determine the real-time status of a pending or recently completed transfer request.

* Authorisation:
  - This operation MUST be accessible to both the sponsoring client and the gaining client.
  - Server policy determines whether other clients may query transfer status and what information is returned.
* Input: None
* Output: Transfer Process Object

6.1.1.3. Delete (Transfer Cancel)

* Identifier: delete

The Delete operation allows the initiating client to cancel its own pending transfer request.

* Authorisation:

- Only the initiating client (the client that originally requested the transfer).

* Input: None

* Output: Transfer Process Object or nothing

6.1.1.4. Approve

* Identifier: approve

The Approve operation allows the appropriate client to accept a pending transfer request.

* Authorisation:

  - Pull transfer:
    o only the sponsoring client
  - Push transfer:
    o only the designated gaining client

* Input: None

* Output: Transfer Process Object

6.1.1.5. Reject

* Identifier: reject

The Reject operation allows the appropriate client to decline a pending transfer request.

* Authorisation:
  - Pull transfer:
    o only the sponsoring client
  - Push transfer:
    o only the designated gaining client

The following transient data elements are defined for this operation:

* Reason

  - Identifier: reason
  - Cardinality: 0-1
  - Data Type: String
  - Description: A human-readable text describing the rationale for rejecting the transfer request.

- Constraints: In EPP Compatibility Profile this data element MUST NOT be used

* Output: Transfer Process Object

6.2. Restore Process Object

* Name: Restore Process Object
* Identifier: restoreProcess
* Description: Represents the current state of a restore request for an object that has entered the Redemption Grace Period (RGP).
* Data Elements:
  - Restore Status
    - Identifier: restoreStatus
    - Cardinality: 1
    - Mutability: read-only
    - Data Type: String
    - Description: The current state of the restore process.
    - Constraints: The value MUST be one of: "pendingRestore", "restored", "rgpPendingDelete"
  - Request Date
    - Identifier: requestDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp
    - Description: The date and time when the restore request was submitted.
    - Constraints: MUST NOT be present if no restore request has been submitted yet.
  - Report Date
    - Identifier: reportDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp
    - Description: The date and time when the most recent restore report was accepted by the server.
    - Constraints: MUST NOT be present if no restore report has been accepted yet.
  - Report Due Date
    - Identifier: reportDueDate
    - Cardinality: 0-1
    - Mutability: read-only
    - Data Type: Timestamp
    - Description: The date and time by which a restore report must be submitted before the object reverts to redemptionPeriod state. Only present when the object is in pendingRestore state.

- Constraints: MUST NOT be present when restoreStatus is not "pendingRestore".

6.2.1. Operations

6.2.1.1. Create (Restore Request)

* Identifier: create

The Create operation initiates the recovery of an object in the redemptionPeriod state by creating a Restore Process Object. The server MUST reject this operation if the owning object is not in the redemptionPeriod state.

* Input: Restore Process Object (create-only and read-write elements)

* Output: Restore Process Object

* Authorisation:

  - Only the sponsoring client is authorised to perform this operation.

The following transient data elements are defined for this operation:

* Restore Report
  - Identifier: restoreReport
  - Cardinality: 0-1
  - Data Type: Restore Report Object
  - Description: An OPTIONAL inline restore report. If provided, the server processes the request and the report atomically. If the server does not require a restore report, this element MUST NOT be present and the restore is completed immediately. If the server requires a report and this element is absent, the object transitions to pendingRestore state awaiting a subsequent Report operation.
  - Constraints:
    - MUST NOT be provided if the server does not require a restore report.
    - In EPP Compatibility Profile, corresponds to op="request" (without report) or a combined op="request" followed immediately by op="report" as defined in [RFC3915].

6.2.1.2. Read (Restore Query)

* Identifier: read

The Read operation allows the sponsoring client to retrieve the current state of the RGP process for an object.

* Authorisation:
  - Only the sponsoring client is authorised to perform this operation.
* Input: Object Identifier
* Output: Restore Process Object

In EPP Compatibility Profile this operation is not supported.

6.2.1.3. Report

* Identifier: report

This operation is OPTIONAL, only for servers which support or require submission of a restore report.

The Report operation submits the restore report required by the RGP process for an object in the pendingRestore state. A report MAY be submitted more than once if corrections are required. The server MUST reject this operation if the object is not in the pendingRestore state.

* Input: Object Identifier, Restore Report Object
* Output: Restore Process Object
* Authorisation:
  - Only the sponsoring client is authorised to perform this operation.

The following transient data elements are defined for this operation:

* Restore Report
  - Identifier: restoreReport
  - Cardinality: 1
  - Data Type: Restore Report Object
  - Description: The restore report to be submitted as part of the RGP process.
  - Constraints:
    o In EPP Compatibility Profile, corresponds to op="report" as defined in [RFC3915].

7. Domain Name Data Object

7.1. Object Description

* Name: Domain Name Data Object
* Identifier: domainName
* Description: A Domain Name data object represents a domain name and contains the data required for its provisioning and management in the registry.

7.2. Data Elements

The following data elements are defined for the Domain Name Data Object.

* Name

  - Identifier: name
  - Cardinality: 1
  - Mutability: create-only
  - Data Type: String
  - Description: The fully qualified name of the domain object.
  - Constraints:
    - The value MUST be a fully qualified domain name that conforms to the syntax described in [RFC1035].
    - A server MAY restrict allowable domain names to a particular top-level domain, second-level domain, or other domain for which the server is authoritative.
    - The trailing dot required when these names are stored in a DNS zone is implicit and MUST NOT be provided when exchanging host and domain names.

* Provisioning Metadata

  - Identifier: provMetadata
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Provisioning Metadata Object
  - Description: Standard metadata about the object’s lifecycle and ownership.
  - Constraints: (None)

* Status

  - Identifier: status
  - Cardinality: 0+
  - Mutability: read-only
  - Data Type: Status Object

- Description: The current status descriptors associated with the domain.
- Constraints:
  - Possible combinations of Status Object Labels is specified in Section 2.3 of [RFC5731].

| TBC: IANA registry for statuses?

* Registrant
  - Identifier: registrant
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: Contact Object.
  - Description: The contact object associated with the domain as the registrant.
  - Constraints:
    - The relation MUST correspond to a valid Contact Data Object known to the server.
    - Servers MAY restrict association of a Contact Object of a different sponsoring client.

| TBC: leave registrant here or move it to contacts with a type?

* Contacts
  - Identifier: contacts
  - Cardinality: 0+
  - Mutability: read-write
  - Data Type: LabelledAggregation[Contact Object]
    - Label Description: The role of the associated contact.
    - Label Constraints:
      + List of supported roles is defined by server policy
      + In the EPP Compatibility Profile, the value MUST be one of: "admin", "billing", or "tech"
  - Description: A collection of other contact objects associated with the domain object.
  - Constraints:
    - Maximum number of associated contacts (per role) MAY be restricted by server policy

| TBC: IANA registry for contact role label?

* Nameservers

  - Identifier: nameservers
  - Cardinality: 0+
  - Mutability: read-write
  - Data Type: Aggregation[Host Data Object]

- Description: A collection of nameservers associated with the domain.
- Constraints: (None)

* DNS Data

- Identifier: dns
- Cardinality: 0-1
- Mutability: read-write
- Data Type: DNS Data Object
- Description: DNS resource records and operational controls related to the domain name. This structure follows the unified DNS data model defined in [I-D.simmen-rpp-dns-data], providing a single container for all DNS-related provisioning data including delegation, DNSSEC, and other record types.
- Constraints:
  - The type of the record entries MAY be constrained by the server policy. Typically the values would be limited to allowed parent-side resource record types.
  - In EPP Compatibility Profile, a server MUST support NS, A, and AAAA record types for delegation, and with DNSSEC Extension [RFC5910] a server MUST additionally support DS and/or DNSKEY record types.
  - The names of DNS entries MUST be the domain name itself or subordinate to the domain name and MUST NOT be below zone cut in case of present delegation.
  - A server MAY forbid use of NS/A/AAAA record types if the server only supports host object model through Nameserver property defining those records.

* Subordinate Hosts

- Identifier: subordinateHosts
- Cardinality: 0+
- Mutability: read-only
- Data Type: Aggregation[Host Data Object]
- Description: A collection of subordinate host objects that exist under this domain.
- Constraints: (None)

* Expiry Date

- Identifier: expiryDate
- Cardinality: 0-1
- Mutability: read-only
- Data Type: Timestamp.
- Description: The date and time identifying the end of the domain object’s registration period.

- Constraints: The value is set by the server and cannot be specified by the client.

* Authorisation Information

- Identifier: authInfo
- Cardinality: 0-1
- Mutability: read-write
- Data Type: Authorisation Information Object
- Description: Authorisation information associated with the domain object.
- Constraints: (None)

7.3. Operations

7.3.1. Create Operation

* Identifier: create

The Create operation allows a client to provision a new Domain Name resource. The operation accepts as input all create-only and read-write data elements defined for the Domain Name Data Object.

* Authorisation:
  - Generally each client is authorised to create new domain objects becoming a sponsoring client. This can be however constrained by the server policy in many ways, i.e. by applying rate limiting, billing related constraints or compliance locks.

In addition, the following transient data element is defined for this operation:

* Registration Period
  - Identifier: period
  - Cardinality: 0-1
  - Data Type: Period Object.
  - Description: The initial registration period for the domain name. This value is used by the server to calculate the initial expiryDate of the object. This element is not persisted as part of the object’s state.

7.3.2. Read Operation

* Identifier: read

The Read operation allows a client to retrieve the data elements of a Domain Name resource. The server’s response MAY vary depending on client authorisation and server policy.

* Authorisation:
  - Sponsoring client:
    o Full object
  - Other client:
    o Without Object Authorisation:
      + Limited object (non-confidential properties) or operation denied
    o With Object Authorisation:
      + Full object, however some properties only authorised to the sponsoring client MAY be redacted according to server policy

The following transient data elements are defined for this operation:

* Hosts Filter
  - Identifier: hostsFilter
  - Cardinality: 0-1
  - Data Type: String
  - Description: Controls which host information is returned with the object.
  - Constraints: The value MUST be one of "all", "del" (delegated), "sub" (subordinate), or "none". The default value is "all".

7.3.3. Update Operation

The Update operation allows a client to modify the read-write data elements of an existing Domain Name resource.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

The following transient data elements are defined for this operation:

* Urgent
  - Identifier: urgent
  - Cardinality: 0-1
  - Data Type: Boolean
  - Description: Requests that the server operator process and implement the update with high priority. "High priority" is relative to standard server operator policies determined using an out-of-band mechanism. In EPP Compatibility Profile this corresponds to the "urgent" attribute of the <secDNS:update> element defined in [RFC5910]. The default value is false.
  - Constraints:
    o A server that does not support this parameter MUST return an error if it is set to true.
    o A server that supports this parameter but cannot fulfil a specific urgent request MUST return an error.

7.3.4. Delete Operation

* Identifier: delete

The Delete operation allows a client to remove an existing Domain Name resource. The operation targets a specific data object identified by its name.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

The server SHOULD reject a delete request if subordinate host objects are associated with the domain name.

The error response SHOULD indicate the related subordinate host objects.

7.3.5. Renew Operation

* Identifier: renew

The Renew operation allows a client to extend the validity period of an existing Domain Name resource. The operation targets a specific data object identified by its name.

* Authorisation:

  - Only sponsoring client is authorised to perform this operation

* Input: Domain Name

* Output: Full object (read-write and read-only properties), or a minimum set of properties affected by the operation (Expiry Date).

The following transient data elements are defined for this operation:

* Current Expiry Date

  - Identifier: currentExpiryDate
  - Cardinality: 1
  - Data Type: Timestamp
  - Description: The current expiry date of the domain name. The server MUST validate this against the object’s current expiryDate to prevent unintended duplicate renewals.

* Renewal Period

  - Identifier: renewalPeriod

- Cardinality: 0-1
- Data Type: Period Object
- Description: The duration to be added to the object’s registration period. This value is used by the server to calculate the new expiryDate. The default value MAY be defined by server policy. The number of units available MAY be subject to limits imposed by the server.

7.3.6. Transfer Operations

The Domain Name Data Object supports the common transfer operations defined in the Section 2.5.4. The transfer of a domain name changes the sponsoring client of the domain object.

Transfer of a domain object MUST implicitly transfer all host objects that are subordinate to the domain object. For example, if domain object "example.com" is transferred and host object "ns1.example.com" exists, the host object MUST be transferred as part of the "example.com" transfer process.

In addition to the common Transfer Process Object elements, the following object-specific create-only data element is defined and can be provided when creating a Transfer Process Object for a domain name:

* Transfer Period
  - Identifier: transferPeriod
  - Cardinality: 0-1
  - Mutability: create-only
  - Data Type: Period Object
  - Description: The number of units to be added to the registration period of the domain object upon successful completion of the transfer. The number of units available MAY be subject to limits imposed by the server.
  - Constraints: (None)

In addition to the common Transfer Process Object elements, the following object-specific read-only data element is included in the output of domain transfer operations:

* Expiry Date
  - Identifier: expiryDate
  - Cardinality: 0-1
  - Mutability: read-only
  - Data Type: Timestamp
  - Description: The end of the domain object’s registration period if the transfer caused or causes a change in the validity period.

Subordinate host objects MUST be transferred implicitly when the domain object is transferred.

7.3.7. Restore Operations

The Domain Name Data Object supports the restore operations defined in the section. These operations are OPTIONAL and are only available when the RGP feature is supported.

No domain-specific transient data elements extend the common restore operations beyond those defined in the Section 2.5.5.

8. Contact Data Object

8.1. Object Description

* Name: Contact Data Object
* Identifier: contact
* Description: A Contact Data Object represents the social information for an individual or organisation associated with other objects.

8.2. Data Elements

The following data elements are defined for the Domain Name Data Object.

* Handle ID
  - Identifier: id
  - Cardinality: 1
  - Mutability: create-only
  - Data Type: Identifier.
  - Description: External unique identifier of the contact object.
  - Constraints:
    o This value MUST be supported to be provided by the client.
    o Servers MAY support server-side generation of this value.

* Provisioning Metadata
  - Identifier: provMetadata
  - Cardinality: 1
  - Mutability: read-only
  - Data Type: Provisioning Metadata Object
  - Description: Standard metadata about the object’s lifecycle and ownership.
  - Constraints: (None)

* Status

  - Identifier: status
  - Cardinality: 0+
  - Mutability: read-only
  - Data Type: Status Object
  - Description: The current status descriptors associated with the contact.
  - Constraints:
    - Possible combinations of Domain Status Labels is specified in Section 2.2 of [RFC5733]
    - The value MUST be one of the status tokens defined in the IANA registry for domain statuses.
    - The initial value list MAY be as defined in [RFC5733]. In this case the values MUST have the same semantics.

* Postal Information

  - Identifier: postalInfo
  - Cardinality: 1-2
  - Mutability: read-write
  - Data Type: DictionaryAggregation[Postal Info Object]
    - Label Description: type of contact data localisation
    - Label Constraints: Allowed values: "int" for "internationalised" all-ASCII version of an address and "loc" for localised forms with possible non-ASCII character sets.
  - Description: Contains postal-address information.
  - Constraints: There MUST be no more that 1 element of type "int" and one element of type "loc".

* Voice Phone Number

  - Identifier: voice
  - Cardinality: 0+
  - Mutability: read-write
  - Data Type: Phone Number
  - Description: Voice phone number associated with the contact
  - Constraints: (None)

* Fax Phone Number

  - Identifier: fax
  - Cardinality: 0+
  - Mutability: read-write
  - Data Type: Phone Number
  - Description: Fax number associated with the contact
  - Constraints: (None)

* E-mail

  - Identifier: email
  - Cardinality: 0+
  - Mutability: read-write
  - Data Type: String.
  - Description: The contact’s email address.
  - Constraints: Email address syntax is defined in [RFC5322].

* Authorisation Information

  - Identifier: authInfo
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: Authorisation Information
  - Description: Authorisation information associated with the contact object.
  - Constraints: (None)

* Disclose

  - Identifier: disclose
  - Cardinality: 0-1
  - Mutability: read-write
  - Data Type: Disclose Object.
  - Description: Identifies elements that require exceptional server-operator handling to allow or restrict disclosure to third parties.

  | TBC: IANA registry for statuses?

8.3. Operations

8.3.1. Create Operation

The Create operation allows a client to provision a new Contact resource. The operation accepts as input all create-only and read-write data elements defined for the Contact Data Object.

* Authorisation:
  - Generally each client is authorised to create new contact objects becoming a sponsoring client. This can be however constrained by the server policy, e.g. by applying rate limiting or compliance locks.

In EPP Compatibility Profile, the following data elements MUST be provided:

* Handle ID (id)
* At least one Postal Information entry (postalInfo) containing a Name (name) and an Address (addr) with City (city) and Country Code (cc)
* E-mail (email)
* Authorisation Information (authInfo)

8.3.2. Read Operation

The Read operation allows a client to retrieve the data elements of a Contact resource. The server's response MAY vary depending on client authorisation and server policy.

* Authorisation:
  - Sponsoring client:
    o Full object
  - Other client:
    o Without Object Authorisation:
      + Limited object (non-confidential properties) or operation denied
    o With Object Authorisation:
      + Full object, however some properties only authorised to the sponsoring client MAY be redacted according to server policy

Authorisation Information (authInfo) MUST NOT be provided in the response if the querying client is not the current sponsoring client.

When constructing the response, the server MUST respect the disclosure policies defined by the Disclose Object (disclose), whether set by the server operator's default data-collection policy or by the sponsoring client for the contact. Data elements marked for non-disclosure MUST NOT be included in responses to unauthorised clients.

8.3.3. Update Operation

The Update operation allows a client to modify the attributes of an existing Contact resource.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

The following aspects of the contact object MAY be modified:

* Status values that are client-manageable (prefixed with "client") MAY be added or removed.

* Postal Information, Voice Phone Number, Fax Phone Number, E-mail, Authorisation Information, and Disclose preferences MAY be changed.

A client MUST NOT add, delete or alter values for statuses managed by the server (prefixed with "server"). A server MAY add, delete or alter status values set by a client, subject to server policy.

8.3.4. Delete Operation

The Delete operation allows a client to remove an existing Contact resource. The operation targets a specific data object identified by its Handle ID.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

The server SHOULD reject a delete request if the contact object is associated with other known objects (e.g., domain names). An associated contact SHOULD NOT be deleted until associations with other known objects have been broken.

The error response SHOULD indicate the existing object associations.

8.3.5. Transfer Operations

The Contact Data Object supports the common transfer operations defined in the Section 2.5.4. The transfer of a contact changes the sponsoring client of the contact object.

No object-specific transient data elements are defined for contact transfer operations beyond the common transfer data elements.

9. Host Data Object

9.1. Object Description

* Name: Host Data Object
* Identifier: host
* Description: A Host Data Object represents a name server that provides DNS services for a domain name.

9.2. Data Elements

The following data elements are defined for the Host Data Object.

* Host Name

- Identifier: hostName
- Cardinality: 1
- Mutability: read-write
- Data Type: String
- Description: Fully qualified name of a host.
- Constraints: The value MUST be a syntactically valid host name.

* Provisioning Metadata

- Identifier: provMetadata
- Cardinality: 1
- Mutability: read-only
- Data Type: Provisioning Metadata Object
- Description: Standard metadata about the object’s lifecycle and ownership.
- Constraints: (None)

* Status

- Identifier: status
- Cardinality: 0+
- Mutability: read-only
- Data Type: Status Object
- Description: The current status descriptors associated with the host.
- Constraints: Possible combinations of Host Status Labels is specified in Section 2.3 of [RFC5732]

* DNS Data

- Identifier: dns
- Cardinality: 0-1
- Mutability: read-write
- Data Type: DNS Data Object
- Description: DNS resource records and operational controls related to the host. This structure follows the unified DNS data model defined in [I-D.simmen-rpp-dns-data].
- Constraints:
  - The names of DNS entries MUST be the host name itself or subordinate to the host name.
  - In EPP Compatibility Profile the record entries MUST be limited to A and AAAA entries for IPv4 and IPv6 glue records respectively.

9.3. Operations

9.3.1. Create Operation

The Create operation allows a client to provision a new Host Data Object. The operation accepts as input all create-only and read-write data elements defined for the Host Data Object.

* Authorisation:
  - Generally each client is authorised to create new host objects becoming a sponsoring client. This can be however constrained by the server policy, e.g. by applying rate limiting or compliance locks.

If the host name exists in a namespace for which the server is authoritative, then the superordinate domain of the host MUST be known to the server before the host object can be created.

In EPP Compatibility Profile, IP addresses are REQUIRED only as needed to produce DNS glue records. If the host name exists in a namespace for which the server is authoritative and is subordinate to an existing domain, IP addresses SHOULD be provided. If the host name is external to the server’s namespace, IP addresses are not required by the DNS and MAY be omitted.

9.3.2. Read Operation

The Read operation allows a client to retrieve the data elements of a Host Data Object.

* Authorisation:
  - Any client is authorised to retrieve the full object. In EPP Compatibility Profile, host objects do not carry authorisation information and there is no distinction based on client identity as described in Section 3.1.2 of [RFC5732].

9.3.3. Update Operation

The Update operation allows a client to modify the attributes of an existing Host Data Object. The operation targets a specific data object identified by its host name.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

Host name changes MAY require the addition or removal of IP addresses to be accepted by the server. IP address association MAY be subject to server policies for provisioning hosts as name servers.

Host name changes can have an impact on associated objects that refer to the host object. A Host Name change SHOULD NOT require additional updates of associated objects to preserve existing associations, with one exception: changing an external host object that has associations with objects that are sponsored by a different client. Attempts to update such hosts directly MUST fail. The change can be provisioned by creating a new external host with a new name and any needed new attributes, and subsequently updating the other objects sponsored by the client.

9.3.4. Delete Operation

The Delete operation allows a client to remove an existing Host Data Object. The operation targets a specific data object identified by its host name.

* Authorisation:
  - Only sponsoring client is authorised to perform this operation

The server SHOULD reject a delete request if the host object is associated with any other object, such as a domain name object. Deleting a host object without first breaking existing associations can cause DNS resolution failure for domain objects that refer to the deleted host object.

> | TODO: consider RFC 9874 / BCP 244 for host deletions practices

The error response SHOULD indicate the related associated objects.

9.3.5. Restore Operations

The Host Data Object supports the restore operations defined in the Section 2.5.5. These operations are OPTIONAL and are only available when the RGP feature for Host Data Object is supported by the server.

No domain-specific transient data elements extend the common restore operations beyond those defined in the Section 2.5.5.

10. IANA Considerations

10.1. RPP Data Object Registry

This document establishes the "RESTful Provisioning Protocol (RPP) Data Object Registry". This registry serves as a catalogue of all data objects, component objects, data elements, and operations used within RPP.

10.1.1. Registration Policy

The policy for adding new objects, data elements, or operations to this registry is "Specification Required" [RFC8126].

Standardised RPP extensions that introduce new data objects, add data elements to existing objects, or define new operations or operation parameters MUST register these additions in this registry. Each such registration MUST reference the specification that defines the extension.

Private (non-standardised) extensions are not required to register in this registry.

10.1.2. Registry Structure

The registry is organised as a collection of Object definitions. Each Object definition MUST include:

* A header containing the Object Identifier, Object Name, Object Type (Resource, Process or Component), a brief description, and a reference to its defining specification.

* A "Data Elements" table listing all persisted data elements associated with the object. Each entry MUST specify the element’s Identifier, Name, Cardinality, Mutability, Data Type, description, and a reference to the specification that defines it.

* An "Operations" section (applicable only for Object Types Resource or Process). For each operation, the registry MUST provide:

  - The Operation’s Name, a description, and a reference to the specification that defines it.
  - A "Parameters" table listing all data elements that are provided as input to the operation but are not persisted as part of the object’s state. Each entry MUST specify the parameter’s Identifier, Name, Cardinality, Data Type, description, and a reference to the specification that defines it.

Extensions MAY add new data elements, operations, or operation parameters to existing Object definitions in the registry. Each such addition MUST reference the extension specification that introduces it, allowing implementations to distinguish core protocol elements from extension-defined elements.

10.1.3. Initial Registrations

The initial contents of the RPP Data Object Registry are defined below.

Object: period

Object Name: Period Object

Object Type: Component

Description: Represents a duration of time.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
value	Value	1	read-write	Integer	The numeric value of the period.
unit	Unit	1	read-write	String	The unit of the period.

Table 1

Object: dnsRecord

Object Name: DNS Resource Record Object

Object Type: Component

Description: Represents a single DNS resource record.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
name	Name	1	read-write	String	The owner name of the DNS entry.
class	Class	0-1	read-write	String	The DNS resource record class.
type	Type	1	read-write	String	The DNS resource record type, indicating the format of the RDATA field.
rdata	RDATA	1	read-write	Object	The actual payload data of the DNS record. Structure depends on the record type.

Table 2

Object: dnsControls

Object Name: DNS Operational Controls Object

Object Type: Component

Description: Contains operational control parameters that a client MAY use to influence server-side DNS behaviour for a set of DNS records.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
ttl	TTL	0-1	read-write	Dictionary [Integer]	Controls the caching behaviour of DNS resource records, keyed by lower-case record type name.
maxSigLifetime	Maximum Signature Lifetime	0-1	read-write	Dictionary [Integer]	Maximum number of seconds after signature generation when the parent’s signature on signed DNS data should expire, keyed by lower-case record type name.

Table 3

Object: dnsData

Object Name: DNS Data Object

Object Type: Component

Description: A container for DNS resource records and associated operational controls for a provisioned object.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
records	Records	0+	read-write	Composition [DNS Resource Record Object]	An array of DNS resource records associated with the provisioned object.
controls	Controls	0-1	read-write	DNS Operational Controls Object	Operational control parameters for the DNS records.

Table 4

Object: authInfo

Object Name: Authorisation Information

Object Type: Component

Description: Contains authorisation credentials for an operation.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
method	Method	1	create-only	String	The identifier of the RPP authorisation method.
authdata	Authorisation Information	1	create-only	String	The value of the authorisation information. It might be as simple as password string, but also more complex values like public key certificates or tokens encoded as string are possible.

Table 5

Object: status

Object Name: Status Object

Object Type: Component

Description: Represents one of the status values associated with the provisioning object.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
label	Label	1	create-only	String	machine-readable enum label of a status
reason	Reason	0-1	create-only	String	a human-readable text that describes the rationale for the status applied to the object.
due	Due	0-1	read-write	Timestamp	a timestamp, when this status is going to be removed automatically, or changed to other status. This field can be used to express lifecycle related information

Table 6

Object: provMetadata

Object Name: Provisioning Metadata Object

Object Type: Component

Description: Contains standard metadata about the lifecycle and ownership of a provisioned object.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
repositoryId	Repository ID	0-1	read-only	Identifier	A server-assigned unique identifier for the object.
spClientId	Sponsoring Client ID	1	read-only	Client Identifier	The identifier of the client that is the current sponsor of the object.
crClientId	Creating Client ID	0-1	read-only	Client Identifier	The identifier of the client that created the object.
crDate	Creation Date	0-1	read-only	Timestamp	The date and time of object creation.
upClientId	Updating Client ID	0-1	read-only	Client Identifier	The identifier of the client that last updated the object.
upDate	Update Date	0-1	read-only	Timestamp	The date and time of the most recent object modification.
trDate	Transfer Date	0-1	read-only	Timestamp	The date and time of the most recent successful object transfer.

Table 7

Object: transferProcess

Object Name: Transfer Process Object

Object Type: Process

Description: Represents a transfer request for a provisioned object. Creating this object initiates the transfer. Approve and Reject are additional operations; Delete corresponds to cancel.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
transferDir	Transfer Direction	0-1	create-only	String	The direction of the transfer ("pull" or "push"). If omitted, server policy determines default.
gainingClientId	Gaining Client ID	0-1	create-only	Client Identifier	The designated gaining client. REQUIRED for push transfers; MUST NOT be provided for pull.
trStatus	Transfer Status	1	read-only	String	The state of the transfer request.
reqClientId	Requesting Client ID	1	read-only	Client Identifier	The identifier of the client that

					initiated the transfer request.
requestDate	Request Date	1	read-only	Timestamp	The date and time that the transfer was requested.
actClientId	Acting Client ID	1	read-only	Client Identifier	The identifier of the client that should or did act on the request.
actionDate	Action Date	1	read-only	Timestamp	The response deadline (if pending) or completion date.

Table 8

Operations

Operation: Create

Operation Identifier: create

Description: Initiates a transfer of a Domain Name resource by creating a Transfer Process Object. Transfer direction and gaining client are provided as create-only data elements.

Parameters

Identifier	Name	Card.	Data Type	Description
transferPeriod	Transfer Period	0-1	period	The duration to add to the registration period upon transfer.

Table 9

Operation: Transfer Read

Operation Identifier: read

Description: Queries the status of a transfer of a Domain Name resource.

Parameters: (None)

Operation: Transfer Delete

Operation Identifier: delete

Description: Cancels a pending transfer of a Domain Name resource.

Parameters: (None)

Operation: Transfer Approve

Operation Identifier: approve

Description: Approves a pending transfer of a Domain Name resource.

Parameters: (None)

Operation: Transfer Reject

Operation Identifier: reject (on Transfer Process Object)

Description: Rejects a pending transfer of a Domain Name resource.

Parameters

Identifier	Name	Card.	Data Type	Description
reason	Reason	0-1	String	A human-readable text describing the rationale for rejection.

Table 10

| TODO: IANA table: Postal Address Object TODO: IANA table: Postal Info Object TODO: IANA table: Disclose Object

Object: restoreProcess

Object Name: Restore Process Object

Object Type: Process

Description: Represents the current state of a restore request for an object that has entered the Redemption Grace Period (RGP). Returned as output of all restore operations. This object is OPTIONAL and is only used when the RGP feature is supported.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
restoreStatus	Restore Status	1	read-only	String	The current state of the restore process.
requestDate	Request Date	0-1	read-only	Timestamp	The date and time when the restore request was submitted. Absent if no request has been submitted.
reportDate	Report Date	0-1	read-only	Timestamp	The date and time when the most recent restore report was accepted. Absent if no report has been accepted.
reportDueDate	Report Due Date	0-1	read-only	Timestamp	The deadline for submitting a restore report before the object reverts to redemptionPeriod. Present only when status is pendingRestore.

Table 11

Operations

Operation: Restore Create

Operation Identifier: create

Description: Initiates recovery of a domain name in the redemptionPeriod state by creating a Restore Process Object. This operation is OPTIONAL and is only available when the RGP feature is supported.

Parameters

Identifier	Name	Card.	Data Type	Description
restoreReport	Restore Report	0-1	Restore Report Object	An inline restore report.

Table 12

Operation: Restore Read

Operation Identifier: read (on Restore Process Object)

Description: Retrieves the current state of the RGP restore process for a domain name. This operation is OPTIONAL and is only available when the RGP feature is supported.

Parameters: (None)

Operation: Restore Report

Operation Identifier: report

Description: Submits the restore report for a domain name in the pendingRestore state. This operation is OPTIONAL and is only available when the RGP feature is supported and the server requires a restore report.

Parameters

Identifier	Name	Card.	Data Type	Description
restoreReport	Restore Report	1	Restore Report Object	The restore report to be submitted.

Table 13

Object: restoreReport

Object Name: Restore Report Object

Object Type: Component

Description: Contains the redemption grace period restore report submitted by the sponsoring client as required by the RGP process. This object is OPTIONAL and is only used when the RGP feature is supported and a restore report is required by server policy.

Reference: [This-ID]

Data Elements

Element Identifier	Element Name	Card.	Mutability	Data Type	Description
preData	Pre-Delete Data	0-1	read-write	String	A copy of the registration data that existed for the object prior to deletion.
postData	Post-Restore Data	0-1	read-write	String	A copy of the registration data that exists for the object at the time the restore report is submitted.
deleteTime	Delete Time	0-1	read-write	Timestamp	The date and time when the object delete request was sent to the server.
restoreTime	Restore Time	0-1	read-write	Timestamp	The date and time when the original restore request operation was sent to

					the server.
restoreReason	Restore Reason	0-1	read-write	String	A brief explanation of the reason for restoring the object.
statements	Statements	0+	read-write	String	Mandatory client statements required by the RGP process. At least one and at most two statements MUST be provided.
other	Other	0-1	read-write	String	Any additional information needed to support the statements provided by the client.

Table 14

Object: domainName

Object Name: Domain Name Data Object

Object Type: Resource

Description: Represents a domain name and its associated data.

Reference: [This-ID]

Data Elements

Identifier	Name	Card.	Mutability	Data Type	Description
name	Name	1	create-only	String	The fully qualified name of the domain object.
provMetadata	Provisioning Metadata	1	read-only	Provisioning Metadata Object	Standard metadata about object lifecycle and ownership.
status	Status	0+	read-only	Status Object	The current status descriptors for the domain.
registrant	Registrant	0-1	read-write	Contact Object	The registrant contact ID.
contacts	Contacts	0+	read-write	LabelledAggregation [Contact Object]	Associated contact objects.
nameservers	Nameservers	0+	read-write	Aggregation [Host Data Object]	A collection of nameservers associated with the domain.
dns	DNS Data	0-1	read-write	DNS Data Object	DNS resource records and operational controls related to the domain name.
subordinateHosts	Subordinate Hosts	0+	read-only	Aggregation [Host Data Object]	Subordinate host names.
expiryDate	Expiry Date	0-1	read-only	Timestamp	Expiry

					timestamp.
authInfo	Authorisation Info	0-1	read-write	authInfo	Authorisation information for the object.

Table 15

Operations

Operation: Create

Operation Identifier: create

Description: Provisions a new Domain Name resource.

Parameters

Identifier	Name	Card.	Data Type	Description
period	Registration Period	0-1	period	The initial registration period for the domain name.

Table 16

Operation: Read

Operation Identifier: read

Description: Retrieves the data elements of a Domain Name resource.

Parameters

Identifier	Name	Card.	Data Type	Description
hostsFilter	Hosts Filter	0-1	String	Controls which host information is returned.
queryAuthInfo	Query Authorisation Information	0-1	authInfo	Credentials to authorise access to full object data.

Table 17

Operation: Update

Description: Modifies the read-write data elements of a Domain Name resource.

Parameters

Identifier	Name	Card.	Data Type	Description
urgent	Urgent	0-1	Boolean	Requests high-priority processing of the update by the server.

Table 18

Operation: Delete

Operation Identifier: delete

Description: Removes an existing Domain Name resource.

Parameters: (None)

Operation: Renew

Operation Identifier: renew

Description: Extends the validity period of a Domain Name resource.

Parameters

Identifier	Name	Card.	Data Type	Description
currentExpiryDate	Current Expiry Date	1	Timestamp	The expected current expiry date, for validation.
renewalPeriod	Renewal Period	0-1	period	The duration to add to the registration period.

Table 19

| TODO: IANA table: Contact Data Object

Object: host

Object Name: Host Data Object

Object Type: Resource

Description: Represents a name server that provides DNS services for a domain name.

Reference: [This-ID]

Data Elements

Identifier	Name	Card.	Mutability	Data Type	Description
hostName	Host Name	1	read-write	String	Fully qualified name of a host.
provMetadata	Provisioning Metadata	1	read-only	Provisioning Metadata Object	Standard metadata about object lifecycle and ownership.
status	Status	0+	read-only	Status Object	The current status descriptors for the host.
dns	DNS Data	0-1	read-write	DNS Data Object	DNS resource records and operational controls related to the host.

Table 20

Operations

Operation: Create

Description: Provisions a new Host Data Object.

Parameters: (None)

Operation: Read

Description: Retrieves the data elements of a Host Data Object.

Parameters: (None)

Operation: Update

Description: Modifies the attributes of a Host Data Object.

Parameters: (None)

Operation: Delete

Description: Removes an existing Host Data Object.

Parameters: (None)

11. Security Considerations

| TODO: write security considerations, if any

12. Changes History

draft-kowalik-rpp-data-objects -02 - -03

* add Object and Dictionary[Value Type] primitive data types
* change "Aggregation/Composition Dictionary" to "Dictionary Aggregation/Composition" (Issue #32)
* describe operations for contacts #15
* describe operations for hosts #16
* add Domain Update operation with urgent transient parameter from [RFC5910]
* add identifiers to all operations
* abstract common provisioning metadata into reusable component object
* define common RGP/restore process and Restore Process Object
* define common transfer operations and Transfer Process Object with support for pull and push transfers #23
* add domain-specific transfer operations with implicit renewal and subordinate host transfer
* add host/domain relationship terminology from RFC 5732
* restructure DNS data model aligned with draft-simmen-rpp-dns-data-01: redefine DNS Resource Record Object, add DNS Operational Controls Object, add DNS Data Object; update Domain, Host, and Nameserver objects to use it
* add DNSSEC support based on [RFC5910]: DS and DNSKEY record types with structured RDATA fields
* add NS, A, and AAAA RDATA structures and record type constraints to EPP compatibility profile for DNS Resource Record Object
* expand extensibility section with standardised/private extension mechanisms, extension points, and operation extensibility
* update IANA registration policy and registry structure to accommodate extension registrations

draft-kowalik-rpp-data-objects -01 - -02

* ontology of allowed basic datatypes #4
* add examples of associations #31

13. References

13.1. Normative References

[I-D.ietf-rpp-architecture]
Kowalik, P. and M. Wullink, "RPP Architecture", Work in Progress, Internet-Draft, draft-ietf-rpp-architecture-01, 27 February 2026, <https://datatracker.ietf.org/doc/html/draft-ietf-rpp-architecture-01>.

[I-D.kowalik-rpp-architecture]
Kowalik, P. and M. Wullink, "RPP Architecture", Work in Progress, Internet-Draft, draft-kowalik-rpp-architecture-03, 10 October 2025, <https://datatracker.ietf.org/doc/html/draft-kowalik-rpp-architecture-03>.

[ISO3166-1]
International Organization for Standardization, "Codes for the representation of names of countries and their subdivisions -- Part 1: Country codes", ISO Standard 3166, November 2000.

[ITU.E164.2005]
International Telecommunication Union, "The international public telecommunication numbering plan", ITU-T Recommendation E.164, February 2005.

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, DOI 10.17487/RFC1035, November 1987, <https://www.rfc-editor.org/info/rfc1035>.

[RFC1738] Berners-Lee, T., Masinter, L., and M. McCahill, "Uniform Resource Locators (URL)", RFC 1738, DOI 10.17487/RFC1738, December 1994, <https://www.rfc-editor.org/info/rfc1738>.

[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>.

[RFC3915] Hollenbeck, S., "Domain Registry Grace Period Mapping for the Extensible Provisioning Protocol (EPP)", RFC 3915, DOI 10.17487/RFC3915, September 2004, <https://www.rfc-editor.org/info/rfc3915>.

[RFC5322] Resnick, P., Ed., "Internet Message Format", RFC 5322, DOI 10.17487/RFC5322, October 2008, <https://www.rfc-editor.org/info/rfc5322>.

[RFC5730] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)", STD 69, RFC 5730, DOI 10.17487/RFC5730, August 2009, <https://www.rfc-editor.org/info/rfc5730>.

[RFC5731] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Domain Name Mapping", STD 69, RFC 5731, DOI 10.17487/RFC5731, August 2009, <https://www.rfc-editor.org/info/rfc5731>.

[RFC5732] Hollenbeck, S., "Extensible Provisioning Protocol (EPP) Host Mapping", STD 69, RFC 5732, DOI 10.17487/RFC5732, August 2009, <https://www.rfc-editor.org/info/rfc5732>.

[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>.

[RFC5952] Kawamura, S. and M. Kawashima, "A Recommendation for IPv6 Address Text Representation", RFC 5952, DOI 10.17487/RFC5952, August 2010, <https://www.rfc-editor.org/info/rfc5952>.

[RFC8126] Cotton, M., Leiba, B., and T. Narten, "Guidelines for Writing an IANA Considerations Section in RFCs", BCP 26, RFC 8126, DOI 10.17487/RFC8126, June 2017, <https://www.rfc-editor.org/info/rfc8126>.

[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>.

13.2. Informative References

[I-D.simmen-rpp-dns-data]
Simmen, C. and P. Kowalik, "DNS data representation for use in RESTful Provisioning Protocol (RPP)", Work in Progress, Internet-Draft, draft-simmen-rpp-dns-data-01, 20 October 2025, <https://datatracker.ietf.org/doc/html/draft-simmen-rpp-dns-data-01>.

[RFC3596] Thomson, S., Huitema, C., Ksinant, V., and M. Souissi, "DNS Extensions to Support IP Version 6", STD 88, RFC 3596, DOI 10.17487/RFC3596, October 2003, <https://www.rfc-editor.org/info/rfc3596>.

[RFC4034] Arends, R., Austein, R., Larson, M., Massey, D., and S. Rose, "Resource Records for the DNS Security Extensions", RFC 4034, DOI 10.17487/RFC4034, March 2005, <https://www.rfc-editor.org/info/rfc4034>.

[RFC5910] Gould, J. and S. Hollenbeck, "Domain Name System (DNS) Security Extensions Mapping for the Extensible Provisioning Protocol (EPP)", RFC 5910, DOI 10.17487/RFC5910, May 2010, <https://www.rfc-editor.org/info/rfc5910>.

Authors' Addresses

Pawel Kowalik
DENIC
Email: pawel.kowalik@denic.de
URI: https://denic.de/

Maarten Wullink
SIDN Labs
Email: maarten.wullink@sidn.nl
URI: https://sidn.nl/