Network Working Group
Internet-Draft
Intended status: Standards Track
Expires: June 3, 2009

M. Hunt
New Zealand Registry Services

November 30, 2008

System for Managing a Shared Domain Registry
draft-nzrs-srs-01

Status of this Memo

Abstract

This document describes the typical usage and communication protocol of a client/server shared registry system for management of domain name registrations. The system described is a "thick registry" system, providing support for the storage and management of both the technical and the registrant contact information concerning domain registrations. The system relies on the existing HTTP and HTTPS infrastructure for transport and secure transfer to avoid having to implement a dedicated protocol and server environment. A bespoke XML format is used to communicate between clients and the SRS server.

Comments are solicited and should be addressed to the mailing list at srsstandards-l@nzrs.net.nz and/or the author. The mailing list

management page can be found at
<https://mailman.nzrs.net.nz/cgi-bin/mailman/listinfo/
srsstandards-l>.

Table of Contents

1. Introduction

This document describes the Shared Registry System (SRS), a system for managing a shared domain name registry. This system broadly satisfies the requirements for a generic registry-registrar protocol as defined in RFC 3375 [RFC3375]. As this system only addresses the issue of managing a shared registry service and uses HTTP [RFC2616] as its transport layer, implementations are expected to be relatively easy.

This document also describes the communication protocol used by the SRS system. This protocol uses messages in an XML format for system requests and responses. The DTD for the SRS protocol is provided in its entirety as an appendix (Appendix C) to this document, and individual parts of the protocol are covered in depth throughout the document.

The SRS works in a connection-oriented fashion with no session state. The registrar sends a request document to the registry containing commands to be executed by the SRS and the result of processing these commands is returned as the response. Each registrar request document receives a response document containing the result of processing all of the requests in a single request document.

Public Key Infrastructure (PKI) is used to manage issues of security, authentication of requests and non-repudiation of actions. Exchange of keys between the registry and registrars is outside the scope of this document.

A system built using this protocol is used by .nz Registry Services (NZRS) and a sample implementation [1] consisting of client and server software is available from the Sourceforge web site.

2. Conventions used in this Document

The definitions of Registry, Registrar, and Registrant from RFC 3375 [RFC3375] are used in this document and are included below.

SRS: a software implementation of the shared registry system described here.

Registry: An entity that provides back-end domain name registration services to registrars, managing a central repository of information associated with domain name delegations. A registry is typically responsible for publication and distribution of zone files used by the Domain Name System.

Registrar: An entity that provides front-end domain name registration services to registrants, providing a public interface to registry services.

Registrant: An entity that registers domain names in a registry through the services provided by a registrar. Registrants include individuals, organizations, and corporations.

UDAI: The Unique Domain Authentication Identifier (UDAI) is a domain code that is generated by the SRS and stored by the registrar and registrant to restrict updates to domain details. The value of the UDAI is not stored within the SRS, but a message digest of the value is kept to check the integrity of domain update and transfer requests. The UDAI SHOULD be an ASCII [US-ASCII] encoded character string.

2.1. Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in RFC 2119 [RFC2119].

3. System Description

The SRS provides the functions required to support all of the usual business functions of a domain registration service, including:

- creation, maintenance, querying and deletion of domain details
- querying of public details of a domain - to support a public WHOIS service
- transfer of domains between registrars
- creation, maintenance, querying and deletion of registrar details
- regular zone file creation for domains delegated to appear in the domain name system (DNS)
- registrar account query, maintenance, and message polling
- creation and cancellation of scheduled jobs in the SRS to support business functions (such as building zone files and releasing and renewing domains)
- management and configuration of the SRS by the registry

o billing and accounting functions to work with the registry's accounting system

Direct and indirect interactions with the SRS may be split into three main groups:

o the public (including registrants),

o registrars,

o and the registry.

Their interactions with the SRS are summarized below.

```
                     +-----------+
                     |  Public/  |
                     |registrants|
                     |           |                   Public
                     +-----------+
                       ^   ^   ^
                 WHOIS |   |   | DNS queries
         +-------------+   |   +-------------+
         |                 v                 |
         |           +-----------+           |
         |           |Registrar  |           |
         |           |service    |           |
         |           |           |           |       Registrars
         |           +-----------+           |
         |                 ^                 |
         |                 | HTTP/HTTPS      |
         |                 | (SRS protocol)  |
         v                 v                 v
   +-----------+     +-----------+     +-----------+
   |Whois      |     |SRS        |     |DNS        |
   |server     |<--->|           |---->|           |
   |           |     |           |     |           | Registry
   +-----------+     +-----------+     +-----------+
```

3.1. The Public

The public (and registrants - the domain registering public) have no direct access to the SRS server. They interact with the system through:

o querying domain ownership details in the public WHOIS system,

o performing hostname lookups to access services hosted on the internet,

- o registering domains by establishing a business relationship with a registrar,
- o transferring their registered domains between registrars,
- o and maintaining their own domain details through systems provided by a registrar.

3.2. Registrars

Registrars generate most of the work that the SRS server handles. They are expected to maintain their own registrar details within the SRS as an aspect of their business relationship with the registry. They also offer services to the public, or a restricted section of the public, to manage domain registrations. These public services are likely to include:

- o checking availability of domain names,
- o registering domain names,
- o transferring domain names from another registrar,
- o cancelling domain registrations,
- o billing for domain registrations,
- o and providing a service to allow registrants to maintaining domain details.

In addition to their customer services, registrars are expected to regularly poll the SRS server using the GetMessages (Section 6.3.1) request, to retrieve details of actions relevant to their business with the registry that they have not initiated directly (for instance, registrant-initiated transfers of domains from them to another registrar, and actions performed by the registry on their behalf).

3.3. The Registry

Generally the registry will be responsible for maintaining all of the information associated with domain name registrations (including both the technical information required to produce zone files and the contact information for registrars and registrants), running the SRS service to support registrars' and its own maintenance of domains, running a public WHOIS service compliant with RFC 3912 [RFC3912], regular backups, data security, and running name servers for their domains.

By running a "thick registry" system, the registry provides security, stability and backup of the registry information, and ensures that registrants are protected against registrar failures.

The information for the public WHOIS service and the name server zone files is derived from the data held in the SRS server. Most registered domains will be delegated to other nameservers.

3.4. The Domain Registration Cycle

Registrants may register domains through registrars according to the limitations of the available subdomains allowed by the registry, and any policy decisions that the registry applies to domain naming.

Registries are encouraged to apply a system of grace periods to the registration cycle of domains once they have initially been registered. The .nz Registry Service applies the following grace periods:

- registration grace period
- renewal grace period
- auto-renew grace period
- redemption grace period

Each grace period exists for a specific period of time that is typically measured in days. The duration of each grace period is a matter of registry operational policy that is not addressed in this document.

3.4.1. Registration Grace Period

The registration grace period applies after the initial registration of a domain name. If the domain name is cancelled by the registrar during this period, all registrar account transactions (billing) for the domain will also be cancelled. The domain name is released back to the available pool of names. A registrant may not transfer the management of a domain to another registrar during this period.

The registry MAY restrict domain cancellations in this term to prevent a single registrar from cancelling a particular domain more than once during the registration grace period.

3.4.2. Renewal Grace Period

The renewal grace period applies after a domain name registration period is explicitly extended (renewed) by the registrar. If the domain name is cancelled by the registrar during this period, the registrar account transaction for the renewal will also be cancelled and the BilledUntil date of the domain rolled back to the previous value.

3.4.3. Auto-renew Grace Period

The auto-renew grace period applies after a domain name registration period expires and is extended (renewed) automatically by the registry. If the domain name is cancelled by the registrar during this period, the registrar account transaction for the renewal will also be cancelled and the BilledUntil date of the domain rolled back to the previous value.

3.4.4. Redemption Grace Period

The redemption grace period applies after a domain registration is cancelled by a registrant. It defines the length of time that a domain will stay in the cancelled state ("PendingRelease" status). After this time expires, the registry will release the domain name back to the available pool of names.

4. Authentication, Security and Authorization

4.1. Authentication

A two factor authentication system is used to establish the identity of the user that makes a request:

- A unique numeric identifier issued to each registrar by the registry

- An OpenPGP-compatible signature of the request body

Registrars are issued with a unique numeric identifier when their account is first created in the SRS. This registrar identifier MUST be included on client requests to allow the SRS server to identify the client. Failure to provide a correct identifier as part of the request SHALL result in the SRS server returning an error response.

The registrar must also maintain at least one public, private OpenPGP-compatible key pair for authentication. One or more public keys are provided to the registry when the registrar account is first created and the registrar MUST sign requests using one of its registered private keys. This information is used for authentication and to ensure non-repudiation of requests. The registrar may provide multiple public keys to ease the process of expiring old or revoked keys without interrupting the work of the registrar.

If a request does not have a signature, or the signature does not confirm that the identity of the registrar that signed the request matches the registrar identifier attached to the request, then the SRS server SHALL return an error response.

Responses from the SRS server MUST also be signed. Responses are signed by the registry using the registry's private key. The registry public key MUST be made easily available to registrars to allow authentication of response messages. This ensures that registrars can be confident that the responses to their requests are authentic, and have not been altered in transit.

4.2. Security

The request and response signature mechanism provides a means of ensuring that messages have not been tampered with in transit. In addition to this, all requests for private data (data that cannot be retrieved using the public WHOIS system) MUST use an encrypted HTTP connection (HTTPS) for data security. If a request for private data is received via an unencrypted HTTP connection, the SRS server SHALL return an error response.

Only the Whois (Section 6.1.4) request may be issued over an unencrypted HTTP connection.

4.3. Authorization

SRS implementations SHOULD impose a permissions model to restrict the SRS requests that users are allowed to access. Action and role types are defined in the protocol definition for this purpose. If the originating user does not possess all of the permissions required to complete a request, the server SHOULD reject the transaction.

Transactions sent to the server MUST identify the registrar making the request. Registrars SHALL NOT be permitted to perform functions using an effective registrar other than their own. Any such requests received by the registry SHALL be rejected.

The registry SHOULD only access the SRS server through the same interface provided to registrars and SHOULD have one or more well-known registrar identifiers allocated to itself for the purpose of maintaining the registry. No changes should be made to public or private data in the SRS using other means of access that the registry may have available (for example, direct access to a database store).

The registry MAY perform SRS functions using an arbitrary effective registrar value.

Registrants and the public have no direct access to the SRS.

5. Communication

All communication with the SRS is performed using XML [W3C.REC-xml] documents encoded in UTF-8 [RFC3629] and sent using HTTP [RFC2616] or HTTPS [RFC2818]. The request body MAY contain multiple independent requests to be performed on the SRS. The response MAY include a response to each request in the XML document, or a single error response. The SRS SHOULD process the requests in the order that they are received in the request body.

The user should ensure that:

- requests in an XML document are ordered logically to prevent errors due to sequencing (for example, attempting to update a domain record before creating it)
- the number of requests per XML document is limited to ensure acceptable processing time and response size

Both request and response messages MUST be accompanied by a digital signature of the complete XML request body (the value of the r parameter to the HTTP messages detailed below). The digital signature authentication method follows the specification in section 2.2 of OpenPGP Message Format [RFC4880] and MUST be encoded in ASCII Armor (see section 6.2 of [RFC4880]) for inclusion in the HTTP request. The SRS protocol is a signature-only application of OpenPGP.

SRS implementations MAY define a limit on the response size to support service level agreements on response time. SRS implementations MAY reject requests in an XML document if they would otherwise exceed a defined limit on the response size or response time.

The VerMajor and VerMinor attributes are required in the first element of each request and response to allow clients and servers to modify their behaviour dependent on the version of the protocol that they support. The meaning of the two fields is:

VerMajor: Major version number of the protocol. This number only changes when major changes are made to the system that are not backward compatible with previous versions; for example, client/server interface changes (changes to the protocol DTD).

VerMinor: Minor version number of the protocol. This number changes for minor updates to the protocol that are backward compatible with previous versions with the same major version number.

An SRS server MUST reject any request made with a major version number that is greater than the SRS server's own supported version. SRS server implementations MAY support requests made with a major version number that is less than the SRS server's own supported version.

5.1. Request Format

Requests MUST include three parameters in the HTTP POST request:

Parameter	Description
n	The registry-assigned registrar identifier.
r	The XML request, encoded in UTF-8.
s	An OpenPGP-compatible signature of the XML request document, signed with the registrar's private key. Presented in ASCII armored format and encoded in UTF-8.

The XML request document MUST be formatted using the NZSRSRequest element. The NZSRSRequest element MAY contain one or more requests. Valid requests names are defined by the Action (Section 9.1.6) entity.

Syntax:

```
<!ELEMENT NZSRSRequest ((%Action;)+)>
<!ATTLIST NZSRSRequest
        VerMajor    (1|2)          #REQUIRED
        VerMinor    %Number;       #REQUIRED
        RegistrarId %RegistrarId; #IMPLIED>
```

Example of a request body prior to encoding:

```
n=50041&r=
<NZSRSRequest VerMajor="2" VerMinor="47" RegistrarId="50041">
    <Whois DomainName="example.org" />
</NZSRSRequest>&s=-----BEGIN PGP SIGNATURE-----...
```

5.2. Response Format

The body of the SRS server response MUST include two parameters:

Parameter	Description
r	The XML response, encoded in UTF-8.
s	An OpenPGP-compatible signature of the XML response document, signed with the registry's private key. Presented in ASCII armored format and encoded in UTF-8.

The XML response document MUST be formatted using the NZSRSResponse element. The NZSRSResponse element MAY contain one or more Response (Section 5.2.1) elements or an Error (Section 7.7) element.

Syntax:

```
<!ELEMENT NZSRSResponse (Response+|Error)>
<!ATTLIST NZSRSResponse
        VerMajor    (2)           #REQUIRED
        VerMinor    %Number;      #REQUIRED
        RegistrarId %RegistrarId; #IMPLIED>
```

Example of a decoded response message:

```
r=<?xml version="1.0" encoding="utf-8"?>
<NZSRSResponse VerMajor="2" VerMinor="47" RegistrarId="50041">
    <Response Action="Whois" FeId="4" FeSeq="137"
      OrigRegistrarId="50041" RecipientRegistrarId="50041" Rows="1">
        <FeTimeStamp Year="2007" Month="10" Day="19" Hour="14"
           Minute="7" Second="51" TimeZoneOffset="+13:00" />
        <Domain DomainName="example.org">...</Domain>
    </Response>
</NZSRSResponse>&s=-----BEGIN PGP SIGNATURE-----...
```

5.2.1. Response

The response element acts as a container for all response types, including errors to individual requests in an XML document. If an error is encountered in parsing the request, or the client specified a major protocol version higher than that supported by the server, the SRS server MUST return the Error (Section 7.7) element with no Response container.

The response element may contain any of the following:

- o multiple responses in response to a GetMessages (Section 6.3.1) request, or
- o a single action response (Section 9.13.1) specific to the type of action request.

The Response element must specify:

Attribute	Description
Action	The name of the request action (Section 9.1.6) that this is a response to.
FeId	The unique identifier of the SRS server that handled the request.
FeSeq	The unique identifier for the request on the server that handled the request.
OrigRegistrarId	The unique identifier for the user that submitted the request to the SRS server.

The Response element may also specify:

Attribute	Description
TransId	The identifier (either an ActionId or a QryId) for the transaction submitted by the user that originated the request.
Rows	The number of results returned in the current response body.
Count	The total number of matches found in the SRS that satisfied the request query conditions.
MoreRowsAvailable	A boolean value indicating whether this response was truncated by an SRS limit on the number of results returned.

RecipientRegistrarId	The unique identifier for the user that the response is returned to.

The Response element MUST contain an FeTimeStamp (Section 8.35) element, which shows the time and date that the request was processed by the SRS server, and also MAY contain one of the SRS response types (Section 7).

Syntax:

```
<!ELEMENT Response (FeTimeStamp,
                    (Response*|%ActionResponse;)?
                   )>
<!ATTLIST Response
        Action (%Action;|UnknownTransaction|DomainTransfer) #REQUIRED
        FeId                 %Number;      #REQUIRED
        FeSeq                %Number;      #REQUIRED
        OrigRegistrarId      %RegistrarId; #REQUIRED
        TransId              %UID;         #IMPLIED
        Rows                 %Number;      #IMPLIED
        Count                %Number;      #IMPLIED
        MoreRowsAvailable    %Boolean;     #IMPLIED
        RecipientRegistrarId %RegistrarId; #IMPLIED>
```

5.2.2. Error

An Error response may be returned for a whole request - for instance, when the request body failed validation or incorrect authentication was provided - or for one or more requests from the XML document body. In either case, the Error element will be used.

This document does not specify the error codes and situations. All SRS server errors SHOULD be treated as a failed request by the client and the SRS server MUST NOT change any stored details if it returns an error to the client request.

6. SRS Requests

The SRS defines request elements that support the running of the shared domain registry from both a technical and a business perspective. Implementations SHOULD ensure that access to requests and data is restricted to comply with legal obligations and the registry's own business requirements.

To support the implementation of a flexible permissions model for SRS users, the requests are grouped into five major categories:

Domain query: requests that allow users to query domain related information stored in the SRS

Domain write: requests that allow users to create and update domain related information in the SRS

Registrar query: requests that allow users to query registrar related information stored in the SRS

Registrar write: requests that allow users to create and update registrar related information in the SRS

Registry: requests that support registry business functions, SRS system settings, running jobs on the SRS, and billing functions

Typically, registrars will have access to the domain query, domain write and registrar query requests, as well as the ability to maintain their own registrar information within the SRS. Administrative users will be able to create registrars and run registry requests - normally this will be restricted to the registry itself.

All requests that result in an update to data held by the SRS server MUST provide an ActionId to identify the request. The combination of RegistrarId and ActionId for a request MUST be unique. This allows requests to be fully identified by the user that made the request and the ActionId that the user assigned to it. SRS server implementations SHOULD maintain a full audit trail by logging all update requests and their outcomes.

If the SRS server receives a request with the same RegistrarId and ActionId as a previous request, but different request details, it MUST return an Error (Section 7.7) response. If the SRS server receives a request with the same RegistrarId and ActionId as a previous request, and identical request details, it MUST respond by returning the response that was sent to the original request.

6.1. Domain Query Actions

The domain query requests are:

Action	Description
ActionDetailsQry	Retrieve the XML content and signature for a previous SRS request and the response that it received.

DomainDetailsQry	Retrieve the current stored details for a domain.
UDAIValidQry	Check the validity of a UDAI code for a domain.
Whois	Retrieve the public WHOIS data for a domain.

6.1.1. ActionDetailsQry

The ActionDetailsQry request allows the user to retrieve the original XML document and signature for a previous request to the SRS, and also the XML document and signature for the response that was returned. The user must supply the correct ActionId of the original request. Registrars SHOULD be restricted to only retrieving the details of requests that they were the originating registrar for.

6.1.1.1. Request

The ActionDetailsQry request is an empty element that must specify:

Attribute	Description
ActionId	An action identifier. This is the action identifier for a previous request to the SRS.

Additionally, the request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
OriginatingRegistrarId	A user identifier. The identifier of the registrar that originally requested the action that matches the provided ActionId.

Syntax:

```
<!ELEMENT ActionDetailsQry EMPTY>
<!ATTLIST ActionDetailsQry
        QryId                  %UID; #IMPLIED
        ActionId               %UID; #REQUIRED
        OriginatingRegistrarId %UID; #IMPLIED>
```

6.1.1.2. Response

The response to an ActionDetailsQry request will be either an Error (Section 7.7) element or a RawRequest and a RawResponse (Section 7.8) element containing the XML and the registrar signature of the original request and response. If a QryId was provided in the request, it is returned in the TransId attribute of the Response element.

6.1.2. DomainDetailsQry

The DomainDetailsQry request allows the user to retrieve the stored details (including details that will not be shown for a public WHOIS query) for one or more domains. The request can be used to view historical details for a domain and may return a full update history for a domain.

Registrars are expected to only request information on domains that they currently manage. SRS implementations SHOULD restrict access to non-public domain information to the managing registrar. If historical data is requested for a period when the registrar did not manage a particular domain - for example, in the case of domain transfers - then only the public details as returned by the Whois (Section 6.1.4) request shall be returned for this time.

6.1.2.1. Request

The DomainDetailsQry request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
Status	A text filter. Matched against the registration status of domains. Valid values are "Active" and "PendingRelease".
Delegate	A boolean value. Matched against the delegation status of domains.

Term	A numeric value. Matched against the billing term for domains.
RegistrantRef	A text filter. Matched against the registrar-provided customer reference of a domain's registrant.
MaxResults	A numeric value. Used to specify the maximum number of domain records to return. If not specified, the default defined by the server implementation will be used.
SkipResults	A numeric value that defaults to 0. Used to specify the number of records at the start of a results set to be skipped before returning domain results. Can be used with MaxResults to retrieve a large result set in pages.
CountResults	A boolean value that defaults to false. If true, the response should only return the number of matches for the query and no specific domain details. If MaxResults or SkipResults is defined and CountResults is true then the server SHALL return an error response.

For the optional Status, Delegate, Term, and RegistrantRef filters, if the attribute is not specified, then the results will not be filtered on these fields.

The request may also include elements to specify other filters to limit the results returned by the SRS server and to change the list of fields that are returned in the results. All of the filters are optional and, with the exception of the DomainNameFilter, may only occur once in the request. The DomainNameFilter may occur multiple times in the request and details of domains that match any of the provided filters will be returned in the results.

The elements that can be included in the DomainDetailsQry request are:

- DomainNameFilter (Section 8.15)

- NameServerFilter (Section 8.23)

- RegistrantContactFilter (Section 8.11)

- AdminContactFilter (Section 8.11)

- TechnicalContactFilter (Section 8.11)

- ResultDateRange (Section 8.13)
- SearchDateRange (Section 8.13)
- ChangedInDateRange (Section 8.13)
- RegisteredDateRange (Section 8.13)
- LockedDateRange (Section 8.13)
- CancelledDateRange (Section 8.13)
- BilledUntilDateRange (Section 8.13)
- AuditTextFilter (Section 8.6)
- ActionIdFilter (Section 8.2)
- FieldList (Section 8.19)

The FieldList element specifies the fields to be returned for each domain in the result set. If no FieldList is specified, the SRS server will only return the DomainName and Status for each domain.

Syntax:

```
<!ELEMENT DomainDetailsQry (DomainNameFilter*,NameServerFilter?,
                            RegistrantContactFilter?,
                            AdminContactFilter?,
                            TechnicalContactFilter?,
                            ResultDateRange?,SearchDateRange?,
                            ChangedInDateRange?,
                            RegisteredDateRange?,LockedDateRange?,
                            CancelledDateRange?,
                            BilledUntilDateRange?,
                            AuditTextFilter?,ActionIdFilter?,
                            FieldList?)>
<!ATTLIST DomainDetailsQry
        QryId         %UID;              #IMPLIED
        Status        (%RegDomainStatus;) #IMPLIED
        Delegate      %Boolean;          #IMPLIED
        Term          %Term;             #IMPLIED
        RegistrantRef %UID;              #IMPLIED
        MaxResults    %Number;           #IMPLIED
        SkipResults   %Number;           #IMPLIED
        CountResults  %Boolean;          "0">
```

6.1.2.2. Response

The response to a DomainDetailsQry request will be either an Error (Section 7.7) element, or zero or more Domain (Section 7.5) elements that matched the filters specified in the request. If a QryId was provided in the request, it is returned in the TransId attribute of the Response element.

If a FieldList was specified in the request, the fields provided in any returned Domain elements will match the list provided unless restricted due to ownership by a registrar other than the registrar making the request - in such cases only the public Whois information shall be returned. DomainName will always be returned for all matched domains.

By default, only the DomainName and Status details shall be returned for each Domain.

The ResultDateRange element in a DomainDetailsQry request has a number of affects on the response:

- o the results shall be extracted by first selecting all of the domains that were managed by the requesting registrar during the given date range and then applying the other filters to this new result set to produce the final response
- o the EffectiveDateRange will also be returned for each result domain, regardless of whether it was specified by a FieldList attribute
- o the MaxResults and SkipResults elements in the request (if specified) are treated as a maximum number of distinct domains and a number of distinct domains to skip - the actual number of results returned may exceed MaxResults due to having multiple entries for each domain returned with different effective date ranges

6.1.3. UDAIValidQry

The UDAIValidQry request allows users to check a UDAI strings against the details stored by the SRS for a domain. Changes to domains may only be made with a valid UDAI.

6.1.3.1. Request

The UDAIValidQry request is an empty element that must specify:

Attribute	Description
DomainName	A text string. The name of the domain to check.
UDAI	A text string. The UDAI to be checked.

Additionally, the request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.

Syntax:

```
<!ELEMENT UDAIValidQry EMPTY>
<!ATTLIST UDAIValidQry
        QryId      %UID;         #IMPLIED
        DomainName %DomainName; #REQUIRED
        UDAI       %UID;         #REQUIRED>
```

6.1.3.2. Response

The response to a UDAIValidQry request will be a UDAIValid (Section 7.13) element. If a QryId was provided in the request, it is returned in the TransId attribute of the Response element.

6.1.4. Whois

The Whois request allows users to retrieve the public information stored on domains. The details requested may be the full public details, or a simple check of the status of the domain.

6.1.4.1. Request

The Whois request is an empty element that must specify:

Attribute	Description
DomainName	A text string. The domain name for which the details are requested.

Additionally, the request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
FullResult	A boolean value. Indicates whether the full domain details should be returned. If false, only the status of the domain shall be returned. Defaults to true.
SourceIP	An IP address. The source IP address of the request. This may be used to monitor and limit Whois requests.

Syntax:

```
<!ELEMENT Whois EMPTY>
<!ATTLIST Whois
        QryId      %UID;         #IMPLIED
        FullResult %Boolean;     "1"
        SourceIP   CDATA         #IMPLIED
        DomainName %DomainName; #REQUIRED>
```

6.1.4.2. Response

The response to a Whois request will be either an Error (Section 7.7) element, or a Domain (Section 7.5) element. If a QryId was provided in the request, it is returned in the TransId attribute of the Response element.

The amount of detail returned for matched domains will depend on the request details and the domain status:

- o if the domain is not currently registered, only the domain's name and status ("Available") will be returned,
- o if the request is not for a full result, only the domain status will be returned,
- o otherwise, the Domain element will be returned with any of the public details for which the SRS server has a value stored

The public details for a domain should be set by the registry's policy. For the .nz registry, the public details are:

- DomainName
- Status
- RegisteredDate
- CancelledDate
- LockedDate
- BilledUntil
- EffectiveFrom
- Delegate
- RegistrarPublicContact
- RegistrantContact
- AdminContact
- TechnicalContact
- NameServers
- AuditDetails

6.2. Domain Write Actions

The domain write requests are:

Request	Description
DomainCreate	Register a new domain in the SRS.
DomainUpdate	Update the stored details of an existing domain in the SRS.

6.2.1. DomainCreate

The DomainCreate request allows users to request the creation of a new domain in the SRS. The domain created will be managed by the registrar that makes the request - or by the effective registrar for the request, if the request is made by the registry. This request will only be successful if the domain name is available.

6.2.1.1. Request

The DomainCreate request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
DomainName	A text string. The domain name to be created. Must conform to the format and syntax in RFC 1035 [RFC1035], RFC 1123 [RFC1123], and RFC 2181 [RFC2181].
Term	A numeric value. The number of months to bill when the domain is registered or renewed.

Additionally, the request may specify:

Attribute	Description
RegistrantRef	A customer identifier. Assigned to the registrant by the registrar.
Delegate	A boolean value that defaults to true. Indicates whether the domain should be delegated to appear in the DNS. Requires details of name servers to be provided with the domain creation request.

The request must also include a RegistrantContact (Section 8.10) details element for the domain, and may include elements defining the AdminContact (Section 8.10), TechnicalContact (Section 8.10), and NameServers (Section 8.22) for the domain, and also AuditText (Section 8.5) for the transaction.

Syntax:

```
<!ELEMENT DomainCreate (RegistrantContact,AdminContact?,
                        TechnicalContact?,NameServers?,AuditText?)>
<!ATTLIST DomainCreate
        ActionId      %UID;         #REQUIRED
        DomainName    %DomainName;  #REQUIRED
        RegistrantRef %UID;         #IMPLIED
        Term          %Term;        #REQUIRED
        Delegate      %Boolean;     "1">
```

6.2.1.2. Response

The response to a DomainCreate request will be either an Error (Section 7.7) element, or a full Domain (Section 7.5) element. The Domain element will only be returned if the domain creation is successful.

6.2.2. DomainUpdate

The DomainUpdate request allows users to update the details of existing domains and to perform various update functions, including:

- transferring a domain
- cancelling a domain
- un-cancelling a domain
- renewing a domain
- changing the delegation status of a domain
- requesting a new domain UDAI

Domain details may be updated individually or combined in a single request that updates multiple parts of the domain information.

In the absence of a valid UDAI, domain updates are restricted to the registrar that currently manages a domain, or to suitably authorized registry users. If a registrant provides the UDAI for their domain to a registrar other than the one that currently manages the domain, the other registrar will be able to perform an update to the domain. If the UDAI is valid, then the new registrar may make any required updates to the domain details and the SRS server will automatically initiate a domain transfer as part of processing the request.

A domain can be transferred regardless of the status of the domain (that is, the status of the domain can be "Active" or "PendingRelease" at the time of the transfer), however, the domain MUST NOT be locked. A locked domain cannot be transferred until it is unlocked. Registrants SHOULD have access to the UDAI for their domains to allow them to transfer their domains freely between registrars.

6.2.2.1. Request

The DomainUpdate request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

Additionally, the request may specify:

Attribute	Description
UDAI	A text string. The UDAI for the domain.
NewUDAI	A boolean value. Allows the user to request generation of a new UDAI for the domain. The server MAY provide a new UDAI value regardless of the value of this parameter.
RegistrantRef	A customer identifier. Assigned to the registrant by the registrar.
Term	A numeric value. The number of months to bill when the domain is registered or renewed.
Delegate	A boolean value. Indicates whether the domain should be delegated to appear in the DNS. Requires details of name servers to be available to the SRS server.
Renew	A boolean value. Indicates that the domain should be billed for a further billing period immediately, rather than waiting for it to expire.
NoAutoRenew	A boolean value. Indicates that the domain should not automatically renew at the expiry date. Only the registry can change the automatic renewal status for a domain.
Lock	A boolean value. Specifies the lock status of the domain. Locked domains cannot be updated until they are unlocked by the registry. If the attribute is not specified, the current lock setting will remain unchanged. SRS implementations MAY allow registrars to change the lock status of domains.

Cancel	A boolean value. Specifies a request to change the status of a domain. If true, a domain in "Active" status will change to "PendingRelease"; domains in any other status will remain unchanged. If false, a domain in "PendingRelease" status will change to "Active"; domains in any other status will remain unchanged. If the attribute is not specified, the current status will remain unchanged.
Release	A boolean value. Specifies that the domain should become "Available" after the cancellation grace period expires.
FullResult	A boolean value. Specifies that a full domain result is required (if true), or only the changed fields (if false). Defaults to true.

For the boolean attributes, if the attribute is not specified in the update request, then no change will be made to the related domain details.

The DomainUpdate request must also include a DomainNameFilter (Section 8.15) element to identify the domain(s) to be updated and may include elements defining the new details for RegistrantContact (Section 8.10), AdminContact (Section 8.10), TechnicalContact (Section 8.10), and NameServers (Section 8.22) for the domain, and also AuditText (Section 8.5) for the transaction.

Syntax:

```
<!ELEMENT DomainUpdate (DomainNameFilter+,RegistrantContact?,
                        AdminContact?,TechnicalContact?,
                        NameServers?,AuditText?)>
<!ATTLIST DomainUpdate
        ActionId      %UID;     #REQUIRED
        UDAI          %UID;     #IMPLIED
        NewUDAI       %Boolean; #IMPLIED
        RegistrantRef %UID;     #IMPLIED
        Term          %Term;    #IMPLIED
        Delegate      %Boolean; #IMPLIED
        Renew         %Boolean; #IMPLIED
        NoAutoRenew   %Boolean; #IMPLIED
        Lock          %Boolean; #IMPLIED
        Cancel        %Boolean; #IMPLIED
        Release       %Boolean; #IMPLIED
        FullResult    %Boolean; "1">
```

6.2.2.2. Response

The response to a DomainUpdate request will be either an Error (Section 7.7) element, or a Domain (Section 7.5) element. By default the Domain element returned will include the following fields:

- DomainName
- AuditDetails
- Status
- Delegate
- Term
- NameServers
- RegistrantRef
- RegistrarId
- RegistrantContact
- AdminContact
- TechnicalContact
- BilledUntil
- RegisteredDate
- CancelledDate
- LockedDate

If the FullResult parameter in the request is false, the DomainName and AuditDetails fields will be returned along with any of the other fields above that have been updated by the transaction.

6.3. Registrar Query Actions

The registrar query requests are:

Request	Description
GetMessages	Retrieve responses generated by the SRS server for a registrar.
RegistrarAccountQry	Request details of transactions in the account of a registrar.
RegistrarDetailsQry	Request the registrar details stored by the SRS server.

6.3.1. GetMessages

The GetMessages request allows the user to retrieve messages that they may have been unaware of, or to confirm the status of a specific transaction that had been sent previously. Messages that registrars may miss are likely to include requests that are made by the registry on the registrar's behalf, and domain transfer requests issued by other registrars transferring domains at the request of a registrant.

The GetMessages request MUST NOT be accompanied by any other requests within a request document.

6.3.1.1. Request

The GetMessages request that may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
OriginatingRegistrarId	A user identifier. The identifier of the registrar that originally requested the action. Also accepts the special "other registrar" (Section 9.12.2) value.
ActionId	An action identifier. This is the action identifier for a previous request to the SRS.
RecipientRegistrarId	A user identifier. The registrar identifier of the registrar that the messages retrieved were intended for. Only registry users can retrieve messages intended for a user other than themselves.

MaxResults	A numeric value. Used to specify the maximum number of messages to return. If not specified, the default defined by the server implementation will be used.
SkipResults	A numeric value. Used to specify the number of records at the start of a results set to be skipped before returning domain results. Can be used with MaxResults to retrieve a large result set in pages.
CountResults	A boolean value that defaults to false. If true, the response should only return the number of matches for the query and no messages. If MaxResults or SkipResults is defined and CountResults is true then the server SHALL return an error response.

The GetMessages request may also include a TransDateRange (Section 8.13) element and an AuditTextFilter (Section 8.6) element.

Either an ActionId or a TransDateRange MUST be specified in order for the request to succeed.

Syntax:

```
<!ELEMENT GetMessages (TransDateRange?,AuditTextFilter?)>
<!ATTLIST GetMessages
        QryId                  %UID;                  #IMPLIED
        OriginatingRegistrarId %RegistrarIdOrOTHERS; #IMPLIED
        ActionId               %UID;                  #IMPLIED
        RecipientRegistrarId   %RegistrarId;          #IMPLIED
        MaxResults             %Number;               #IMPLIED
        SkipResults            %Number;               #IMPLIED
        CountResults           %Boolean;              "0">
```

6.3.1.2. Response

The response to a GetMessages request will be either an Error (Section 7.7) element or one or more Response (Section 5.2.1) elements containing the details of messages that match the request specification.

6.3.2. RegistrarAccountQry

The RegistrarAccountQry request allows the user to retrieve the details of billing transactions made in the account of a registrar.

6.3.2.1. Request

The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
RegistrantRef	A text filter. Matched against the registrar-provided customer reference of a registrant.
DomainName	A text filter. Matched against the domain name that billing amounts relate to.
InvoiceId	An invoice identifier. Used to match billing records that have been associated with an invoice.
MaxResults	A numeric value. Used to specify the maximum number of messages to return. If not specified, the default defined by the server implementation will be used.
SkipResults	A numeric value. Used to specify the number of records at the start of a results set to be skipped before returning domain results. Can be used with MaxResults to retrieve a large result set in pages.
TransStatus	A BillStatus (Section 9.2.2) value. Used to filter on the bill status of billing amounts.
CountResults	A boolean value that defaults to false. If true, the response should only return the number of matches for the query and no messages. If MaxResults or SkipResults is defined and CountResults is true then the server SHALL return an error response.

The RegistrarAccountQry request may also include a TransDateRange (Section 8.13) element and an InvoiceDateRange (Section 8.13) element.

Syntax:

```
<!ELEMENT RegistrarAccountQry (TransDateRange?,InvoiceDateRange?)>
<!ATTLIST RegistrarAccountQry
        QryId         %UID;           #IMPLIED
        RegistrantRef %UID;           #IMPLIED
        DomainName    %DomainName;    #IMPLIED
        InvoiceId     %UID;           #IMPLIED
        MaxResults    %Number;        #IMPLIED
        SkipResults   %Number;        #IMPLIED
        TransStatus   (%BillStatus;) #IMPLIED
        CountResults  %Boolean;       "0">
```

6.3.2.2. Response

The response to a RegistrarAccountQry request will be either an Error (Section 7.7) element or a set of BillingTrans (Section 7.3) elements.

6.3.3. RegistrarDetailsQry

The RegistrarDetailsQry request allows the user to retrieve the stored details for a registrar account. Registrars SHOULD be restricted to only retrieving details of their own account. The registry may retrieve information about any registrar.

6.3.3.1. Request

The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
RegistrarId	A user identifier. The registrar identifier of the registrar to retrieve stored details for.
NameFilter	A text filter. Matched against the registrar names stored in the SRS server.

The RegistrarDetailsQry request may also include a ResultDateRange (Section 8.13) element.

If the ResultDateRange element is supplied, it will cause the server to return multiple records for each registrar, describing all the changes that the registrar went through during the given period.

Otherwise, if it is not specified, only the latest data for each registrar that matches the filters will be returned. When a ResultDateRange is requested, the Allowed2LDs and the Roles are not returned (there is no historical data available for these fields).

Syntax:

```
<!ELEMENT RegistrarDetailsQry (ResultDateRange?)>
<!ATTLIST RegistrarDetailsQry
        QryId       %UID;          #IMPLIED
        RegistrarId %RegistrarId; #IMPLIED
        NameFilter  CDATA         #IMPLIED>
```

6.3.3.2. Response

The response to a RegistrarDetailsQry request will be either an Error (Section 7.7) element or one or more Registrar (Section 7.9) elements matching the provided filter details.

6.4. Registrar Write Actions

Registrar's typically only have access to a single requests for the maintenance of their details stored at the SRS:

Request	Description
RegistrarUpdate	Updates the details stored about a registrar.

6.4.1. RegistrarUpdate

The RegistrarUpdate request allows the user to update their own account details and allows the registry to maintain registrar information for any registrar.

6.4.1.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

Additionally, the request may specify:

Attribute	Description
Name	A text string. The registrar's identifier in the registry's accounting system.
AccRef	A text string. The registrar's identifier in the registry's accounting system.
URL	A text string. The registrar's public web site address.

The RegistrarUpdate request may also include new details for any of the related registrar information elements:

- RegistrarPublicContact (Section 8.10)
- RegistrarSRSContact (Section 8.10)
- DefaultTechnicalContact (Section 8.10)
- EncryptKeys (Section 8.17)
- Allowed2LDs (Section 8.3)
- Roles (Section 8.29)
- AuditText (Section 8.5)

If an attribute or element is not provided in the request, then the related details SHALL NOT be updated by the SRS server.

Syntax:

```
<!ELEMENT RegistrarUpdate (RegistrarPublicContact?,
                           RegistrarSRSContact?,
                           DefaultTechnicalContact?,
                           EncryptKeys?,
                           Allowed2LDs?,
                           Roles?,
                           AuditText?)>
<!ATTLIST RegistrarUpdate
        ActionId %UID; #REQUIRED
        Name     CDATA #IMPLIED
        AccRef   CDATA #IMPLIED
        URL      CDATA #IMPLIED>
```

6.4.1.2. Response

The response to a RegistrarUpdate request will be either an Error (Section 7.7) element or a Registrar (Section 7.9) element showing the updated state of the user details.

6.5. Registry Actions

The registry requests are used by the registry to manage the business functions of the SRS. Registrars MUST NOT have access to these functions.

The registry requests are:

Request	Description
AccessControlListAdd	Adds one or more entries to the registry's access control list.
AccessControlListQry	Queries the registry's access control list.
AccessControlListRemove	Removes one or more entries from the registry's access control list.
AdjustRegistrarAccount	Creates billing transactions in order to adjust the details of a registrar's account with the registry.
BilledUntilAdjustment	Adjusts the end date of billing for a domain name.
BillingExtract	Extracts, and potentially adds invoice details to, set of billing transactions.
BuildDnsZoneFiles	Builds the registry's DNS zone files including details of all of the currently delegated domain.
DeferredIncomeDetailQry	Reports on the billing transactions that contribute to an amount of deferred income for an income month.
DeferredIncomeSummaryQry	Reports on all deferred income for an income month.
GenerateDomainReport	Generates a report on all of the domains registered in the SRS.
QryBillingAmount	Queries the per-term charge for domain registration over time.
RegistrarCreate	Creates a new registrar in the SRS and assigns initial details to the record.
RunLogCreate	Creates an entry in the SRS server log.

RunLogQry	Queries details of entries in the SRS server log.
ScheduleCancel	Cancels a scheduled process in the SRS server.
ScheduleCreate	Creates a scheduled process in the SRS server.
ScheduleQry	Queries details of a scheduled process in the SRS server.
ScheduleUpdate	Updates the details of a scheduled process in the SRS server.
SetBillingAmount	Inserts a new per-term charge for domain registrations.
SysParamsQry	Queries the details of configurable system parameters in the SRS server.
SysParamsUpdate	Updates the details of configurable system parameters in the SRS server.

6.5.1. AccessControlListAdd

The AccessControlListAdd request allows the registry to add entries to its access control lists.

6.5.1.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
Resource	A text string. An identifier for the system or resource to which the access control list entries should be added. For example, this might specify a service name such as "whoisd".
Type	A text string. A description of the function of the access control list to which the access control lists should be added. For example, "allow" or "deny".

Additionally, the request may specify:

Attribute	Description
FullResult	A boolean value. Indicates whether the result returned should contain full details of entries in the modified access control lists.

The AccessControlListAdd request must also include one or more AccessControlListEntry (Section 8.1) elements to specify the entries to be added to the access control lists, and an AuditText (Section 8.5) element giving a description for the addition.

Syntax:

```
<!ELEMENT AccessControlListAdd (AccessControlListEntry+,
                                AuditText)>
<!ATTLIST AccessControlListAdd
        Resource   CDATA     #REQUIRED
        Type       CDATA     #REQUIRED
        ActionId   %UID;     #REQUIRED
        FullResult %Boolean; "1">
```

6.5.1.2. Response

The response to an AccessControlListAdd request will be either an Error (Section 7.7) or one or more AccessControlList (Section 7.1) elements. If FullResult was specified in the query, then the returned AccessControlList elements will contain all of the AccessControlListEntry elements belonging to the modified access control lists.

6.5.2. AccessControlListQry

The AccessControlListQry request allows the registry to retrieve details of the current state of access control lists.

6.5.2.1. Request

The request may specify:

Attribute	Description
Resource	A text string. Used to match the system or resource name of the access control list.
Type	A text string. Used to match the type of the access control list.

FullResult	A boolean value. Indicates whether the result returned should contain full details of entries in the matched access control lists.

Syntax:

```
<!ELEMENT AccessControlListQry EMPTY>
<!ATTLIST AccessControlListQry
        Resource   CDATA     #IMPLIED
        Type       CDATA     #IMPLIED
        FullResult %Boolean; "0">
```

6.5.2.2. Response

The response to an AccessControlListQry request will be either an Error (Section 7.7) element or zero or more AccessControlList (Section 7.1) elements - dependent on the number of access control lists that matched the query specification.

The effect of the FullResult, Resource, and Type parameters on the results is illustrated by the following table:

FullResult	Resource and/or Type	Results
Unspecified	Unspecified	AccessControlList elements for all lists.
Specified	Unspecified	AccessControlList elements and AccessControlListEntry elements for all lists.
Unspecified	Specified	AccessControlList elements and AccessControlListEntry elements for matched lists (no EffectiveDate elements for entries).
Specified	Specified	AccessControlList elements and AccessControlListEntry elements with their EffectiveDate elements for matched lists.

6.5.3. AccessControlListRemove

The AccessControlListRemove request allows the registry to remove entries from access control lists.

6.5.3.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
Resource	A text string. An identifier for the system or resource of the access control list from which the entries should be removed.
Type	A text string. A description of the function of the access control list from which the entries should be removed.

Additionally, the request may specify:

Attribute	Description
FullResult	A boolean value. Indicates whether the result returned should contain full details of entries in the modified access control lists.

The AccessControlListRemove request must also include one or more AccessControlListEntry (Section 8.1) elements to specify the entries to be removed from the access control lists, and an AuditText (Section 8.5) element giving a description for the removal.

Syntax:

```
<!ELEMENT AccessControlListRemove (AccessControlListEntry+,
                                   AuditText)>
<!ATTLIST AccessControlListRemove
        Resource   CDATA     #REQUIRED
        Type       CDATA     #REQUIRED
        ActionId   %UID;     #REQUIRED
        FullResult %Boolean; "1">
```

6.5.3.2. Response

The response to an AccessControlListRemove request will be either an Error (Section 7.7) element or one or more AccessControlList (Section 7.1) elements. If FullResult was specified in the query, then the returned AccessControlList elements will contain all of the

AccessControlListEntry elements belonging to the access control lists.

6.5.4. AdjustRegistrarAccount

The AdjustRegistrarAccount request allows the registry to adjust the account of a registrar by creating a billing transaction in the registrar's name. This allows the registry to make adjustments by crediting or debiting the registrar account, for instance, in the case of a refunded payment.

6.5.4.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
RegistrarId	A user identifier. The identifier of the registrar whose account is to be adjusted.
DomainName	A text string. The name of the domain for which the registrar's account is being adjusted.
Months	An integer. The number of months of payment to credit to, or debit from the registrar's account for the specified domain.
ActionType	A text string. A valid accounting action; either 'Credit' or 'Debit'.

The AdjustRegistrarAccount request must also include the date details for the billing transaction as a TransactionDate (Section 8.35) element, a BillPeriodStart (Section 8.35) element, and a BillPeriodEnd (Section 8.35) element, and also an AuditText (Section 8.5) element providing audit details for the adjustment.

Syntax:

```
<!ELEMENT AdjustRegistrarAccount (TransactionDate,
                                  BillPeriodStart,
                                  BillPeriodEnd,
                                  AuditText)>
<!ATTLIST AdjustRegistrarAccount
        RegistrarId %RegistrarId;        #REQUIRED
        DomainName  %DomainName;         #REQUIRED
        ActionId    %UID;                #REQUIRED
        Months      %Number;             #REQUIRED
        ActionType  (%AccountingAction;) #REQUIRED>
```

6.5.4.2. Response

The response to a AdjustRegistrarAccount request will be either an Error (Section 7.7) element or a BillingTrans (Section 7.3) element for the new transaction details provided.

6.5.5. BilledUntilAdjustment

The BilledUntilAdjustment request allows the registry to change the date that a domain has been billed until, for example, when a domain is renewed by the registrar before the automatic renewal date arrives.

6.5.5.1. Request

The request must specify:

Attribute	Description
DomainName	A text string. The name of the domain to adjust the billed until date for.
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The BilledUntilAdjustment request must also include a NewBilledUntilDate (Section 8.35) element providing the new date for the domain billed until record, and an AuditText (Section 8.5) element describing the reason for the change.

Syntax:

```
<!ELEMENT BilledUntilAdjustment (NewBilledUntilDate,AuditText)>
<!ATTLIST BilledUntilAdjustment
        DomainName %DomainName; #REQUIRED
        ActionId   %UID;        #REQUIRED>
```

6.5.5.2. Response

The response to a BilledUntilAdjustment request will be either an Error (Section 7.7) element or a full Domain (Section 7.5) element showing all details for the updated domain, including the new BilledUntil date.

6.5.6. BillingExtract

The BillingExtract request allows the registry to retrieve details of billing transactions and to assign invoice numbers and dates to groups of billing transactions.

This transaction is expected to support the registry's customer invoicing and accounting systems. The registry first extracts the details of fees that have been incurred through domain registrations and renewals for a period. These transactions may be entered into the registry's accounting system to be issued as an invoice to the registrar concerned. Once the invoice has been issued, the BillingExtract may be run again as an invoice extract to assign the accounting system invoice number and maybe the invoice date to the requests and mark the transactions as having been billed.

The registry may use this request to retrieve details of billing transactions for any registrar.

6.5.6.1. Request

The BillingExtract request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

Additionally, the request may also include:

Attribute	Description
RegistrarId	A user identifier. The registrar identifier of the registrar that incurred the billing transactions to be retrieved.
ConfirmedTrans	A boolean value. Indicates that the SRS should only extract details of confirmed billing transactions.
InsideGracePeriod	A boolean value. Indicates whether the SRS should extract details of billing transactions that have been issued for domains that are within their registration or renewal grace period.
InvoiceExtract	A boolean value. Indicates whether the request should apply the supplied invoice identifier, and an invoice date if one is provided, to the matching transactions.
InvoiceId	An invoice identifier. The identifier from the registry's accounting system to be applied to the extracted billing transactions.

The request must also include a TransDateRange (Section 8.13) element and may include an InvoiceDate (Section 8.35) element. The TransDateRange element specifies the start date and end date to be included in the billing extract. The InvoiceDate (Section 8.35) element, if provided, specifies a date to be applied to the extracted invoice details - this date is only applied to the extracted details if the InvoiceExtract attribute was provided with a true value.

Syntax:

```
<!ELEMENT BillingExtract (TransDateRange,InvoiceDate?)>
<!ATTLIST BillingExtract
        ActionId          %UID;     #REQUIRED
        InvoiceExtract    %Boolean; #IMPLIED
        RegistrarId       %UID;     #IMPLIED
        ConfirmedTrans    %Boolean; #IMPLIED
        InsideGracePeriod %Boolean; #IMPLIED
        InvoiceId         CDATA     #IMPLIED>
```

6.5.6.2. Response

The response to a BillingExtract request will be either an Error (Section 7.7) element or a set of BillingTrans (Section 7.3) elements that matched the request terms.

If an invoice extract was requested, the returned transactions will be marked with the provided invoice identifier and maybe the optional invoice date.

6.5.7. BuildDnsZoneFiles

The BuildDnsZoneFiles request allows the registry to initiate the creation of DNS zone and configuration files for all registered domains that have been set to delegate. The SRS server SHALL generate a run-log entry for this request recording the final status and MAY include statistics on the domains exported to the DNS.

The SRS server MUST NOT allow multiple instances of this request to run simultaneously.

6.5.7.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The BuildDnsZoneFiles request may also include a RunDate (Section 8.35) element to specify the date and time to run the zone build process.

Syntax:

```
<!ELEMENT BuildDnsZoneFiles (RunDate)>
<!ATTLIST BuildDnsZoneFiles
        ActionId %UID; #REQUIRED>
```

6.5.7.2. Response

The response to a BuildDnsZoneFiles request will be either an Error (Section 7.7) element or a RunLog (Section 7.10) element containing details produced by the zone file build process.

6.5.8. DeferredIncomeDetailQry

The DeferredIncomeDetailQry request allows the registry to obtain a report of billing transactions that contribute to an amount of deferred income (for example, to provide a breakdown of the billing transactions that make up an amount of deferred income reported by the DeferredIncomeSummaryQry (Section 6.5.9) request).

6.5.8.1. Request

The request must specify:

Attribute	Description
BaseMonth	An integer. The latest month (January = 1, December = 12) of the base year to consider billing transactions for when calculating deferred income. Billing transactions up to, and including, this month will be included.
BaseYear	An integer. The latest year to consider billing transactions for when calculating deferred income.
IncomeMonth	An integer. The month (January = 1, December = 12) of the income year to calculate deferred income for.
IncomeYear	An integer. The year to calculate deferred income for.

Additionally, the request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.

Syntax:

```
<!ELEMENT DeferredIncomeDetailQry EMPTY>
<!ATTLIST DeferredIncomeDetailQry
        BaseMonth   CDATA #REQUIRED
        BaseYear    CDATA #REQUIRED
        IncomeMonth CDATA #REQUIRED
        IncomeYear  CDATA #REQUIRED
        QryId       %UID; #IMPLIED>
```

6.5.8.2. Response

The response to a DeferredIncomeDetailQry request will be either an Error (Section 7.7) element or a set of BillingTrans (Section 7.3) elements consisting of all of the billing transactions that contributed to deferred income for the specified period.

6.5.9. DeferredIncomeSummaryQry

The DeferredIncomeSummaryQry request allows the registry to generate a report on the amount of deferred income that can be realised in any given month for the period provided in the request.

6.5.9.1. Request

The request must specify:

Attribute	Description
BaseMonth	An integer. The latest month (January = 1, December = 12) of the base year to consider billing transactions for when calculating deferred income. Billing transactions up to, and including, this month will be included.
BaseYear	An integer. The latest year to consider billing transactions for when calculating deferred income.
IncomeMonth	An integer. The month (January = 1, December = 12) of the income year to calculate deferred income for.
IncomeYear	An integer. The year to calculate deferred income for.

Additionally, the request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.

Syntax:

```
<!ELEMENT DeferredIncomeSummaryQry EMPTY>
<!ATTLIST DeferredIncomeSummaryQry
        BaseMonth   CDATA #REQUIRED
        BaseYear    CDATA #REQUIRED
        IncomeMonth CDATA #REQUIRED
        IncomeYear  CDATA #REQUIRED
        QryId       %UID; #IMPLIED>
```

6.5.9.2. Response

The response to a DeferredIncomeSummaryQry request will be either an Error (Section 7.7) element or a set of DeferredRegistrarIncome (Section 7.4) elements containing the deferred income contribution for each registrar for the specified period.

6.5.10. GenerateDomainReport

The GenerateDomainReport request allows the registry to initiate the creation of a domain report containing details of the domains registered in the SRS system. SRS implementations may vary in the information that they include in this report.

6.5.10.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The GenerateDomainReport request must also include a RunDate (Section 8.35) element providing the date and time at which the domain report creation process should run.

Syntax:

```
<!ELEMENT GenerateDomainReport (RunDate)>
<!ATTLIST GenerateDomainReport
        ActionId %UID; #REQUIRED>
```

6.5.10.2. Response

The response to a GenerateDomainReport request will be either an Error (Section 7.7) element or a RunLog (Section 7.10) element containing details produced by the domain report creation process.

6.5.11. QryBillingAmount

The QryBillingAmount request allows the registry to query all of the billing amounts in the system (including all historical billing amounts, the current effective billing amount and any future billing amounts). It does not set a new billing amount.

6.5.11.1. Request

The QryBillingAmount request has no required parameters. The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.

Syntax:

```
<!ELEMENT QryBillingAmount EMPTY>
<!ATTLIST QryBillingAmount
        QryId %UID; #IMPLIED>
```

6.5.11.2. Response

The QryBillingAmount response will either be an Error (Section 7.7) element or a set of BillingAmount (Section 7.2) elements. If a QryId was provided with the request, this will be returned in the TransId attribute of the Response (Section 5.2.1) element.

6.5.12. RegistrarCreate

The RegistrarCreate request allows the user to create a new registrar user account in the SRS server and to provide details for it.

6.5.12.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
Name	A text string. The name of the registrar.
AccRef	A text string. The registrar's identifier in the registry's accounting system.
RegistrarId	A user identifier. The unique number to be assigned to the new registrar.

Additionally, the request may specify:

Attribute	Description
URL	A text string. The registrar's public web site address.

The RegistrarCreate request must also include a RegistrarPublicContact (Section 8.10) element specifying the public contact details for the registrar, a RegistrarSRSContact (Section 8.10) element specifying the contact details for the registry's use, a DefaultTechnicalContact (Section 8.10) element which will be applied to domains registered through the registrar if no other technical contact is specified, and an EncryptKeys (Section 8.17) element providing one or more public keys that the registrar will use.

The RegistrarCreate request may also include the Allowed2LDs (Section 8.3) element specifying the domains in which the registrar may manage domain registrations, a Roles (Section 8.29) element defining system access roles for the registrar, and the AuditText (Section 8.5) element to provide details on the addition of the new registrar account.

Syntax:

```
<!ELEMENT RegistrarCreate (RegistrarPublicContact,
                           RegistrarSRSContact,
                           DefaultTechnicalContact,
                           EncryptKeys,
                           Allowed2LDs?,
                           Roles?,
                           AuditText?)>
<!ATTLIST RegistrarCreate
        ActionId    %UID;          #REQUIRED
        Name        CDATA          #REQUIRED
        AccRef      CDATA          #REQUIRED
        RegistrarId %RegistrarId;  #REQUIRED
        URL         CDATA          #IMPLIED>
```

6.5.12.2. Response

The response to a RegistrarCreate request will be either an Error (Section 7.7) element or a Registrar (Section 7.9) element showing the details for the new registrar account.

6.5.13. RunLogCreate

The RunLogCreate request allows the registry to request creation of a log entry in the SRS server's run log. This is useful for allowing batch processes that interface with the SRS server to log messages to a standard location.

6.5.13.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The RunLogCreate request must also include a FirstRunDate (Section 8.35) element specifying the date at which the process was initiated, and a RunLog (Section 7.10) element containing the details of the log entry to be created.

Syntax:

```
<!ELEMENT RunLogCreate (FirstRunDate,RunLog)>
<!ATTLIST RunLogCreate
        ActionId %UID; #REQUIRED>
```

6.5.13.2. Response

The response to a RunLogCreate request will be either an Error (Section 7.7) element or a RunLog (Section 7.10) element containing the details of the log entry that was created.

6.5.14. RunLogQry

The RunLogQry request allows the registry to query the SRS server for details of log entries created by processes that interact with the SRS server.

6.5.14.1. Request

The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
ProcessName	A text string. The name of a scheduled job process that created the run log. This will be matched against existing run log entries to limit the results returned.
Parameters	A text string. The parameters with which a process was called. This will be matched against existing run log entries to limit the results returned.

The RunLogQry request may also include a LogDateRange (Section 8.13) element defining the period for which to return run log entry details.

If no limiting terms are specified, the SRS server will return all run log entries. The server MAY impose a limit on the number of entries that are returned in a single response.

Syntax:

```
<!ELEMENT RunLogQry (LogDateRange?)>
<!ATTLIST RunLogQry
        QryId       %UID; #IMPLIED
        ProcessName CDATA #IMPLIED
        Parameters  CDATA #IMPLIED>
```

6.5.14.2. Response

The response to a RunLogQry request will be either an Error (Section 7.7) element or a set of RunLog (Section 7.10) elements matching the provided terms.

6.5.15. ScheduleCancel

The ScheduleCancel request allows the registry to cancel a scheduled process that has been registered in the SRS server using the ScheduleCreate (Section 6.5.16) request.

6.5.15.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
ProcessName	A text string. The name of a scheduled job process to be cancelled.

Additionally, the request may specify:

Attribute	Description
Parameters	A text string. The parameters with which a process to be cancelled was called.

The ScheduleCancel request must also include a FirstRunDate (Section 8.35) element identifying the first run date of the scheduled job process to be cancelled, and may include a AuditText (Section 8.5) element providing the reasons for the job cancellation.

Syntax:

```
<!ELEMENT ScheduleCancel (FirstRunDate,AuditText?)>
<!ATTLIST ScheduleCancel
        ActionId    %UID;            #REQUIRED
        ProcessName (%ScheduledJob;) #REQUIRED
        Parameters  CDATA            #IMPLIED>
```

6.5.15.2. Response

The response to a ScheduleCancel request will be either an Error (Section 7.7) element or a Schedule (Section 7.11) element giving the details of the cancelled job, including the new cancellation date.

6.5.16. ScheduleCreate

The ScheduleCreate request allows the registry to insert a scheduled job entry into the SRS server to be run at a specified time, and possibly at regular intervals after that time. This may be used to set times for running processes that support the day-to-day work of the registry. These processes are defined by the ScheduledJob (Section 9.15.1) entity.

6.5.16.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
ProcessName	A text string. The name of the process to be executed at the designated time.
Frequency	An text string. The delay between repeat executions of the process, after the initial run date. The format for this is implementation dependent but it is recommended that a simple text description be used; as examples: "15 minutes", "1 hour", "1 day". Registries should ensure that the frequency is not shorter than the run-time for the process.

Additionally, the request may specify:

Attribute	Description
Parameters	A text string. The parameters to be passed to the process when it is executed.

The ScheduleCreate request must also include a FirstRunDate (Section 8.35) element providing the date and time at which the process should first be executed. The request may also include a FinalRunDate (Section 8.35) element providing the last date and time at which a repeating process should be executed, and an AuditText (Section 8.5) element providing details about the process creation request.

Syntax:

```
<!ELEMENT ScheduleCreate (FirstRunDate,FinalRunDate?,AuditText?)>
<!ATTLIST ScheduleCreate
        ProcessName (%ScheduledJob;) #REQUIRED
        Frequency   CDATA            #REQUIRED
        Parameters  CDATA            #IMPLIED
        ActionId    %UID;            #REQUIRED>
```

6.5.16.2. Response

The response to a ScheduleCreate request will be either an Error (Section 7.7) element or a Schedule (Section 7.11) element with the newly created scheduled job details.

6.5.17. ScheduleQry

The ScheduleQry request allows the registry to query details of scheduled jobs in the SRS system. The results may be filtered to return only scheduled events with particular characteristics.

6.5.17.1. Request

The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.
ProcessName	A text string. Used to match against the name of a scheduled process.

Attribute	Description
Parameters	A text string. Used to match against the parameters specified for a scheduled process.

The ScheduleQry request may also include an ActiveOn (Section 8.35) element providing a date and time at which a process must have been active (between the first run date and the last run date of a repeating process), and a FirstRunDate (Section 8.35) element to match against the first run dates of scheduled jobs.

Syntax:

```
<!ELEMENT ScheduleQry (ActiveOn?,FirstRunDate?)>
<!ATTLIST ScheduleQry
        QryId       %UID; #IMPLIED
        ProcessName CDATA #IMPLIED
        Parameters  CDATA #IMPLIED>
```

6.5.17.2. Response

The response to a ScheduleQry request will be either an Error (Section 7.7) element or a set of Schedule (Section 7.11) elements, one for each schedule that matched the request parameters.

6.5.18. ScheduleUpdate

The ScheduleUpdate request allows the registry to amend the details for an existing Schedule (Section 7.11) entry in the SRS server.

6.5.18.1. Request

The request may specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.
ProcessName	A text string. The name of the process to be executed at the designated time.

Additionally, the request may specify:

Attribute	Description
Parameters	A text string. The parameters to be passed to the process when it is executed.

The ScheduleUpdate request must also include a FirstRunDate (Section 8.35) element to specify the new first run date for the process. The request may include a LastRunDate (Section 8.35) element to specify the most recent date at which a repeating process ran and an AuditText (Section 8.5) element providing details about the schedule update request.

Syntax:

```
<!ELEMENT ScheduleUpdate (FirstRunDate,LastRunDate?,AuditText?)>
<!ATTLIST ScheduleUpdate
        ActionId    %UID;             #REQUIRED
        Parameters  CDATA             #IMPLIED
        ProcessName (%ScheduledJob;) #REQUIRED>
```

6.5.18.2. Response

The response to a ScheduleUpdate request will be either an Error (Section 7.7) element or a Schedule (Section 7.11) element with the new scheduled job details.

6.5.19. SetBillingAmount

The SetBillingAmount request allows the registry to set a new per-domain monthly charge for registered domains within the SRS.

Each change to the monthly charge MUST also include a date that the new charge will become effective. SRS implementations SHOULD ensure that the effective date is not in the past (that is, charges should not be altered retroactively). Future billing amount changes may be amended by issuing a new SetBillingAmount request with the same effective date.

If the request succeeds, the response will provide a list of all stored billing amounts and their effective dates. On failure an error response is returned.

6.5.19.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The SetBillingAmount request must also include a BillingAmount (Section 7.2) element specifying the new per-domain monthly charge and the effective date for the new charge.

Syntax:

```
<!ELEMENT SetBillingAmount (BillingAmount)>
<!ATTLIST SetBillingAmount
        ActionId %UID; #REQUIRED>
```

Example:

```
<SetBillingAmount ActionId="000000123">
   <BillingAmount Amount="7.50">
       <EffectiveDate Year="2007" Month="11" Day="19" Hour="12"
          Minute="00" Second="00" TimeZoneOffset="+13:00" />
   </BillingAmount>
</SetBillingAmount>
```

6.5.19.2. Response

The response to a SetBillingAmount request will be either an Error (Section 7.7) element or a set of BillingAmount (Section 7.2) elements, one for each billing amount in the history of the SRS server and the new value.

Example:

```
<Response Action="SetBillingAmount" FeId="4" FeSeq="214"
  OrigRegistrarId="50041">
    <BillingAmount Amount="6.00">
        <EffectiveDate Year="2004" Month="01" Day="01" Hour="04"
           Minute="00" Second="00" TimeZoneOffset="+13:00" />
    </BillingAmount>
    <BillingAmount Amount="4.00">
        <EffectiveDate Year="2005" Month="01" Day="01" Hour="04"
           Minute="00" Second="00" TimeZoneOffset="+13:00" />
    </BillingAmount>
    <BillingAmount Amount="4.50">
        <EffectiveDate Year="2006" Month="01" Day="01" Hour="04"
           Minute="00" Second="00" TimeZoneOffset="+13:00" />
    </BillingAmount>
    <BillingAmount Amount="7.50">
        <EffectiveDate Year="2007" Month="11" Day="19" Hour="12"
           Minute="00" Second="00" TimeZoneOffset="+13:00" />
    </BillingAmount>
</Response>
```

6.5.20. SysParamsQry

The SysParamsQry request allows the registry to obtain a list of the current state of the SRS server's configurable parameters.

6.5.20.1. Request

The request may specify:

Attribute	Description
QryId	A query identifier. This has no meaning within the SRS; if provided, it will be returned in the response to this request.

Syntax:

```
<!ELEMENT SysParamsQry EMPTY>
<!ATTLIST SysParamsQry
        QryId %UID; #IMPLIED>
```

6.5.20.2. Response

The response to a SysParamsQry request will be either an Error (Section 7.7) element or a set of SysParam (Section 7.12) elements, one for each user-configurable system parameter in the SRS server implementation.

6.5.21. SysParamsUpdate

The SysParamsUpdate request allows the registry to update the value of any user configurable parameters in the SRS server implementation.

6.5.21.1. Request

The request must specify:

Attribute	Description
ActionId	An action identifier. This is a unique identifier for the request to the SRS server.

The SysParamsUpdate request must include one or more SysParam (Section 7.12) elements containing the new parameter details.

The SysParamsUpdate request may also include an AuditText (Section 8.5) element giving further details of the configuration changes.

Syntax:

```
<!ELEMENT SysParamsUpdate (SysParam+,AuditText?)>
<!ATTLIST SysParamsUpdate
        ActionId %UID; #REQUIRED>
```

6.5.21.2. Response

The response to a SysParamsUpdate request will be either an Error (Section 7.7) element or a set of SysParam (Section 7.12) elements, one for each changed system parameter, showing the new value.

7. SRS Response Types

7.1. AccessControlList

The AccessControlList element is used by the SRS to return details of the registry's access control lists.

The AccessControlList element must specify:

Attribute	Description
ActionId	An action identifier. The unique identifier for the request to the SRS server that prompted return of access control list details.
Resource	A text string. The system that the access control list is intended to be applied to.
Type	A text string. The type of access control list that is being returned. Registries may maintain different lists for different purposes, for example, blacklists and whitelists for controlling access.

The AccessControlList element may include one or more AccessControlListEntry (Section 8.1) elements.

Syntax:

```
<!ELEMENT AccessControlList (AccessControlListEntry*)>
<!ATTLIST AccessControlList
        Resource CDATA #REQUIRED
        Type     CDATA #REQUIRED>
```

Example:

```
<AccessControlList Resource="whoisd" Type="blacklist">
    <AccessControlListEntry Address="192.0.2.5", "Blacklisted for
        abuse." />
    <AccessControlListEntry Address="192.0.2.9", "Blacklisted for
        abuse." />
    <AccessControlListEntry Address="192.0.2.237", "Blacklisted at
        owner's request" />
</AccessControlList>
```

7.2. BillingAmount

The BillingAmount element is used by the SRS to return details of the amount charged for registered domains and the date that the amount became effective in the SRS.

The element content consists of an Amount attribute containing the charge in the registry's chosen currency, and may also include an EffectiveDate (Section 8.35) element containing the date and time that the charge became (or will become, for future changes) effective. The Amount value should be validated as an acceptable currency value.

The BillingAmount element must specify:

Attribute	Description
Amount	A numeric currency value. The amount charged per-term for domain registration and renewal.

The BillingAmount element may also include an EffectiveDate (Section 8.35) element.

Syntax:

```
<!ELEMENT BillingAmount (EffectiveDate?)>
<!ATTLIST BillingAmount
        Amount %Dollars; #REQUIRED>
```

Example:

```
<BillingAmount Amount="9.95">
    <EffectiveDate Year="2004" Month="01" Day="01" Hour="04"
        Minute="00" Second="00" TimeZoneOffset="+13:00" />
</BillingAmount>
```

7.3. BillingTrans

The BillingTrans element is used by the SRS to return details of charges that have been incurred by registrars over a given billing period.

The BillingTrans element must specify:

Attribute	Description
RegistrarId	A user identifier. The identifier of the registrar that manages the domain that the billing amount relates to.
Type	A text string. The type of transaction that the billing amount relates to, for instance, "Create" or "Renew" a domain.
TransStatus	A text string. The status of the billing transaction; valid values are defined by the BillStatus (Section 9.2.2) entity.
DomainName	A text string. The domain name that the billing transaction relates to.
BillingTerm	An integer. The number of months of payment for which the billing transaction was issued.
Amount	A numeric currency value. The amount of money attached to the billing transaction.

The BillingTrans element may also specify:

Attribute	Description
RegistrantRef	A customer identifier. A reference assigned to the registrant by the registrar.
InvoiceId	An invoice identifier. The identifier from the registry's accounting system that relates to the billing transaction.

The BillingTrans element must also include a TransDate (Section 8.35) element containing the date on which the billing transaction (domain creation, renewal, etc.) occurred, a BillPeriodStart (Section 8.35) and a BillPeriodEnd (Section 8.35) element containing the start and end of the billing period to which the transaction belongs. The element may include an InvoiceDate (Section 8.35) element containing the date of the invoice to which the billing transaction has been assigned.

Syntax:

```
<!ELEMENT BillingTrans (InvoiceDate?,TransDate,BillPeriodStart,
                        BillPeriodEnd)>
<!ATTLIST BillingTrans
        RegistrarId   %RegistrarId;  #REQUIRED
        Type          CDATA          #REQUIRED
        TransStatus   (%BillStatus;) #REQUIRED
        DomainName    %DomainName;   #REQUIRED
        RegistrantRef %UID;          #IMPLIED
        BillingTerm   %Term;         #REQUIRED
        InvoiceId     %UID;          #IMPLIED
        Amount        %Dollars;      #REQUIRED>
```

Example:

```
<BillingTrans Amount="1.50" BillingTerm="1" DomainName="example.org"
            RegistrantRef="Example registrant" RegistrarId="50041"
            TransStatus="PendingConfirmation" Type="Create">
    <TransDate Day="21" Hour="17" Minute="08" Month="2"
             Second="31" TimeZoneOffset="+13:00" Year="2008" />
    <BillPeriodStart Day="21" Hour="17" Minute="08" Month="2"
             Second="31" TimeZoneOffset="+13:00" Year="2008" />
    <BillPeriodEnd Day="21" Hour="17" Minute="08" Month="3"
             Second="31" TimeZoneOffset="+13:00" Year="2008" />
</BillingTrans>
```

7.4. DeferredRegistrarIncome

The DeferredRegistrarIncome element allows the SRS server to return details of deferred income that may be realised in a particular month by considering the domain registration payments that have been made by a registrar up to and including a base month.

The DeferredRegistrarIncome element must specify:

Attribute	Description
RegistrarId	A user identifier. The identifier of the registrar that the deferred income amount relates to.
BilledAmount	A numeric currency value. The amount of a billed amount that can be realised in the income month, considering all billing transactions up to and including the base month.
BilledCount	

The DeferredRegistrarIncome element must also include BaseMonth (Section 8.8) and BaseYear (Section 8.9) element to define the last month of billing transactions to include for calculating deferred income amounts, and IncomeMonth (Section 8.20) and IncomeYear (Section 8.21) elements to define the month to calculate realised deferred income for.

Syntax:

```
<!ELEMENT DeferredRegistrarIncome (BaseMonth,BaseYear,IncomeMonth,
                                   IncomeYear)>
<!ATTLIST DeferredRegistrarIncome
        RegistrarId  %RegistrarId; #REQUIRED
        BilledAmount %Dollars;     #REQUIRED
        BilledCount  %Number;      #REQUIRED>
```

7.5. Domain

The Domain element can contain full details of a domain and its status within the SRS server and is used for creating and updating domains, as well as for reporting the current domain details.

The Domain element must specify:

Attribute	Description
DomainName	A text string. The domain name. Must conform to the format and syntax in RFC 1035 [RFC1035], RFC 1123 [RFC1123], and RFC 2181 [RFC2181].

The Domain element may also specify:

Attribute	Description
Delegate	A boolean value. Indicates whether the domain should be delegated to appear in the DNS.
RegistrantRef	A customer identifier. Assigned to the registrant by the registrar.
RegistrarId	A user identifier. The unique identifier for the registrar that manages the domain.
RegistrarName	A text string. The name of the registrar that manages the domain.
Status	A text string. The registration status of the domain. Valid values are defined by the RegDomainStatus (Section 9.9.1) entity.

Term	A numeric value. The number of months to bill when the domain is renewed.
UDAI	A text string. The UDAI to be checked.

The Domain element may also include elements to define the contact details, name servers for the domain and various dates relating to the registration. These optional elements are:

- NameServers (Section 8.22)
- RegistrantContact (Section 8.10)
- RegistrarPublicContact (Section 8.10)
- AdminContact (Section 8.10)
- TechnicalContact (Section 8.10)
- BilledUntil (Section 8.35)
- RegisteredDate (Section 8.35)
- CancelledDate (Section 8.35)
- LockedDate (Section 8.35)
- AuditDetails (Section 8.4)

Syntax:

```
<!ELEMENT Domain (NameServers?,RegistrantContact?,
                 RegistrarPublicContact?,AdminContact?,
                 TechnicalContact?,BilledUntil?,
                 RegisteredDate?,CancelledDate?,
                 LockedDate?,AuditDetails?)>
<!ATTLIST Domain
        DomainName    %DomainName;      #REQUIRED
        RegistrantRef %UID;             #IMPLIED
        RegistrarName CDATA             #IMPLIED
        Status        (%DomainStatus;)  #IMPLIED
        Delegate      %Boolean;         #IMPLIED
        Term          %Term;            #IMPLIED
        RegistrarId   %RegistrarId;     #IMPLIED
        UDAI          %UID;             #IMPLIED>
```

Example:

```
<Domain Delegate="1" DomainName="mydomain.example.org"
      RegistrantRef="RS1001" RegistrarId="50041" Status="Active"
      Term="1" UDAI="ke783oe9">
    <NameServers>
        <Server FQDN="ns1.mydomain.example.org" />
        <Server FQDN="ns2.mydomain.example.org" />
    </NameServers>
    <RegistrantContact Name="John Smith"
        Email="j.smith@mydomain.example.org">
        <PostalAddress Address1="14 Rural Street" Address2="Suburbia"
            City="Wellington" CountryCode="NZ" PostalCode="6135" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111234" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111235" />
    </RegistrantContact>
    <AdminContact Name="ISP Administrator"
        Email="admin@myisp.example.org">
        <PostalAddress Address1="1 Main Road" Address2="Central"
            City="Wellington" CountryCode="NZ" PostalCode="6001" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111123" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111124" />
    </AdminContact>
    <TechnicalContact Name="ISP Technician"
        Email="tech@myisp.example.org">
        <PostalAddress Address1="1 Main Road" Address2="Central"
            City="Wellington" CountryCode="NZ" PostalCode="6001" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111125" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111126" />
    </TechnicalContact>
    <BilledUntil Hour="10" Minute="00" Second="00"
        Day="1" Month="2" Year="2009" TimeZoneOffset="+13:00" />
    <RegisteredDate Hour="10" Minute="00" Second="00"
        Day="1" Month="2" Year="2008" TimeZoneOffset="+13:00" />
    <AuditDetails ActionId="Domain create for mydomain.example.org"
        RegistrarId="50041">
        <AuditTime>
            <From Hour="10" Minute="00" Second="00" Day="1"
                Month="2" Year="2008" TimeZoneOffset="+13:00" />
        </AuditTime>
        <AuditText>Domain created for John Smith, RS8974</AuditText>
    </AuditDetails>
</Domain>
```

7.6. DomainTransfer

The DomainTransfer element is used by the SRS to inform a registrar of the transfer of one of the domains that they were the managing registrar for. This element will be returned in response to a GetMessages (Section 6.3.1) request.

The DomainTransfer element must specify:

Attribute	Description
RegistrarName	A text string. The name of the registrar that the domain has been transferred to.
Minute	An integer value. The minute in which the domain transfer occurred.
Hour	An integer value. The hour in which the domain transfer occurred.
Day	An integer value. The day in which the domain transfer occurred.
Month	An integer value. The month in which the domain transfer occurred.
Year	An integer value. The year in which the domain transfer occurred.

The DomainTransfer element may also specify:

Attribute	Description
Second	An integer value. The second in which the domain transfer occurred.
TimeZoneOffset	An integer value. The TimeZoneOffset from UTC of the SRS server that processed the domain transfer.

The DomainTransfer element must also include one or more TransferredDomain (Section 8.36) elements to define the domain names that have been transferred to another registrar.

Syntax:

```
<!ELEMENT DomainTransfer (TransferredDomain+)>
<!ATTLIST DomainTransfer
        RegistrarName CDATA #REQUIRED
        %TimeStamp;>
```

Example:

```
<DomainTransfer RegistrarName="My ISP" Hour="10" Minute="00"
    Second="00" Day="1" Month="4" Year="2008" TimeZoneOffset="+13" />
    <TransferredDomain>domain.co.example</TransferredDomain>
</DomainTransfer>
```

7.7. Error

The Error element may be used to present details of errors that occur when processing individual requests or a whole transaction.

The Error element must specify:

Attribute	Description
ErrorId	SRS system identifier for the error that occurred.
Severity	Indicative severity level for the error.
Hint	SRS server hint for addressing the error.

The Error element also contains a Description (Section 8.14) element and may also provide ErrorDetails (Section 8.18) elements with extra description of the error.

Syntax:

```
<!ELEMENT Error (Description, ErrorDetails*)>
<!ATTLIST Error
        ErrorId      %UID;    #REQUIRED
        Severity     %Number; #REQUIRED
        Hint         %UID;    #REQUIRED>
```

Example:

```
<Error Hint="MALFORMED_REQUEST_ERROR" ErrorId="INVALID_FIELD"
     Severity="er">
<Description>Invalid value in field</Description>
<ErrorDetails>DomainName</ErrorDetails>
</Error>
```

7.8. RawRequest and RawResponse

The RawRequest and RawResponse elements allow the SRS server to return the details of a previous request and response. The details are returned as XML (Section 8.37) and Signature (Section 8.34) elements with content encoded as parsed character data.

Syntax:

```
<!ELEMENT RawRequest (XML,Signature)>

<!ELEMENT RawResponse (XML,Signature)>
```

Example:

```
<Response Action="ActionDetailsQry" FeId="4" FeSeq="137"
  OrigRegistrarId="50041">
    <RawRequest>
        <XML>&lt;NZSRSRequest VerMajor="2" ...</XML>
        <Signature>-----BEGIN PGP SIGNATURE-----...</Signature>
    </RawRequest>
    <RawResponse>
        <XML>&lt;NZSRSResponse VerMajor="2" ...</XML>
        <Signature>-----BEGIN PGP SIGNATURE-----...</Signature>
    </RawResponse>
</Response>
```

7.9. Registrar

The Registrar element is used to contain the details for a registrar.

The Registrar element must specify:

Attribute	Description
RegistrarId	A user identifier. The unique number to be assigned to the new registrar.
Name	A text string. The name of the registrar.
AccRef	A text string. The registrar’s identifier in the registry’s accounting system.

The Registrar element may also specify:

Attribute	Description
URL	A text string. The public web site address for the registrar.

The Registrar element must also include a RegistrarPublicContact (Section 8.10) element specifying the public contact details for the registrar, a RegistrarSRSContact (Section 8.10) element specifying the contact details for the registry's use, a DefaultTechnicalContact (Section 8.10) element which will be applied to domains registered through the registrar if no other technical contact is specified, and an EncryptKeys (Section 8.17) element providing one or more public keys that the registrar will use.

The Registrar element may also include the Allowed2LDs (Section 8.3) element specifying the subdomains managed by the registry in which the registrar may manage domain registrations, a Roles (Section 8.29) element defining system access roles for the registrar, and the AuditText (Section 8.5) element to provide details on the addition of the new registrar account.

Syntax:

```
<!ELEMENT Registrar (RegistrarPublicContact,RegistrarSRSContact,
                     DefaultTechnicalContact,EncryptKeys,
                     Allowed2LDs?,Roles?,AuditDetails?)>
<!ATTLIST Registrar
        RegistrarId %RegistrarId; #REQUIRED
        Name        CDATA         #REQUIRED
        AccRef      CDATA         #REQUIRED
        URL         CDATA         #IMPLIED>
```

Example:

```
<Registrar AccRef="Reg8974" Name="My ISP" RegistrarId="50041">
    <RegistrarPublicContact Name="Customer Service"
        Email="custserv@myisp.example.org">
        <PostalAddress Address1="1 Main Road" Address2="Central"
            City="Wellington" CountryCode="NZ" PostalCode="6001" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111123" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111124" />
    </RegistrarPublicContact>
    <RegistrarSRSContact Name="Admin"
        Email="admin@myisp.example.org">
        <PostalAddress Address1="1 Main Road" Address2="Central"
            City="Wellington" CountryCode="NZ" PostalCode="6001" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111125" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111126" />
    </RegistrarSRSContact>
    <DefaultTechnicalContact Name="ISP Technician"
        Email="tech@myisp.example.org">
        <PostalAddress Address1="1 Main Road" Address2="Central"
            City="Wellington" CountryCode="NZ" PostalCode="6001" />
        <Phone AreaCode="4" CountryCode="64" LocalNumber="111123" />
        <Fax AreaCode="4" CountryCode="64" LocalNumber="111124" />
    </DefaultTechnicalContact>
```

```
        <EncryptKeys>
            <EncryptKey>-----BEGIN PGP PUBLIC KEY BLOCK-----
   Version: GnuPG v1.4.6 (GNU/Linux)

   mQGiBEfpmesRBADBkcApvlyH67pTIDObNgUm40u5WuLMkBvP8a9RWJNACdbUjMj+
   ...
   =nDaR
   -----END PGP PUBLIC KEY BLOCK-----</EncryptKey>
        </EncryptKeys>
        <Allowed2LDs>
            <SecondLD DomainName="co.example" />
            <SecondLD DomainName="org.example" />
        </Allowed2LDs>
        <Roles>
            <Role RoleName="CancelDomain" />
            <Role RoleName="Connect" />
            <Role RoleName="CreateDomain" />
            <Role RoleName="Query" />
            <Role RoleName="Registrar" />
            <Role RoleName="TransferDomain" />
            <Role RoleName="UncancelDomain" />
            <Role RoleName="UpdateDomain" />
            <Role RoleName="UpdateRegistrar" />
            <Role RoleName="Whois" />
        </Roles>
        <AuditDetails ActionId="Registrar update My ISP"
            RegistrarId="50041">
            <AuditTime>
                <From Hour="17" Minute="09" Second="10" Day="21"
                    Month="2" Year="2008" TimeZoneOffset="+13:00" />
            </AuditTime>
            <AuditText>Update contact details</AuditText>
        </AuditDetails>
   </Registrar>
```

7.10. RunLog

The RunLog element is used to provide details of log messages from processes, such as batch processes, that interact with the SRS server. This includes the scheduled job processes that support the running of the registry, as well as any other systems that wish to log messages in the SRS server log.

The RunLog element must specify:

Attribute	Description
ActionStatus	A text string. The status of the process that created the log.
ProcessName	A text string. The name of the process that created the run log entry.

The RunLog element may also specify:

Attribute	Description
Parameters	A text string. The parameters with which the process was called.
Control	A text string. A free-text field for storing extra information about the process run. For example, SRS server implementations may store the name of the server that executed the process and the completion time of the process here.

Syntax:

```
<!ELEMENT RunLog (RunLogTimeStamp,RunLogDetails?)>
<!ATTLIST RunLog
        ProcessName  CDATA #REQUIRED
        Parameters   CDATA #IMPLIED
        ActionStatus CDATA #REQUIRED
        Control      CDATA #IMPLIED>
```

7.11. Schedule

The Schedule element is used to return details of scheduled events within the SRS system.

The Schedule element must specify:

Attribute	Description
ProcessName	A text string. The name of the scheduled job process.
Frequency	An text string. The delay between repeat executions of the process, after the initial run date.

CreateByRegistrarId	A user identifier. The registry's registrar identifier that was used to create the scheduled job.
CreateActionId	An action identifier. This is the unique identifier for the request to the SRS server that created the scheduled job.

The Schedule element may also specify:

Attribute	Description
Parameters	A text string. The parameters to be passed to the process when it is executed.
CancelByRegistrarId	A user identifier. The registry's registrar identifier that was used to cancel the scheduled job.
CancelActionId	An action identifier. This is the unique identifier for the request to the SRS server that cancelled the scheduled job.

The Schedule element must also include a FirstRunDate (Section 8.35) element specifying the first date and time at which the process should be executed, and may also include a FinalRunDate (Section 8.35) element specifying the last time at which the process should run (for a repeating process), a CreateAuditText (Section 8.12) element providing details on the creation of the scheduled job entry, and a CancelAuditText (Section 8.7) element providing details on the cancellation of the scheduled job entry.

Syntax:

```
<!ELEMENT Schedule (FirstRunDate,FinalRunDate?,CreateAuditText?,
                   CancelAuditText?)>
<!ATTLIST Schedule
        ProcessName         (%ScheduledJob;) #REQUIRED
        Frequency           CDATA            #REQUIRED
        Parameters          CDATA            #IMPLIED
        CreateByRegistrarId %UID;            #REQUIRED
        CreateActionId      %UID;            #REQUIRED
        CancelByRegistrarId %UID;            #IMPLIED
        CancelActionId      %UID;            #IMPLIED>
```

Example:

```
<Schedule CreateActionId="BUILDZONES-20060601"
          CreateByRegistrarId="50001" Frequency="01:00:00"
          Parameters="" ProcessName="BuildDnsZoneFiles">
    <FirstRunDate Day="21" Hour="17" Minute="10" Month="2"
                  Second="30" TimeZoneOffset="+13:00" Year="2008" />
    <CreateAuditText>Registry regular zone build</CreateAuditText>
</Schedule>
```

7.12. SysParam

The SysParam element is used to return details of system settings and any audit details relating to them.

The SysParam element must specify:

Attribute	Description
Name	A text string. The name of the system parameter.

The SysParam element must include a ParamValue (Section 8.24) element containing the value of the system parameter, and may include an AuditDetails (Section 8.4) element containing details about the most recent change to the parameter.

Syntax:

```
<!ELEMENT SysParam (ParamValue, AuditDetails?)>
<!ATTLIST SysParam
        Name CDATA #REQUIRED>
```

7.13. UDAIValid

The UDAIValid response element is used to return the result of checking the validity of a UDAI string for a given DomainName. The element has a single boolean attribute containing the result of the UDAI validation.

The UDAIValid element must specify:

Attribute	Description
Valid	A boolean value. The validity status of the tested UDAI value.

Syntax:

```
<!ELEMENT UDAIValid EMPTY>
<!ATTLIST UDAIValid
        Valid %Boolean; #REQUIRED>
```

Example:

```
<UDAIValid Valid="1" />
```

8. Information Elements

This section documents elements from the XML DTD that are not used as request or response elements. These information elements contain data passed between the client and server during normal registry usage. The elements are used by requests and responses throughout the system.

8.1. AccessControlListEntry

The AccessControlListEntry contains details for a single entry in the registry's access control list and may also include an EffectiveDate (Section 8.35) element containing the date and time that the access control list entry became effective. It has two attributes:

Address: An identifier to be used to match and control client access to network services (for example, an IP address or a range of IP addresses).

Comment: A brief comment giving the reason for entry in the access control list.

The Address is a required element and should contain a value that can be used to control access to network services. For example, a range of IPv4 addresses in the format: 192.0.2.0/24.

The Comment is an optional element which can be used to maintain a record of the reason that Addresses were added to the access control list.

The AccessControlListEntry element is used by the AccessControlList (Section 7.1) element to return details of access control lists, and by the AccessControlListAdd (Section 6.5.1) and AccessControlListRemove (Section 6.5.3) queries to specify entries to add to and remove from access control lists.

Syntax:

```
<!ELEMENT AccessControlListEntry (EffectiveDate?)>
<!ATTLIST AccessControlListEntry
        Address CDATA #REQUIRED
        Comment CDATA #IMPLIED>
```

8.2. ActionIdFilter

The ActionIdFilter element contains a pattern match string to match against the stored action identifiers of previous requests made to the SRS. The content should be a valid text filter (Section 10).

Syntax:

```
<!ELEMENT ActionIdFilter (#PCDATA)>
```

8.3. Allowed2LDs

The Allowed2LDs element contains zero or more SecondLD (Section 8.31) elements and is used to hold a list of the subdomains managed by the registry for which a registrar may manage domain registrations.

Syntax:

```
<!ELEMENT Allowed2LDs (SecondLD*)>
```

8.4. AuditDetails

The AuditDetails element contains the registrar and action identifiers of actions for system auditing purposes. The element may also contain details of the time of the action in an AuditTime (Section 8.35) element and a text description in an AuditText (Section 8.5) element.

Syntax:

```
<!ELEMENT AuditDetails (AuditTime?,AuditText?)>
<!ATTLIST AuditDetails
        RegistrarId CDATA #IMPLIED
        ActionId    %UID; #IMPLIED>
```

8.5. AuditText

The AuditText element is a simple element that contains parsed character data that describes an action in the SRS. For instance, it may contain a text description of the purpose of a request.

Syntax:

```
<!ELEMENT AuditText (#PCDATA)>
```

8.6. AuditTextFilter

The AuditTextFilter element is a simple element that contains parsed character data. The content of this element is used as a pattern match string to match against audit text for previous transactions made to the SRS. The content should be a valid text filter (Section 10).

Syntax:

```
<!ELEMENT AuditTextFilter (#PCDATA)>
```

8.7. CancelAuditText

The CancelAuditText element is a simple element that contains parsed character data. The content of this element is used to hold the text of an audit message provided during cancellation of a scheduled process in the SRS. It is used by the Schedule (Section 7.11) response element.

Syntax:

```
<!ELEMENT CancelAuditText (#PCDATA)>
```

8.8. BaseMonth

The BaseMonth element is an empty element. It has a single attribute, Month, that must contain a valid month value in the range 1 to 12.

Syntax:

```
<!ELEMENT BaseMonth EMPTY>
<!ATTLIST BaseMonth
        Month %Number; #REQUIRED>
```

8.9. BaseYear

The BaseYear element is an empty element. It has a single attribute, Year, that must contain a valid year value.

Syntax:

```
<!ELEMENT BaseYear EMPTY>
<!ATTLIST BaseYear
        Year %Number; #REQUIRED>
```

8.10. Contact Details Elements

The SRS protocol employs many contact details elements that all share a common definition. The contact details elements are:

- AdminContact
- DefaultTechnicalContact
- RegistrantContact
- RegistrarPublicContact
- RegistrarSRSContact
- TechnicalContact

The contact details elements are defined using the Contact (Section 9.4.1) entity definition for the element content, and the ContactAttr (Section 9.4.2) entity for the element attribute definitions.

The contact details elements have Name and Email attributes to store the contact name and email address; either attribute may be omitted where appropriate. The contact details elements may also include elements for PostalAddress (Section 8.25), Phone (Section 8.27), and Fax (Section 8.27) details.

Syntax:

```
<!ELEMENT AdminContact (%Contact;)>
<!ATTLIST AdminContact %ContactAttr;>

<!ELEMENT DefaultTechnicalContact (%Contact;)>
<!ATTLIST DefaultTechnicalContact %ContactAttr;>

<!ELEMENT RegistrantContact (%Contact;)>
<!ATTLIST RegistrantContact %ContactAttr;>

<!ELEMENT RegistrarPublicContact (%Contact;)>
<!ATTLIST RegistrarPublicContact %ContactAttr;>

<!ELEMENT RegistrarSRSContact (%Contact;)>
<!ATTLIST RegistrarSRSContact %ContactAttr;>

<!ELEMENT TechnicalContact (%Contact;)>
<!ATTLIST TechnicalContact %ContactAttr;>
```

Example:

```
<AdminContact Name="John Doe" Email="john_doe@example.org">
    <PostalAddress Address1="1 Example Street" Address2=""
        City="Example" CountryCode="NZ" PostalCode="99001"
        Province="" />
    <Phone CountryCode="99" AreaCode="8" LocalNumber="555-5678"/>
    <Fax CountryCode="99" AreaCode="8" LocalNumber="555-5679"/>
</AdminContact>
```

8.11. Contact Details Search Filters

The contact details search filter elements all share a common definition referenced from the ContactFilter (Section 9.5.1) entity definition for the element content, and the ContactFilterAttr (Section 9.5.2) entity for the element attribute definitions. The contact details filters are used to provide fields for matching against stored contact details when using the DomainDetailsQry (Section 6.1.2) request. Only the details provided in the filter are used to match against stored values in the SRS, this enables general matches, for instance, to select all domains with a particular administrative contact name.

The contact details filter elements have the same structure as the contact details elements. Name and Email attributes specify the match values for contact name and email address; either attribute may be omitted where appropriate. The contact details filter elements may also include filter details elements for PostalAddress

(Section 8.26), Phone (Section 8.27), and Fax (Section 8.27) details.

Syntax:

```
<!ELEMENT AdminContactFilter (%ContactFilter;)>
<!ATTLIST AdminContactFilter %ContactAttrFilter;>

<!ELEMENT RegistrantContactFilter (%ContactFilter;)>
<!ATTLIST RegistrantContactFilter %ContactAttrFilter;>

<!ELEMENT TechnicalContactFilter (%ContactFilter;)>
<!ATTLIST TechnicalContactFilter %ContactAttrFilter;>
```

8.12. CreateAuditText

The CreateAuditText element is a simple element that contains parsed character data. The content of this element is used to hold the text of an audit message provided during creation of a scheduled process in the SRS. It is used by the Schedule (Section 7.11) response element.

Syntax:

```
<!ELEMENT CreateAuditText (#PCDATA)>
```

8.13. Date Range Elements

The SRS protocol employs many date range elements that all share a common definition. The date range elements are:

- AuditTime
- BilledUntilDateRange
- CancelledDateRange
- ChangedInDateRange
- InvoiceDateRange
- LockedDateRange
- LogDateRange
- RegisteredDateRange
- ResultDateRange

- SearchDateRange

- TransDateRange

The content of the date range elements is defined using the DateRange (Section 9.7.4) entity. The date range elements can contain two optional elements, a From (Section 8.35) date and a To (Section 8.35) date, that define the start and end of a range of dates.

If the From date is omitted, SRS implementations SHOULD use the earliest action in the system as the From date value. If the To date is omitted, SRS implementation SHOULD use the current date as the To date value.

Syntax:

```
<!ELEMENT AuditTime            (%DateRange;)>

<!ELEMENT BilledUntilDateRange (%DateRange;)>

<!ELEMENT CancelledDateRange   (%DateRange;)>

<!ELEMENT ChangedInDateRange   (%DateRange;)>

<!ELEMENT InvoiceDateRange     (%DateRange;)>

<!ELEMENT LockedDateRange      (%DateRange;)>

<!ELEMENT LogDateRange         (%DateRange;)>

<!ELEMENT RegisteredDateRange  (%DateRange;)>

<!ELEMENT ResultDateRange      (%DateRange;)>

<!ELEMENT SearchDateRange      (%DateRange;)>

<!ELEMENT TransDateRange       (%DateRange;)>
```

Example of a ResultDateRange element:

```
<ResultDateRange>
    <From Year="2008" Month="01" Day="01" Hour="00"
        Minute="00" Second="00" TimeZoneOffset="+13:00" />
    <To Year="2008" Month="01" Day="31" Hour="23"
        Minute="59" Second="59" TimeZoneOffset="+13:00" />
</ResultDateRange>
```

8.14. Description

The Description element is a simple element that contains parsed character data. The content of this element is a text description of an error condition. It is used by the Error (Section 7.7) element.

Syntax:

```
<!ELEMENT Description (#PCDATA)>
```

8.15. DomainNameFilter

The DomainNameFilter element is a simple element that contains parsed character data. The content of this element is used as a pattern match string to match against the names of domains registered in the SRS. The content should be a valid text filter (Section 10) for matching against domain names.

Syntax:

```
<!ELEMENT DomainNameFilter (#PCDATA)>
```

8.16. EncryptKey

The EncryptKey element is a simple element that contains parsed character data. The content of this element will be an ASCII armored copy of a registrar's OpenPGP-compatible public key. It is used by the EncryptKeys (Section 8.17) element.

Syntax:

```
<!ELEMENT EncryptKey (#PCDATA)>
```

8.17. EncryptKeys

The EncryptKeys element is a simple simple container for EncryptKey (Section 8.16) elements. It is used when creating and updating registrars and when the SRS returns details of registrars.

Syntax:

```
<!ELEMENT EncryptKeys (EncryptKey*)>
```

8.18. ErrorDetails

The ErrorDetails element is a simple element that contains parsed character data. The content of this element is a text description providing further details about an error condition. It is used by the Error (Section 7.7) element.

Syntax:

```
<!ELEMENT ErrorDetails (#PCDATA)>
```

8.19. FieldList

The FieldList element is an empty element with a set of Boolean (Section 9.3.3) attributes. It is used in the DomainDetailsQry (Section 6.1.2) request to specify the domain details to return for matching domains in the SRS.

The domain details that may be requested and their FieldList attributes are:

Status: The registration status of the domain ("Active" or "PendingRelease").

NameServers: The list of name servers that the domain is delegated to.

RegisteredDate: The date and time that the domain was registered.

AdminContact: The domain's administrative contact details.

RegistrantContact: The domain's registrant contact details.

TechnicalContact: The domain's technical contact details.

LockedDate: The date that a locked domain was locked.

Delegate: Whether the domain is delegated in the DNS system.

RegistrarId: The registry-assigned registrar number.

RegistrarName: The name of the registrar that manages the domain.

RegistrantRef: The personal reference for the registrant of the domain.

LastActionId: The unique identifier for the last transaction that amended the domain details in the SRS.

ChangedByRegistrarId: The identifier of the registrar that last updated the domain.

Term: The billing term for the domain registration.

BilledUntil: The date and time that the domain has been billed until.

CancelledDate: The date that a cancelled domain registration was cancelled.

AuditText: The audit text for changes to the domain. This is an optional reference for the user.

EffectiveFrom: The date that this domain record was replaced.

Syntax:

```
<!ELEMENT FieldList EMPTY>
<!ATTLIST FieldList
        Status               %Boolean; #IMPLIED
        NameServers          %Boolean; #IMPLIED
        RegistrantContact    %Boolean; #IMPLIED
        RegisteredDate       %Boolean; #IMPLIED
        AdminContact         %Boolean; #IMPLIED
        TechnicalContact     %Boolean; #IMPLIED
        LockedDate           %Boolean; #IMPLIED
        Delegate             %Boolean; #IMPLIED
        RegistrarId          %Boolean; #IMPLIED
        RegistrarName        %Boolean; #IMPLIED
        RegistrantRef        %Boolean; #IMPLIED
        LastActionId         %Boolean; #IMPLIED
        ChangedByRegistrarId %Boolean; #IMPLIED
        Term                 %Boolean; #IMPLIED
        BilledUntil          %Boolean; #IMPLIED
        CancelledDate        %Boolean; #IMPLIED
        AuditText            %Boolean; #IMPLIED
        EffectiveFrom        %Boolean; #IMPLIED>
```

8.20. IncomeMonth

The IncomeMonth element is an empty element. It has a single attribute, Month, that must contain a valid month value in the range 1 to 12.

Syntax:

```
<!ELEMENT IncomeMonth EMPTY>
<!ATTLIST IncomeMonth
        Month %Number; #REQUIRED>
```

8.21. IncomeYear

The IncomeYear element is an empty element. It has a single attribute, Year, that must contain a valid year value.

Syntax:

```
<!ELEMENT IncomeYear EMPTY>
<!ATTLIST IncomeYear
        Year %Number; #REQUIRED>
```

8.22. NameServers

The NameServers element is a simple container for Server (Section 8.32) elements. It is used when creating and updating domains and when the SRS returns details of the name servers for domains. SRS implementations MAY set a limit on the number of server elements that can be provided.

Syntax:

```
<!ELEMENT NameServers (Server*)>
```

8.23. NameServerFilter

The NameServerFilter element is a simple container for ServerFilter (Section 8.33) elements. It is used in the DomainDetailsQry (Section 6.1.2) request to specify name server details to match against the details of domains in the SRS.

If multiple ServerFilter elements are provided, the SRS should return domain results that match any of the provided filters (assuming that other filter restrictions are also met).

Syntax:

```
<!ELEMENT NameServerFilter (ServerFilter+)>
```

8.24. ParamValue

The ParamValue element is a simple element that contains parsed character data. The content of this element is the value of an SRS system parameter. It is used by the SysParam (Section 7.12) element.

Syntax:

```
<!ELEMENT ParamValue       (#PCDATA)>
```

8.25. PostalAddress

The PostalAddress element is an empty element with attributes for describing postal address details. SRS implementations SHOULD validate the CountryCode as a valid ISO 3166 [ISO.3166.1988] two-letter country code.

Syntax:

```
<!ELEMENT PostalAddress EMPTY>
<!ATTLIST PostalAddress
        Address1    CDATA #IMPLIED
        Address2    CDATA #IMPLIED
        City        CDATA #IMPLIED
        Province    CDATA #IMPLIED
        CountryCode CDATA #IMPLIED
        PostalCode  CDATA #IMPLIED>
```

8.26. PostalAddressFilter

The PostalAddressFilter element is an empty element with attributes for describing postal address details to match against details held by the SRS.

Syntax:

```
<!ELEMENT PostalAddressFilter EMPTY>
<!ATTLIST PostalAddressFilter
        Address1    CDATA #IMPLIED
        Address2    CDATA #IMPLIED
        City        CDATA #IMPLIED
        Province    CDATA #IMPLIED
        CountryCode CDATA #IMPLIED
        PostalCode  CDATA #IMPLIED>
```

8.27. Telephone Number Details

There are two telephone number details elements that share a common definition:

- Fax

- Phone

The telephone number details elements are empty elements that use the PhoneAttr (Section 9.16.1) entity for attribute declarations.

Syntax:

```
<!ELEMENT Fax EMPTY>
<!ATTLIST Fax %PhoneAttr;>

<!ELEMENT Phone EMPTY>
<!ATTLIST Phone %PhoneAttr;>
```

8.28. Role

The Role element is an empty element. It has a single attribute, RoleName, that must contain a value as defined by the Role (Section 9.14.1) entity.

Syntax:

```
<!ELEMENT Role EMPTY>
<!ATTLIST Role
        RoleName (%Role;) #REQUIRED>
```

8.29. Roles

The Roles element is a simple container for Role (Section 8.28) elements. It is used when creating and updating registrars and when the SRS returns details of registrars.

Syntax:

```
<!ELEMENT Roles (Role*)>
```

8.30. RunLogDetails

The RunLogDetails element is a simple element that contains parsed character data. The content of this element is the value of an SRS system parameter. It is used by the SysParam (Section 7.12) element and contains process data to be stored in the SRS log.

Syntax:

```
<!ELEMENT RunLogDetails (#PCDATA)>
```

8.31. SecondLD

The SecondLD element is an empty element. It has a single attribute, DomainName, that will contain a subdomain for which a given registrar has domain registration privileges. It is used by the Allowed2LDs (Section 8.3) element.

The SecondLD element MAY contain subdomains at any level in the domain name system hierarchy that the registry supports registrations for. It is not technically restricted to second level domains.

Syntax:

```
<!ELEMENT SecondLD EMPTY>
<!ATTLIST SecondLD
        DomainName %DomainName; #REQUIRED>
```

8.32. Server

The Server element is an empty element. It has three attributes:

FQDN: The fully qualified domain name of the server.

IP4Addr: The IPv4 address of the server.

IP6Addr: The IPv6 address of the server.

The fully qualified domain name is a required attribute. The IP address fields should only be included if the fully qualified domain name is within the domain that the server is a name server for.

The IP4Addr value should be given in standard dotted-decimal notation, for example, 192.0.2.14

The IP6Addr value should be given in any of the formats described in RFC 4291 [RFC4291], for example:

Fully specified (preferred form): 2001:DB8:0:0:0:0:C000:20E

Compressed form: 2001:DB8::C000:20E

Alternative form: 2001:DB8:0:0:0:0:192.0.2.14 (or 2001:DB8::192.0.2.14)

The Server element is used by the NameServers (Section 8.22) element to specify name server details.

Syntax:

```
<!ELEMENT Server EMPTY>
<!ATTLIST Server
        FQDN    CDATA #REQUIRED
        IP4Addr CDATA #IMPLIED
        IP6Addr CDATA #IMPLIED>
```

8.33. ServerFilter

The ServerFilter element is an empty element. It has the same attributes as the Server (Section 8.32) element, however all of the attributes are optional in the ServerFilter element. The content of the ServerFilter attributes is used to match against name server details held by the SRS.

The ServerFilter element is used by the NameServerFilter (Section 8.23) element to specify the server details to match.

Syntax:

```
<!ELEMENT ServerFilter EMPTY>
<!ATTLIST ServerFilter
        FQDN    CDATA #IMPLIED
        IP4Addr CDATA #IMPLIED
        IP6Addr CDATA #IMPLIED>
```

8.34. Signature

The Signature element is a simple element that contains parsed character data. The content of this element is the OpenPGP-compatible signature of a request message in ASCII armored format and encoded in UTF-8. It is used by the RawRequest and RawResponse (Section 7.8) elements when returning details of previous SRS transactions.

Syntax:

```
<!ELEMENT Signature (#PCDATA)>
```

8.35. Timestamp Elements

The SRS protocol employs many timestamp elements that all share a common definition. The timestamp elements are:

- ActiveOn
- BilledUntil
- BillPeriodEnd
- BillPeriodStart
- CancelledDate

o EffectiveDate

o FeTimeStamp

o FinalRunDate

o FirstRunDate

o From

o InvoiceDate

o LastRunDate

o LockedDate

o NewBilledUntilDate

o RegisteredDate

o RunDate

o RunLogTimeStamp

o To

o TransactionDate

o TransDate

The timestamp elements are empty elements that use the TimeStamp (Section 9.7.3) entity for attribute declarations.

Syntax:

```
<!ELEMENT ActiveOn            EMPTY>
<!ATTLIST ActiveOn            %TimeStamp;>

<!ELEMENT BilledUntil         EMPTY>
<!ATTLIST BilledUntil         %TimeStamp;>

<!ELEMENT BillPeriodEnd       EMPTY>
<!ATTLIST BillPeriodEnd       %TimeStamp;>

<!ELEMENT BillPeriodStart     EMPTY>
<!ATTLIST BillPeriodStart     %TimeStamp;>

<!ELEMENT CancelledDate       EMPTY>
```

```
   <!ATTLIST CancelledDate      %TimeStamp;>

   <!ELEMENT EffectiveDate      EMPTY>
   <!ATTLIST EffectiveDate      %TimeStamp;>

   <!ELEMENT FeTimestamp        EMPTY>
   <!ATTLIST FeTimestamp        %TimeStamp;>

   <!ELEMENT FinalRunDate       EMPTY>
   <!ATTLIST FinalRunDate       %TimeStamp;>

   <!ELEMENT FirstRunDate       EMPTY>
   <!ATTLIST FirstRunDate       %TimeStamp;>

   <!ELEMENT From               EMPTY>
   <!ATTLIST From               %TimeStamp;>

   <!ELEMENT InvoiceDate        EMPTY>
   <!ATTLIST InvoiceDate        %TimeStamp;>

   <!ELEMENT LastRunDate        EMPTY>
   <!ATTLIST LastRunDate        %TimeStamp;>

   <!ELEMENT LockedDate         EMPTY>
   <!ATTLIST LockedDate         %TimeStamp;>

   <!ELEMENT NewBilledUntilDate EMPTY>
   <!ATTLIST NewBilledUntilDate %TimeStamp;>

   <!ELEMENT RegisteredDate     EMPTY>
   <!ATTLIST RegisteredDate     %TimeStamp;>

   <!ELEMENT RunDate            EMPTY>
   <!ATTLIST RunDate            %TimeStamp;>

   <!ELEMENT RunLogTimeStamp    EMPTY>
   <!ATTLIST RunLogTimeStamp    %TimeStamp;>

   <!ELEMENT To                 EMPTY>
   <!ATTLIST To                 %TimeStamp;>

   <!ELEMENT TransactionDate    EMPTY>
   <!ATTLIST TransactionDate    %TimeStamp;>

   <!ELEMENT TransDate          EMPTY>
   <!ATTLIST TransDate          %TimeStamp;>
```

Example of an EffectiveDate element:

```
<EffectiveDate Year="2007" Month="11" Day="19" Hour="12"
    Minute="00" Second="00" TimeZoneOffset="+13:00" />
```

8.36. TransferredDomain

The TransferredDomain element is a simple element that contains parsed character data. The content of the element is the domain name of a domain that has been transferred from one registrar to another. It is used by the DomainTransfer (Section 7.6) element.

Syntax:

```
<!ELEMENT TransferredDomain (#PCDATA)>
```

8.37. XML

The XML element is a simple element that contains parsed character data. The content of this element is the XML content of a request or response message encoded in UTF-8. It is used by the RawRequest and RawResponse (Section 7.8) elements when returning details of previous SRS transactions.

Syntax:

```
<!ELEMENT XML (#PCDATA)>
```

9. Internal Entities

The XML DTD for the SRS communications protocol uses internal entities to provide common definitions in the DTD. These entities also enhance the readability of the DTD.

9.1. Actions

9.1.1. DomainWriteAction

The DomainWriteAction entity defines the valid values for attributes that hold the name of a domain write request.

Definition:

```
<!ENTITY % DomainWriteAction "DomainCreate|
                              DomainUpdate">
```

9.1.2. DomainQueryAction

The DomainQueryAction entity defines the valid values for attributes that hold the name of a domain query request.

Definition:

```
<!ENTITY % DomainQueryAction "Whois|
                               DomainDetailsQry|
                               ActionDetailsQry|
                               UDAIValidQry">
```

9.1.3. RegistrarWriteAction

The RegistrarWriteAction entity defines the valid values for attributes that hold the name of a registrar write request.

Definition:

```
<!ENTITY % RegistrarWriteAction "RegistrarCreate|
                                  RegistrarUpdate">
```

9.1.4. RegistrarQueryAction

The RegistrarQueryAction entity defines the valid values for attributes that hold the name of a registrar query request.

Definition:

```
<!ENTITY % RegistrarQueryAction "RegistrarDetailsQry|
                                  RegistrarAccountQry|
                                  GetMessages">
```

9.1.5. RegistryAction

The RegistryAction entity defines the valid values for attributes that hold the name of a registry request.

Definition:

```
<!ENTITY % RegistryAction "SysParamsUpdate|
                           SysParamsQry|
                           RunLogCreate|
                           RunLogQry|
                           ScheduleCreate|
                           ScheduleCancel|
                           ScheduleQry|
                           ScheduleUpdate|
                           BillingExtract|
                           SetBillingAmount|
                           QryBillingAmount|
                           DeferredIncomeSummaryQry|
                           DeferredIncomeDetailQry|
                           BilledUntilAdjustment|
                           BuildDnsZoneFiles|
                           GenerateDomainReport|
                           AdjustRegistrarAccount|
                           AccessControlListQry|
                           AccessControlListAdd|
                           AccessControlListRemove">
```

9.1.6. Action

The Action entity defines the valid values for attributes that hold the name of an SRS request. It is defined in terms of the various action entity definitions.

Definition:

```
<!ENTITY % Action "%DomainWriteAction;|
                   %DomainQueryAction;|
                   %RegistrarWriteAction;|
                   %RegistrarQueryAction;|
                   %RegistryAction;"
```

9.2. Accounts

9.2.1. AccountingAction

The AccountingAction entity defines the valid values for attributes that hold the name of an accounting action type.

Definition:

```
<!ENTITY % AccountingAction "Credit|
                             Debit>
```

9.2.2. BillStatus

The BillStatus entity defines the valid values for attributes that hold the name of a billing status.

Definition:

```
<!ENTITY % BillStatus "PendingConfirmation|
                       Confirmed">
```

9.3. Booleans

9.3.1. True

The True entity defines a fixed integer value for true boolean values in the SRS protocol.

Definition:

```
<!ENTITY % True "1">
```

9.3.2. False

The False entity defines a fixed integer value for false boolean values in the SRS protocol.

Definition:

```
<!ENTITY % False "0">
```

9.3.3. Boolean

The Boolean entity defines the valid values for attributes that hold a boolean value.

Definition:

```
<!ENTITY % Boolean "(%False;|%True;)">
```

9.4. Contact Details

9.4.1. Contact

The Contact entity defines a content model for elements that contain contact details.

Definition:

```
<!ENTITY % Contact "PostalAddress?,Phone?,Fax?">
```

9.4.2. ContactAttr

The ContactAttr entity defines the attribute specification for elements that contain contact details.

Definition:

```
<!ENTITY % ContactAttr "Name  CDATA #IMPLIED
                        Email CDATA #IMPLIED">
```

9.5. Contact Details Filters

9.5.1. ContactFilter

The ContactFilter entity defines a content model for elements that contain contact filter details.

Definition:

```
<!ENTITY % ContactFilter "PostalAddressFilter?,Phone?,Fax?">
```

9.5.2. ContactFilterAttr

The ContactFilterAttr entity defines the attribute specification for elements that contain contact filter details.

Definition:

```
<!ENTITY % ContactAttrFilter "Name  CDATA #IMPLIED
                              Email CDATA #IMPLIED">
```

9.6. Currency

9.6.1. Dollars

The Dollars entity defines a value definition for attributes that contain currency values. SRS implementations SHOULD ensure that values given and received in attributes of this type conform to valid values in their registry’s working currency.

Definition:

```
<!ENTITY % Dollars "CDATA">
```

9.7. Dates And Times

9.7.1. Date

The Date entity defines the attribute specification for elements that contain date details.

Definition:

```
<!ENTITY % Date "Year  %Number; #REQUIRED
                 Month %Number; #REQUIRED
                 Day   %Number; #REQUIRED">
```

9.7.2. Time

The Time entity defines the attribute specification for elements that contain time details. SRS implementations MAY use their local timezone offset from UTC as a default value for TimeZoneOffset if this field is not populated.

Definition:

```
<!ENTITY % Time "Hour           %Number; #REQUIRED
                 Minute         %Number; #REQUIRED
                 Second         %Number; #IMPLIED
                 TimeZoneOffset CDATA    #IMPLIED">
```

9.7.3. TimeStamp

The TimeStamp entity defines a composite attribute specification for elements that contain a combined date and time value. It is defined in terms of the Date (Section 9.7.1) and Time (Section 9.7.2) entities.

Definition:

```
<!ENTITY % TimeStamp "%Date; %Time;">
```

9.7.4. DateRange

The DateRange entity defines a content model for elements that contain a start date and time and an end date and time. Such elements consist of an optional From (Section 8.35) element and an optional To (Section 8.35) element. Both the From and To element are empty elements with attributes as defined in the TimeStamp (Section 9.7.3) entity.

Definition:

```
<!ENTITY % DateRange "From?,To?">
```

9.8. Domain Names

9.8.1. DomainName

The DomainName entity defines the valid value definition for attributes that hold a domain name.

Definition:

```
<!ENTITY % DomainName "CDATA">
```

9.9. Domain Status

9.9.1. RegDomainStatus

The RegDomainStatus entity defines the valid values for attributes that hold the status of registered domains.

Definition:

```
<!ENTITY % RegDomainStatus "Active|
                            PendingRelease">
```

9.9.2. DomainStatus

The DomainStatus entity defines the valid values for attributes that hold the status of both registered and unregistered domains.

Definition:

```
<!ENTITY % DomainStatus "%RegDomainStatus;|
                         Available">
```

9.10. Duration

9.10.1. Term

The Term entity defines the valid value definition for attributes that hold a timespan. Generally this will be an integer number of months representing the registry’s billing cycle for registered domains.

Definition:

```
<!ENTITY % Term "%Number;">
```

9.11. Numeric

9.11.1. Number

The Number entity defines the valid value definition for attributes that hold a number. SRS implementations SHOULD ensure that data provided in such attributes is numeric.

Definition:

```
<!ENTITY % Number "CDATA">
```

9.12. Registrar Identifiers

9.12.1. RegistrarId

The RegistrarId entity defines the valid value definition for attributes that hold a registrar identifier. Registrar identifiers in the SRS are numeric and this entity is defined in terms of the Number (Section 9.11.1) entity.

Registrar identifiers are assigned to registrars for performing registrar functions and to the registry for performing registry functions.

The registry may have more than one identity, with each identity having separate roles and permissions in the system. The registry's registrar identifiers should be known to registrars so that registrars can identify actions performed by the registry.

Definition:

```
<!ENTITY % RegistrarId "%Number;">
```

9.12.2. RegistrarIdOrOTHERS

The RegistrarIdOrOTHERS entity defines the valid value definition for attributes that hold a registrar identifier or the special value "OTHERS". SRS Implementations SHOULD ensure that data provided in such attributes is either a valid registrar identifier or the string "OTHERS". The value "OTHERS" may be used instead of a registrar identifier in some cases, for example when using the GetMessages (Section 6.3.1) request, to specify a registrar identifier other than the registrar making the request.

Definition:

```
<!ENTITY % RegistrarIdOrOTHERS "CDATA">
```

9.13. Responses

9.13.1. ActionResponse

The ActionResponse entity defines the valid values for attributes that hold the name of an action response type.

Definition:

```
<!ENTITY % ActionResponse "Error|
                           Domain*|
                           UDAIValid|
                           DomainTransfer|
                           BillingTrans*|
                           DeferredRegistrarIncome*|
                           Registrar*|
                           SysParam*|
                           RunLog*|
                           Schedule*|
                           AccessControlList*|
                           (RawRequest,RawResponse)|
                           BillingAmount*">
```

9.14. System Roles

9.14.1. Role

The Role entity defines the valid values for attributes that hold the name of a system role type. Roles may be used by implementations to restrict the requests that users (typically registrars) have access to.

Definition:

```
<!ENTITY % Role "Registrar|
                 Registry|
                 Whois|
                 Query|
                 CreateDomain|
                 UpdateDomain|
                 TransferDomain|
                 CancelDomain|
                 UncancelDomain|
                 UpdateRegistrar|
                 Administer|
                 Supervisor|
                 Connect|
                 ReleaseDomain|
                 QueryACL|
                 UpdateACL">
```

9.15. Scheduled Processes

9.15.1. ScheduledJob

The ScheduledJob entity defines the valid values for attributes that hold the name of a scheduled job type.

Definition:

```
<!ENTITY % ScheduledJob "BuildDnsZoneFiles|
                         ReleaseDomains|
                         RenewDomains|
                         GenerateDomainReport|
                         GenerateStatsReport"
```

9.16. Telephone Numbers

9.16.1. PhoneAttr

The PhoneAttr entity defines the attribute specification for elements that telephone or fax number details.

Definition:

```
<!ENTITY % PhoneAttr "CountryCode %NumberFilter; #IMPLIED
                      AreaCode    %NumberFilter; #IMPLIED
                      LocalNumber %NumberFilter; #IMPLIED">
```

9.17. Unique Identifiers

9.17.1. UID

The UID entity defines a value definition for attributes that contain unique identifier values.

Definition:

```
<!ENTITY % UID "CDATA">
```

10. Text Filter

The text filter provides support for case-insensitive matching of a character string against information held in the SRS. Two wildcard characters are supported:

Wildcard	Description
?	matches any single character
*	matches zero or more characters

Note: when a text filter pattern is used for matching against domain names, the wildcard characters will not match the dot-separator character. The only exception to this is when the * wildcard is used at the start of the text filter pattern, in this case the wildcard may match any characters, as shown in the examples below.

Text filter pattern matching examples for domain names:

Text filter pattern	Domain	Result
alpha.*.org	alpha.example.org	Match
alpha.*.org	alpha.beta.example.org	No match
*.example.org	alpha.example.org	Match
*.example.org	alpha.beta.example.org	Match
*.example.org	example.org	No match
alpha.*	alpha.example	Match
alpha.*	alpha.co.example	No match
*	any.example.org	Match (* on its own will match all domains in the SRS)

11. Security Considerations

Care should be taken to ensure that no private data is sent over any unsecured transport. HTTPS MUST be used for all transactions involving private data - that is, data that cannot be retrieved using the public WHOIS system.

The exchange of public keys between registry and registrar when a new registrar is created in the SRS SHOULD be handled in a secure way with careful identity verification by both parties.

Storage of private keys by the registry and the registrar must be handled carefully to avoid compromise. Keys should be changed on an occasional basis, according to current best practice, to ensure that the SRS remains secure. The registrar may provide the registry with more than one valid public key to support migration from one key to another and the expiration of old keys.

Registrars MUST be restricted to only viewing data that is publicly available (that is, data that can be retrieved using the public WHOIS system), and data that they manage. Registrars MUST NOT be able to access through the SRS registry information that is managed by other registrars and is not available through the public WHOIS system.

The registry SHOULD access all requests and data through the SRS system. This allows the implementation to ensure that there is only a single way to access the system functions and data.

12. IANA Considerations

This document has no actions for IANA.

13. References

13.1. Normative References

[RFC1035] Mockapetris, P., "Domain names - implementation and specification", STD 13, RFC 1035, November 1987.

[RFC1123] Braden, R., "Requirements for Internet Hosts - Application and Support", STD 3, RFC 1123, October 1989.

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

[RFC2181] Elz, R. and R. Bush, "Clarifications to the DNS

Specification", RFC 2181, July 1997.

[RFC4880] Callas, J., Donnerhacke, L., Finney, H., Shaw, D., and R. Thayer, "OpenPGP Message Format", RFC 4880, November 2007.

[RFC2616] Fielding, R., Gettys, J., Mogul, J., Frystyk, H., Masinter, L., Leach, P., and T. Berners-Lee, "Hypertext Transfer Protocol -- HTTP/1.1", RFC 2616, June 1999.

[RFC2818] Rescorla, E., "HTTP Over TLS", RFC 2818, May 2000.

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

[W3C.REC-xml] Bray, T., Paoli, J., Sperberg-McQueen, C., and E. Maler, "Extensible Markup Language (XML) 1.0 (2nd ed)", W3C REC-xml, October 2000, <http://www.w3.org/TR/REC-xml>.

[ISO.3166.1988] International Organization for Standardization, "Codes for the representation of names of countries, 3rd edition", ISO Standard 3166, August 1988.

13.2. Informative References

[RFC3912] Daigle, L., "WHOIS Protocol Specification", RFC 3912, September 2004.

[RFC4291] Hinden, R. and S. Deering, "IP Version 6 Addressing Architecture", RFC 4291, February 2006.

[RFC3375] Hollenbeck, S., "Generic Registry-Registrar Protocol Requirements", RFC 3375, September 2002.

[US-ASCII] American National Standards Institute, "Coded Character Set - 7-bit American Standard Code for Information Interchange", ANSI X3.4, 1986, 1968.

URIs

[1] <http://sourceforge.net/projects/dnrs/>

Appendix A. Example

This example shows a typical request and response document. The example does not show the full format of the multipart request body that must be sent to the server for a request.

A.1. Whois request

Example of a Whois request (formatted for readability):

```
C: n=50041&r=<!DOCTYPE NZSRSRequest SYSTEM "protocol.dtd">
C: <NZSRSRequest VerMajor="2" VerMinor="0" RegistrarId="500411">
C:    <Whois FullResult="0" DomainName="example.org"/>
C: </NZSRSRequest>
C: &s=-----BEGIN PGP SIGNATURE-----
C: Version: Crypt::OpenPGP 1.03
C:
C: iQBGBAARAgAGBQJHtLqmAAoJECvUc6XZCUibvs4An2qp5xHugp7tOjO4pmxTqb3k
C: N63OAJ44F+/O3bsRhIGoqrCjvg1hlFAK0A==
C: =5C/k
C: -----END PGP SIGNATURE-----

S: r=<?xml version="1.0"?>
S: <!DOCTYPE NZSRSResponse SYSTEM "protocol.dtd">
S: <NZSRSResponse VerMajor="2" VerMinor="0" RegistrarId="1">
S:    <Response Action="Whois" FeId="7" FeSeq="1996051"
S:        OrigRegistrarId="1" RecipientRegistrarId="1">
S:       <FeTimeStamp Day="15" Hour="11" Minute="03" Month="2"
S:             Second="18" TimeZoneOffset=" 13:00" Year="2008"/>
S:       <Domain DomainName="example.org" Status="Active"/>
S:    </Response>
S: </NZSRSResponse>
S: &s=-----BEGIN PGP SIGNATURE-----
S: Version: Crypt::OpenPGP 1.03
S:
S: iQBGBAARAgAGBQJHtLqmAAoJENi/K6P6QHemSbgAn2W3SQFFZzI1GKbGbiJZtq4w
S: k7SxAJ491nPIkU/8kJvJ+No+Ysiph19Whw==
S: =r5Vn
S: -----END PGP SIGNATURE-----
```

Appendix B. Acknowledgements

The author would like to thank the following groups and individuals for reviewing this document in draft format and providing very helpful comments and advice.

- The SRS Implementation Team at Catalyst IT Limited - Steven Craig, Evan Giles, John Praill, Andrew Ruthven, and Sam Vilain.

   o Dave Baker of .nz Registry Services

Appendix C. DTD

   Complete DTD for the SRS protocol:

```
   <!-- DTD for NZ Shared Registry System Protocol -->

   <!-- Entity declarations -->

   <!ENTITY % Number "CDATA">

   <!ENTITY % NumberFilter "CDATA">

   <!ENTITY % Dollars "CDATA">

   <!ENTITY % True "1">

   <!ENTITY % False "0">

   <!ENTITY % Boolean "(%False;|%True;)">

   <!ENTITY % Date "Year  %Number; #REQUIRED
           Month %Number; #REQUIRED
           Day   %Number; #REQUIRED">

   <!ENTITY % Time "Hour           %Number; #REQUIRED
                    Minute         %Number; #REQUIRED
                    Second         %Number; #IMPLIED
                    TimeZoneOffset CDATA    #IMPLIED">

   <!ENTITY % TimeStamp "%Date; %Time;">

   <!ENTITY % DomainName "CDATA">

   <!ENTITY % UID "CDATA">

   <!ENTITY % Term "%Number;">

   <!ENTITY % RegistrarId "%Number;">

   <!ENTITY % RegistrarIdOrOTHERS "CDATA">

   <!ENTITY % DomainWriteAction "DomainCreate|DomainUpdate">

   <!ENTITY % DomainQueryAction "Whois|DomainDetailsQry|
```

```
                             ActionDetailsQry|UDAIValidQry">

   <!ENTITY % RegistrarWriteAction "RegistrarCreate|RegistrarUpdate">

   <!ENTITY % RegistrarQueryAction "RegistrarDetailsQry|
                                    RegistrarAccountQry|GetMessages">

   <!ENTITY % RegistryAction "SysParamsUpdate|SysParamsQry|
                              RunLogCreate|RunLogQry|
                              ScheduleCreate|ScheduleCancel|
                              ScheduleQry|ScheduleUpdate|
                              BillingExtract|SetBillingAmount|
                              QryBillingAmount|
                              DeferredIncomeSummaryQry|
                              DeferredIncomeDetailQry|
                              BilledUntilAdjustment|
                              BuildDnsZoneFiles|
                              GenerateDomainReport|
                              AdjustRegistrarAccount|
                              AccessControlListQry|
                              AccessControlListAdd|
                              AccessControlListRemove">

   <!ENTITY % Action "%DomainWriteAction;|%DomainQueryAction;|
           %RegistrarWriteAction;|%RegistrarQueryAction;|
           %RegistryAction;">

   <!ENTITY % ActionResponse "Error|Domain*|UDAIValid|DomainTransfer|
                              BillingTrans*|DeferredRegistrarIncome*|
                              Registrar*|SysParam*|RunLog*|Schedule*|
                              AccessControlList*|
                              (RawRequest,RawResponse)|
                              BillingAmount*">

   <!ENTITY % RegDomainStatus "Active|PendingRelease">

   <!ENTITY % DomainStatus "%RegDomainStatus;|Available">

   <!ENTITY % BillStatus "PendingConfirmation|Confirmed">

   <!ENTITY % ScheduledJob "BuildDnsZoneFiles|ReleaseDomains|
                            RenewDomains|GenerateDomainReport|
                            GenerateStatsReport">

   <!ENTITY % Contact "PostalAddress?,Phone?,Fax?">

   <!ENTITY % ContactAttr "Name  CDATA #IMPLIED
                           Email CDATA #IMPLIED">
```

```
   <!ENTITY % ContactFilter "PostalAddressFilter?,Phone?,Fax?">

   <!ENTITY % ContactAttrFilter "Name  CDATA #IMPLIED
                                 Email CDATA #IMPLIED">

   <!ENTITY % PhoneAttr "CountryCode %NumberFilter; #IMPLIED
                         AreaCode    %NumberFilter; #IMPLIED
                         LocalNumber %NumberFilter; #IMPLIED">

   <!ENTITY % Role "Registrar|Registry|Whois|Query|CreateDomain|
                    UpdateDomain|TransferDomain|CancelDomain|
                    UncancelDomain|UpdateRegistrar|Administer|
                    Supervisor|Connect|ReleaseDomain|QueryACL|
                    UpdateACL">

   <!ENTITY % DateRange "From?,To?">

   <!ENTITY % AccountingAction "Credit|Debit">

   <!-- DATA ELEMENTS -->

   <!-- Parsed character data elements -->

   <!ELEMENT ActionIdFilter (#PCDATA)>

   <!ELEMENT AuditText (#PCDATA)>

   <!ELEMENT AuditTextFilter (#PCDATA)>

   <!ELEMENT CancelAuditText (#PCDATA)>

   <!ELEMENT CreateAuditText (#PCDATA)>

   <!ELEMENT Description (#PCDATA)>

   <!ELEMENT DomainNameFilter (#PCDATA)>

   <!ELEMENT EncryptKey (#PCDATA)>

   <!ELEMENT ErrorDetails (#PCDATA)>

   <!ELEMENT ParamValue (#PCDATA)>

   <!ELEMENT RunLogDetails (#PCDATA)>

   <!ELEMENT Signature (#PCDATA)>

   <!ELEMENT TransferredDomain (#PCDATA)>
```

```
   <!ELEMENT XML (#PCDATA)>

   <!-- Contact information: for registrant, admin, technical  -->
   <!ELEMENT RegistrantContact       (%Contact;)>
   <!ATTLIST RegistrantContact       %ContactAttr;>

   <!ELEMENT RegistrantContactFilter (%ContactFilter;)>
   <!ATTLIST RegistrantContactFilter %ContactAttrFilter;>

   <!ELEMENT AdminContact            (%Contact;)>
   <!ATTLIST AdminContact            %ContactAttr;>

   <!ELEMENT AdminContactFilter      (%ContactFilter;)>
   <!ATTLIST AdminContactFilter      %ContactAttrFilter;>

   <!ELEMENT TechnicalContact        (%Contact;)>
   <!ATTLIST TechnicalContact        %ContactAttr;>

   <!ELEMENT TechnicalContactFilter  (%ContactFilter;)>
   <!ATTLIST TechnicalContactFilter  %ContactAttrFilter;>

   <!ELEMENT RegistrarPublicContact  (%Contact;)>
   <!ATTLIST RegistrarPublicContact  %ContactAttr;>

   <!ELEMENT DefaultTechnicalContact (%Contact;)>
   <!ATTLIST DefaultTechnicalContact %ContactAttr;>

   <!ELEMENT RegistrarSRSContact     (%Contact;)>
   <!ATTLIST RegistrarSRSContact     %ContactAttr;>

   <!-- Postal address -->
   <!ELEMENT PostalAddress EMPTY>
   <!ATTLIST PostalAddress
           Address1    CDATA #IMPLIED
           Address2    CDATA #IMPLIED
           City        CDATA #IMPLIED
           Province    CDATA #IMPLIED
           CountryCode CDATA #IMPLIED
           PostalCode  CDATA #IMPLIED>

   <!-- Postal address filter -->
   <!ELEMENT PostalAddressFilter EMPTY>
   <!ATTLIST PostalAddressFilter
           Address1    CDATA #IMPLIED
           Address2    CDATA #IMPLIED
           City        CDATA #IMPLIED
           Province    CDATA #IMPLIED
           CountryCode CDATA #IMPLIED
```

```
          PostalCode  CDATA #IMPLIED>

   <!-- Telephone numbers -->
   <!ELEMENT Phone            EMPTY>
   <!ATTLIST Phone            %PhoneAttr;>

   <!ELEMENT Fax              EMPTY>
   <!ATTLIST Fax              %PhoneAttr;>

   <!-- Time stamps (date and time) -->
   <!ELEMENT ActiveOn EMPTY>
   <!ATTLIST ActiveOn %TimeStamp;>

   <!ELEMENT BillPeriodEnd EMPTY>
   <!ATTLIST BillPeriodEnd %TimeStamp;>

   <!ELEMENT BillPeriodStart EMPTY>
   <!ATTLIST BillPeriodStart %TimeStamp;>

   <!ELEMENT BilledUntil EMPTY>
   <!ATTLIST BilledUntil %TimeStamp;>

   <!ELEMENT EffectiveDate EMPTY>
   <!ATTLIST EffectiveDate %TimeStamp;>

   <!ELEMENT FeTimeStamp EMPTY>
   <!ATTLIST FeTimeStamp %TimeStamp;>

   <!ELEMENT FinalRunDate EMPTY>
   <!ATTLIST FinalRunDate %TimeStamp;>

   <!ELEMENT FirstRunDate EMPTY>
   <!ATTLIST FirstRunDate %TimeStamp;>

   <!ELEMENT From EMPTY>
   <!ATTLIST From %TimeStamp;>

   <!ELEMENT InvoiceDate EMPTY>
   <!ATTLIST InvoiceDate %TimeStamp;>

   <!ELEMENT LastRunDate EMPTY>
   <!ATTLIST LastRunDate %TimeStamp;>

   <!ELEMENT NewBilledUntilDate EMPTY>
   <!ATTLIST NewBilledUntilDate %TimeStamp;>

   <!ELEMENT RunDate EMPTY>
   <!ATTLIST RunDate %TimeStamp;>
```

```
   <!ELEMENT RunLogTimeStamp EMPTY>
   <!ATTLIST RunLogTimeStamp %TimeStamp;>

   <!ELEMENT LockedDate EMPTY>
   <!ATTLIST LockedDate %TimeStamp;>

   <!ELEMENT RegisteredDate EMPTY>
   <!ATTLIST RegisteredDate %TimeStamp;>

   <!ELEMENT CancelledDate  EMPTY>
   <!ATTLIST CancelledDate  %TimeStamp;>

   <!ELEMENT To EMPTY>
   <!ATTLIST To %TimeStamp;>

   <!ELEMENT TransDate EMPTY>
   <!ATTLIST TransDate %TimeStamp;>

   <!ELEMENT TransactionDate EMPTY>
   <!ATTLIST TransactionDate %TimeStamp;>

   <!-- Date ranges  -->
   <!ELEMENT AuditTime (%DateRange;)>

   <!ELEMENT BilledUntilDateRange (%DateRange;)>

   <!ELEMENT CancelledDateRange (%DateRange;)>

   <!ELEMENT ChangedInDateRange (%DateRange;)>

   <!ELEMENT LockedDateRange (%DateRange;)>

   <!ELEMENT LogDateRange (%DateRange;)>

   <!ELEMENT InvoiceDateRange (%DateRange;)>

   <!ELEMENT RegisteredDateRange (%DateRange;)>

   <!ELEMENT ResultDateRange (%DateRange;)>

   <!ELEMENT SearchDateRange (%DateRange;)>

   <!ELEMENT TransDateRange (%DateRange;)>

   <!-- Base month for deferred income calculations -->
   <!ELEMENT BaseMonth EMPTY>
   <!ATTLIST BaseMonth
           Month %Number; #REQUIRED>
```

```
   <!-- Base year for deferred income calculations -->
   <!ELEMENT BaseYear EMPTY>
   <!ATTLIST BaseYear
           Year %Number; #REQUIRED>

   <!-- Income month for deferred income calculations -->
   <!ELEMENT IncomeMonth EMPTY>
   <!ATTLIST IncomeMonth
           Month %Number; #REQUIRED>

   <!-- Income year for deferred income calculations -->
   <!ELEMENT IncomeYear EMPTY>
   <!ATTLIST IncomeYear
           Year %Number; #REQUIRED>

   <!-- Name server container -->
   <!ELEMENT NameServers (Server*)>

   <!-- Server -->
   <!ELEMENT Server EMPTY>
   <!ATTLIST Server
           FQDN    CDATA #REQUIRED
           IP4Addr CDATA #IMPLIED
           IP6Addr CDATA #IMPLIED>

   <!-- Name server filter -->
   <!ELEMENT NameServerFilter    (ServerFilter+)>

   <!-- Server filter -->
   <!ELEMENT ServerFilter EMPTY>
   <!ATTLIST ServerFilter
           FQDN    CDATA #IMPLIED
           IP4Addr CDATA #IMPLIED
           IP6Addr CDATA #IMPLIED>

   <!-- Domain registration permission list -->
   <!ELEMENT Allowed2LDs (SecondLD*)>
   <!ELEMENT SecondLD EMPTY>
   <!ATTLIST SecondLD
           DomainName %DomainName; #REQUIRED>

   <!-- Registrar/user role definition -->
   <!ELEMENT Roles (Role*)>
   <!ELEMENT Role EMPTY>
   <!ATTLIST Role
           RoleName (%Role;) #REQUIRED>

   <!-- Encryption keys -->
```

```
   <!ELEMENT EncryptKeys (EncryptKey*)>

   <!-- Audit details element -->
   <!ELEMENT AuditDetails (AuditTime?,AuditText?)>
   <!ATTLIST AuditDetails
           RegistrarId CDATA #IMPLIED
           ActionId    %UID; #IMPLIED>

   <!-- Access control list element -->
   <!ELEMENT AccessControlList (AccessControlListEntry*)>
   <!ATTLIST AccessControlList
           Resource CDATA #REQUIRED
           Type     CDATA #REQUIRED>

   <!-- Access control list entry element -->
   <!ELEMENT AccessControlListEntry (EffectiveDate?)>
   <!ATTLIST AccessControlListEntry
           Address               CDATA                 #REQUIRED
           Comment               CDATA                 #IMPLIED>

   <!-- Query field list -->
   <!ELEMENT FieldList EMPTY>
   <!ATTLIST FieldList
           Status               %Boolean; #IMPLIED
           NameServers          %Boolean; #IMPLIED
           RegistrantContact    %Boolean; #IMPLIED
           RegisteredDate       %Boolean; #IMPLIED
           AdminContact         %Boolean; #IMPLIED
           TechnicalContact     %Boolean; #IMPLIED
           LockedDate           %Boolean; #IMPLIED
           Delegate             %Boolean; #IMPLIED
           RegistrarId          %Boolean; #IMPLIED
           RegistrarName        %Boolean; #IMPLIED
           RegistrantRef        %Boolean; #IMPLIED
           LastActionId         %Boolean; #IMPLIED
           ChangedByRegistrarId %Boolean; #IMPLIED
           Term                 %Boolean; #IMPLIED
           BilledUntil          %Boolean; #IMPLIED
           CancelledDate        %Boolean; #IMPLIED
           AuditText            %Boolean; #IMPLIED
           EffectiveFrom        %Boolean; #IMPLIED>

   <!-- REQUEST AND RESPONSE ROOT ELEMENTS -->

   <!-- Root request element -->
   <!ELEMENT NZSRSRequest ((%Action;)+)>
   <!ATTLIST NZSRSRequest
           VerMajor    (1|2)         #REQUIRED
```

```
           VerMinor    %Number;      #REQUIRED
           RegistrarId %RegistrarId; #IMPLIED>

   <!-- Root response element -->
   <!ELEMENT NZSRSResponse (Response+|Error)>
   <!ATTLIST NZSRSResponse
           VerMajor    (2)           #REQUIRED
           VerMinor    %Number;      #REQUIRED
           RegistrarId %RegistrarId; #IMPLIED>

   <!-- SUCCESS RESPONSE CONTAINER -->
   <!-- Response -->
   <!ELEMENT Response (FeTimeStamp, (Response*|%ActionResponse;)?)>
   <!ATTLIST Response
           Action               (%Action;|UnknownTransaction|
                                 DomainTransfer)     #REQUIRED
           FeId                 %Number;             #REQUIRED
           FeSeq                %Number;             #REQUIRED
           OrigRegistrarId      %RegistrarId;        #REQUIRED
           TransId              %UID;                #IMPLIED
           Rows                 %Number;             #IMPLIED
           Count                %Number;             #IMPLIED
           MoreRowsAvailable    %Boolean;            #IMPLIED
           RecipientRegistrarId %RegistrarId;        #IMPLIED>

   <!-- REQUEST ELEMENTS -->

   <!-- Query previous request and response details
        Response: ((RawRequest,RawResponse)|Error) -->
   <!ELEMENT ActionDetailsQry EMPTY>
   <!ATTLIST ActionDetailsQry
           QryId                  %UID; #IMPLIED
           ActionId               %UID; #REQUIRED
           OriginatingRegistrarId %UID; #IMPLIED>

   <!-- Query domain details
        Response: (Domain*|Error) -->
   <!ELEMENT DomainDetailsQry (DomainNameFilter*,NameServerFilter?,
           RegistrantContactFilter?,
           AdminContactFilter?,TechnicalContactFilter?,
           ResultDateRange?,SearchDateRange?,
           ChangedInDateRange?,
           RegisteredDateRange?,LockedDateRange?,
           CancelledDateRange?,BilledUntilDateRange?,
           AuditTextFilter?,ActionIdFilter?,FieldList?)>
   <!ATTLIST DomainDetailsQry
           QryId        %UID;               #IMPLIED
           Status       (%RegDomainStatus;) #IMPLIED
```

```
          Delegate      %Boolean;           #IMPLIED
          Term          %Term;              #IMPLIED
          RegistrantRef %UID;               #IMPLIED
          MaxResults    %Number;            #IMPLIED
          SkipResults   %Number;            #IMPLIED
          CountResults  %Boolean;           "0">

   <!-- Query if the given UDAI matches the UDAI for a domain
        Response: (UDAIValid) -->
   <!ELEMENT UDAIValidQry EMPTY>
   <!ATTLIST UDAIValidQry
          QryId      %UID;         #IMPLIED
          DomainName %DomainName;  #REQUIRED
          UDAI       %UID;         #REQUIRED>

   <!-- Retrieve public details for a domain (WHOIS)
        Response: (Domain|Error) -->
   <!ELEMENT Whois EMPTY>
   <!ATTLIST Whois
          QryId      %UID;         #IMPLIED
          FullResult %Boolean;     "1"
          SourceIP   CDATA         #IMPLIED
          DomainName %DomainName;  #REQUIRED>

   <!-- Create new domain record
        Response: (Domain|Error) -->
   <!ELEMENT DomainCreate (RegistrantContact,
          AdminContact?,
          TechnicalContact?,
          NameServers?,
          AuditText?)>
   <!ATTLIST DomainCreate
          ActionId      %UID;         #REQUIRED
          DomainName    %DomainName;  #REQUIRED
          RegistrantRef %UID;         #IMPLIED
          Term          %Term;        #REQUIRED
          Delegate      %Boolean;     "1">

   <!-- Update domain record(s)
        Response: (Domain+|Error) -->
   <!ELEMENT DomainUpdate (DomainNameFilter+,RegistrantContact?,
                          AdminContact?,TechnicalContact?,
                          NameServers?,AuditText?)>
   <!ATTLIST DomainUpdate
          ActionId      %UID;     #REQUIRED
          UDAI          %UID;     #IMPLIED
          NewUDAI       %Boolean; #IMPLIED
          RegistrantRef %UID;     #IMPLIED
```

```
          Term           %Term;    #IMPLIED
          Delegate       %Boolean; #IMPLIED
          Renew          %Boolean; #IMPLIED
          NoAutoRenew    %Boolean; #IMPLIED
          Lock           %Boolean; #IMPLIED
          Cancel         %Boolean; #IMPLIED
          Release        %Boolean; #IMPLIED
          FullResult     %Boolean; "1">

   <!-- Query access control list entries
        Response: (AccessControlList*|Error) -->
   <!ELEMENT AccessControlListQry EMPTY>
   <!ATTLIST AccessControlListQry
          Resource   CDATA     #IMPLIED
          Type       CDATA     #IMPLIED
          FullResult %Boolean; "0">

   <!-- Remove access control list entries
        Response: (AccessControlList+|Error) -->
   <!ELEMENT AccessControlListRemove (AccessControlListEntry+,
                                      AuditText)>
   <!ATTLIST AccessControlListRemove
          Resource   CDATA     #REQUIRED
          Type       CDATA     #REQUIRED
          ActionId   %UID;     #REQUIRED
          FullResult %Boolean; "1">

   <!-- Add access control list entries
        Response: (AccessControlList*|Error) -->
   <!ELEMENT AccessControlListAdd (AccessControlListEntry+,
                                   AuditText)>
   <!ATTLIST AccessControlListAdd
          Resource   CDATA     #REQUIRED
          Type       CDATA     #REQUIRED
          ActionId   %UID;     #REQUIRED
          FullResult %Boolean; "1">

   <!-- Get SRS generated messages for a registrar
        Response: (Response|Error) -->
   <!ELEMENT GetMessages (TransDateRange?,AuditTextFilter?)>
   <!ATTLIST GetMessages
          QryId                  %UID;                  #IMPLIED
          OriginatingRegistrarId %RegistrarIdOrOTHERS; #IMPLIED
          ActionId               %UID;                  #IMPLIED
          RecipientRegistrarId   %RegistrarId;          #IMPLIED
          MaxResults             %Number;               #IMPLIED
          SkipResults            %Number;               #IMPLIED
          CountResults           %Boolean;              "0">
```

```
<!-- Query registrar details
     Response: (Registrar*|Error) -->
<!ELEMENT RegistrarDetailsQry (ResultDateRange?)>
<!ATTLIST RegistrarDetailsQry
        QryId       %UID;          #IMPLIED
        RegistrarId %RegistrarId; #IMPLIED
        NameFilter  CDATA          #IMPLIED>

<!-- Query registrar account (obtain billing records)
     Response: (BillingTrans*|Error) -->
<!ELEMENT RegistrarAccountQry (TransDateRange?,InvoiceDateRange?)>
<!ATTLIST RegistrarAccountQry
        QryId         %UID;           #IMPLIED
        RegistrantRef %UID;           #IMPLIED
        DomainName    %DomainName;    #IMPLIED
        InvoiceId     %UID;           #IMPLIED
        MaxResults    %Number;        #IMPLIED
        SkipResults   %Number;        #IMPLIED
        TransStatus   (%BillStatus;) #IMPLIED
        CountResults  %Boolean;       "0">

<!-- Insert details for a new registrar
     Response: (Registrar|Error) -->
<!ELEMENT RegistrarCreate (RegistrarPublicContact,
                           RegistrarSRSContact,
                           DefaultTechnicalContact,EncryptKeys,
                           Allowed2LDs?,Roles?,AuditText?)>
<!ATTLIST RegistrarCreate
        ActionId    %UID;          #REQUIRED
        Name        CDATA          #REQUIRED
        AccRef      CDATA          #REQUIRED
        RegistrarId %RegistrarId; #REQUIRED
        URL         CDATA          #IMPLIED>

<!-- Update existing registrar details
     Response: (Registrar|Error) -->
<!ELEMENT RegistrarUpdate (RegistrarPublicContact?,
                           RegistrarSRSContact?,
                           DefaultTechnicalContact?,EncryptKeys?,
                           Allowed2LDs?,Roles?,AuditText?)>
<!ATTLIST RegistrarUpdate
        ActionId %UID; #REQUIRED
        Name     CDATA #IMPLIED
        AccRef   CDATA #IMPLIED
        URL      CDATA #IMPLIED>

<!-- Adjust a registrar's account by adding billing transactions
     Response: (BillingTrans|Error) -->
```

```
   <!ELEMENT AdjustRegistrarAccount (TransactionDate,BillPeriodStart,
                                     BillPeriodEnd,AuditText)>
   <!ATTLIST AdjustRegistrarAccount
           RegistrarId %RegistrarId;        #REQUIRED
           DomainName  %DomainName;         #REQUIRED
           ActionId    %UID;                #REQUIRED
           Months      %Number;             #REQUIRED
           ActionType  (%AccountingAction;) #REQUIRED>

   <!-- Adjust the billed until date for a domain
        Response: (Domain|Error) -->
   <!ELEMENT BilledUntilAdjustment (NewBilledUntilDate,AuditText)>
   <!ATTLIST BilledUntilAdjustment
           DomainName %DomainName; #REQUIRED
           ActionId   %UID;        #REQUIRED>

   <!-- Obtain billing records (optionally assign them to an invoice)
        Response: (BillingTrans*|Error) -->
   <!ELEMENT BillingExtract (TransDateRange,InvoiceDate?)>
   <!ATTLIST BillingExtract
           ActionId          %UID;     #REQUIRED
           InvoiceExtract    %Boolean; #IMPLIED
           RegistrarId       %UID;     #IMPLIED
           ConfirmedTrans    %Boolean; #IMPLIED
           InsideGracePeriod %Boolean; #IMPLIED
           InvoiceId         CDATA     #IMPLIED>

   <!-- Initiate process to rebuild the DNS zone files
        Response: (RunLog|Error) -->
   <!ELEMENT BuildDnsZoneFiles (RunDate)>
   <!ATTLIST BuildDnsZoneFiles
           ActionId %UID; #REQUIRED>

   <!-- Query billing transaction details of deferred income
        Response: (BillingTrans*|Error) -->
   <!ELEMENT DeferredIncomeDetailQry EMPTY>
   <!ATTLIST DeferredIncomeDetailQry
           BaseMonth   CDATA #REQUIRED
           BaseYear    CDATA #REQUIRED
           IncomeMonth CDATA #REQUIRED
           IncomeYear  CDATA #REQUIRED
           QryId       %UID; #IMPLIED>

   <!-- Query summary details of deferred income
        Response: (BillingTrans*|Error) -->
   <!ELEMENT DeferredIncomeSummaryQry EMPTY>
   <!ATTLIST DeferredIncomeSummaryQry
           BaseMonth   CDATA #REQUIRED
```

```
          BaseYear    CDATA #REQUIRED
          IncomeMonth CDATA #REQUIRED
          IncomeYear  CDATA #REQUIRED
          QryId       %UID; #IMPLIED>

   <!-- Initiate process to create a domain report
        Response: (RunLog|Error) -->
   <!ELEMENT GenerateDomainReport (RunDate)>
   <!ATTLIST GenerateDomainReport
          ActionId %UID; #REQUIRED>

   <!-- Query domain billing amounts in the system
        Response: (BillingAmount*|Error) -->
   <!ELEMENT QryBillingAmount EMPTY>
   <!ATTLIST QryBillingAmount
          QryId %UID; #IMPLIED>

   <!-- Create a new run log entry
        Response: (RunLog|Error) -->
   <!ELEMENT RunLogCreate (FirstRunDate,RunLog)>
   <!ATTLIST RunLogCreate
          ActionId %UID; #REQUIRED>

   <!-- Query existing run log entries
        Response: (RunLog*|Error) -->
   <!ELEMENT RunLogQry (LogDateRange?)>
   <!ATTLIST RunLogQry
          QryId       %UID; #IMPLIED
          ProcessName CDATA #IMPLIED
          Parameters  CDATA #IMPLIED>

   <!-- Cancel a scheduled job entry
        Response: (Schedule|Error) -->
   <!ELEMENT ScheduleCancel (FirstRunDate,AuditText?)>
   <!ATTLIST ScheduleCancel
          ActionId    %UID;            #REQUIRED
          ProcessName (%ScheduledJob;) #REQUIRED
          Parameters  CDATA            #IMPLIED>

   <!-- Create a new scheduled job entry
        Response: (Schedule|Error) -->
   <!ELEMENT ScheduleCreate (FirstRunDate,FinalRunDate?,AuditText?)>
   <!ATTLIST ScheduleCreate
          ProcessName (%ScheduledJob;) #REQUIRED
          Frequency   CDATA            #REQUIRED
          Parameters  CDATA            #IMPLIED
          ActionId    %UID;            #REQUIRED>
```

```
   <!-- Query details of existing scheduled job entries
        Response: (Schedule*|Error) -->
   <!ELEMENT ScheduleQry (ActiveOn?,FirstRunDate?)>
   <!ATTLIST ScheduleQry
           QryId       %UID; #IMPLIED
           ProcessName CDATA #IMPLIED
           Parameters  CDATA #IMPLIED>

   <!-- Update an existing scheduled job entry
        Response: (Schedule|Error) -->
   <!ELEMENT ScheduleUpdate (FirstRunDate,LastRunDate?,AuditText?)>
   <!ATTLIST ScheduleUpdate
           ActionId    %UID;             #REQUIRED
           Parameters  CDATA             #IMPLIED
           ProcessName (%ScheduledJob;) #REQUIRED>

   <!-- Create a new billing amount
        Response: (BillingAmount*|Error) -->
   <!ELEMENT SetBillingAmount (BillingAmount)>
   <!ATTLIST SetBillingAmount
           ActionId %UID; #REQUIRED>

   <!-- Query details of configurable system parameters
        Response: (SysParam*|Error) -->
   <!ELEMENT SysParamsQry EMPTY>
   <!ATTLIST SysParamsQry
           QryId %UID; #IMPLIED>

   <!-- Update details of configurable system parameters
        Response: (SysParam*|Error) -->
   <!ELEMENT SysParamsUpdate (SysParam+,AuditText?)>
   <!ATTLIST SysParamsUpdate
           ActionId %UID; #REQUIRED>

   <!-- RESPONSE ELEMENTS -->

   <!-- Billing amount -->
   <!ELEMENT BillingAmount (EffectiveDate)>
   <!ATTLIST BillingAmount
           Amount %Dollars; #REQUIRED>

   <!-- Billing transaction -->
   <!ELEMENT BillingTrans (InvoiceDate?,TransDate,BillPeriodStart,
                           BillPeriodEnd)>
   <!ATTLIST BillingTrans
           RegistrarId   %RegistrarId;  #REQUIRED
           Type          CDATA          #REQUIRED
           TransStatus   (%BillStatus;) #REQUIRED
```

```
           DomainName    %DomainName;   #REQUIRED
           RegistrantRef %UID;          #IMPLIED
           BillingTerm   %Term;         #REQUIRED
           InvoiceId     %UID;          #IMPLIED
           Amount        %Dollars;      #REQUIRED>

   <!-- Deferred registrar income amount -->
   <!ELEMENT DeferredRegistrarIncome (BaseMonth,BaseYear,
                                      IncomeMonth,IncomeYear)>
   <!ATTLIST DeferredRegistrarIncome
           RegistrarId  %RegistrarId; #REQUIRED
           BilledAmount %Dollars;     #REQUIRED
           BilledCount  %Number;      #REQUIRED>

   <!-- Domain element -->
   <!ELEMENT Domain (NameServers?,RegistrantContact?,
                     RegistrarPublicContact?,AdminContact?,
                     TechnicalContact?,BilledUntil?,RegisteredDate?,
                     CancelledDate?,LockedDate?,AuditDetails?)>
   <!ATTLIST Domain
           DomainName    %DomainName;     #REQUIRED
           RegistrantRef %UID;            #IMPLIED
           RegistrarName CDATA            #IMPLIED
           Status        (%DomainStatus;) #IMPLIED
           Delegate      %Boolean;        #IMPLIED
           Term          %Term;           #IMPLIED
           RegistrarId   %RegistrarId;    #IMPLIED
           UDAI          %UID;            #IMPLIED>

   <!-- Domain transfer message -->
   <!ELEMENT DomainTransfer (TransferredDomain+)>
   <!ATTLIST DomainTransfer
           RegistrarName CDATA #REQUIRED
           %TimeStamp;>

   <!-- Error -->
   <!ELEMENT Error (Description, ErrorDetails*)>
   <!ATTLIST Error
           ErrorId  %UID;    #REQUIRED
           Severity %Number; #REQUIRED
           Hint     %UID;    #REQUIRED>

   <!-- Raw request -->
   <!ELEMENT RawRequest (XML,Signature)>

   <!-- Raw response -->
   <!ELEMENT RawResponse (XML,Signature)>
```

```
<!-- Registrar details  -->
<!ELEMENT Registrar (RegistrarPublicContact,RegistrarSRSContact,
                     DefaultTechnicalContact,EncryptKeys,
                     Allowed2LDs?,Roles?,AuditDetails?)>
<!ATTLIST Registrar
        RegistrarId %RegistrarId; #REQUIRED
        Name        CDATA         #REQUIRED
        AccRef      CDATA         #REQUIRED
        URL         CDATA         #IMPLIED>

<!-- Run Log -->
<!ELEMENT RunLog (RunLogTimeStamp, RunLogDetails?)>
<!ATTLIST RunLog
        ProcessName  CDATA #REQUIRED
        Parameters   CDATA #IMPLIED
        ActionStatus CDATA #REQUIRED
        Control      CDATA #IMPLIED>

<!-- Schedule -->
<!ELEMENT Schedule (FirstRunDate,FinalRunDate?,CreateAuditText?,
                    CancelAuditText?)>
<!ATTLIST Schedule
        ProcessName         (%ScheduledJob;) #REQUIRED
        Frequency           CDATA            #REQUIRED
        Parameters          CDATA            #IMPLIED
        CreateByRegistrarId %UID;            #REQUIRED
        CreateActionId      %UID;            #REQUIRED
        CancelByRegistrarId %UID;            #IMPLIED
        CancelActionId      %UID;            #IMPLIED>

<!-- System parameter -->
<!ELEMENT SysParam (ParamValue, AuditDetails?)>
<!ATTLIST SysParam
        Name CDATA #REQUIRED>

<!-- UDAI validity -->
<!ELEMENT UDAIValid EMPTY>
<!ATTLIST UDAIValid
        Valid %Boolean; #REQUIRED>
```

Author's Address

Matthew Hunt
New Zealand Registry Services
Level 9
Exchange Place
5-7 Willeston Street
Wellington
New Zealand

Email: matt@nzrs.net.nz
URI: http://www.nzrs.net.nz/