

# Extensible Provisioning Protocol (EPP) v1.0 Registrar Acceptance Criteria



November 17, 2006  
Version 1.0.5

Technical Support: [techsupport@pir.org](mailto:techsupport@pir.org) / +1.416.646.3308  
<http://www.pir.org>

# Contents

## 1. Introduction

- 1.1 Purpose
- 1.2 Formatting Conventions
- 1.3 Accounts
- 1.4 Additional Requirements
- 1.5 Successful Command & Test Completion
- 1.6 Passing the Test
- 1.7 Contact and Nameserver Policy

## 2. EPP Communications

### 2.1 Session Management

- 2.1.1 Start Session
- 2.1.2 Authentication
- 2.1.3 Change Password

### 2.2 Domain Name Operations

- 2.2.1 Create Domain with 4 Contacts and 2 Name Servers
- 2.2.2 Create Domain with Minimum Registration Period
- 2.2.3 Create Domain with Maximum Registration Period
- 2.2.4 Create Domain with Maximum Number of Name Servers
- 2.2.5 Create Domain with Minimum Length Domain Name (3 Chars + .org)
- 2.2.6 Create Domain with Maximum Length Domain Name (63 Chars + .org)
- 2.2.7 Check Domain (Domain Not Available)
- 2.2.8 Check Domain (Domain Available)
- 2.2.9 Check Domain with Maximum Length Domain Name (Domain Not Available)
- 2.2.10 Query Domain
- 2.2.11 Delete Domain
- 2.2.12 Renew Domain
- 2.2.13 Renew Domain to Maximum Registration Period
- 2.2.14 Transfer a Domain
- 2.2.15 Approval Domain Transfer
- 2.2.16 Rejection Domain Transfer
- 2.2.17 Change Domain Name Servers
- 2.2.18 Change Domain Contact
- 2.2.19 Change Domain Status

### 2.3 Name Server Operations

- 2.3.1 Create Name Server
- 2.3.2 Create Name Server with Maximum Length Host Name (63 Chars + "." + 63 Chars + ".org")
- 2.3.3 Create Name Server with Maximum Allowable IPs
- 2.3.4 Create Nameserver (Foreign Registry)
- 2.3.5 Check Name Server (Name Server Known)

- 2.3.6 Check Name Server (Name Server Unknown)
- 2.3.7 Query Name Server
- 2.3.8 Delete Name Server
- 2.3.9 Change Name Server (Add IP Address)
- 2.3.10 Change Name Server (Remove IP Address)

## **2.4 Contact Operations**

- 2.4.1 Create Contact
- 2.4.2 Check Contact (Contact Known)
- 2.4.3 Check Contact (Contact Unknown)
- 2.4.4 Query Contact
- 2.4.5 Transfer Contact CONFIDENTIAL
- 2.4.6 Query Contact Transfer Status
- 2.4.7 Approve Contact Transfer
- 2.4.8 Reject Contact Transfer
- 2.4.9 Change Contact (Change Element)
- 2.4.10 Change Contact (Remove Element)
- 2.4.11 Delete Contact

## **2.5 Client Error Handling**

- 2.5.1 Correctly Handle 2003 Exception
- 2.5.2 Correctly Handle 2005 Exception
- 2.5.3 Correctly Handle 2306 Exception
- 2.5.4 Correctly Handle 2105 Exception
- 2.5.5 Correctly Handle 2303 Exception
- 2.5.6 Correctly Handle 2305 Exception
- 2.5.7 Correctly Handle 2201 Exception

## **2.6 Empty Element Commands**

- 2.6.1 Keep Session Alive
- 2.6.2 Request Message Queue Information
- 2.6.3 Ack Queued Message

## **2.7 End Session**

## **Appendix A - Seeded Registry information**

This document is made available to the registrars that have entered into Registry-Registrar Agreements with Public Interest Registry (PIR), manager of the registry of .ORG. The contents of this document are proprietary information of Afilias, Limited. This information may be used by recipient only for the purpose for which it was transmitted and shall be returned upon request to PIR or when no longer needed by recipient.

Contents Copyright Afilias Limited 2006  
All rights reserved

## 1. Introduction

### 1.1 Purpose

This document describes the basic operations that a Registrar's client application must perform to be accepted by the Registry. Each of the following sections describes the actions that the client must perform to demonstrate correct implementation of the Extensible Provisioning Protocol (EPP) v1.0 and interactions with the Registry. Registrars should have a detailed knowledge of the following internet RFCs before attempting the test:

*Extensible Provisioning Protocol (EPP) RFC: 3730*

*Extensible Provisioning Protocol Domain Name Mapping RFC: 3731*

*Extensible Provisioning Protocol Host Mapping RFC: 3732*

*Extensible Provisioning Protocol Contact Mapping RFC: 3733*

*Extensible Provisioning Protocol Transport Over TCP RFC: 3734*

The tests presented herein verify the correct interface with the Registry for standard Registrar operations. They do not cover all possible error and unusual conditions. The Registrar client application is responsible for correctly handling all unusual error conditions.

### 1.2 Formatting Conventions

Proper completion of the test requires that all commands and data must be entered exactly as given in this document. Any deviations will be considered a failure. The following items show the formatting conventions included in this document for required input and output values and for variable input and output responses.

Regular text in this format represents expected system input and output values that the client system will send to the server and that the server system will display in response to an action or actions provided by the Registrar. The following example illustrates an expected system output.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

When **bold** text is located in Regular text, this represents a required input value that the Registrar must provide - the Registrar must enter the text exactly as shown. The following example illustrates the format for the required input values.

Domain Name: **epptest1.org**

Italicized text in output represents data returned from the server, which may or may not be the exact values represented in this document. It is the responsibility of the client to interpret these values properly and possibly reuse these for subsequent commands.

```
<domain:exDate>2004-02-11T22:07:28.0Z</domain:exDate>
```

### 1.3 Accounts

For the duration of the test, the Registrar will use a seeded test account, called ClientB. The Registrar will provide the Registry Technical Support Group with a valid email address. Standard registry transfer notifications, processed by the registry during the initial test seeding (\*\*see appendix for details\*\*), will be sent to this e-mail address for Registrar reference. Upon the scheduling of a test, the Registry Technical Support Group will provide hostnames and port numbers for the Registrar's client connection.

#### 1.4 Additional Requirements

Registry Operator will prime the Test Registry with data required to complete this test. Please refer to Appendix A if you wish to review this data. Do not attempt to enter this data into the Test Registry.

#### 1.5 Successful Command & Test Completion

While performing this test, if the response to a command is not exactly as shown, then stop your test and contact PIR technical support.

#### 1.6 Passing the Test

The Registrar must complete the test perfectly (with no typos or breaking the sequence of operations) from start to finish, including the session keep alive portion in section 2.6, within the allotted time.

#### 1.7 Contact and Name Server Policy Requirements

There are certain policies that are enforced in the .org implementation of EPP:

A minimum of 4 contacts (including 1 Registrant and at least 1 of each Admin, Billing and Technical contacts) must be provided during the create domain transaction.

For the purpose of this test, all domains must be created with at least 2 name servers. Registrars may, however, when working with the "live" registry, create domains with fewer than 2 name servers, though DNS resolution depends upon a minimum of 2 assigned name servers.

## 2. EPP Communications

Registrar to Registry communications utilize the Extensible Provisioning Protocol (EPP) mapped over TCP (Transport Control Protocol). EPP commands are formulated using the Extensible Markup Language (XML). The Registrars' application client must utilize XML to send commands to the Registry and utilize an XML parser to interpret the server's responses. EPP itself relies exclusively upon user authentication for security. Additional security is provided by the use of Transport Layer Security (TLS), for session cryptography. Clients must communicate with the EPP server using a commercial or open source implementation of TLS, such as OpenSSL. Additional information concerning mapping EPP over TCP is available in 'Extensible Provisioning Protocol

Transport Over TCP RFC 3734. Additional information concerning the TLS may be found in RFC 2246.

## 2.1 Session Management

### 2.1.1 Start Session

After making an initial connection to the Registry, the server shall reply with a greeting. A Registrar must receive the greeting message before attempting authentication and/or other supplementary commands.

### 2.1.2 Authentication

After the initial greeting the registrar client shall send the Login command to authenticate itself to the test registry with the following information:

Client ID: **ClientB**  
Password: **foo-BAR2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.1.3 Change Password

To change a Registrar's password, an additional field is required in the Login command. At this point, the client must logout (keep the connection open), then log back in, and pass the following information to the Login command.

Client ID: **ClientB**  
Password: **foo-BAR2**  
New Password: **bar-FOO2**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.2 Domain Name Operations

The following tests exercises EPP commands that revolve around Domain Name creation and manipulation.

### 2.2.1 Create Domain with 4 Contacts and 2 Name Servers

Create a new Domain and associate 2 Name Servers and 4 Contacts to it by supplying the following elements to the Create command.

Domain Name: **eppvalid1.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID : **EPPOTE-C2**  
Domain Admin contact ID : **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.2 Create Domain with Minimum Registration Period

Enter the following data to the Create command.

Domain Name: **epp1year.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Domain Period (Years): **1**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<domain:exDate>2004-02-11T22:07:28.0Z</domain:exDate>
```

### 2.2.3 Create Domain with Maximum Registration Period

Enter the following data to the Create command.

Domain Name: **epp10years.org**  
Domain Server: **ns1.eppvalid.org**

Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Domain Period (Years): **10**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<domain:exDate>2013-02-11T22:07:28.0Z</domain:exDate>
```

#### 2.2.4 Create Domain with Maximum Number of Name Servers

Using the Create command, add the following Domain along with the thirteen Name Servers.

Domain Name: **eppmaxhost.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Name Server: **ns1.eppvalid.org**  
Name Server: **ns2.eppvalid.org**  
Name Server: **ns3.eppvalid.org**  
Name Server: **ns4.eppvalid.org**  
Name Server: **ns5.eppvalid.org**  
Name Server: **ns6.eppvalid.org**  
Name Server: **ns7.eppvalid.org**  
Name Server: **ns8.eppvalid.org**  
Name Server: **ns9.eppvalid.org**  
Name Server: **ns10.eppvalid.org**  
Name Server: **ns11.eppvalid.org**  
Name Server: **ns12.eppvalid.org**  
Name Server: **ns13.eppvalid.org**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.5 Create Domain with Minimum Length Domain Name (3 Chars + .org)

Using the Create command, pass the following data.

Domain Name: **epp.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

### 2.2.6 Create Domain with Maximum Length Domain Name (63 Chars + .org)

Using the Create command, enter the following data.

Domain Name:  
**eppabcdefghijklmnopqrstuvwxyabcdefghijklmnopqrstuvwxyabcdefghijklmnop  
g**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

### 2.2.7 Check Domain (Domain Not Available)

Use the Check command with the following data to determine that the domain is not available:

Domain Name: **eppvalid.org**



Verify that the following response is received:

Domain Name: eppinfodomain.org  
Client ID: ClientB  
Domain Status: ok  
Domain Contact (Registrant) ID: EPPOTE-C2  
Domain Admin Contact: EPPOTE-C3  
Domain Billing Contact: EPPOTE-C4  
Domain Technical Contact: EPPOTE-C5  
Domain Name Server: ns1.eppinfodomain.org  
Domain Name Server: ns2.eppinfodomain.org  
Domain Host Server: ns1.eppinfodomain.org  
Domain Host Server: ns2.eppinfodomain.org  
Auth info: my secret  
Created By: ClientB  
Created Date: 1999-04-03T22:00:00.0Z  
Expiration Date: 2005-04-03T22:00:00.0Z  
Last Updated By: ClientB

You will need the Admin Contact ID for section 2.2.15.

### 2.2.11 Delete Domain

Issue the Delete command with the following data:

Domain Name: **eppdeleteme.org**

Verify that the following response is received:

```
<result code='1001'><msg lang='en-US'>Command completed  
successfully</msg>
```

### 2.2.12 Renew Domain

First, get the Expiration Date of the Domain by issuing the info command with the following data.

Domain Name: **epprenewable.org**

Examine the Expiration Date returned from the previous command (output should be similar to the following).

Domain Expiration Date: *2005-04-03T22:00:00.0Z*

Issue the Renew command with the following data.

Domain Name: **epprenewable.org**

Current Expiration Date: *2005-04-03* (returned in the previous info command)

Domain Years Period: **5**

Verify the output so that the expected Expiration Date is correct.

Domain Name: epprenewable.org

Expiration Date: *2010-04-03T22:00:00.0Z*

### 2.2.13 Renew Domain to Maximum Registration Period

Enter the following information to the Renew command.

Domain Name: **epprenewable.org**

Current Expiration Date: *2010-04-03* (returned in the previous renew command)

Domain Period (Years): **3**

Verify the change by issuing the info command and compare the output here.

Domain Name: epprenewable.org

Expiration Date: *2013-04-03T22:00:00.0Z*

### 2.2.14 Transfer a Domain

Use the Transfer command with the op="request" attribute, with the following information to request a transfer of a domain

Domain Name: **eppttransfer1.org**

Auth info: **my secretY**

Verify that the following response is received:

```
<result code='1001'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.15 Approve Domain Transfer

Another Registrar has made a transfer request for one of ClientB's domains. This section involves the approval of this transfer request. Check the status of the transfer using the Transfer command with the op="query" attribute and the following information:

Domain Name: **eppttransfer2.org**  
Auth info: **my secretX**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

Transfer Status: pending

Approve the transfer by using the Transfer command with the op="approve" attribute and the following information:

Domain Name: **eppttransfer2.org**  
Auth info: **my secretX**

Verify the following output:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.16 Reject Domain Transfer

Another Registrar has made a transfer request for one of ClientB's domain's. This section involves the rejection of this transfer request. Reject the transfer by using the Transfer command with the op="reject" attribute and the following information:

Domain Name: **eppttransfer3.org**  
Auth org: **my secretX**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.2.17 Change Domain Name Servers

Enter the following information to the Update command.

Domain Name: **eppinfodomain.org**

Remove Name Server: **ns1.eppinfodomain.org**

Remove Name Server: **ns2.eppinfodomain.org**

Add Name Server: **ns1.eppvalid.org**

Add Name Server: **ns2.eppvalid.org**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

### 2.2.18 Change Domain Contact

Issue the Update command with the following data. Remove the Admin Contact that was shown in section 2.2.10 and add a new Contact (this new contact already exists - it was seeded in the database by the registry operator prior to the test).

Domain Name: **eppinfodomain.org**

Remove Admin Contact ID: XXXX-00 (get this ID from section 2.2.10)

Add Admin Contact ID: **EPPOTE-C4**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

### 2.2.19 Change Domain Status

Change the status of a Domain by issuing the Update command with the following values.

Domain Name: **eppinfodomain.org**

Add Domain Status: **clientUpdateProhibited**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

## 2.3 Name Server Operations

The following tests exercise EPP commands that revolve around Domain Name Server creation and manipulation.

### 2.3.1 Create Name Server

Supply the following information to the Create command.

Host: **ns1.eppnewname.org**  
Host Address: **192.168.10.11**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.2 Create Name Server with Maximum Length Host Name (63 Chars + "." + 63 Chars + ".org")

Supply the following information to the Create command.

Host: **abcdefghijklmnopqrstuvwxyabcdefghijklmnopqrstuvwxyabcdefghijklmnop  
.eppabcdefghijklmnopqrstuvwxyabcdefghijklmnopqrstuvwxyabcdefghijklmnop  
g**  
Host Address: **192.168.10.12**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.3 Create Name Server with Maximum Allowable IPs

Supply the following information to the create command:

Host: **ns2.eppnewname.org**  
Host Address: **192.168.10.1**  
Host Address: **192.168.10.2**  
Host Address: **192.168.10.3**  
Host Address: **192.168.10.4**  
Host Address: **192.168.10.5**  
Host Address: **192.168.10.6**  
Host Address: **192.168.10.7**  
Host Address: **192.168.10.8**  
Host Address: **192.168.10.9**  
Host Address: **192.168.10.10**

Host Address: **192.168.10.11**  
Host Address: **192.168.10.12**  
Host Address: **192.168.10.13**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

2.3.4 Create Name Server (Foreign Registry)  
Supply the following to the create command:

Host: **ns1.eppvalid.com**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

2.3.5 Check Name Server (Name Server Known)

Use the Check command to check for a Name Server.

Host Name: **ns1.eppvalid.org**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<host:name avail='0'>
```

2.3.6 Check Name Server (Name Server Unknown)

Enter the following data to the Check command.

Host Name: **ns1.eppavailable.org**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<host:name avail='1'>
```

### 2.3.7 Query Name Server

Supply the following values to the info command.

Host Name: **ns1.eppvalid.org**

Verify that the following response is received:

Host Name: ns1.eppvalid.org  
Client ID: ClientB  
Host IP Address: 192.168.2.11  
Created By: ClientB  
Created Date: 1999-04-03T22:00:00.0Z  
Client Trans ID: 11AA  
Server Trans ID: 22BB  
Status: ok

### 2.3.8 Delete Name Server

Use the Delete command with the following values.

Host Name: **ns1.eppdelns.org**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.9 Change Name Server (Add IP Address)

Supply the following information to the Update command.

Host Name: **ns12.eppvalid.org**  
Add IP Address: **192.1.2.3**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

### 2.3.10 Change Name Server (Remove IP Address)

Supply the following information to the Update command.

Host Name: **ns12.eppvalid.org**  
Remove IP Address: **192.1.2.3**

Verify that the following response is received.

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

## 2.4 Contact Operations

The following tests exercises EPP commands that revolve around Contact creation and manipulation.

### 2.4.1 Create Contact

Supply the following information to the Create command.

Contact ID: **EPPOTE-C6**  
Contact Name: **John Doe**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jdoe@eppvalid.org**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

The contact id submitted with this create command will be necessary for the completion of sections 2.4.2, 2.4.4, 2.4.7, and 2.4.8

### 2.4.2 Check Contact (Contact Known)

Use the Check command with the following arguments.

ID: XXXX-00 (use the Contact ID from section 2.4.1)

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<contact:id  
avail='0'>
```

### 2.4.3 Check Contact (Contact Unknown)

Use the Check command with the following arguments.

**ID: EPPOTE-C8**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
<contact:id avail='1'>
```

### 2.4.4 Query Contact

Supply the following information to the org command. Use the ID created in section 2.4.1.

ID: XXXX-00 (use the ContactID from the contact created in 2.4.1)

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

```
ContactID: XXXX-00  
Contact Name: John Doe  
Contact Organization: Example Corp, Inc  
Contact Address Street1: 123 Example St.  
Contact Address Street2: Suite 100  
Contact Address City: Anytown  
Contact Address State/Province: Any Prov  
Contact Address Postal Code: A1A1A1  
Contact Address Country: CA  
Contact Voice: +1.4165555555
```

Contact Voice Extension: 1111  
Contact Fax: +1.4165555556  
Contact Email: jdoe@eppvalid.org  
Auth info: my secret  
Status: ok

#### 2.4.5 Transfer Contact

This section tests the client's ability to request the transfer of a contact owned by another registrar. Please note that this contact was seeded in the database by PIR prior to the start of the test. Supply the following information to the Transfer command with the op="request" attribute and the following information.

ID: **EPPOTE-C7**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

#### 2.4.6 Query Contact Transfer Status

Use the Transfer command's op="query" attribute, along with the following information.

ID: **EPPOTE-C7**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

Contact Transfer Status: pending

#### 2.4.7 Approve Contact Transfer

Another Registrar has an outstanding Transfer Request of one of ClientB's Contacts. This section involves the approval of the transfer request. Supply the following information to the Transfer command with the op="approve" attribute.

ID: **EPPOTE-C1-approv**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.4.8 Reject Contact Transfer

Another Registrar has an outstanding Transfer Request of one of ClientB's Contacts. This section involves the rejection of the transfer request. Supply the following information to the Transfer command with the op="reject" attribute.

ID: **EPPOTE-C1-reject**

Auth info: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.4.9 Change Contact (Change Element)

Supply the following information to the Update command. Use the ID created in section 2.4.1.

ID: XXXX-00 (use the ID from the contact created in 2.4.1)

Contact Name: **Mr. Contact**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.4.10 Change Contact (Remove Element)

Supply the following information to the Update command. To remove a value, overwrite it as a NULL value. Use the ID created in section 2.4.1.

ID: XXXX-00 (use the ID from the contact created in 2.4.1) Contact Fax:

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg>
```

#### 2.4.11 Delete Contact

First, create a simple Contact by supplying the following information to the Create command:

Contact ID: **EPPOTE-C8**  
Contact Name: **Delete Me**  
Contact Organization: **Example Corp. Inc**  
Contact Address Street1: **123 Example St.**  
Contact Address Street2: **Suite 100**  
Contact Address City: **Anytown**  
Contact Address State/Province: **Any Prov**  
Contact Address Postal Code: **A1A1A1**  
Contact Address Country: **CA**  
Contact Voice: **+1.4165555555**  
Contact Voice Extension: **1111**  
Contact Fax: **+1.4165555556**  
Contact Email: **jdoe@eppvalid.org**  
Auth org: **my secret**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

Now delete the Contact by supplying the following information to the Contact Delete command.

ID: **EPPOTE-C8**

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed  
successfully</msg>
```

## 2.5 Client Error Handling

The following section exercises the client's ability to correctly handle common EPP exceptions. The client should remain connected to the Test Registry despite the receipt of exceptions. A definition of each exception code is provided.

### 2.5.1 Correctly Handle 2003 Exception

2003 "Required parameter missing" This response code **MUST** be returned when

a server receives a command for which a required parameter value has not been provided.

Submit the following using the create command (do NOT submit a value for auth info):

Domain Name: **eppexception.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**

Verify that the following response is received:

```
<result code='2003'><msg lang='en-US'>Required parameter missing</msg>
```

### 2.5.2 Correctly Handle 2005 Exception

2005 "Parameter value syntax error" This response code MUST be returned when a server receives a command containing a parameter whose value is improperly formed. The error value SHOULD be returned via an element in the EPP response.

Submit the following using the create command:

Domain Name: **-\*eppiaminvalid.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='2005'><msg lang='en-US'>Parameter value syntax error</msg>
```

### 2.5.3 Correctly Handle 2306 Exception

2306 "Parameter value policy error" This response code MUST be returned when a server receives a command containing a parameter value that is syntactically valid, but semantically invalid due to local policy. For example, the server MAY support a subset of a range of valid protocol parameter values. The error value

SHOULD be returned via an element in the EPP response. Submit the following using the create command:

Domain Name: **eppexception.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C2**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**  
Domain Period (Years): **99**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='2306'><msg lang='en-US'>Parameter value policy error</msg>
```

#### 2.5.4 Correctly Handle 2105 Exception

2105 "Object is not eligible for renewal" This response code MUST be returned when a client attempts to an object that is not eligible for renewal in accordance with server policy.

Submit the following using the renew command:

Domain Name: **eppinfodomain.org**  
Expiration Date: **2014-10-01**

Verify that the following response is received:

```
<result code='2105'><msg lang='en-US'>Object is not eligible for renewal</msg>
```

#### 2.5.5 Correctly Handle 2303 Exception

2303 "Object does not exist" This response code MUST be returned when a server receives a command to transform an object that does not exist in the repository.

Submit the following using the create command:

Domain Name: **eppexception.org**  
Domain Server: **ns1.eppvalid.org**  
Domain Server: **ns2.eppvalid.org**  
Domain registrant contact ID: **EPPOTE-C99**  
Domain Admin contact ID: **EPPOTE-C3**  
Domain Billing contact ID: **EPPOTE-C4**  
Domain Technical contact ID: **EPPOTE-C5**

Domain Period (Years): **2**  
Auth info: **my secret**

Verify that the following response is received:

```
<result code='2303'><msg lang='en-US'>Object does not exist</msg>
```

### 2.5.6 Correctly Handle 2305 Exception

2305 "Object association prohibits operation" This response code **MUST** be returned when a server receives a command to transform an object that can not be completed due to dependencies on other objects that are associated with the target object. For example, a server **MAY** disallow commands while an object has active associations with other objects.

Submit the following to the delete command:

Contact ID: **EPPOTE-C2**

Verify that the following response is received:

```
<result code='2305'><msg lang='en-US'>Object association prohibits operation</msg>
```

### 2.5.7 Correctly Handle 2201 Exception

2201 "Authorization error" This response code **MUST** be returned when a server notes a client authorization error when executing a command. This error is used to note that a client lacks privileges to execute the requested command.

Submit the following to the delete command:

Domain Name: **eppttransfer2.org**

Verify that the following response is received:

```
<result code='2201'><msg lang='en-US'>Authorization error</msg>
```

## 2.6 Empty Element Commands

This section exercise the client's ability to utilize commands that must be represented as empty elements, with no child objects.

### 2.6.1 Keep Session Alive

For this test, the client must keep the current session open to the Registry for at

least 30 minutes before proceeding to Section 2.6.2. Use the Hello command at intervals under 10 minutes to maintain client connectivity.

## 2.6.2 Request Message Queue Information

Clients may use the poll command to retrieve messages queued by the server. Issue the poll command with the op=request attribute to retrieve queue information, and the first message within the queue.

Verify that the following response is received:

```
<response><result code='1301'><msg lang='en-US'>Command completed successfully; ack to dequeue</msg></result><msgQ count='48' id='43'><msg lang='en-US'>Transfer Requested.</msg>
```

Note: the value returned for 'id' will be necessary for section 2.6.3

## 2.6.3 Ack Queued Message

Issue the poll command with the op=ack attribute to acknowledge receipt of the first message, and remove it from the queue.

Verify that the following response is received:

```
<result code='1000'><msg lang='en-US'>Command completed successfully</msg></result><msgQ count='47' id='45'>
```

## 2.7 End Session

For a client Registrar to end communications with the Registry, the Logout command is used with no arguments.

If successful, the Registry will send the following response and then end the session.

```
<result code='1500'><msg lang='en-US'>Command completed successfully; ending session</msg>
```

At this point, contact PIR Technical Support at techsupport@pir.org or +1.4166463308 and inform them you have completed this test.

## Appendix A - Seeded Registry information

The OT&E test requires the creation and manipulation of several EPP objects

prior to the client's initial connection. PIR will perform the necessary operations before the client's initial connection. The data within this appendix is included for informational purposes only.

\*\*\* Registrar: Do not attempt to enter this data into the Test Registry. \*\*\*

Registrar: ClientB  
Password: foo-BAR2

Contacts: The seven seeded contacts use the following common values:

Contact ID: XXXX-00  
Contact Name: Test Contact  
Contact Organization: Example Corp. Inc  
Contact Address Street: 123 Example St.  
Contact Address Street: Suite 100  
Contact Address City: Anytown  
Contact Address State/Province: Any Prov  
Contact Address Postal Code: A1A1A1  
Contact Address Country: CA  
Contact Voice: +1.4165555555  
Contact Voice Extension: 1111  
Contact Fax: +1.4165555556  
Contact Email: jdoe@ppvalid.org  
Auth info: my secret

The unique contact ID values for each of the seven seeded contacts are as follows:

ID: EPPOTE-C7  
Owned by ClientY

ID: EPPOTE-C1-approv  
Owned by ClientB

\*\*This contact has pending transfer status, initiated by ClientY\*\*

ID: EPPOTE-C1-reject  
Owned by ClientB

\*\*This contact has pending transfer status, initiated by ClientY\*\*

ID: EPPOTE-C2  
Owned by ClientB

ID: EPPOTE-C3

Owned by ClientB

ID: EPPOTE-C4  
Owned by ClientB

ID: EPPOTE-C5  
Owned by ClientB

Domain: eppdelns.org  
Owned by ClientB

Host: ns1.eppdelns.org  
Owned by ClientB

Domain: eppvalid.org  
Owned by: ClientB

Host: ns1.eppvalid.org 192.168.10.11 Owned by ClientB  
Host: ns2.eppvalid.org 192.168.10.12 Owned by ClientB  
Host: ns3.eppvalid.org 192.168.10.13 Owned by ClientB  
Host: ns4.eppvalid.org 192.168.10.14 Owned by ClientB  
Host: ns5.eppvalid.org 192.168.10.15 Owned by ClientB  
Host: ns6.eppvalid.org 192.168.10.16 Owned by ClientB  
Host: ns7.eppvalid.org 192.168.10.17 Owned by ClientB  
Host: ns8.eppvalid.org 192.168.10.18 Owned by ClientB  
Host: ns9.eppvalid.org 192.168.10.19 Owned by ClientB  
Host: ns10.eppvalid.org 192.168.10.20 Owned by ClientB  
Host: ns11.eppvalid.org 192.168.10.21 Owned by ClientB  
Host: ns12.eppvalid.org 192.168.10.22 Owned by ClientB  
Host: ns13.eppvalid.org 192.168.10.23 Owned by ClientB

Domain: eppinfodomain.org  
Owned by ClientB

Hosts: ns1.eppinfodomain.org  
ns2.eppinfodomain.org

Domain: eppdeleteme.org  
Owned by ClientB

Domain: epprenewable.org  
Owned by ClientB

Domain: eppnewname.org  
Owned by ClientB

Domain: eptransfer1.org

Auth info: my secretY

\*\* Owned by ClientY \*\*

Domain: eppttransfer2.org

\*\* This domain has pending transfer status, initiated by ClientY\*\*

Auth Info: my secretX

Owned by ClientB

Domain: eppttransfer3.org

\*\*This domain has pending transfer status, initiated by ClientY\*\*

Auth ID: my secretX

Owned by ClientB