



**Domain Name Registration Reseller**

**Application Programming Interface**

**Reference Manual**

# Table of Contents

## 1.0 Introduction

- 1.1 Technology
- 1.2 General Requirements

## 2.0 Structure and Rules

- 2.1 Request and Response
- 2.2 XML Structure
- 2.3 XSD Schema
  - 2.3.1 Request-type XSD files
  - 2.3.2 Response-type XSD files

## 3.0 Functionality

- 3.1 Overview
- 3.2 Register Domain
- 3.3 Transfer Domain
- 3.4 Renew Domain
- 3.5 Delete Domain
- 3.6 Modify Domain
- 3.7 Check Domain Availability
- 3.8 Obtain Whois Information
- 3.9 Complex Combinations

## 4.0 Response Code Reference

- 4.1 Error Response Code
- 4.2 Registry Response Code

## Appendix A Request and response XML examples

- Register Domain
- Transfer Domain
- Renew Domain
- Delete Domain
- Modify Domain
- Check Domain Availability
- Obtain Whois Information
- Complex Request
- Example Responses

## Appendix B Request and response XSD

## ● 1.0 Introduction

Welcome to a new way of reselling Domain Name Registration and supporting services. This guide will take you through the steps required to develop your own custom domain name registration system where you will be the provider of essential Internet related services to your customers. The DomainPeople Application Programming Interface (DomainPeople API) is used by resellers to integrate DomainPeople's services into their web site, providing domain name registration services to their customer base while using their own unique look and style.

The DomainPeople API is easy to incorporate into an existing web site and connects to our API server using a simple SSL connection. We have included sample code and provide an online resource area to help resellers save time and money by making the DomainPeople API easy to use. The time it takes to customize a site using the DomainPeople API depends on the programmer's experience and the existing functionality of the web site. Factors such as an existing e-commerce solution will speed up the process. Additional time will be required if you are just starting to develop an e-commerce site from the ground up.

### **Domain Name Registration (DNR) Services**

The DomainPeople API enables resellers to offer their customers DNR services including the registration, renewal, transfer, and modification of domain name registration.

### **Supported TLDs**

The following domain extensions can be offered through the DomainPeople API:

- .com
- .net
- .org
- .info
- .biz
- .name
- .us
- .ca
- .cn

**Registration:** You can offer domain registration to your customers for any of the TLDs listed above. Please Note: Some TLDs have specific requirements that a Registrant must meet such as a US nexus requirement for .us domains, and a Canadian Presence for .ca domains. Please refer to our online resource area for more details.

**Renewals:** Your customers can renew their domain names at anytime for periods up to a maximum allowable registration term defined by the Registry. Most domains can be registered for a maximum of 10 years.

**Transfers:** Registrants can transfer their domain name from one registrar to another. An additional year is added to the registration when this service is used.

**Domain Modifications:** Modifications to the domain name registration data can be easily maintained and managed through the API. Your customers can modify their Registrant, admin, technical, and billing contacts, plus the name server information through an online interface that you provide them with on your site.

## **Additional Services**

DomainPeople will be adding new features and services that will be made available to resellers using the DomainPeople API. Please check the online reseller resource area for descriptions of upcoming new services and features. Deployment dates will be emailed to you well in advance to afford you adequate time to market these products and prepare your systems.

## **How the DomainPeople API works**

The DomainPeople API defines a set of commands that are relayed to and from the DomainPeople registration servers to facilitate the domain name registration and additional services needs of your customers. Commands are sent using standard XML communication over a persistent SSL connection. You design your CGI program to take the data submitted through your web site's order forms and include that data within the XML commands sent them over an SSL connection to the DomainPeople DNR server. The command access is validated and the request is fulfilled by returning the results over the same SSL connection.

### **1.1 Technology**

The two major technologies that are being used are: SSL, and XML (which includes XSD).

SSL is a "Secure Sockets Layer" technology. It is the industry-standard method for protecting web communications. The SSL security protocol provides data encryption, server authentication, message integrity, and optional client authentication for a TCP/IP connection.

XML is the way by which the API Server accepts requests to be performed. The formats for XML documents that are accepted by the API Server are found in the XSD files. XSD is a "XML Schema definition" language, and it defines the rules, as to what a valid XML document looks like.

### **1.2 General Requirements**

There are two general requirements to be able to use the API Server, namely: being able to HTTP Post over an SSL connection and being able to generate an XML document.

Both posting over an SSL connection and XML documents are platform independent, hence there are no restrictions on which operating system to use.

## ● 2.0 Structure and Rules

### 2.1 Request and Response

For every Request sent to the API Server to execute a command, there is one corresponding Response. Both Request and Response will be in XML format, see section 2.2 for details on the structures.

There are four different types of Request: RegRequest, CheckRequest, WhoisRequest, or DomainModRequest. View the XSD files for each request type, or the Examples in Section 5, to see the structure of the request object. (Also, refer to section 2.3 for a brief description of what an XSD file is).

There are two Response types: StatusResponse and WhoisResponse.

A StatusResponse is returned in response to a RegRequest, CheckRequest or DomainModRequest. A StatusRequest contains statusItem children elements that contain information about a single domain that was passed in the Request. The code and message fields provided will indicate if the Request was successful or not. If not successful the message field should be able to indicate what went wrong and why.

A WhoisResponse contains Whois information about a domain found in the DomainPeople, Inc WHOIS database. This information provided in the WhoisResponse includes creation, modification and expiration date, contact and name server information.

### 2.2 XML Structure

All Request objects must be in an XML format.

The root of each Request will be an 'API Protocol' element, which will contain information about the: version, Schema instance, and Schema location (the name of the corresponding xsd file).

Nested in each API Protocol element is the request-type, which contains the user's login name and password.

Each request-type has sub-elements that contain the data used by the API Server when it processes the request.

View the XSD files for each request-type, or the Examples in Section 5, to see the different elements and attributes that need to be supplied.

### 2.3 XSD Schema

XSD stands for XML Schema definition language. An XSD is a schema definition for an XML instance document. That means that when creating an XML Document you

need to refer to its schema to see what elements and attributes are required and where they are required in the document.

From the definition of the request structure, found in its xsd file, one should be able to devise an XML document that will be valid, when passed to the API Server.

We have seven different xsd files: apitypes, checkrequest, domainmodrequest, regrequest, whoisrequest, statusresponse and whoisresponse.

### 2.3.1 Request-type XSD files

The checkrequest.xsd, domainmodrequest.xsd, regrequest.xsd, and whoisrequest.xsd files include the file apitypes.xsd. Apitypes.xsd declares all the generic elements that aren't request-type specific. For example, the file includes the schema for a contact element which is used in many request and response bodies.

In each of checkrequest, domainmodrequest, regrequest and whoisrequest xsd files, we have specific information about the structure of the given request. Also indicated are, which sub-elements are required and which aren't.

### 2.3.2 Response-type XSD files

In each of: statusresponse and whoisresponse xsd files, we have specific information about the structure of the given response. Also indicated are, which sub-elements are required and which aren't.

## ● 3.0 Functionality

### 3.1 Overview

The DomainPeople API consists of 7 functions or actions. They are register, transfer, renew, delete, modify domain, check domain availability and obtain Whois information.

Each action or a combination of actions are sent to the DomainPeople API server in the form of an XML document. This constitutes a request.

In return, there is one XML response for each request made to the DomainPeople API server.

The following table maps the correct request and response XML for each function or action of the DomainPeople API.

	request	response
Register Domain	regrequest.xml	statusresponse.xml
Transfer Domain	regrequest.xml	statusresponse.xml
Renew Domain	regrequest.xml	statusresponse.xml
Delete Domain	regrequest.xml	statusresponse.xml
Modify Domain	domainmodrequest.xml	statusresponse.xml
Domain Avail.	checkrequest.xml	statusresponse.xml
Whois Info	whoisrequest.xml	whoisresponse.xml

Each of the above XML requests or responses is strictly governed by a corresponding XSD available in Appendix B - Request and Response XSD

It should be noted that attributes and elements of each XML request or response must be kept in the order as they appear in the XSD.

The DomainPeople API OTE service is available for testing at [raindance.domainpeople.com](https://raindance.domainpeople.com), port 2000. Requests and responses are communicated through HTTPS POST only.

The procedures set out in section 3.2 through 3.9 follows the same pattern. Gather the necessary information. Fill out and send the XML document to the DomainPeople API server.

With the exception of check domain availability, all successful operation will return an error code of 3000. Any error code other than 3000 indicates operation failure. In which case, check the verbose error message for details.

As for check domain availability, 3022 indicates domain available, 3023 indicates domain unavailable, and 3024 indicates that an error occurred and the domain availability status is unknown.

Reference section 4.1 and 4.2 for a list of possible return codes and return messages.

## 3.2 Register Domain

To register a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] Login name for the customer account. (optional – not supported at this time)
- [c] Name servers for the domain(s) to be registered. (optional)
- [d] Contact information for registrant, admin, technical and billing.
- [e] Each contact information has the following contact data.
  - First Name
  - Last Name
  - Organization
  - Address 1
  - Address 2 (optional)
  - City
  - State
  - Country in 2 letter ISO format
  - Postal Code
  - Telephone Number
  - Fax Number (optional)
  - Email
  - Language Code in 2 letter ISO format
- [f] One or more domain names and the term for each.
- [g] Optionally attach a reference id.

For certain TLDs such as .CA, .US and .PRO, additional information such as legal type, purpose or Nexus category must be provided. Consult apitypes.xsd in appendix B for specifics.

With the above information, populate the XML document to be sent. The correct XML format to use for domain registration is defined in regrequest.xml. The action attribute of each domain element should be set to "register".

Reference appendix A - Register Domain for an example with the following information.

```
DomainPeople Partner Login [username]
DomainPeople Partner Password [88bear99]
Customer Login [johnsmith]
Name Server 1 [ns1.netnation.com]
Name Server 2 [ns2.netnation.com]
Registrant Contact Information:
- First Name [john]
- Last Name [smith]
- Organization [john smith]
- Address 1 [4850 12 - Ave]
- Address 2 []
- City [Vancouver]
- State [BC]
- Country [CA]
- Postal Code[V7E4X3]
```

- Telephone Number [604 5551212]
- Fax Number []
- Email [johnsmith@example.com]
- Language Code [EN]

Admin Contact Information [Same as registrant contact]

Technical Contact Information [Same as registrant contact]

Billing Contact Information [Same as registrant contact]

Domain Name / Registration Term: [john-smith-domain.com / 1]

Domain Name / Registration Term: [john-smith-domain.net / 1]

Domain Name / Registration Term: [john-smith-domain.org / 1]

Reference Number: [A12345678]

### 3.3 Transfer Domain

To transfer a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] Login name for the customer account. (optional – not supported at this time)
- [c] Name servers for the domain(s) to be transferred. (optional – not supported)
- [d] Contact information for registrant, admin, technical and billing.
- [e] Each contact information has the following contact data.
  - First Name
  - Last Name
  - Organization
  - Address 1
  - Address 2 (optional)
  - City
  - State
  - Country in 2 letter ISO format
  - Postal Code
  - Telephone Number
  - Fax Number (optional)
  - Email
  - Language Code in 2 letter ISO format
- [f] One or more domain names to be transferred.
- [g] Optionally attach a reference id.

For certain TLDs such as .CA, .US and .PRO, additional information such as legal type, purpose or Nexus category must be provided. Consult apitypes.xsd in appendix B for specifics.

With the above information, populate the XML document to be sent. The correct XML format to use for domain transfer is defined in regrequest.xml. The action attribute of each domain element should be set to "transfer".

For security reason, some registries require a password in order to authorize the transfer of a domain. In which case, specify the password as the value of the authInfo attribute of each domain element.

The registration term is extended by one year once the transfer completes. It is not possible to add more than a single year in the transfer process.

Reference appendix A - Transfer Domain for an example with the following information.

```
DomainPeople Partner Login [username]
DomainPeople Partner Password [88bear99]
Customer Login [johnsmith]
Name Server 1 [ns1.netnation.com]
Name Server 2 [ns2.netnation.com]
Registrant Contact Information:
- First Name [john]
- Last Name [smith]
```

- Organization [john smith]
- Address 1 [4850 12 - ave]
- Address 2 []
- City [Vancouver]
- State [BC]
- Country [CA]
- Postal Code[V7E4X3]
- Telephone Number [604 5551212]
- Fax Number []
- Email [johnsmith@example.com]
- Language Code [EN]

Admin Contact Information [Same as registrant contact]

Technical Contact Information [Same as registrant contact]

Billing Contact Information [Same as registrant contact]

Domain Name / Registration Term: [john-smith-domain.com / 1]

Domain Name / Registration Term: [john-smith-domain.net / 1]

Domain Name / Registration Term: [john-smith-domain.org / 1]

Reference Number: [A12345678]

### 3.4 Renew Domain

To renew a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] Login name for the customer account. (optional – not supported at this time)
- [c] One or more domain names to be renewed, the number of years to renew and the old expiry date for each domain name.
- [d] Optionally attach a reference id.

Note that contacts and name servers are not required.

With the above information, populate the XML document to be sent. The correct XML format to use for domain renewal is defined in `regrequest.xml`. The action attribute of each domain element should be set to "renew".

Reference appendix A - Renew Domain for an example with the following information.

DomainPeople Partner Login [username]  
DomainPeople Partner Password [88bear99]  
Customer Login [johnsmith]  
Domain Name / Registration Term / Old Expiry Date:  
[john-smith-domain.com / 1 / 2007-10-09]  
Reference Number: [A12345678]

### 3.5 Delete Domain

To delete a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] Login name for the customer account. (optional – not supported at this time)
- [c] One or more domain names to be deleted.
- [d] Optionally attach a reference id.

Note that contacts and name servers are not required.

With the above information, populate the XML document to be sent. The correct XML format to use for domain deletion is defined in `regrequest.xml`. The action attribute of each domain element should be set to "delete".

Reference appendix A - Delete Domain for an example with the following information.

DomainPeople Partner Login [username]  
DomainPeople Partner Password [88bear99]  
Customer Login [johnsmith]  
Domain Name [john-smith-domain.com]  
Reference Number: [A12345678]

## 3.6 Modify Domain

To modify a domain or multiple domains contact info and/or name servers, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] Login name for the customer account.
- [c] Name servers for the domain(s) to be modified. (optional)
- [d] Contact information for registrant, admin, technical or billing. (optional)
- [e] Each contact information has the following contact data.
  - First Name
  - Last Name
  - Organization
  - Address 1
  - Address 2 (optional)
  - City
  - State
  - Country in 2 letter ISO format
  - Postal Code
  - Telephone Number
  - Fax Number (optional)
  - Email
  - Language Code in 2 letter ISO format
- [f] One or more domain names to be modified.
- [g] Optionally attach a reference id.

For certain TLDs such as .CA, .US and .PRO, additional information such as legal type, purpose or Nexus category must be provided. Consult apitypes.xsd in appendix B for specifics.

With the above information, populate the XML document to be sent. The correct XML format to use for domain modification is defined in domainmodrequest.xml.

*Please note, that the Registration Organization name can only be changed using RNCA.*

Reference appendix A - Modify Domain for an example with the following information.

DomainPeople Partner Login [username]  
DomainPeople Partner Password [88bear99]  
Customer Login [johnsmith]  
Name Server 1 [ns1.netnation.com]  
Name Server 2 [ns2.netnation.com]  
Registrant Contact Information:

- First Name [john]
- Last Name [smith]
- Organization [john smith]
- Address 1 [4850 12 - ave]
- Address 2 []
- City [Vancouver]

- State [BC]
- Country [CA]
- Postal Code[V7E4X3]
- Telephone Number [604 5551212]
- Fax Number []
- Email [johnsmith@example.com]
- Language Code [EN]

Admin Contact Information [Same as registrant contact]

Technical Contact Information [Same as registrant contact]

Billing Contact Information [Same as registrant contact]

Domain Name / Registration Term: [john-smith-domain.com / 1]

Domain Name / Registration Term: [john-smith-domain.net / 1]

Domain Name / Registration Term: [john-smith-domain.org / 1]

Reference Number: [A12345678]

### 3.7 Check Domain Availability

To check the availability of a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] One or more domain names to be checked.
- [c] Optionally attach a reference id.

With the above information, populate the XML document to be sent. The correct XML format to use for check domain availability is defined in checkrequest.xml.

Reference appendix A - Check Domain Availability for an example with the following information.

DomainPeople Partner Login [username]  
DomainPeople Partner Password [88bear99]  
Domain Name [john-smith-domain.com]  
Domain Name [john-smith-domain.net]  
Domain Name [john-smith-domain.org]  
Reference Number: [A12345678]

### 3.8 Obtain Whois Information

To obtain Whois information of a domain or multiple domains, the following information is required.

- [a] Login name and password for your DomainPeople partner account.
- [b] One or more domain names to be looked up.
- [c] Optionally attach a reference id.

With the above information, populate the XML document to be sent. The correct XML format to use for Whois information is defined in whoisrequest.xml.

Reference appendix A - Obtain Whois Information for an example with the following information.

DomainPeople Partner Login [username]  
DomainPeople Partner Password [88bear99]  
Domain Name [john-smith-domain.com]  
Domain Name [john-smith-domain.net]  
Domain Name [john-smith-domain.org]  
Reference Number: [A12345678]

### 3.9 Complex Combinations

The DomainPeople API has the capability to accept mixed complex requests. A complex request can combine multiple functions or actions into a single XML request.

Documentation for complex action combinations is pending.

## ● 4.0 Response Code Reference

### 4.1 Error Response Code

- 3000 : Success
- 3001 : No Such customer
- 3002 : Invalid Account Login
- 3003 : No Such Contact ID
- 3004 : System Error
- 3005 : No Such domain
- 3006 : Corba Server Down
- 3007 : Corba Server Error
- 3008 : The request timed out
- 3009 : Too many requests from this partner
- 3010 : Server capacity has been reached, try later
- 3011 : The request that you sent was empty
- 3012 : Missing legal type for .ca domain
- 3013 : Missing NEXUS\_CATEGORY for .us domain
- 3014 : Missing add purpose code
- 3015 : XML Error
- 3016 : Invalid term was passed with the domain
- 3017 : Saving contact failed
- 3018 : Since you live in Canada or the US your phone number needs to be 10 digits in length
- 3019 : Missing/invalid legal type for .ca registration in registrant contact
- 3021 : Missing/invalid addPurpose or nexusCategory for .us registration in registrant contact
- 3022 : This domain is available
- 3023 : This domain is taken
- 3024 : System unavailable, domain status unknown
- 3025 : No contact info was entered
- 3026 : Could not get the domain's price, possible non supported domain extension
- 3027 : useCustomer should be set when copying a customer contact
- 3028 : Invalid domain status, or the domain might not be registered
  
- 3100 : Invalid Input
- 3101 : Invalid Country Code
- 3102 : Invalid Organization Name
- 3103 : Invalid First Name
- 3104 : Invalid Last Name
- 3105 : Invalid Phone
- 3106 : Invalid Address1
- 3107 : Invalid City
- 3108 : Invalid State/Province
- 3109 : Invalid Zip/Postal Code
- 3110 : Invalid Email address

## 4.2 Registry Response Code

Additional errors maybe returned from the registry. They follow the EPP response code standard.

<http://www.ietf.org/internet-drafts/draft-ietf-provreg-epp-09.txt>

- x0xx -- Protocol syntax
- x1xx -- Server policy and implementation rules
- x2xx -- Security and authorization
- x3xx -- Data management
- x4xx -- Server and internal errors
- x5xx -- Connection management

## ● Appendix A Request and response XML examples

### Register Domain

#### Request:

```
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="regrequest.xsd">
  <regRequest password="88bear99" user="username">
    <useCustomer login="johnsmith"/>
    <nameserver name="ns1.domainpeople.com"/>
    <nameserver name="ns2.domainpeople.com"/>
    <contact type="registrant">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
        country="CA" languageCode="en" organization="Philip Rittscher3"
        state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="admin">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
        country="CA" email="johnsmith@example.com" fax="" firstName="John"
        languageCode="en" lastName="Smith" organization="Philip Rittscher3"
        phone="604-5551-212" state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="technical">
      <contactref><copy>admin</copy></contactref>
    </contact>
    <contact type="billing">
      <contactref><copy>admin</copy></contactref>
    </contact>
    <domain action="register" name="john-smith-domain.com"
      oldexpiry="2003-10-10" term="1"/>
    <domain action="register" name="john-smith-domain.net"
      oldexpiry="2003-10-10" term="1"/>
    <domain action="register" name="john-smith-domain.org"
      oldexpiry="2003-10-10" term="1"/>
  </regRequest>
</apiProtocol>
```

#### Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067296267864"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.com"
    objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.org"
    objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.net"
    objType="domain"/>
</apiProtocol>
```

## Transfer Domain

### Transfer Request:

```
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="regrequest.xsd">
  <regRequest password="88bear99" user="username">
    <useCustomer login="johnsmith"/>
    <nameserver name="ns1.domainpeople.com"/>
    <nameserver name="ns2.domainpeople.com"/>
    <contact type="registrant">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
        country="CA" languageCode="en" organization="Philip Rittscher3"
        state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="admin">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
        country="CA" email="johnsmith@example.com" fax="" firstName="John"
        languageCode="en" lastName="Smith" organization="Philip Rittscher3"
        phone="604-5551-212" state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="technical">
      <contactref><copy>admin</copy></contactref>
    </contact>
    <contact type="billing">
      <contactref><copy>admin</copy></contactref>
    </contact>
    <domain action="transfer" name="john-smith-domain.com"
      oldexpiry="2003-10-10" term="1"/>
    <domain action="transfer" name="john-smith-domain.net"
      oldexpiry="2003-10-10" term="1"/>
    <domain action="transfer" name="john-smith-domain.org"
      oldexpiry="2003-10-10" term="1"/>
  </regRequest>
</apiProtocol>
```

### Transfer Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067296267892"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.net"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.org"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.com"
objType="domain"/>
</apiProtocol>
```

## Renew Domain

### Renew Request:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067296267895"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.org"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.net"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.com"
objType="domain"/>
</apiProtocol>
```

[mirrong@disco etc]\$ less renew\_example.xml

```
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="regrequest.xsd">
  <regRequest password="88bear99" user="username">
    <useCustomer login="johnsmith"/>
    <domain action="renew" name="john-smith-domain.com"
oldexpiry="2004-10-27" term="1"/>
    <domain action="renew" name="john-smith-domain.net"
oldexpiry="2004-10-27" term="1"/>
    <domain action="renew" name="john-smith-domain.org"
oldexpiry="2004-10-27" term="1"/>
  </regRequest>
</apiProtocol>
```

### Renew Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067296267895"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.org"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.net"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.com"
objType="domain"/>
</apiProtocol>
```

## Delete Domain

### Delete Request:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="regrequest.xsd">
  <regRequest password="88bear99" user="username">
    <domain action="delete" name="john-smith-domain.com"/>
    <domain action="delete" name="john-smith-domain.net"/>
    <domain action="delete" name="john-smith-domain.org"/>
  </regRequest>
</apiProtocol>
```

```
</regRequest>
</apiProtocol>
```

### Delete Response:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067364314401"/>
  <statusItem code="3000" message="Success" objName="john-smith-domain.com"
objType="domain"/>
  <statusItem code="2303" message="Object does not exist"
  objName="john-smith-domain.org" objType="domain"/>
  <statusItem code="3000" message="Success" objName="john-smith-domain.net"
objType="domain"/>
</apiProtocol>
```

## Modify Domain

### Modify Request:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="domainmodrequest.xsd">
  <domainModRequest password="88bear99" reference="123:abc"
user="username">
  <domain name="dp-test-test-username.com"/>
  <useCustomer login="johnsmith"/>
  <nameserver name="ns1.domainpeople.com"/>
  <nameserver name="ns2.domainpeople.com"/>
  <contact type="admin">
  <contactdata address1="123 12th Ave" address2="" city="Vancouver"
country="CA" email="john_smith@username.com" fax="" firstName="John"
languageCode="en" lastName="Smith" organization="John Smith Inc."
phone="6045435464" state="BC" zip="v7f4h4"/>
  </contact>
  <contact type="technical">
  <contactdata address1="123 12th Ave" address2="" city="Vancouver"
country="CA" email="john_smith@username.com" fax="" firstName="John"
languageCode="en" lastName="Smith" organization="John Smith Inc."
phone="6045435464" state="BC" zip="v7f4h4"/>
  </contact>
  <contact type="billing">
  <contactdata address1="123 12th Ave" address2="" city="Vancouver"
country="CA" email="john_smith@username.com" fax="" firstName="John"
languageCode="en" lastName="Smith" organization="John Smith Inc."
phone="6045435464" state="BC" zip="v7f4h4"/>
  </contact>
  <contact type="registrant">
  <contactdata address1="123 12th Ave" address2="" city="Vancouver"
country="CA" languageCode="en" organization="John Smith Inc." state="BC"
zip="v7f4h4"/>
  </contact>
```

```
</domainModRequest>
</apiProtocol>
```

#### **Modify Response:**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"><statusResponse
reference="123:abc" requestId="1067364314399"/><statusItem code="3000"
message="Success" objName="dp-test-test-username.com"
objType="domain"/></apiProtocol>
```

## **Check Domain Availability**

#### **Check Domain Availability Request:**

```
<?xml version="1.0" encoding="UTF-8"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation='checkrequest.xsd'>
<checkRequest user="username" password="88bear99" reference="12312:aSD">
  <domain name="john-smith-domain.com"/>
  <domain name="john-smith-domain.net"/>
  <domain name="john-smith-domain.org"/>
</checkRequest>
</apiProtocol>
```

#### **Check Domain Availability Response:**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse reference="12312:aSD" requestId="1067296267896"/>
  <statusItem code="-23" message="This domain is taken"
  objName="john-smith-domain.com" objType="domain"/>
  <statusItem code="-24" message="System unavailable, domain status unknown"
  objName="john-smith-domain.net" objType="domain"/>
  <statusItem code="-22" message="This domain is available"
  objName="john-smith-domain.org" objType="domain"/>
</apiProtocol>
```

## **Obtain Whois Information**

#### **Whois Request:**

```
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="whoisRequest.xsd">
  <whoisRequest password="88bear99" user="username">
    <domain name="john-smith-domain.com"/>
  </whoisRequest>
</apiProtocol>
```

#### **Whois Response:**

```
<?xml version="1.0" encoding="ISO-8859-1"?>
```

```

<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <whoisResponse requestId="1067307947338"/>
  <domain created="2003-10-27" domainId="412667" expires="2005-10-27"
  name="john-smith-domain.com" updated="2003-10-27">
    <contact type="admin">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" email="johnsmith@example.com" firstName="John"
      languageCode="en" lastName="Smith" organization="Philip Rittscher3"
      phone="6045551212" state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="billing">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" email="johnsmith@example.com" firstName="John"
      languageCode="en" lastName="Smith" organization="Philip Rittscher3"
      phone="6045551212" state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="registrant">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" email="" firstName="" languageCode="en" lastName=""
      organization="Philip Rittscher3" phone="" state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="technical">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" email="johnsmith@example.com" firstName="John"
      languageCode="en" lastName="Smith" organization="Philip Rittscher3"
      phone="6045551212" state="BC" zip="v7e4x3"/>
    </contact>
  </domain>
</apiProtocol>

```

## Complex Request

### Complex Request:

```

<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="regrequest.xsd">
  <regRequest password="88bear99" user="username">
    <useCustomer login="johnsmith"/>
    <nameserver name="ns1.domainpeople.com"/>
    <nameserver name="ns2.domainpeople.com"/>
    <contact type="registrant">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" languageCode="en" organization="Philip Rittscher3"
      state="BC" zip="v7e4x3"/>
    </contact>
    <contact type="admin">
      <contactdata address1="4850 12 - ave" address2="" city="Vancouver"
      country="CA" email="johnsmith@example.com" fax="" firstName="John"
      languageCode="en" lastName="Smith" organization="Philip Rittscher3"
      phone="604-5551-212" state="BC" zip="v7e4x3"/>
    </contact>
  </regRequest>
</apiProtocol>

```

```

<contact type="technical">
  <contactref><copy>admin</copy></contactref>
</contact>
<contact type="billing">
  <contactref><copy>admin</copy></contactref>
</contact>
<domain action="register" name="john-smith-domain.biz"
  oldexpiry="2003-10-10" term="1"/>
<domain action="transfer" name="john-smith-domain.info"
  oldexpiry="2003-10-10" term="1" authInfo="myAuthInfo"/>
</regRequest>
</apiProtocol>

```

### Complex Response:

```

<?xml version="1.0" encoding="ISO-8859-1"?>
<apiProtocol version="1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:noNamespaceSchemaLocation="statusresponse.xsd">
  <statusResponse requestId="1067307947339"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.info"
objType="domain"/>
  <statusItem code="0" message="Success" objName="john-smith-domain.biz"
objType="domain"/>
</apiProtocol>

```

## Example Responses

### Registered a domain successfully:

```

<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"> <statusResponse
reference="123:abc" requestId="1067283503398"/> <statusItem code="0"
message="Success" objName="mirron1067283496076.com"
objType="domain"/></apiProtocol>

```

### Registering a domain with no contact, failure:

```

<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <statusItem code="-
25" message="No contact info was entered" objName="admin" objType="contact"/>
<statusItem code="-25" message="No contact info was entered" objName="billing"
objType="contact"/> <statusItem code="-25" message="No contact info was
entered" objName="registrant" objType="contact"/> <statusItem code="-25"
message="No contact info was entered" objName="technical"
objType="contact"/></apiProtocol>

```

### Transferring a domain successfully:

```

<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"> <statusResponse
reference="123:abc" requestId="1067283503400"/> <statusItem code="0"
message="Success" objName="mirron1067283801676.com"
objType="domain"/></apiProtocol>

```

**Transferring a domain with no contact, failure:**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <statusItem code="-
25" message="No contact info was entered" objName="admin" objType="contact"/>
<statusItem code="-25" message="No contact info was entered" objName="billing"
objType="contact"/> <statusItem code="-25" message="No contact info was
entered" objName="registrant" objType="contact"/> <statusItem code="-25"
message="No contact info was entered" objName="technical"
objType="contact"/></apiProtocol>
```

**Renewing a domain successfully:**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"> <statusResponse
reference="123:abc" requestId="1067283503402"/> <statusItem code="0"
message="Success" objName="mirron1066924885881.com"
objType="domain"/></apiProtocol>
```

**Renewing a domain for an invalid term (there mistake):**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <error>
<causetext>cvc-minInclusive-valid: Value '-1' is not facet-valid with respect to
minInclusive '1' for type 'positiveInteger'.</causetext> <text>XML Error:cvc-
minInclusive-valid: Value '-1' is not facet-valid with respect to minInclusive '1' for
type 'positiveInteger'.</text> <code>-15</code> <!--
domainpeople.api.ApiHandler.run(ApiHandler.java:168)java.lang.Thread.run(Thread.
java:534)CAUSED BY:org.apache.xerces.parsers.DOMParser.parse(Unknown
Source)domainpeople.api.ApiParser.parse(ApiParser.java:85)domainpeople.api.ApiHa
ndler.run(ApiHandler.java:95)java.lang.Thread.run(Thread.java:534)-->
</error></apiProtocol>
```

**Applying a check request for a domain that is taken:**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"> <statusResponse
reference="123:abc" requestId="1067283503408"/> <statusItem code="-23"
message="This domain is taken" objName="ontheveranda.com"
objType="domain"/></apiProtocol>
```

**Applying a check request for a domain with an extension that we aren't registering:**

```
<?xml version="1.0" encoding="ISO-8859-1"?><apiProtocol version="1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="statusresponse.xsd"> <statusResponse
reference="123:abc" requestId="1067283503410"/> <statusItem code="-24"
message="System unavailable, domain status unknown" objName="friends.tv"
objType="domain"/></apiProtocol>
```

## ● Appendix B Request and response XSD

### apitypes.xsd

```
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
<xs:simpleType name="languageType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="en" />
    <xs:enumeration value="fr" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="domainType">
  <xs:restriction base="xs:string">
    <xs:pattern value="([a-z0-9]|[a-z0-9][a-z0-9\-\]{0,61}[a-z0-9])\.[a-z]+" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="contactType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="admin" />
    <xs:enumeration value="technical" />
    <xs:enumeration value="billing" />
    <xs:enumeration value="registrant" />
    <xs:enumeration value="customer" />
  </xs:restriction>
</xs:simpleType>

  <xs:simpleType name="emailType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[a-zA-Z0-9_\-\.]{1,65}@(([a-z0-9]|[a-z0-9][a-z0-9\-\]{0,61}[a-z0-9])\.[a-z]+" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="countryType">
    <xs:restriction base="xs:string">
      <xs:pattern value="[A-Z]{2}" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="serviceType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="mailforward" />
      <xs:enumeration value="urlforward" />
      <xs:enumeration value="frameforward" />
      <xs:enumeration value="customdns" />
    </xs:restriction>
  </xs:simpleType>

  <xs:simpleType name="addPurposeType">
    <xs:restriction base="xs:string">
      <xs:pattern value="P[1-5]" />
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

```

<xs:simpleType name="legalTypeType">
  <xs:restriction base="xs:string">
    <xs:pattern
value="CCO|CCT|RES|GOV|EDU|ASS|PRT|TDM|TRS|TRD|LGR|OMK|LAM|HOP|PLT|A
BO|INB|MAJ" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="nexusCategoryType">
  <xs:restriction base="xs:string">
    <xs:pattern value="C11|C12|C21|C31|C32" />
  </xs:restriction>
</xs:simpleType>

<xs:element name="contactdata">
  <xs:complexType>
    <xs:attribute name="firstName" type="xs:string"
use='optional' />
    <xs:attribute name="lastName" type="xs:string"
use='optional' />
    <xs:attribute name="organization" type="xs:string"
use='required' />
    <xs:attribute name="address1" type="xs:string"
use='required' />
    <xs:attribute name="address2" type="xs:string"
use='optional' />
    <xs:attribute name="city" type="xs:string" use='required' />
    <xs:attribute name="state" type="xs:string" use='required' />
    <xs:attribute name="country" type="xs:string"
use='required' />
    <xs:attribute name="zip" type="xs:string" use='optional' />
    <xs:attribute name="phone" type="xs:string" use='optional' />
    <xs:attribute name="fax" type="xs:string" use='optional' />
    <xs:attribute name="email" type="emailType" use='optional' />
    <xs:attribute name="languageCode" type="languageType"
use='optional' />
    <xs:attribute name="addPurpose" type="addPurposeType"
use='optional' />
    <xs:attribute name="legalType" type="legalTypeType"
use='optional' />
    <xs:attribute name="nexusCategory" type="nexusCategoryType"
use='optional' />
  </xs:complexType>
</xs:element>

<xs:element name="contact">
  <xs:complexType>
    <xs:choice>
      <xs:element ref="contactdata" minOccurs='0'
maxOccurs='1' />
      <xs:element ref="contactref" minOccurs='0' maxOccurs='1' />
    </xs:choice>
    <xs:attribute name="type" type="contactType"
use='required' />

```

```

</xs:complexType>
</xs:element>

<xs:element name="contactref">
  <xs:complexType>
    <xs:choice>
      <xs:element name="contactId" type="xs:positiveInteger" />
      <xs:element name="copy" type="contactType" />
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:simpleType name="actionType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="register" />
    <xs:enumeration value="renew" />
    <xs:enumeration value="transfer" />
    <xs:enumeration value="delete" />
  </xs:restriction>
</xs:simpleType>

<xs:simpleType name="referenceType">
  <xs:restriction base="xs:string">
    <xs:maxLength value="64"/>
  </xs:restriction>
</xs:simpleType>

</xs:schema>

```

## regrequest.xsd

```

<?xml version="1.0" encoding="utf-8"?>
<!--
  The registrant contact element, should NOT require:
  ã,*   first name
  ã,*   last name
  ã,*   phone
  ã,*   fax
  ã,*   email

```

### Registration:

- domain name
- domain action
- registration term
- all contact information
- name servers (optional)

### Renewal:

- domain name
- domain action
- old domain expiry date
- domain renewal term

- DOES NOT require contact
- DOES NOT require nameservers

**Transfer:**

- domain name
- domain action
- domain expiry date
- all contact information
- nameservers aren't necessary
- DOES NOT require term

**Delete:**

- domain name
- domain action
- DOESN'T REQUIRE ANY OTHER FIELDS

-->

```
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
<xs:include schemaLocation="apitypes.xsd"/>
<xs:element name="apiProtocol">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="regRequest" minOccurs='1'
        maxOccurs='1' />
    </xs:sequence>
    <xs:attribute name="version" type="xs:decimal"
      use='required' />
  </xs:complexType>
</xs:element>
<xs:element name="regRequest">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="createCustomer" minOccurs='0'
        maxOccurs='1' />
      <xs:element ref="useCustomer" minOccurs='0'
        maxOccurs='1' />
      <xs:element ref="nameserver" minOccurs='0'
        maxOccurs='10' />
      <xs:element ref="contact" minOccurs='0' maxOccurs='4' />
      <xs:element ref="domain" minOccurs='1'
        maxOccurs='unbounded' />
    </xs:sequence>
    <xs:attribute name="user" type="xs:string" use='required' />
    <xs:attribute name="password" type="xs:string"
      use='required' />
    <xs:attribute name="reference" type="referenceType" use='optional' />
  </xs:complexType>
</xs:element>
<xs:element name="nameserver">
  <xs:complexType>
    <xs:attribute name="name" type="xs:string" use='required' />
    <!--
  <xs:attribute name="ip" type="xs:string" use='optional' />
  -->
```

```

</xs:complexType>
</xs:element>
<xs:element name="createCustomer">
  <xs:complexType>
    <xs:attribute name="login" type="xs:string" use='required' />
    <xs:attribute name="password" type="xs:string"
      use='required' />
  </xs:complexType>
</xs:element>
<xs:element name="useCustomer">
  <xs:complexType>
    <xs:attribute name="login" type="xs:string" use='required' />
  </xs:complexType>
</xs:element>
<xs:element name="domain">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="authInfo" type="xs:string" minOccurs='0'
        maxOccurs='1' />
      <xs:element ref="nameserver" minOccurs='0'
        maxOccurs='10' />
      <!--
      <xs:element ref="service" minOccurs='0'
        maxOccurs='unbounded' />
      -->
    </xs:sequence>
    <xs:attribute name="authInfo" type="xs:string" use='optional' />
    <xs:attribute name="name" type="domainType" use='required' />
    <xs:attribute name="action" type="actionType"
      use='required' />
    <xs:attribute name="term" type="xs:positiveInteger"
      use='optional' />
    <xs:attribute name="oldexpiry" type="xs:date"
      use='optional' />
  </xs:complexType>
</xs:element>
<xs:element name="service">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="serviceitem" minOccurs='0'
        maxOccurs='unbounded' />
    </xs:sequence>
    <xs:attribute name="name" type="serviceType"
      use='required' />
    <xs:attribute name="value" type="xs:string" use='optional' />
    <xs:attribute name="quantity" type="xs:positiveInteger"
      use='optional' />
    <xs:attribute name="term" type="xs:positiveInteger"
      use='optional' />
  </xs:complexType>
</xs:element>
<xs:element name="serviceitem">
  <xs:complexType>

```

```

    <xs:attribute name="name" type="xs:string" use='required' />
    <xs:attribute name="value" type="xs:string" use='required' />
  </xs:complexType>
</xs:element>
</xs:schema>

```

## statusresponse.xsd

```

<?xml version="1.0" encoding="utf-8"?>
<!-- Schema for the generic response -->
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
  <xs:element name="apiProtocol">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="statusResponse" minOccurs='1'
maxOccurs='1' />
      </xs:sequence>
      <xs:attribute name="version" type="xs:decimal" use='required' />
    </xs:complexType>
  </xs:element>
  <xs:element name="statusResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="status" minOccurs='1' maxOccurs='1' />
      </xs:sequence>
      <xs:attribute name="requestId" type="xs:nonNegativeInteger"
use='required' />
      <xs:attribute name="reference" type="referenceType" use='optional' />
    </xs:complexType>
  </xs:element>
  <xs:element name="status">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="statusItem" minOccurs='0'
maxOccurs='unbounded'
/>
      </xs:sequence>
      <xs:attribute name="code" type="xs:nonNegativeInteger"
use='required' />
      <xs:attribute name="message" type="xs:string" use='required' />
    </xs:complexType>
  </xs:element>
  <xs:element name="statusItem">
    <xs:complexType>
      <xs:attribute name="objType" type="xs:string" use='required' />
      <xs:attribute name="objName" type="xs:string" use='required' />
      <xs:attribute name="code" type="xs:nonNegativeInteger"
use='required' />
      <xs:attribute name="message" type="xs:string" use='required' />
    </xs:complexType>
  </xs:element>

```

```
</xs:schema>
```

## domainmodrequest.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
<xs:include schemaLocation="apitypes.xsd"/>
  <xs:element name="apiProtocol">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="domainModRequest" minOccurs='1'
          maxOccurs='1' />
      </xs:sequence>
      <xs:attribute name="version" type="xs:decimal"
        use='required' />
    </xs:complexType>
  </xs:element>
  <xs:element name="domainModRequest">
    <xs:complexType>
<xs:sequence>
      <xs:element ref="domain" minOccurs='1'
        maxOccurs='unbounded' />
      <xs:element ref="useCustomer" minOccurs='0'
        maxOccurs='1' />
      <xs:element ref="nameserver" minOccurs='0'
        maxOccurs='10' />
      <xs:element ref="contact" minOccurs='0' maxOccurs='4' />
</xs:sequence>
      <xs:attribute name="user" type="xs:string" use='required' />
      <xs:attribute name="password" type="xs:string"
        use='required' />
      <xs:attribute name="reference" type="referenceType" use='optional' />
    </xs:complexType>
  </xs:element>
  <xs:element name="nameserver">
    <xs:complexType>
      <xs:attribute name="name" type="xs:string" use='required' />
      <xs:attribute name="ip" type="xs:string" use='optional' />
    </xs:complexType>
  </xs:element>
  <xs:element name="useCustomer">
    <xs:complexType>
      <xs:attribute name="login" type="xs:string" use='required' />
    </xs:complexType>
  </xs:element>

  <xs:element name="domain">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="service" minOccurs='0'
          maxOccurs='unbounded' />

```

```

    </xs:sequence>
    <xs:attribute name="name" type="domainType" use='required' />
  </xs:complexType>
</xs:element>

<xs:element name="service">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="serviceitem" minOccurs='0'
        maxOccurs='unbounded' />
    </xs:sequence>
    <xs:attribute name="name" type="serviceType"
      use='required' />
    <xs:attribute name="value" type="xs:string" use='optional' />
    <xs:attribute name="quantity" type="xs:positiveInteger"
      use='optional' />
    <xs:attribute name="term" type="xs:positiveInteger"
      use='optional' />
  </xs:complexType>
</xs:element>

<xs:element name="serviceitem">
  <xs:complexType>
    <xs:attribute name="name" type="xs:string" use='required' />
    <xs:attribute name="value" type="xs:string" use='required' />
  </xs:complexType>
</xs:element>

</xs:schema>

```

## checkrequest.xsd

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
  <xs:include schemaLocation="apitypes.xsd"/>
  <xs:element name="apiProtocol">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="checkRequest" minOccurs='1'
          maxOccurs='1' />
      </xs:sequence>
      <xs:attribute name="version" type="xs:decimal"
        use='required' />
    </xs:complexType>
  </xs:element>
  <xs:element name="checkRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="domain" minOccurs='1'
          maxOccurs='unbounded' />
      </xs:sequence>
    </xs:complexType>
  </xs:element>

```

```

    <xs:attribute name="user" type="xs:string" use='required' />
    <xs:attribute name="password" type="xs:string"
    use='required' />
    <xs:attribute name="reference" type="referenceType" use='optional' />
  </xs:complexType>
</xs:element>

<xs:element name="domain">
  <xs:complexType>
    <xs:attribute name="name" type="domainType" use='required' />
  </xs:complexType>
</xs:element>
</xs:schema>

```

## whoisrequest.xsd

```

<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs='http://www.w3.org/2001/XMLSchema'>
<xs:include schemaLocation="apitypes.xsd"/>

  <xs:element name="apiProtocol">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="whoisRequest" minOccurs='1'
        maxOccurs='1' />
      </xs:sequence>
      <xs:attribute name="version" type="xs:decimal"
      use='required' />
    </xs:complexType>
  </xs:element>

  <xs:element name="whoisRequest">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="domain" minOccurs='1' maxOccurs='unbounded' />
      </xs:sequence>
      <xs:attribute name="user" type="xs:string" use='required' />
      <xs:attribute name="password" type="xs:string" use='required' />
      <xs:attribute name="reference" type="referenceType" use='optional' />
    </xs:complexType>
  </xs:element>

  <xs:element name="domain">
    <xs:complexType>
      <xs:attribute name="name" type="domainType" use='required' />
    </xs:complexType>
  </xs:element>
</xs:schema>

```

## whoisresponse.xsd

```
<?xml version="1.0" encoding="utf-8"?>
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema">
  <xs:element name="apiProtocol">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="whoisResponse" minOccurs="1" maxOccurs="1" />
      </xs:sequence>
      <xs:attribute name="version" type="xs:decimal" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="whoisResponse">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="domain" minOccurs="1" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="requestId" type="xs:nonNegativeInteger" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="domain">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="customer" minOccurs="0" maxOccurs="1" />
        <xs:element ref="nameserver" minOccurs="0" maxOccurs="10" />
        <xs:element ref="contact" minOccurs="4" maxOccurs="4" />
        <xs:element ref="service" minOccurs="0" maxOccurs="unbounded" />
      </xs:sequence>
      <xs:attribute name="name" type="domainType" use="required" />
      <xs:attribute name="created" type="xs:date" use="required" />
      <xs:attribute name="updated" type="xs:date" use="required" />
      <xs:attribute name="expires" type="xs:date" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="contact">
    <xs:complexType>
      <xs:choice>
        <xs:element ref="contactdata" minOccurs="0" maxOccurs="1" />
        <xs:element ref="contactref" minOccurs="0" maxOccurs="1" />
      </xs:choice>
      <xs:attribute name="type" type="contactType" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="nameserver">
    <xs:complexType>
      <xs:attribute name="name" type="xs:string" use="required" />
      <xs:attribute name="ip" type="xs:string" use="required" />
    </xs:complexType>
  </xs:element>
  <xs:element name="customer">
```

```

<xs:complexType>
  <xs:attribute name="login" type="xs:string" use='required' />
</xs:complexType>
</xs:element>
<xs:element name="useCustomer">
  <xs:complexType>
    <xs:attribute name="login" type="xs:string" use='required' />
  </xs:complexType>
</xs:element>
<xs:element name="contactdata">
  <xs:complexType>
    <xs:attribute name="firstName" type="xs:string" use='required' />
    <xs:attribute name="lastName" type="xs:string" use='required' />
    <xs:attribute name="organization" type="xs:string" use='required' />
    <xs:attribute name="address1" type="xs:string" use='required' />
    <xs:attribute name="address2" type="xs:string" use='optional' />
    <xs:attribute name="city" type="xs:string" use='required' />
    <xs:attribute name="state" type="xs:string" use='required' />
    <xs:attribute name="country" type="xs:string" use='required' />
    <xs:attribute name="zip" type="xs:string" use='required' />
    <xs:attribute name="phone" type="xs:string" use='required' />
    <xs:attribute name="fax" type="xs:string" use='optional' />
    <xs:attribute name="email" type="emailType" use='required' />
    <xs:attribute name="languageCode" type="languageType" use='required' />
    <xs:attribute name="addPurpose" type="addPurposeType" use='optional' />
    <xs:attribute name="legalType" type="legalTypeType" use='optional' />
    <xs:attribute name="nexusCategory" type="nexusCategoryType" use='optional' />
  </xs:complexType>
</xs:element>
<xs:element name="contactref">
  <xs:complexType>
    <xs:choice>
      <xs:element name="contactId" type="xs:positiveInteger" />
      <xs:element name="copy" type="contactType" />
    </xs:choice>
  </xs:complexType>
</xs:element>
<xs:element name="service">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="serviceitem" minOccurs='0' maxOccurs='unbounded' />
    </xs:sequence>
    <xs:attribute name="name" type="serviceType" use='required' />
    <xs:attribute name="value" type="xs:string" use='optional' />
    <xs:attribute name="quantity" type="xs:positiveInteger" use='optional' />
    <xs:attribute name="term" type="xs:positiveInteger" use='optional' />
  </xs:complexType>

```

```

</xs:complexType>
</xs:element>
<xs:element name="serviceitem">
  <xs:complexType>
    <xs:attribute name="name" type="xs:string" use='required' />
    <xs:attribute name="value" type="xs:string" use='required' />
  </xs:complexType>
</xs:element>
<xs:simpleType name="contactType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="admin" />
    <xs:enumeration value="technical" />
    <xs:enumeration value="billing" />
    <xs:enumeration value="registrant" />
    <xs:enumeration value="customer" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="languageType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="en" />
    <xs:enumeration value="fr" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="emailType">
  <xs:restriction base="xs:string">
    <xs:pattern
      value="[a-zA-Z0-9_\-\.]{1,65}@((([a-z0-9]|[a-z0-9][a-z0-9\-\-]{0,61})[a-z0-9])\.)+[a-z]+"
    />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="countryType">
  <xs:restriction base="xs:string">
    <xs:pattern value="[A-Z]{2}" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="domainType">
  <xs:restriction base="xs:string">
    <xs:pattern
      value="((([a-z0-9]|[a-z0-9][a-z0-9\-\-]{0,61})[a-z0-9])\.)+[a-z]+"
    />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="actionType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="register" />
    <xs:enumeration value="renew" />
    <xs:enumeration value="transfer" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="serviceType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="mailforward" />
  </xs:restriction>
</xs:simpleType>

```

```

    <xs:enumeration value="urlforward" />
    <xs:enumeration value="frameforward" />
    <xs:enumeration value="customdns" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="addPurposeType">
  <xs:restriction base="xs:string">
    <xs:pattern value="P[1-5]" />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="legalTypeType">
  <xs:restriction base="xs:string">
    <xs:pattern
value="CCO|CCT|RES|GOV|EDU|ASS|PRT|TDM|TRS|TRD|LGR|OMK|LAM|HOP|PLT|A
BO|INB|MAJ"
    />
  </xs:restriction>
</xs:simpleType>
<xs:simpleType name="nexusCategoryType">
  <xs:restriction base="xs:string">
    <xs:pattern value="C11|C12|C21|C31|C32" />
  </xs:restriction>
</xs:simpleType>
</xs:schema>

```