Extensible Provisioning Protocol (EPP) Contact Mapping

Status of This Memo

Copyright Notice

Abstract

This document describes an Extensible Provisioning Protocol (EPP)
mapping for the provisioning and management of individual or
organizational social information identifiers (known as "contacts")
stored in a shared central repository. Specified in Extensible
Markup Language (XML), the mapping defines EPP command syntax and
semantics as applied to contacts. This document obsoletes RFC 3733.

Table of Contents

1. Introduction

   This document describes a personal and organizational identifier
   mapping for version 1.0 of the Extensible Provisioning Protocol
   (EPP). This mapping is specified using the Extensible Markup
   Language (XML) 1.0 as described in [W3C.REC-xml-20040204] and XML
   Schema notation as described in [W3C.REC-xmlschema-1-20041028] and
   [W3C.REC-xmlschema-2-20041028]. This document obsoletes RFC 3733
   [RFC3733].

   [RFC4930] provides a complete description of EPP command and response
   structures. A thorough understanding of the base protocol
   specification is necessary to understand the mapping described in
   this document.

   XML is case sensitive. Unless stated otherwise, XML specifications
   and examples provided in this document MUST be interpreted in the
   character case presented to develop a conforming implementation.

1.1. Conventions Used in This Document

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

   In examples, "C:" represents lines sent by a protocol client and "S:"
   represents lines returned by a protocol server. Indentation and
   white space in examples are provided only to illustrate element
   relationships and are not a REQUIRED feature of this protocol.

2. Object Attributes

   An EPP contact object has attributes and associated values that can
   be viewed and modified by the sponsoring client or the server. This
   section describes each attribute type in detail. The formal syntax
   for the attribute values described here can be found in the "Formal
   Syntax" section of this document and in the appropriate normative
   references.

2.1. Contact and Client Identifiers

   All EPP contacts are identified by a server-unique identifier.
   Contact 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
   [RFC4930].

2.2. Status Values

A contact object MUST always have at least one associated status value. Status values can be set only by the client that sponsors a contact object and by the server on which the object resides. A client can change the status of a contact object using the EPP <update> command. Each status value MAY be accompanied by a string of human-readable text that describes the rationale for the status applied to the object.

A client MUST NOT alter status values set by the server. A server MAY alter or override status values set by a client subject to local server policies. The status of an object MAY change as a result of either a client-initiated transform command or an action performed by a server operator.

Status values that can be added or removed by a client are prefixed with "client". Corresponding status values that can be added or removed by a server are prefixed with "server". Status values that do not begin with either "client" or "server" are server-managed.

Status Value Descriptions:

- clientDeleteProhibited, serverDeleteProhibited

  Requests to delete the object MUST be rejected.

- clientTransferProhibited, serverTransferProhibited

  Requests to transfer the object MUST be rejected.

- clientUpdateProhibited, serverUpdateProhibited

  Requests to update the object (other than to remove this status) MUST be rejected.

- linked

  The contact object has at least one active association with another object, such as a domain object. Servers SHOULD provide services to determine existing object associations.

- ok

  This is the normal status value for an object that has no pending operations or prohibitions. This value is set and removed by the server as other status values are added or removed.

   - pendingCreate, pendingDelete, pendingRenew, pendingTransfer,
      pendingUpdate

      A transform command has been processed for the object, but the
      action has not been completed by the server. Server operators can
      delay action completion for a variety of reasons, such as to allow
      for human review or third-party action. A transform command that
      is processed, but whose requested action is pending, is noted with
      response code 1001.

   When the requested action has been completed, the pendingCreate,
   pendingDelete, pendingTransfer, or pendingUpdate status value MUST be
   removed. All clients involved in the transaction MUST be notified
   using a service message that the action has been completed and that
   the status of the object has changed.

   "ok" status MAY only be combined with "linked" status.

   "linked" status MAY be combined with any status.

   "pendingDelete" status MUST NOT be combined with either
   "clientDeleteProhibited" or "serverDeleteProhibited" status.

   "pendingTransfer" status MUST NOT be combined with either
   "clientTransferProhibited" or "serverTransferProhibited" status.
   "pendingUpdate" status MUST NOT be combined with either
   "clientUpdateProhibited" or "serverUpdateProhibited" status.

   The pendingCreate, pendingDelete, pendingTransfer, and pendingUpdate
   status values MUST NOT be combined with each other.

   Other status combinations not expressly prohibited MAY be used.

2.3. Individual and Organizational Names

   Individual and organizational names associated with a contact are
   represented using character strings. These strings have a specified
   minimum length and a specified maximum length. Individual and
   organizational names MAY be provided in both UTF-8 [RFC3629] and a
   subset of UTF-8 that can be represented in 7-bit ASCII depending on
   local needs.

2.4. Address

   Every contact has associated postal address information. A postal
   address contains OPTIONAL street information, city information,
   OPTIONAL state/province information, an OPTIONAL postal code, and a
   country identifier. Address information MAY be provided in both

UTF-8 and a subset of UTF-8 that can be represented in 7-bit ASCII
depending on local needs.

2.4.1. Street, City, and State or Province

Contact street, city, and state or province information is
represented using character strings. These strings have a specified
minimum length and a specified maximum length.

2.4.2. Postal Code

Contact postal codes are represented using character strings. These
strings have a specified minimum length and a specified maximum
length.

2.4.3. Country

Contact country identifiers are represented using two-character
identifiers specified in [ISO.3166.1997].

2.5. Telephone Numbers

Contact telephone number structure is derived from structures defined
in [ITU.E164.2005]. Telephone numbers described in this mapping 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"
attribute is provided to note telephone extension information.

2.6. Email Addresses

Email address syntax is defined in [RFC0822]. This mapping does not
prescribe minimum or maximum lengths for character strings used to
represent email addresses.

2.7. Dates and Times

Date and time attribute values MUST be represented in Universal
Coordinated Time (UTC) using the Gregorian calendar. The extended
date-time form using upper case "T" and "Z" characters defined in
[W3C.REC-xmlschema-2-20041028] MUST be used to represent date-time
values as XML Schema does not support truncated date-time forms or
lower case "T" and "Z" characters.

2.8. Authorization Information

   Authorization information is associated with contact objects to
   facilitate transfer operations. Authorization information is
   assigned when a contact object is created, and it might be updated in
   the future. This specification describes password-based
   authorization information, though other mechanisms are possible.

2.9. Disclosure of Data Elements and Attributes

   The EPP core protocol requires a server operator to announce data
   collection policies to clients; see Section 2.4 of [RFC4930]. In
   conjunction with this disclosure requirement, this mapping includes
   data elements that allow a client to identify elements that require
   exceptional server operator handling to allow or restrict disclosure
   to third parties.

   A server operator announces a default disclosure policy when
   establishing a session with a client. When an object is created or
   updated, the client can specify contact attributes that require
   exceptional disclosure handling using an OPTIONAL <contact:disclose>
   element. Once set, disclosure preferences can be reviewed using a
   contact information query. A server operator MUST reject any
   transaction that requests disclosure practices that do not conform to
   the announced data collection policy with a 2308 error response code.

   If present, the <contact:disclose> element MUST contain a "flag"
   attribute. The "flag" attribute contains an XML Schema boolean
   value. A value of "true" or "1" (one) notes a client preference to
   allow disclosure of the specified elements as an exception to the
   stated data collection policy. A value of "false" or "0" (zero)
   notes a client preference to not allow disclosure of the specified
   elements as an exception to the stated data collection policy.

   The <contact:disclose> element MUST contain at least one of the
   following child elements:

   <contact:name type="int"/>
   <contact:name type="loc"/>
   <contact:org type="int"/>
   <contact:org type="loc"/>
   <contact:addr type="int"/>
   <contact:addr type="loc"/>
   <contact:voice/>
   <contact:fax/>
   <contact:email/>

   Example <contact:disclose> element, flag="0":

   <contact:disclose flag="0">
    <contact:email/>
    <contact:voice/>
   </contact:disclose>

   In this example, the contact email address and voice telephone number
   cannot be disclosed. All other elements are subject to disclosure in
   accordance with the server's data collection policy.

   Example <contact:disclose> element, flag="1":

   <contact:disclose flag="1">
    <contact:name type="int"/>
    <contact:org type="int"/>
    <contact:addr type="int"/>
   </contact:disclose>

   In this example, the internationalized contact name, organization,
   and address information can be disclosed. All other elements are
   subject to disclosure in accordance with the server's data collection
   policy.

   Client identification features provided by the EPP <login> command
   and contact authorization information are used to determine if a
   client is authorized to perform contact information query commands.
   These features also determine if a client is authorized to receive
   data that is otherwise marked for non-disclosure in a query response.

3. EPP Command Mapping

   A detailed description of the EPP syntax and semantics can be found
   in [RFC4930]. The command mappings described here are specifically
   for use in provisioning and managing contact objects via EPP.

3.1. EPP Query Commands

   EPP provides three commands to retrieve contact information: <check>
   to determine if a contact object can be provisioned within a
   repository, <info> to retrieve detailed information associated with a
   contact object, and <transfer> to retrieve contact object transfer
   status information.

3.1.1. EPP <check> Command

   The EPP <check> command is used to determine if an object can be
   provisioned within a repository. It provides a hint that allows a
   client to anticipate the success or failure of provisioning an object
   using the <create> command as object provisioning requirements are
   ultimately a matter of server policy.

   In addition to the standard EPP command elements, the <check> command
   MUST contain a <contact:check> element that identifies the contact
   namespace. The <contact:check> element contains the following child
   elements:

   - One or more <contact:id> elements that contain the server-unique
      identifier of the contact objects to be queried.

   Example <check> command:

   C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
   C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
   C: <command>
   C: <contact:check
   C: xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
   C: <contact:id>sh8013</contact:id>
   C: <contact:id>sah8013</contact:id>
   C: <contact:id>8013sah</contact:id>
   C: </contact:check>
   C: <clTRID>ABC-12345</clTRID>
   C: </command>
   C:</epp>

   When a <check> command has been processed successfully, the EPP
   <resData> element MUST contain a child <contact:chkData> element that
   identifies the contact namespace. The <contact:chkData> element
   contains one or more <contact:cd> elements that contain the following
   child elements:

   - A <contact:id> element that identifies the queried object. This
      element MUST contain an "avail" attribute whose value indicates
      object availability (can it be provisioned or not) at the moment
      the <check> command was completed. A value of "1" or "true" means
      that the object can be provisioned. A value of "0" or "false"
      means that the object cannot be provisioned.

- An OPTIONAL <contact:reason> element that MAY be provided when an object cannot be provisioned. If present, this element contains server-specific text to help explain why the object cannot be provisioned. This text MUST be represented in the response language previously negotiated with the client; an OPTIONAL "lang" attribute MAY be present to identify the language if the negotiated value is something other than the default value of "en" (English).

Example <check> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <contact:chkData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:cd>
S:          <contact:id avail="1">sh8013</contact:id>
S:        </contact:cd>
S:        <contact:cd>
S:          <contact:id avail="0">sah8013</contact:id>
S:          <contact:reason>In use</contact:reason>
S:        </contact:cd>
S:        <contact:cd>
S:          <contact:id avail="1">8013sah</contact:id>
S:        </contact:cd>
S:      </contact:chkData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <check> command cannot be processed for any reason.

3.1.2. EPP <info> Command

   The EPP <info> command is used to retrieve information associated
   with a contact object. In addition to the standard EPP command
   elements, the <info> command MUST contain a <contact:info> element
   that identifies the contact namespace. The <contact:info> element
   contains the following child elements:

   - A <contact:id> element that contains the server-unique identifier
      of the contact object to be queried.

   - An OPTIONAL <contact:authInfo> element that contains authorization
      information associated with the contact object. If this element
      is not provided or if the authorization information is invalid,
      server policy determines if the command is rejected or if response
      information will be returned to the client.

   Example <info> command:

   C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
   C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
   C: <command>
   C: <info>
   C: <contact:info
   C: xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
   C: <contact:id>sh8013</contact:id>
   C: <contact:authInfo>
   C: <contact:pw>2fooBAR</contact:pw>
   C: </contact:authInfo>
   C: </contact:info>
   C: </info>
   C: <clTRID>ABC-12345</clTRID>
   C: </command>
   C:</epp>

   When an <info> command has been processed successfully, the EPP
   <resData> element MUST contain a child <contact:infData> element that
   identifies the contact namespace. The <contact:infData> element
   contains the following child elements:

   - A <contact:id> element that contains the server-unique identifier
      of the contact object.

   - A <contact:roid> element that contains the Repository Object
      IDentifier assigned to the contact object when the object was
      created.

- One or more <contact:status> elements that describe the status of
  the contact object.

- One or two <contact:postalInfo> elements that contain postal
  address information. Two elements are provided so that address
  information can be provided in both internationalized and
  localized forms; a "type" attribute is used to identify the two
  forms. If an internationalized form (type="int") is provided,
  element content MUST be represented in a subset of UTF-8 that can
  be represented in the 7-bit US-ASCII character set. If a
  localized form (type="loc") is provided, element content MAY be
  represented in unrestricted UTF-8. The <contact:postalInfo>
  element contains the following child elements:

  - A <contact:name> element that contains the name of the
    individual or role represented by the contact.

  - An OPTIONAL <contact:org> element that contains the name of the
    organization with which the contact is affiliated.

  - A <contact:addr> element that contains address information
    associated with the contact. A <contact:addr> element contains
    the following child elements:

    - One, two, or three OPTIONAL <contact:street> elements that
      contain the contact's street address.

    - A <contact:city> element that contains the contact's city.

    - An OPTIONAL <contact:sp> element that contains the contact's
      state or province.

    - An OPTIONAL <contact:pc> element that contains the contact's
      postal code.

    - A <contact:cc> element that contains the contact's country
      code.

- An OPTIONAL <contact:voice> element that contains the contact's
  voice telephone number.

- An OPTIONAL <contact:fax> element that contains the contact's
  facsimile telephone number.

- A <contact:email> element that contains the contact's email
  address.

-   A <contact:clID> element that contains the identifier of the
    sponsoring client.

-   A <contact:crID> element that contains the identifier of the
    client that created the contact object.

-   A <contact:crDate> element that contains the date and time of
    contact object creation.

-   A <contact:upID> element that contains the identifier of the
    client that last updated the contact object. This element MUST
    NOT be present if the contact has never been modified.

-   A <contact:upDate> element that contains the date and time of the
    most recent contact object modification. This element MUST NOT be
    present if the contact object has never been modified.

-   A <contact:trDate> element that contains the date and time of the
    most recent successful contact object transfer. This element MUST
    NOT be provided if the contact object has never been transferred.

-   A <contact:authInfo> element that contains authorization
    information associated with the contact object. This element MUST
    NOT be provided if the querying client is not the current
    sponsoring client.

-   An OPTIONAL <contact:disclose> element that identifies elements
    that require exceptional server operator handling to allow or
    restrict disclosure to third parties. See Section 2.9 for a
    description of the child elements contained within the <contact:
    disclose> element.

Example <info> response for an authorized client:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <contact:infData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:roid>SH8013-REP</contact:roid>
S:        <contact:status s="linked"/>
S:        <contact:status s="clientDeleteProhibited"/>
S:        <contact:postalInfo type="int">
```

```
S:            <contact:name>John Doe</contact:name>
S:            <contact:org>Example Inc.</contact:org>
S:            <contact:addr>
S:              <contact:street>123 Example Dr.</contact:street>
S:              <contact:street>Suite 100</contact:street>
S:              <contact:city>Dulles</contact:city>
S:              <contact:sp>VA</contact:sp>
S:              <contact:pc>20166-6503</contact:pc>
S:              <contact:cc>US</contact:cc>
S:            </contact:addr>
S:          </contact:postalInfo>
S:          <contact:voice x="1234">+1.7035555555</contact:voice>
S:          <contact:fax>+1.7035555556</contact:fax>
S:          <contact:email>jdoe@example.com</contact:email>
S:          <contact:clID>ClientY</contact:clID>
S:          <contact:crID>ClientX</contact:crID>
S:          <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:          <contact:upID>ClientX</contact:upID>
S:          <contact:upDate>1999-12-03T09:00:00.0Z</contact:upDate>
S:          <contact:trDate>2000-04-08T09:00:00.0Z</contact:trDate>
S:          <contact:authInfo>
S:            <contact:pw>2fooBAR</contact:pw>
S:          </contact:authInfo>
S:          <contact:disclose flag="0">
S:            <contact:voice/>
S:            <contact:email/>
S:          </contact:disclose>
S:        </contact:infData>
S:      </resData>
S:      <trID>
S:        <clTRID>ABC-12345</clTRID>
S:        <svTRID>54322-XYZ</svTRID>
S:      </trID>
S:   </response>
S:</epp>
```

An EPP error response MUST be returned if an <info> command cannot be processed for any reason.

3.1.3. EPP <transfer> Query Command

The EPP <transfer> command provides a query operation that allows a client to determine real-time status of pending and completed transfer requests. In addition to the standard EPP command elements, the <transfer> command MUST contain an "op" attribute with value "query", and a <contact:transfer> element that identifies the contact namespace. The <contact:transfer> element MUST contain the following child elements:

   - A <contact:id> element that contains the server-unique identifier
      of the contact object to be queried.

   - An OPTIONAL <contact:authInfo> element that contains authorization
      information associated with the contact object. If this element
      is not provided or if the authorization information is invalid,
      server policy determines whether the command is rejected or the
      response information will be returned to the client.

   Example <transfer> query command:

   C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
   C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
   C: <command>
   C: <transfer op="query">
   C: <contact:transfer
   C: xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
   C: <contact:id>sh8013</contact:id>
   C: <contact:authInfo>
   C: <contact:pw>2fooBAR</contact:pw>
   C: </contact:authInfo>
   C: </contact:transfer>
   C: </transfer>
   C: <clTRID>ABC-12345</clTRID>
   C: </command>
   C:</epp>

   When a <transfer> query command has been processed successfully, the
   EPP <resData> element MUST contain a child <contact:trnData> element
   that identifies the contact namespace. The <contact:trnData> element
   contains the following child elements:

   - A <contact:id> element that contains the server-unique identifier
      for the queried contact.

   - A <contact:trStatus> element that contains the state of the most
      recent transfer request.

   - A <contact:reID> element that contains the identifier of the
      client that requested the object transfer.

   - A <contact:reDate> element that contains the date and time that
      the transfer was requested.

   - A <contact:acID> element that contains the identifier of the
      client that SHOULD act upon a PENDING transfer request. For all
      other status types, the value identifies the client that took the
      indicated action.

- A <contact:acDate> element that contains the date and time of a
  required or completed response. For a pending request, the value
  identifies the date and time by which a response is required
  before an automated response action SHOULD be taken by the server.
  For all other status types, the value identifies the date and time
  when the request was completed.

Example <transfer> query response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <contact:trnData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:trStatus>pending</contact:trStatus>
S:        <contact:reID>ClientX</contact:reID>
S:        <contact:reDate>2000-06-06T22:00:00.0Z</contact:reDate>
S:        <contact:acID>ClientY</contact:acID>
S:        <contact:acDate>2000-06-11T22:00:00.0Z</contact:acDate>
S:      </contact:trnData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <transfer> query command
cannot be processed for any reason.

3.2. EPP Transform Commands

EPP provides four commands to transform contact object information:
<create> to create an instance of a contact object, <delete> to
delete an instance of a contact object, <transfer> to manage contact
object sponsorship changes, and <update> to change information
associated with a contact object. This document does not define a
mapping for the EPP <renew> command.

Transform commands are typically processed and completed in real
time. Server operators MAY receive and process transform commands,
but defer completing the requested action if human or third-party

review is required before the requested action can be completed. In
such situations, the server MUST return a 1001 response code to the
client to note that the command has been received and processed, but
the requested action is pending. The server MUST also manage the
status of the object that is the subject of the command to reflect
the initiation and completion of the requested action. Once the
action has been completed, all clients involved in the transaction
MUST be notified using a service message that the action has been
completed and that the status of the object has changed.

3.2.1. EPP <create> Command

The EPP <create> command provides a transform operation that allows a
client to create a contact object. In addition to the standard EPP
command elements, the <create> command MUST contain a <contact:
create> element that identifies the contact namespace. The <contact:
create> element contains the following child elements:

- A <contact:id> element that contains the desired server-unique
   identifier for the contact to be created.

- One or two <contact:postalInfo> elements that contain postal
   address information. Two elements are provided so that address
   information can be provided in both internationalized and
   localized forms; a "type" attribute is used to identify the two
   forms. If an internationalized form (type="int") is provided,
   element content MUST be represented in a subset of UTF-8 that can
   be represented in the 7-bit US-ASCII character set. If a
   localized form (type="loc") is provided, element content MAY be
   represented in unrestricted UTF-8. The <contact:postalInfo>
   element contains the following child elements:

   - A <contact:name> element that contains the name of the
      individual or role represented by the contact.

   - An OPTIONAL <contact:org> element that contains the name of the
      organization with which the contact is affiliated.

   - A <contact:addr> element that contains address information
      associated with the contact. A <contact:addr> element contains
      the following child elements:

      - One, two, or three OPTIONAL <contact:street> elements that
         contain the contact's street address.

      - A <contact:city> element that contains the contact's city.

- An OPTIONAL <contact:sp> element that contains the contact's
  state or province.

- An OPTIONAL <contact:pc> element that contains the contact's
  postal code.

- A <contact:cc> element that contains the contact's country
  code.

- An OPTIONAL <contact:voice> element that contains the contact's
  voice telephone number.

- An OPTIONAL <contact:fax> element that contains the contact's
  facsimile telephone number.

- A <contact:email> element that contains the contact's email
  address.

- A <contact:authInfo> element that contains authorization
  information to be associated with the contact object. This
  mapping includes a password-based authentication mechanism, but
  the schema allows new mechanisms to be defined in new schemas.

- An OPTIONAL <contact:disclose> element that allows a client to
  identify elements that require exceptional server operator
  handling to allow or restrict disclosure to third parties. See
  Section 2.9 for a description of the child elements contained
  within the <contact:disclose> element.

Example <create> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <create>
C:      <contact:create
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact:postalInfo type="int">
C:          <contact:name>John Doe</contact:name>
C:          <contact:org>Example Inc.</contact:org>
C:          <contact:addr>
C:            <contact:street>123 Example Dr.</contact:street>
C:            <contact:street>Suite 100</contact:street>
C:            <contact:city>Dulles</contact:city>
C:            <contact:sp>VA</contact:sp>
C:            <contact:pc>20166-6503</contact:pc>
C:            <contact:cc>US</contact:cc>
C:          </contact:addr>
C:        </contact:postalInfo>
C:        <contact:voice x="1234">+1.7035555555</contact:voice>
C:        <contact:fax>+1.7035555556</contact:fax>
C:        <contact:email>jdoe@example.com</contact:email>
C:        <contact:authInfo>
C:          <contact:pw>2fooBAR</contact:pw>
C:        </contact:authInfo>
C:        <contact:disclose flag="0">
C:          <contact:voice/>
C:          <contact:email/>
C:        </contact:disclose>
C:      </contact:create>
C:    </create>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <create> command has been processed successfully, the EPP
<resData> element MUST contain a child <contact:creData> element that
identifies the contact namespace. The <contact:creData> element
contains the following child elements:

- A <contact:id> element that contains the server-unique identifier
   for the created contact.

- A <contact:crDate> element that contains the date and time of
   contact object creation.

Example <create> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <resData>
S:      <contact:creData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:      </contact:creData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <create> command cannot be processed for any reason.

3.2.2. EPP <delete> Command

The EPP <delete> command provides a transform operation that allows a client to delete a contact object. In addition to the standard EPP command elements, the <delete> command MUST contain a <contact: delete> element that identifies the contact namespace. The <contact: delete> element MUST contain the following child element:

- A <contact:id> element that contains the server-unique identifier of the contact object to be deleted.

A contact object SHOULD NOT be deleted if it is associated with other known objects. An associated contact SHOULD NOT be deleted until associations with other known objects have been broken. A server SHOULD notify clients of object relationships when a <delete> command is attempted and fails due to existing object relationships.

Example <delete> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <delete>
C:      <contact:delete
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:      </contact:delete>
C:    </delete>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When a <delete> command has been processed successfully, a server
MUST respond with an EPP response with no <resData> element.

Example <delete> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1000">
S:      <msg>Command completed successfully</msg>
S:    </result>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <delete> command cannot
be processed for any reason.

3.2.3. EPP <renew> Command

Renewal semantics do not apply to contact objects, so there is no
mapping defined for the EPP <renew> command.

3.2.4. EPP <transfer> Command

   The EPP <transfer> command provides a transform operation that allows
   a client to manage requests to transfer the sponsorship of a contact
   object. In addition to the standard EPP command elements, the
   <transfer> command MUST contain a <contact:transfer> element that
   identifies the contact namespace. The <contact:transfer> element
   contains the following child elements:

   - A <contact:id> element that contains the server-unique identifier
      of the contact object for which a transfer request is to be
      created, approved, rejected, or cancelled.

   - A <contact:authInfo> element that contains authorization
      information associated with the contact object.

   Every EPP <transfer> command MUST contain an "op" attribute that
   identifies the transfer operation to be performed as defined in
   [RFC4930].

   Example <transfer> request command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <transfer op="request">
C:      <contact:transfer
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact:authInfo>
C:          <contact:pw>2fooBAR</contact:pw>
C:        </contact:authInfo>
C:      </contact:transfer>
C:    </transfer>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

   When a <transfer> command has been processed successfully, the EPP
   <resData> element MUST contain a child <contact:trnData> element that
   identifies the contact namespace. The <contact:trnData> element
   contains the same child elements defined for a <transfer> query
   response.

Example <transfer> response:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1001">
S:      <msg>Command completed successfully; action pending</msg>
S:    </result>
S:    <resData>
S:      <contact:trnData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:trStatus>pending</contact:trStatus>
S:        <contact:reID>ClientX</contact:reID>
S:        <contact:reDate>2000-06-08T22:00:00.0Z</contact:reDate>
S:        <contact:acID>ClientY</contact:acID>
S:        <contact:acDate>2000-06-13T22:00:00.0Z</contact:acDate>
S:      </contact:trnData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54322-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

An EPP error response MUST be returned if a <transfer> command cannot
be processed for any reason.

3.2.5. EPP <update> Command

The EPP <update> command provides a transform operation that allows a
client to modify the attributes of a contact object. In addition to
the standard EPP command elements, the <update> command MUST contain
a <contact:update> element that identifies the contact namespace.
The <contact:update> element contains the following child elements:

- A <contact:id> element that contains the server-unique identifier
   of the contact object to be updated.

- An OPTIONAL <contact:add> element that contains attribute values
   to be added to the object.

- An OPTIONAL <contact:rem> element that contains attribute values
   to be removed from the object.

- An OPTIONAL <contact:chg> element that contains object attribute
   values to be changed.

At least one <contact:add>, <contact:rem>, or <contact:chg> element
MUST be provided if the command is not being extended. All of these
elements MAY be omitted if an <update> extension is present. The
<contact:add> and <contact:rem> elements contain the following child
elements:

- One or more <contact:status> elements that contain status values
  to be associated with or removed from the object. When specifying
  a value to be removed, only the attribute value is significant;
  element text is not required to match a value for removal.

A <contact:chg> element contains the following OPTIONAL child
elements. At least one child element MUST be present:

- One or two <contact:postalInfo> elements that contain postal
  address information. Two elements are provided so that address
  information can be provided in both internationalized and
  localized forms; a "type" attribute is used to identify the two
  forms. If an internationalized form (type="int") is provided,
  element content MUST be represented in a subset of UTF-8 that can
  be represented in the 7-bit US-ASCII character set. If a
  localized form (type="loc") is provided, element content MAY be
  represented in unrestricted UTF-8. The <contact:postalInfo>
  element contains the following OPTIONAL child elements:

  - A <contact:name> element that contains the name of the
    individual or role represented by the contact.

  - A <contact:org> element that contains the name of the
    organization with which the contact is affiliated.

  - A <contact:addr> element that contains address information
    associated with the contact. A <contact:addr> element contains
    the following child elements:

    - One, two, or three OPTIONAL <contact:street> elements that
      contain the contact's street address.

    - A <contact:city> element that contains the contact's city.

    - An OPTIONAL <contact:sp> element that contains the contact's
      state or province.

    - An OPTIONAL <contact:pc> element that contains the contact's
      postal code.

    - A <contact:cc> element that contains the contact's country
      code.

- A <contact:voice> element that contains the contact's voice
  telephone number.

- A <contact:fax> element that contains the contact's facsimile
  telephone number.

- A <contact:email> element that contains the contact's email
  address.

- A <contact:authInfo> element that contains authorization
  information associated with the contact object. This mapping
  includes a password-based authentication mechanism, but the schema
  allows new mechanisms to be defined in new schemas.

- A <contact:disclose> element that allows a client to identify
  elements that require exceptional server operator handling to
  allow or restrict disclosure to third parties. See Section 2.9
  for a description of the child elements contained within the
  <contact:disclose> element.

Example <update> command:

```
C:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
C:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
C:  <command>
C:    <update>
C:      <contact:update
C:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
C:        <contact:id>sh8013</contact:id>
C:        <contact:add>
C:          <contact:status s="clientDeleteProhibited"/>
C:        </contact:add>
C:        <contact:chg>
C:          <contact:postalInfo type="int">
C:            <contact:org/>
C:            <contact:addr>
C:              <contact:street>124 Example Dr.</contact:street>
C:              <contact:street>Suite 200</contact:street>
C:              <contact:city>Dulles</contact:city>
C:              <contact:sp>VA</contact:sp>
C:              <contact:pc>20166-6503</contact:pc>
C:              <contact:cc>US</contact:cc>
C:            </contact:addr>
C:          </contact:postalInfo>
C:          <contact:voice>+1.7034444444</contact:voice>
C:          <contact:fax/>
C:          <contact:authInfo>
C:            <contact:pw>2fooBAR</contact:pw>
C:          </contact:authInfo>
C:          <contact:disclose flag="1">
C:            <contact:voice/>
C:            <contact:email/>
C:          </contact:disclose>
C:        </contact:chg>
C:      </contact:update>
C:    </update>
C:    <clTRID>ABC-12345</clTRID>
C:  </command>
C:</epp>
```

When an <update> command has been processed successfully, a server
MUST respond with an EPP response with no <resData> element.

   Example <update> response:

   S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
   S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
   S: <response>
   S: <result code="1000">
   S: <msg>Command completed successfully</msg>
   S: </result>
   S: <trID>
   S: <clTRID>ABC-12345</clTRID>
   S: <svTRID>54321-XYZ</svTRID>
   S: </trID>
   S: </response>
   S:</epp>

   An EPP error response MUST be returned if an <update> command cannot
   be processed for any reason.

3.3. Offline Review of Requested Actions

   Commands are processed by a server in the order they are received
   from a client. Though an immediate response confirming receipt and
   processing of the command is produced by the server, a server
   operator MAY perform an offline review of requested transform
   commands before completing the requested action. In such situations,
   the response from the server MUST clearly note that the transform
   command has been received and processed, but the requested action is
   pending. The status of the corresponding object MUST clearly reflect
   processing of the pending action. The server MUST notify the client
   when offline processing of the action has been completed.

   Examples describing a <create> command that requires offline review
   are included here. Note the result code and message returned in
   response to the <create> command.

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1001">
S:      <msg>Command completed successfully; action pending</msg>
S:    </result>
S:    <resData>
S:      <contact:creData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id>sh8013</contact:id>
S:        <contact:crDate>1999-04-03T22:00:00.0Z</contact:crDate>
S:      </contact:creData>
S:    </resData>
S:    <trID>
S:      <clTRID>ABC-12345</clTRID>
S:      <svTRID>54321-XYZ</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

The status of the contact object after returning this response MUST
include "pendingCreate". The server operator reviews the request
offline, and informs the client of the outcome of the review either
by queuing a service message for retrieval via the <poll> command or
by using an out-of-band mechanism to inform the client of the
request.

The service message MUST contain text in the <response>, <msgQ>,
<msg> element that describes the notification. In addition, the EPP
<resData> element MUST contain a child <contact:panData> element that
identifies the contact namespace. The <contact:panData> element
contains the following child elements:

- A <contact:id> element that contains the server-unique identifier
  of the contact object. The <contact:id> element contains a
  REQUIRED "paResult" attribute. A positive boolean value indicates
  that the request has been approved and completed. A negative
  boolean value indicates that the request has been denied and the
  requested action has not been taken.

- A <contact:paTRID> element that contains the client transaction
  identifier and server transaction identifier returned with the
  original response to process the command. The client transaction
  identifier is OPTIONAL and will only be returned if the client
  provided an identifier with the original <create> command.

- A <contact:paDate> element that contains the date and time
  describing when review of the requested action was completed.

Example "review completed" service message:

```
S:<?xml version="1.0" encoding="UTF-8" standalone="no"?>
S:<epp xmlns="urn:ietf:params:xml:ns:epp-1.0">
S:  <response>
S:    <result code="1301">
S:      <msg>Command completed successfully; ack to dequeue</msg>
S:    </result>
S:    <msgQ count="5" id="12345">
S:      <qDate>1999-04-04T22:01:00.0Z</qDate>
S:      <msg>Pending action completed successfully.</msg>
S:    </msgQ>
S:    <resData>
S:      <contact:panData
S:       xmlns:contact="urn:ietf:params:xml:ns:contact-1.0">
S:        <contact:id paResult="1">sh8013</contact:id>
S:        <contact:paTRID>
S:          <clTRID>ABC-12345</clTRID>
S:          <svTRID>54321-XYZ</svTRID>
S:        </contact:paTRID>
S:        <contact:paDate>1999-04-04T22:00:00.0Z</contact:paDate>
S:      </contact:panData>
S:    </resData>
S:    <trID>
S:      <clTRID>BCD-23456</clTRID>
S:      <svTRID>65432-WXY</svTRID>
S:    </trID>
S:  </response>
S:</epp>
```

4. Formal Syntax

   An EPP object mapping is specified in XML Schema notation. The
   formal syntax presented here is a complete schema representation of
   the object mapping suitable for automated validation of EPP XML
   instances. The BEGIN and END tags are not part of the schema; they
   are used to note the beginning and ending of the schema for URI
   registration purposes.

   BEGIN
   <?xml version="1.0" encoding="UTF-8"?>

   <schema targetNamespace="urn:ietf:params:xml:ns:contact-1.0"
        xmlns:contact="urn:ietf:params:xml:ns:contact-1.0"
        xmlns:epp="urn:ietf:params:xml:ns:epp-1.0"
        xmlns:eppcom="urn:ietf:params:xml:ns:eppcom-1.0"
        xmlns="http://www.w3.org/2001/XMLSchema"
        elementFormDefault="qualified">

```
<!--
Import common element types.
-->
 <import namespace="urn:ietf:params:xml:ns:eppcom-1.0"/>
 <import namespace="urn:ietf:params:xml:ns:epp-1.0"/>

 <annotation>
   <documentation>
     Extensible Provisioning Protocol v1.0
     contact provisioning schema.
   </documentation>
 </annotation>

<!--
Child elements found in EPP commands.
-->
 <element name="check" type="contact:mIDType"/>
 <element name="create" type="contact:createType"/>
 <element name="delete" type="contact:sIDType"/>
 <element name="info" type="contact:authIDType"/>
 <element name="transfer" type="contact:authIDType"/>
 <element name="update" type="contact:updateType"/>

<!--
Utility types.
-->
 <simpleType name="ccType">
   <restriction base="token">
     <length value="2"/>
   </restriction>
 </simpleType>

 <complexType name="e164Type">
   <simpleContent>
     <extension base="contact:e164StringType">
       <attribute name="x" type="token"/>
     </extension>
   </simpleContent>
 </complexType>

 <simpleType name="e164StringType">
   <restriction base="token">
     <pattern value="(\+[0-9]{1,3}\.[0-9]{1,14})?"/>
     <maxLength value="17"/>
   </restriction>
 </simpleType>

 <simpleType name="pcType">
```

```
      <restriction base="token">
        <maxLength value="16"/>
      </restriction>
    </simpleType>

    <simpleType name="postalLineType">
      <restriction base="normalizedString">
        <minLength value="1"/>
        <maxLength value="255"/>
      </restriction>
    </simpleType>

    <simpleType name="optPostalLineType">
      <restriction base="normalizedString">
        <maxLength value="255"/>
      </restriction>
    </simpleType>

  <!--
  Child elements of the <create> command.
  -->
   <complexType name="createType">
     <sequence>
       <element name="id" type="eppcom:clIDType"/>
       <element name="postalInfo" type="contact:postalInfoType"
        maxOccurs="2"/>
       <element name="voice" type="contact:e164Type"
        minOccurs="0"/>
       <element name="fax" type="contact:e164Type"
        minOccurs="0"/>
       <element name="email" type="eppcom:minTokenType"/>
       <element name="authInfo" type="contact:authInfoType"/>
       <element name="disclose" type="contact:discloseType"
        minOccurs="0"/>
     </sequence>
   </complexType>

   <complexType name="postalInfoType">
     <sequence>
       <element name="name" type="contact:postalLineType"/>
       <element name="org" type="contact:optPostalLineType"
        minOccurs="0"/>
       <element name="addr" type="contact:addrType"/>
     </sequence>
     <attribute name="type" type="contact:postalInfoEnumType"
      use="required"/>
   </complexType>
```

```
<simpleType name="postalInfoEnumType">
  <restriction base="token">
    <enumeration value="loc"/>
    <enumeration value="int"/>
  </restriction>
</simpleType>

<complexType name="addrType">
  <sequence>
    <element name="street" type="contact:optPostalLineType"
     minOccurs="0" maxOccurs="3"/>
    <element name="city" type="contact:postalLineType"/>
    <element name="sp" type="contact:optPostalLineType"
     minOccurs="0"/>
    <element name="pc" type="contact:pcType"
     minOccurs="0"/>
    <element name="cc" type="contact:ccType"/>
  </sequence>
</complexType>

<complexType name="authInfoType">
  <choice>
    <element name="pw" type="eppcom:pwAuthInfoType"/>
    <element name="ext" type="eppcom:extAuthInfoType"/>
  </choice>
</complexType>

<complexType name="discloseType">
  <sequence>
    <element name="name" type="contact:intLocType"
     minOccurs="0" maxOccurs="2"/>
    <element name="org" type="contact:intLocType"
     minOccurs="0" maxOccurs="2"/>
    <element name="addr" type="contact:intLocType"
     minOccurs="0" maxOccurs="2"/>
    <element name="voice" minOccurs="0"/>
    <element name="fax" minOccurs="0"/>
    <element name="email" minOccurs="0"/>
  </sequence>
  <attribute name="flag" type="boolean" use="required"/>
</complexType>

<complexType name="intLocType">
  <attribute name="type" type="contact:postalInfoEnumType"
   use="required"/>
</complexType>
```

```
   <!--
   Child element of commands that require only an identifier.
   -->
    <complexType name="sIDType">
      <sequence>
        <element name="id" type="eppcom:clIDType"/>
      </sequence>
    </complexType>

   <!--
   Child element of commands that accept multiple identifiers.
   -->
    <complexType name="mIDType">
      <sequence>
        <element name="id" type="eppcom:clIDType"
         maxOccurs="unbounded"/>
      </sequence>
    </complexType>

   <!--
   Child elements of the <info> and <transfer> commands.
   -->
    <complexType name="authIDType">
      <sequence>
        <element name="id" type="eppcom:clIDType"/>
        <element name="authInfo" type="contact:authInfoType"
         minOccurs="0"/>
      </sequence>
    </complexType>

   <!--
   Child elements of the <update> command.
   -->
    <complexType name="updateType">
      <sequence>
        <element name="id" type="eppcom:clIDType"/>
        <element name="add" type="contact:addRemType"
         minOccurs="0"/>
        <element name="rem" type="contact:addRemType"
         minOccurs="0"/>
        <element name="chg" type="contact:chgType"
         minOccurs="0"/>
      </sequence>
    </complexType>

   <!--
   Data elements that can be added or removed.
   -->
```

```
    <complexType name="addRemType">
      <sequence>
        <element name="status" type="contact:statusType"
        maxOccurs="7"/>
      </sequence>
    </complexType>

  <!--
  Data elements that can be changed.
  -->
   <complexType name="chgType">
     <sequence>
       <element name="postalInfo" type="contact:chgPostalInfoType"
        minOccurs="0" maxOccurs="2"/>
       <element name="voice" type="contact:e164Type"
        minOccurs="0"/>
       <element name="fax" type="contact:e164Type"
        minOccurs="0"/>
       <element name="email" type="eppcom:minTokenType"
        minOccurs="0"/>
       <element name="authInfo" type="contact:authInfoType"
        minOccurs="0"/>
       <element name="disclose" type="contact:discloseType"
        minOccurs="0"/>
     </sequence>
   </complexType>

   <complexType name="chgPostalInfoType">
     <sequence>
       <element name="name" type="contact:postalLineType"
        minOccurs="0"/>
       <element name="org" type="contact:optPostalLineType"
        minOccurs="0"/>
       <element name="addr" type="contact:addrType"
        minOccurs="0"/>
     </sequence>
     <attribute name="type" type="contact:postalInfoEnumType"
      use="required"/>
   </complexType>

  <!--
  Child response elements.
  -->
   <element name="chkData" type="contact:chkDataType"/>
   <element name="creData" type="contact:creDataType"/>
   <element name="infData" type="contact:infDataType"/>
   <element name="panData" type="contact:panDataType"/>
   <element name="trnData" type="contact:trnDataType"/>
```

```
   <!--
   <check> response elements.
   -->
    <complexType name="chkDataType">
      <sequence>
        <element name="cd" type="contact:checkType"
         maxOccurs="unbounded"/>
      </sequence>
    </complexType>

    <complexType name="checkType">
      <sequence>
        <element name="id" type="contact:checkIDType"/>
        <element name="reason" type="eppcom:reasonType"
         minOccurs="0"/>
      </sequence>
    </complexType>

    <complexType name="checkIDType">
      <simpleContent>
        <extension base="eppcom:clIDType">
          <attribute name="avail" type="boolean"
           use="required"/>
        </extension>
      </simpleContent>
    </complexType>

   <!--
   <create> response elements.
   -->
    <complexType name="creDataType">
      <sequence>
        <element name="id" type="eppcom:clIDType"/>
        <element name="crDate" type="dateTime"/>
      </sequence>
    </complexType>

   <!--
   <info> response elements.
   -->
    <complexType name="infDataType">
      <sequence>
        <element name="id" type="eppcom:clIDType"/>
        <element name="roid" type="eppcom:roidType"/>
        <element name="status" type="contact:statusType"
         maxOccurs="7"/>
        <element name="postalInfo" type="contact:postalInfoType"
         maxOccurs="2"/>
```

```
            <element name="voice" type="contact:e164Type"
             minOccurs="0"/>
            <element name="fax" type="contact:e164Type"
             minOccurs="0"/>
            <element name="email" type="eppcom:minTokenType"/>
            <element name="clID" type="eppcom:clIDType"/>
            <element name="crID" type="eppcom:clIDType"/>
            <element name="crDate" type="dateTime"/>
            <element name="upID" type="eppcom:clIDType"
             minOccurs="0"/>
            <element name="upDate" type="dateTime"
             minOccurs="0"/>
            <element name="trDate" type="dateTime"
             minOccurs="0"/>
            <element name="authInfo" type="contact:authInfoType"
             minOccurs="0"/>
            <element name="disclose" type="contact:discloseType"
             minOccurs="0"/>
          </sequence>
        </complexType>

    <!--
    Status is a combination of attributes and an optional human-readable
    message that may be expressed in languages other than English.
    -->
      <complexType name="statusType">
        <simpleContent>
          <extension base="normalizedString">
            <attribute name="s" type="contact:statusValueType"
             use="required"/>
            <attribute name="lang" type="language"
             default="en"/>
          </extension>
        </simpleContent>
      </complexType>

      <simpleType name="statusValueType">
        <restriction base="token">
          <enumeration value="clientDeleteProhibited"/>
          <enumeration value="clientTransferProhibited"/>
          <enumeration value="clientUpdateProhibited"/>
          <enumeration value="linked"/>
          <enumeration value="ok"/>
          <enumeration value="pendingCreate"/>
          <enumeration value="pendingDelete"/>
          <enumeration value="pendingTransfer"/>
          <enumeration value="pendingUpdate"/>
          <enumeration value="serverDeleteProhibited"/>
```

```
            <enumeration value="serverTransferProhibited"/>
            <enumeration value="serverUpdateProhibited"/>
        </restriction>
     </simpleType>

    <!--
    Pending action notification response elements.
    -->
     <complexType name="panDataType">
        <sequence>
          <element name="id" type="contact:paCLIDType"/>
          <element name="paTRID" type="epp:trIDType"/>
          <element name="paDate" type="dateTime"/>
        </sequence>
     </complexType>

     <complexType name="paCLIDType">
        <simpleContent>
          <extension base="eppcom:clIDType">
            <attribute name="paResult" type="boolean"
             use="required"/>
          </extension>
        </simpleContent>
     </complexType>

    <!--
    <transfer> response elements.
    -->
     <complexType name="trnDataType">
        <sequence>
          <element name="id" type="eppcom:clIDType"/>
          <element name="trStatus" type="eppcom:trStatusType"/>
          <element name="reID" type="eppcom:clIDType"/>
          <element name="reDate" type="dateTime"/>
          <element name="acID" type="eppcom:clIDType"/>
          <element name="acDate" type="dateTime"/>
        </sequence>
     </complexType>

    <!--
    End of schema.
    -->
    </schema>
    END
```

5. Internationalization Considerations

   EPP is represented in XML, which provides native support for encoding
   information using the Unicode character set and its more compact
   representations, including UTF-8. Conformant XML processors
   recognize both UTF-8 and UTF-16 [RFC2781]. Though XML includes
   provisions to identify and use other character encodings through use
   of an "encoding" attribute in an <?xml?> declaration, use of UTF-8 is
   REQUIRED with this specification.

   All date-time values presented via EPP MUST be expressed in Universal
   Coordinated Time using the Gregorian calendar. The XML Schema allows
   use of time zone identifiers to indicate offsets from the zero
   meridian, but this option MUST NOT be used with EPP. The extended
   date-time form using upper case "T" and "Z" characters defined in
   [W3C.REC-xmlschema-2-20041028] MUST be used to represent date-time
   values as the XML Schema does not support truncated date-time forms
   or lower case "T" and "Z" characters.

   Humans, organizations, and other entities often need to represent
   social information in both a commonly understood character set and a
   locally optimized character set. This specification provides
   features allowing representation of social information in both a
   subset of UTF-8 for broad readability and unrestricted UTF-8 for
   local optimization.

6. IANA Considerations

   This document uses URNs to describe XML namespaces and XML schemas
   conforming to a registry mechanism described in [RFC3688]. Two URI
   assignments have been registered by the IANA.

   Registration request for the contact namespace:

   URI: urn:ietf:params:xml:ns:contact-1.0

   Registrant Contact: See the "Author's Address" section of this
   document.

   XML: None. Namespace URIs do not represent an XML specification.

   Registration request for the contact XML schema:

   URI: urn:ietf:params:xml:schema:contact-1.0

   Registrant Contact: See the "Author's Address" section of this
   document.

    XML: See the "Formal Syntax" section of this document.

7. Security Considerations

    Authorization information as described in Section 2.8 is REQUIRED to
    create a contact object. This information is used in some query and
    transfer operations as an additional means of determining client
    authorization to perform the command. Failure to protect
    authorization information from inadvertent disclosure can result in
    unauthorized transfer operations and unauthorized information
    release. Both client and server MUST ensure that authorization
    information is stored and exchanged with high-grade encryption
    mechanisms to provide privacy services.

    The object mapping described in this document does not provide any
    other security services or introduce any additional considerations
    beyond those described by [RFC4930] and protocol layers used by EPP.

8. Acknowledgements

    This document was originally written as an individual submission
    Internet-Draft. The PROVREG working group later adopted it as a
    working group document and provided many invaluable comments and
    suggested improvements. The author wishes to acknowledge the efforts
    of WG chairs Edward Lewis and Jaap Akkerhuis for their process and
    editorial contributions.

    Specific suggestions that have been incorporated into this document
    were provided by Chris Bason, Eric Brunner-Williams, Jordyn Buchanan,
    Robert Burbidge, Dave Crocker, Ayesha Damaraju, Anthony Eden, Sheer
    El-Showk, Dipankar Ghosh, Klaus Malorny, Dan Manley, Michael
    Mealling, Patrick Mevzek, Asbjorn Steira, and Rick Wesson.

9. References

9.1. Normative References

    [ISO.3166.1997]
                International Organization for Standardization, "Codes for
                the representation of names of countries and their
                subdivisions -- Part 1: Country codes", ISO Standard 3166,
                October 1997.

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

   [RFC0822] Crocker, D., "Standard for the format of ARPA Internet
              text messages", STD 11, RFC 822, August 1982.

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

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

   [RFC3688] Mealling, M., "The IETF XML Registry", BCP 81, RFC 3688,
              January 2004.

   [RFC4930] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)",
              RFC 4930, May 2007.

   [W3C.REC-xml-20040204]
              Yergeau, F., Maler, E., Sperberg-McQueen, C., Bray, T.,
              and J. Paoli, "Extensible Markup Language (XML) 1.0 (Third
              Edition)", World Wide Web Consortium FirstEdition REC-xml-
              20040204, February 2004,
              <http://www.w3.org/TR/2004/REC-xml-20040204>.

   [W3C.REC-xmlschema-1-20041028]
              Thompson, H., Maloney, M., Mendelsohn, N., and D. Beech,
              "XML Schema Part 1: Structures Second Edition", World Wide
              Web Consortium Recommendation REC-xmlschema-1-20041028,
              October 2004,
              <http://www.w3.org/TR/2004/REC-xmlschema-1-20041028>.

   [W3C.REC-xmlschema-2-20041028]
              Biron, P. and A. Malhotra, "XML Schema Part 2: Datatypes
              Second Edition", World Wide Web Consortium
              Recommendation REC-xmlschema-2-20041028, October 2004,
              <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028>.

9.2. Informative References

   [RFC2781] Hoffman, P. and F. Yergeau, "UTF-16, an encoding of ISO
              10646", RFC 2781, February 2000.

   [RFC3733] Hollenbeck, S., "Extensible Provisioning Protocol (EPP)
              Contact Mapping", RFC 3733, March 2004.

Appendix A. Changes from RFC 3733

   1.   Minor reformatting as a result of converting I-D source format
        from nroff to XML.

   2.   Removed this text from Section 2.2:

        "With one exception, transform commands MUST be rejected when a
        pendingCreate, pendingDelete, pendingTransfer, or pendingUpdate
        status is set. The only exception is that a <transfer> command
        to approve, reject, or cancel a transfer MAY be processed while
        an object is in "pendingTransfer" status."

   3.   Fixed examples in Section 2.9 (added missing "/" characters).

   4.   Changed text in Section 3.1.3 from "A <contact:acID> element
        that contains the identifier of the client that SHOULD act upon
        the transfer request" to "A <contact:acID> element that contains
        the identifier of the client that SHOULD act upon a PENDING
        transfer request. For all other status types, the value
        identifies the client that took the indicated action".

   5.   Fixed the example response in Section 3.2.4 to use the correct
        response code and response text.

   6.   Changed text in Section 3.2.5 from "At least one <contact:add>,
        <contact:rem>, or <contact:chg> element MUST be provided." to
        "At least one <contact:add>, <contact:rem>, or <contact:chg>
        element MUST be provided if the command is not being extended.
        All of these elements MAY be omitted if an <update> extension is
        present."

   7.   Changed text in Section 3.3 (old Section 3.2.6) from this:

        "The server operator reviews the request offline, and informs
        the client of the outcome of the review by queuing a service
        message for retrieval via the <poll> command."

        to this:

        "The server operator reviews the request offline, and informs
        the client of the outcome of the review either by queuing a
        service message for retrieval via the <poll> command or by using
        an out-of-band mechanism to inform the client of the request."

   8.   Removed text describing use of the XML Schema schemaLocation
        attribute. This is an optional attribute that doesn't need to
        be mandated for use in EPP.

9.   Updated text describing a requirement to use UTF-8 encoding in
     the "Internationalization Considerations" section.

10. Removed references to RFC 3339 and replaced them with references
     to the W3C XML Schema specification.

11. Updated the E.164 reference.

12. Updated EPP and XML references. Updated reference from RFC 2279
     to RFC 3629.

Author's Address

   Scott Hollenbeck
   VeriSign, Inc.
   21345 Ridgetop Circle
   Dulles, VA 20166-6503
   US

   EMail: shollenbeck@verisign.com