Order Management Soutions

Neustar (A TransUnion Company).

LSR | Port Out

API Guide

Date: February 2024

© Copyright 2024 Neustar (A TransUnion Company). All Rights Reserved.

***Material herein contained may not be duplicated, redistributed, excerpted, sold, or stored in any information retrieval system without prior express written permission of Neustar (*Neustar (A TransUnion Company).**

TRADEMARKS

Neustar and the Neustar logo are registered trademarks of* Neustar (A TransUnion Company). *All other company, brand, and product names are the trademarks or registered trademarks of their respective companies.

Table of Contents

Chapter 1 About This Guide

The unified API provides a common access method for system access to the Clearinghouse Basic services. This document is a guide for accessing the Clearinghouse using the API to order Local Service (Order and Preorder).

1.1 Document History

The following table lists the change history of the document to date:

Date	Comments
02/12/2024	Document base-lined

1.2 Audience

This document is designed for developers with programming experience and a working knowledge of the terms and procedures used to implement SOAP-RPC.

1.3 Related Documents

This section lists related reference documentation for use in conjunction with this guide.

- *OMS Clearinghouse Database Schema Basic Services*
- *OMS Clearinghouse Standard Reports User Guide*

1.4 Conventions

The following table lists notational conventions found throughout this document:

Convention	Description	Example
Italics	Denotes the introduction of a term or phrase and that its definition is in the vicinity (either right before or right after).	The *engine* is the order management system.
Constant width	Indicates commands, file names, and file and code samples. Might be emphasized with bold.	% xterm -sb -title osagent
Italics, Constant width	Used within command, code, and file samples. Indicates file names or text to replace with words or names that are appropriate to your installation or environment. Might be emphasized with bold.	% perl copy2web.prl <*directory*> AdminPhn=*Administrator's phone number*
<>	Encloses a directory, file name or other information that needs replacement. The actual name should not be enclosed within angle brackets.	D:\<installation root directory>\ If the installation directory is called 'supplier', replace with: D:\supplier\
Hypertext link	Indicates a hypertext link that, if clicked, takes you to either an HTML page or a URL. A default browser must be specified.	Click here to view the link.
Cross-reference	Used to indicate a cross-reference that, if clicked, takes you to the indicated location in the document.	See ***Formatting Text*** on page 13

Convention	Description	Example
♪ NOTE:	A note symbol provides supporting information that is not explicitly addressed in the accompanying text.	♪ NOTE: This symbol indicates supporting information.
Date/Time	The Clearinghouse application is housed and maintained on the east coast. As such, the system records and displays dates and times based on the current eastern time, which, in the summer, is defined more specifically as Eastern Daylight Time (EDT) and as Eastern Standard Time (EST) in the winter.	n/a
XML Notation	It is Neustar standard to represent data values via Node attributes named “value”.	<DSENT value="11-03-2003-0900AM"/>

1.5 Assumptions

The following assumptions were made in the creation of this document:

- Users of this document have programming experience with a working knowledge of the terms and procedures used to implement SOAP-RPC and XML.
- Users of this document are connecting to the Neustar Clearinghouse via the API interface.

Chapter 2 Clearinghouse Overview

2.1 The Clearinghouse Platform

The Clearinghouse provides an advanced set of capabilities that enable a secure, reliable and scalable environment which deliver advanced services to our customers. It allows rapid implementation of new customers and trading partners as well as changes to business rules and workflows. The platform provides transaction routing, guaranteed delivery services, data transformation and support for the conversion and delivery of data to multiple trading partners over multiple protocols. It also supports fallout management services, reports and other service management capabilities.

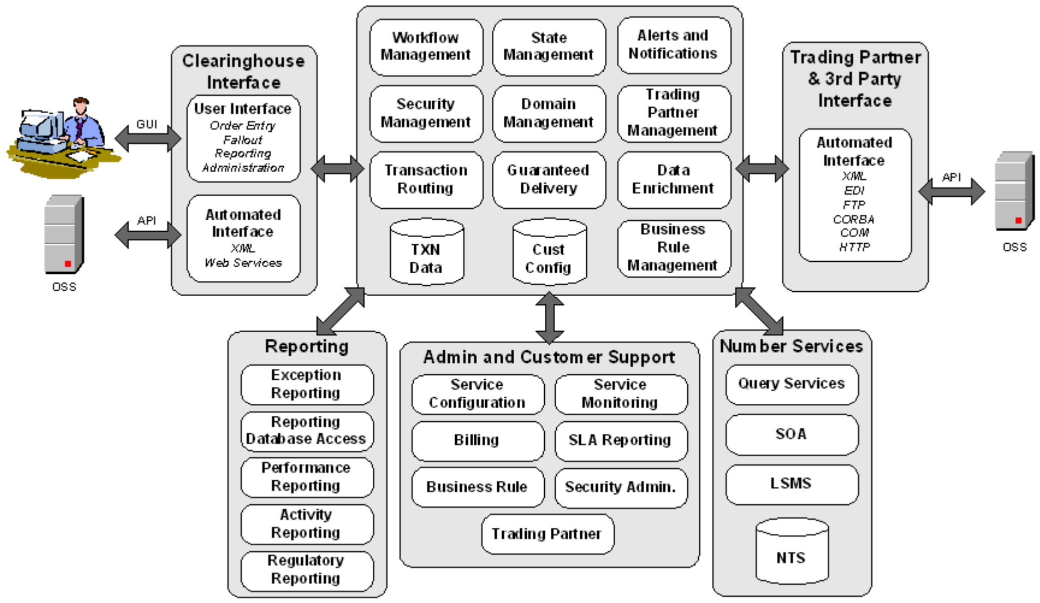


Figure 1: Clearinghouse Platform

Chapter 3 Integration Overview

This chapter provides a description of the SOAP-RPC interface used to communicate with the Clearinghouse. Neustar currently uses SOAP-RPC version 1.1. SOAP resolves integration problems caused by language and platform dependencies, easily allowing you to integrate with the Clearinghouse.

3.1 Accessing the Clearinghouse API

Neustar supports SOAP-RPC (Simple Object Access Protocol - Remote Procedure Call) over HTTPS as a means for systems to interact with the Clearinghouse (submitting requests and receiving responses). Systems can communicate directly with the Clearinghouse via properly formatted XML messages sent via SOAP-RPC. A message sent into this interface is referred to as a *request message* and the message returned from this interface is referred to as a *response message*.

The Clearinghouse is accessed either by the SOAP adapter or the Clearinghouse GUI as shown below in ***Figure 2***.

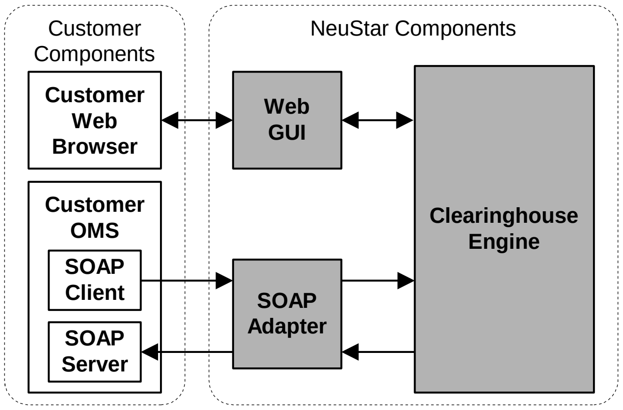


Figure 2: Accessing the Clearinghouse

The Neustar Clearinghouse allows you to send and receive messages with Trading Partners using a single interface. All messages exchanged between the customer and the Clearinghouse are structured as XML documents.

3.2 SOAP-RPC Server-Client Interactions

The SOAP-RPC communication consists of the following primary components.

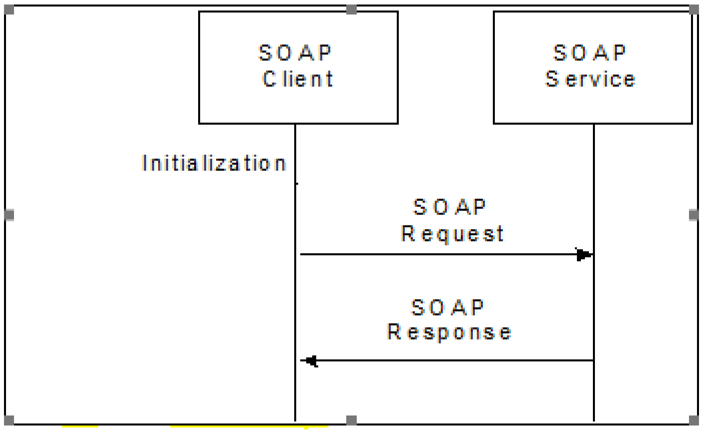
- SOAP client: Calls a method on a service.
- SOAP server: Provides the service and the implementation of the method being called.

Customer interaction with the Clearinghouse API is via SOAP-RPC. SOAP is a lightweight, XML-based protocol for exchanging information in a decentralized and distributed environment. SOAP consists of three primary parts:

- An envelope that defines a framework for describing what is in a message and how to process it.

- A set of encoding rules for expressing instances of application-defined data types.
- A convention for representing remote procedure calls and responses.

For additional information and specifications, see: SOAP Specification:



http://www.w3.org/TR/SOAP/

Figure 3: SOAP Client View

3.3 SOAPRequest Handler

The Clearinghouse Web service is named SOAPRequestHandler. External systems can communicate directly with the Clearinghouse with properly formatted XML messages sent via the SOAP-RPC protocol. The detailed structures of these message elements are specific to each request and response type supported.

The SOAPRequestHandler interface is the WSDL interface exposed to SOAP clients. It defines the methods a client invokes to send messages to the Clearinghouse. The WSDL for SOAPRequest handler is:

```
<wsdl:message name="processAsyncRequest">
 <wsdl:part name="in0" type="xsd:string"/>
 <wsdl:part name="in1" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="processSyncResponse">
 <wsdl:part name="processSyncReturn" type="impl:ArrayOf_xsd_string"/>
</wsdl:message>

<wsdl:message name="processSyncRequest">
 <wsdl:part name="in0" type="xsd:string"/>
 <wsdl:part name="in1" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="processAsyncResponse">
</wsdl:message>

<wsdl:portType name="SOAPRequestHandler">
 <wsdl:operation name="processSync" parameterOrder="in0 in1">
  <wsdl:input name="processSyncRequest" message="impl:processSyncRequest"/>
  <wsdl:output name="processSyncResponse" message="impl:processSyncResponse"/>
 </wsdl:operation>
</wsdl:message>
```

For each method, the first parameter is the request header; an XML structure containing information about the message (see ***Section 4.4***). The second parameter is the request message; an XML structure containing the specific contents of the request. **Table 1** lists the processSync and processAsync invocations used for each of the various Service Types.

Table 1: Basic Service - Request Types

Request Type	Service Type
processAsync	LSR Order LSR Preorder
processSync	Query

The processSync invocation also returns a String array. The array contains two XML strings: a header and a message.

3.4 SOAPResponse Handler - Customer Implementation

The SOAPResponseHandler interface, shown below, is the WSDL interface which your system must implement to provide the Clearinghouse with a way to return asynchronous responses and notifications to you. It defines the method the Clearinghouse invokes to send messages to your system. The WSDL for SOAPResponseHandler is:

```
<wsdl:message name="processEventRequest">
 <wsdl:part name="in0" type="xsd:string"/>
 <wsdl:part name="in1" type="xsd:string"/>
</wsdl:message>

<wsdl:message name="processEventResponse">
</wsdl:message>

<wsdl:portType name="SOAPResponseHandler">
 <wsdl:operation name="processEvent" parameterOrder="in0 in1">
  <wsdl:input name="processEventRequest" message="impl:processEventRequest"/>
  <wsdl:output name="processEventResponse"
message="impl:processEventResponse"/>
 </wsdl:operation>
</wsdl:portType>
```

For each method, the first parameter indicates the source of the event and the second parameter is the event, in XML format.

Figure 4 describes how asynchronous messages from the Clearinghouse are directed to the SOAPResponseHandler (implemented on your side). This use-case assumes there is a SOAPResponseHandler Web service implemented on your side.

The Clearinghouse provides reliable message delivery from the Clearinghouse to you. This service guarantees message delivery by storing messages in a database until they are successfully delivered.

When the Clearinghouse has an event to deliver to you, the Clearinghouse SOAP client saves and attempts to repeatedly deliver the message to your peer web service until the message is successfully delivered. If the SOAPResponse contains a valid response, the message is removed from the queue. If the SOAPResponse contains a SOAPFault, the queue agent subsequently attempts to resend the message a configurable number of times, effectively repeating this process. This process is shown below in ***Figure 4***.

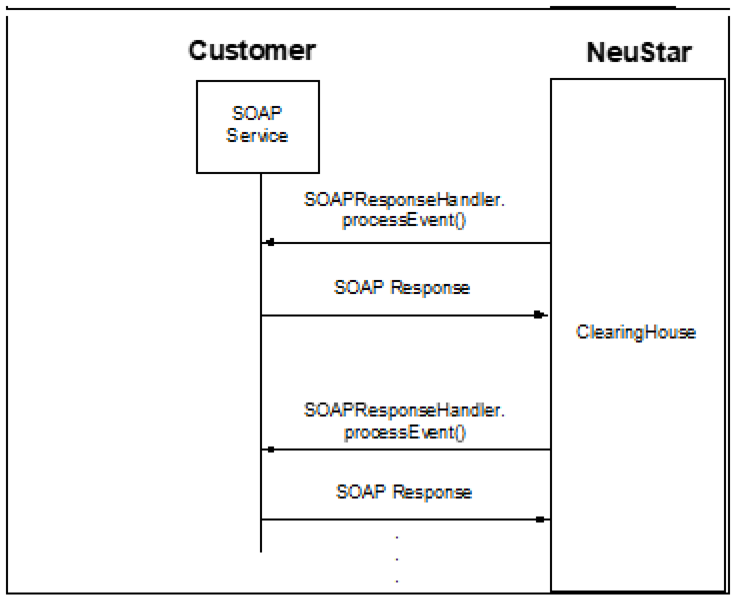

Customer
NeuStar
SOAP Service
SOAPResponseHandler.processEvent()
SOAP Response
ClearingHouse
SOAPResponseHandler.processEvent()
SOAP Response

Figure 4: Receiving a response via the SOAPRequestHandler

3.5 Error Handling – SOAP Fault Codes

There are four error codes associated with SOAP fault, they are:

1. SOAP-ENV:Server.MessageException
2. SOAP-ENV:Server.ProcessingException
3. SOAP-ENV:Server.SecurityException
4. SOAP-ENV:Server.userException

If an error occurs, the SOAP fault code displays the cause.

Upon detecting an error in processing, the Clearinghouse's SOAPRequestHandler generates a SOAPException with one of the SOAP fault code values listed in Table 2. The fault code values also appear in the fault string.

Example fault strings:

SOAP-ENV:Server.MessageException: Could not parse value for node 'Header'

or

```
SOAP-
ENV:Server.ProcessingException:com.nightfire.mgrcore.im.IMProcessingException:
 ERROR: Could not locate context at URL [t3://192.168.8.194:7010]:
javax.naming.CommunicationException [Root exception is
java.net.ConnectException: t3://192.168.8.194:7010: Destination unreachable;
nested exception is: java.net.ConnectException: Connection refused: connect; No
available router to destination]
```

SOAP clients invoking the SOAPRequestHandler should inspect these fault codes and take the appropriate corrective action.

Table 2: SOAP Fault Codes

SOAP Fault Code	Issue and Corrective Action
SOAP-ENV:Server.MessageException	Problem with the request data: Fix data and resubmit
SOAP-ENV:Server.ProcessingException	Transient system issue: Submit request again
SOAP-ENV:Server.SecurityException	Security violation error: Make sure client is authorized to submit request (check in Security Admin GUI)
SOAP-ENV:Server.userException	Unexpected error: Notify Neustar admin

When a request is submitted to the Clearinghouse, business rule validation is immediately performed on the input XML. The SOAP Fault Code SOAPENV:Server.MessageException returns business rule errors. All business rule errors are encoded into an XML document included in the SOAP Fault String. The XML document always follows the same format.

An example of a Fault String, which conveys business rule errors:

```
Fault String: SOAP-ENV:Server.MessageException:
<?xml version="1.0"?>
<Errors>
 <ruleerrorcontainer>
  <ruleerror>
       <RULE_ID value="NP_SERVICEDETAILS_PORTEDNBR_1"/>
       <MESSAGE value="PORTEDNBR is a required field."/>
       <CONTEXT
value="/Request/lsr_order/np/np_servicedetailscontainer/np_servicedetails[1]/PO
RTEDNBR"/>
       <CONTEXT_VALUE value=""/>
  </ruleerror>
  <ruleerror>
   <RULE_ID value="EU_LOCATIONACCESS_ZIP_2" />
   <MESSAGE value="ZIP is required when REQTYP is CB and ACT is V."/>
   <CONTEXT
value="/Request/lsr_order/eu/locationaccesscontainer/locationaccess[*]/ZIP"/>
   <CONTEXT_VALUE value="" />
  </ruleerror>
 </ruleerrorcontainer>
</Errors>
```

♪ NOTE: SOAP Fault Code errors and business rule errors are returned in synchronous fashion.

3.6 Queuing

The interfaces supported by trading partners (and used by the Neustar Clearinghouse) occasionally are off-line due to maintenance and operational issues. During these times, the Neustar Clearinghouse continues to accept SOAP (and GUI) requests from customers. These requests are saved to a database and queued. The orders go through the normal validation and you receive a business rule error, a SOAP Fault Code error or a successful submission notification. However, no Acknowledgement or response is received, as the order is stored in the database until connectivity is reestablished, at which time the requests are transmitted in their received order.

Chapter 4 Creating API Clearinghouse Messages

4.1 Overview of API XML

The API is an Extensible Markup Language (XML) interface that facilitates the integration of the Clearinghouse into a third party system. The API accepts and returns XML messages. Throughout this document, an XML message entering the API from the customer side is called a *request*, and one returning from the supplier side is termed a *response* (regardless of who initiates the exchange).

Messages between customers and trading partners are formatted as XML documents. A reference source for standard XML libraries is provided at http://xml.apache.org.

The structure of XML messages exchanged between a customer and the Clearinghouse is defined in a Document Type Definition (DTD) . The DTD contains the vocabulary and syntax of the document, and specifies a set of rules for the structure of an XML document. The DTD ensures that the Clearinghouse can read the document.

♪ NOTE: The Clearinghouse does not support runtime DTD validation and thus, XML messages should not contain *DOCTYPE* references.

4.2 XML Restrictions

XML requests submitted to the Clearinghouse must adhere to the following restrictions:

4.2.1 Date

All date fields in the U.S. Standard date format: mm-dd-ccyy

Two Digit Month (01-12)
Two Digit Day (01-31)
Two Digit Century (00-99)
Two Digit Year (00-99)

4.2.2 Time

All time fields in the U.S. Standard format: hhmm[AM|PM] or hhmm[AM|PM]-hhmm[AM|PM]

Two Digit Hour (01-12)
Two Digit Minute (00-59)
AM or PM

4.2.3 Date and Time

All date and time fields in the U.S. Standard format: mm-dd-yyyy-hhmm[AM|PM]. Either lowercase or uppercase variables for AM/PM can be used in the API. Mixed case variables, such as Am or pM, are not permitted, however, and will trigger a business rule validation error and return of the XML.

The following XML illustrates standard date and time field formats:

```
<DTSENT value="02-28-1999-1033AM"/>
```

Clearinghouse date and time fields allow a time range. The following are samples of valid field formats:

Date only:	"11-15-2001"
Date and time:	"11-15-2001-0800AM"
Date and time range:	"11-15-2001-0800AM-1100PM"

4.2.4 Integer

All integer fields contain digits.

4.2.5 Telephone Number

Standard format: xxx-xxx-xxxx-xxxx where the first set of digits is the area code, the second set of digits is the exchange, the third set of digits is the number, and the fourth set of digits is the extension. For example, if the telephone number is 510-999-9999-1234, 510 is the area code, 999 is the exchange, 9999 is the number, and 1234 is the extension. The Clearinghouse software is designed and tested to work with numbers from the North American Numbering Plan, therefore, the use of country codes is not allowed.

4.2.6 Special Characters

If any field requires special characters such as &, ‘, “, >, or <, use the corresponding entity shown in ***Table 3*** below. These characters only need replacement when they are embedded values inside the XML, as the XML Syntax uses these characters in its markup.

Table 3: Special characters and corresponding entries

Character	Entity
&	&amp;
‘	&apos;
“	&quot;
>	&gt;
<	&lt;

♪ NOTE: Most XML libraries make these character modifcations for you automatically.

4.2.7 Extra Attributes

Avoid generating “xmlns” attributes as part of XML requests.

4.3 Clearinghouse Transactions

Request is submitted by the client in order to perform some function on the basic services gateway. This section describes the types of requests supported by the Clearinghouse API.

4.3.1 LSR Preorder

The Clearinghouse can perform Preorder inquiry and response functions that take place prior to the ordering of service. The following LSR Preorder transactions are supported:

- **Customer Service Record (CSR)** – Queries the Trading Partner on how the existing service is provisioned.

4.3.2 LSR Order

The Clearinghouse supports the following LSR Order transactions:

- **Port** - Allows you to request an Unbundled Network Element, which represents the capability derived from the central office switch required to permit customers to transmit and receive information over the NSPs public switched network.

- **Simple Port** – involves an account only for a single line, does not involve unbundled network elements, complex switch transactions, or a reseller.

4.4 Request Header

When a request is submitted to the Clearinghouse via the API, the request header dictates the type of operation performed on the request message. The header XML for the request sent to the Gateway is constructed by the customer.

The following XML example illustrates the format of a request header XML document.

```
<header>
 <Request value=”lsr_order”/>
 <Subrequest value=”np_order”/>
 <CustomerIdentifier value=”BWC_BRSPD”/>
 <InterfaceVersion value=”LSOG6”/>
 <Supplier value=”BRSPD_PROD”/>
 <UserIdentifier value=”xxxxx”/>
 <UserPassword value=”xxxxx”/>
 <ApplyBusinessRules value=”Y”/>
 <IsUserPasswordEncoded value="YES"/>
 <Action value=”submit”/>
</header>
```

4.4.1 Header DTD

Neustar header XML is defined in DTD files. These are found and downloaded at https://oms.neustar.biz/convergentCH/content/ch_dl_header.html.The following DTD contains the vocabulary and syntax of Neustar’s XML documents for Header requests for Basic Services.

- OMS Clearinghouse (Basic) - DTD

Chapter 5 Basic Services

Basic Services require you to submit requests directly to the Gateways located in the Clearinghouse. This chapter provides specific details on sending Basic Service Requests and receiving Basic Service Response messages.

5.1 Basic Services

The Basic Services are single transaction processing requests. They include:

- Local Service Preorder (Customer Service Record)
- Local Service Order (LSR)

5.1.1 Header XML Structure

The following is an example section of the Basic Services header XMLRequest type. You may download a sample of it from the Neustar extranet site: https://oms.neustar.biz/convergentCH/content/ch_dl_header.html

```
<header>
         <Request value=”lsr_order”/>
</header>

or

<header>
        <Request value=”lsr_preorder”/>
<header>
```

5.1.1.1 Request and Subrequest Values

The **Request** and **Subrequest** values define the type of request performed. 4 defines valid values for Request and Subrequest in the request header for Basic Services. All requests are invoked via the processAsync() SOAP call. Business Rule errors are returned synchronously. Responses are returned in an asynchronous fashion.

Table 4: Request and Subrequest Values

Gateway Type	Request	LSR Transaction	Subrequest
LSR Preorder	lsr_preorder	Preorder	csr
LSR Order	lsr_order	Simple Port Port	simple_np np_order

5.1.1.2 Action Values

An **Action** field was added to the header to allow for additional API functionality. 5 shows the valid values for the Action field:

Table 5: Valid Values for the Action Field

Field	Valid Values	Description
Action	submit	The 'submit' action value is the standard action of submitting an order to the Clearinghouse. In previous versions of the Clearinghouse, this was the only valid action performed. ♪ NOTE: If the **Action** field is not present in the header, then the default behavior is 'submit'.
	save	The 'save' action value provides the ability to save an LSR Order, LSR PreOrder request in the Clearinghouse. From here, you may work the order via the Clearinghouse GUI. ♪ NOTE: When attempting to save the order via the API, the system checks to ensure that all of the required fields are populated. If all of the required fields are not populated, you receive an error indicating that the order cannot be saved in its current state. ♪ NOTE: If you attempt to re-save an order via the API, the order is saved with the information contained in your save request. Any changes made to the order via the GUI that are not made to your upstream system are not included in the saved record.
	validate	The validate action will check if the request is valid or not without submitting it to the WSP.
	cancel	The 'cancel' action value allows you to declare that an order is no longer valid. If a cancelled order receives a response, the order will be moved into the appropriate state.

♪ NOTE: When the save, validate actions are used, the API call must use the processSync() method.

5.1.2 Response Header

The response header is only returned from a `processSync()` invocation. The information in the request header is returned as-is in the response header.

5.1.3 LSR Preorder Request Structure

The ordering interfaces to different suppliers vary. Fields required by one supplier are often not supported by another. In order to define a programming API into the Clearinghouse that remains constant for a given client while supporting ordering interfaces to all suppliers. In this way, the supplier interfaces are maintained and released independently, without any interdependencies among the various supplier interface components of the gateway.

5.1.4 LSR Preorder DTDs

Table 6 illustrates how the LSR Preorder DTDs define the structure of requests and responses.

Table 6: LSR PreOrder Request Message XML Structure

`<Request>`	Generated inside the root element; conforms to the applicable DTD.

`<lsr_preorder>`
…
`</lsr_preorder>`
	Defined in the generic_<service>_request.dtd
	Contains generic data

`<SupplierLSRPreorderRequest>`
…
`</SupplierLSRPreorderRequest>`
	Defined in the <***supplier***>_<***service***>_request.dtd
	Contains ILEC-specific elements

`</Request>`	

5.1.5 LSR Preorder Response Structure

The messages delivered upon receipt of an asynchronous response from a trading partner contain one XML string consisting of a generic part and a supplier-specific part. While these messages always contain a generic and a supplier part, the contents of the supplier part are sometimes empty.

Table 4: LSR Preorder Response Message XML Structure

`<Response>`	Generated inside the root element; conforms to the applicable DTD.
`<lsr_preorder_response>` … `</lsr_preorder_response>`	Defined in generic_<service>_response.dtd Contains generic data
`<SupplierLSRPreorderResponse>` … `</SupplierLSRPreorderResponse>`	Defined in <supplier>_<service>_response.dtd Contains ILEC-specific elements
`</Response>`	

The following table lists response type values for LSR Preorder.

Table 8: LSR Preorder Response Type Values

Response type Values	
LSR Preorder	csr

5.1.6 LSR Order Request Structure

Table 9 illustrates how the LSR Ordering DTDs define the structure of requests.

Table 9: LSR Order Request Structure

`<Request>`	Generated inside the root element; conforms to the applicable DTD.
`<lsr_order>` … `</lsr_order>`	Defined in the generic_lsr_request.dtd Contains generic data
`<SupplierLSROrderRequest>` … `</SupplierLSROrderRequest>`	Defined in the <supplier>_lsr_request.dtd Contains ILEC-specific elements
`</Request>`	

5.1.7 LSR Order Response Structure

The messages delivered upon receipt of an asynchronous response from a trading partner contain one XML string, consisting of a generic part and a supplier-specific part. While these messages always contain a generic and a supplier part, the contents of the supplier part are sometimes empty.

Table 10: LSR Order Response Message XML Structure

`<Response>`	Generated inside the root element; conforms to the applicable DTD.
`<lsr_order_response>` … `</lsr_order_response>`	Defined in generic_lsr_response.dtd Contains generic data
`<SupplierLSROrderResponse>` … `</SupplierLSROrderResponse>`	Defined in <supplier>_lsr_response.dtd Contains ILEC-specific elements
`</Response>`	

Table 5: LSR Order Response Type Values

Response type Values	
LSR Order	ack negack soc billing_completion focaccept focreject suppaccept suppreject provider_initiated_activity jeopardy

Chapter 6 Request Header and Body

6.1 LSR Header

When a request is submitted to the Clearinghouse via the API, the request header dictates the type of operation performed on the request message. The header XML for the request sent to the Gateway is constructed by the customer.

The following XML example illustrates the format of a request header XML document.

Simple Port:
```
<header>
             <Request value=”lsr_order”/>
             <Subrequest value=”simple_np”/>
             <CustomerIdentifier value=”BWC_BRSPD”/>
              <InterfaceVersion value=”LSOG6”/>
             <Supplier value=”BRSPD_E2E”/>
              <UserIdentifier value=”XXXXX”/>
              <UserPassword value=”XXXXX”/>
              <ApplyBusinessRules value=”Y”/>
             <IsUserPasswordEncoded value="YES"/>
             <Action value=”submit”/>
</header>
```

Port Order:
```
<header>
           <Request value="lsr_order"/>
           <Subrequest value="np_order"/>
           <CustomerIdentifier value=”BWC_BRSPD"/>
           <InterfaceVersion value="LSOG6"/>
           <Supplier value="BRSPD_E2E"/>
           <UserIdentifier value="XXXXX"/>
           <UserPassword value="XXXXX"/>
           <ApplyBusinessRules value="Y"/>
           <IsUserPasswordEncoded value="YES"/>
           <Action value="submit"/>
</header
```

6.2 LSR Request Body

When a request is submitted to the Clearinghouse via the API, the request body dictates the type of operation performed on the request message. The body XML for the request sent to the Gateway is constructed by the customer.

Simple NP:

```
<Request>
	<lsr_order>
		<lsr>
			<lsr_adminsection>
				<DTSENT value="02-08-2024-0219PM"/>
				<REQTYP value="CB"/>
				<VER value="01"/>
				<AN value="12345"/>
				<DDD value="02-13-2024"/>
				<NPDI value="C"/>
				<CCNA value="CCN"/>
				<ACT value="V"/>
				<PON value="TESTPON01"/>
				<CC value="CC01"/>
				<PID value="PID"/>
				<AGAUTH value="Y"/>
				<NNSP value="NNSP"/>
			</lsr_adminsection>
			<contactsection>
				<TELNO value="111-111-0001"/>
				<EMAIL value="test@test.com"/>
			</contactsection>
			<REMARKS value="TEST_Remark"/>
		</lsr>
		<eu>
			<locationaccesscontainer type="container">
				<locationaccess>
					<ZIP value="12345"/>
					<ELT value="B"/>
				</locationaccess>
			</locationaccesscontainer>
		</eu>
		<np>
			<np_servicedetailscontainer type="container">
				<np_servicedetails>
					<PORTEDNBR value="111-001-0001"/>
				</np_servicedetails>
			</np_servicedetailscontainer>
		</np>
	</lsr_order>
</Request>
```

NP Order:

```
<Request>
	<lsr_order>
		<lsr>
			<lsr_adminsection>
				<DTSENT value="02-11-2024-1130AM"/>
				<REQTYP value="CB"/>
				<CCNA value="IUW"/>
				<NNSP value="6214"/>
				<AUTHNM value="DEBRA SMITH"/>
				<ONSPALTSPID value="927D"/>
				<DDD value="02-13-2024"/>
				<ATN value="352-200-2121"/>
				<DATED value="02-11-2024"/>
				<CC value="6214"/>
				<ACT value="V"/>
				<PID value="445522"/>
				<NPDI value="C"/>
				<AN value="974779527"/>
				<VER value="00"/>
				<MI value="C"/>
				<AGAUTH value="Y"/>
				<TOS value="2---"/>
				<ONSP value="927D"/>
				<PON value="6214024042649760"/>
			</lsr_adminsection>
			<REMARKS value="NLSP=6214"/>
			<contactsection>
				<TELNO value="888-338-7678"/>
				<EMAIL value="PortresponseResolve1@abc.com"/>
				<INIT value="GP"/>
			</contactsection>
		</lsr>
		<eu>
			<locationaccesscontainer type="container">
				<locationaccess>
					<LOCNUM value="001"/>
					<NAME value="DEBRA SMITH"/>
					<LCON value="DEBRA SMITH"/>
					<TELNO value="352-200-2121"/>
					<STATE value="FL"/>
					<CITY value="SUMMERFIELD"/>
					<ZIP value="34491"/>
					<SASN value="91ST"/>
					<ELT value="A"/>
				</locationaccess>
			</locationaccesscontainer>
		</eu>
		<np>
			<np_adminsection>
				<NPQTY value="00001"/>
			</np_adminsection>
			<np_servicedetailscontainer type="container">
				<np_servicedetails>
					<LNUM value="00001"/>
					<TDT value="Y"/>
					<PORTEDNBR value="352-200-2121"/>
					<LNA value="V"/>
					<NPI value="C"/>
					<NPT value="D"/>
				</np_servicedetails>
			</np_servicedetailscontainer>
		</np>
```

```
        </lsr_order>
</Request>
```

6.3 LSR Response

Focaccept

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <lsr_order_response>
        <ResponseType value="focaccept"/>
                <focaccept>
                        <lr>
                                <lr_adminsection>
                                <DTSENT value="02-08-2024-1201PM"/>
                                <CCNA value="BCJ"/>
                                <PON value="0004931214"/>
                                <VER value="00"/>
                                <AN value="834740018311213"/>
                                <DD value="03-01-2024"/>
                                <RT value="C"/>
                                <REP_TELNO value="111-111-1111"/>
                                <REP value="REP"/>
                                </lr_adminsection>
                        </lr>
                </focaccept>
        <TXID value="BNDW_SPROD_SPCTM_PROD"/>
        </lsr_order_response>
</Response>
```

FOcreject

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <lsr_order_response>
        <ResponseType value="focreject"/>
                <focreject>
                <REJECTTYPE value="NONFATAL"/>
                        <lr>
                                <lr_adminsection>
                                        <DTSENT value="02-11-2024-0951PM"/>
                                        <CCNA value="BCJ"/>
                                        <PON value="BWC0004931213"/>
                                        <VER value="00"/>
                                        <reasoncontainer type="container">
                                                <reason>
                                                <ERRORCODE value="001"/>
                                                <ERRORTEXT value="TN Not Found"/>
                                                </reason>
                                        </reasoncontainer>
                                        <RT value="E"/>
                                        <REP_TELNO value="111-111-1111"/>
                                        <REP value="REP"/>
                                </lr_adminsection>
                        <REMARKS value=""/>
                        </lr>
                </focreject>
        <TXID value="BNDW_PROD_BRSPD_PROD"/>
        </lsr_order_response>
</Response>
```

Jeopardy

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <lsr_order_response>
        <ResponseType value="jeopardy"/>
                <jeopardy>
                        <lr>
                                <lr_adminsection>
                                        <DTSENT value="09-29-2023-1008AM"/>
                                        <CCNA value="OWS"/>
                                        <PON value="88398379P759853"/>
                                        <VER value="02"/>
                                        <DD value="09-14-2023"/>
                                        <ATN value="360-895-1454"/>
                                        <AN value="3401039079401"/>
                                        <reasoncontainer type="container">
                                        <reason>
                                       <ERRORTEXT value="OTHER - See Remarks"/>
                                       <ERRORCODE value="OTHER"/>
                                        </reason>
```

```
                                                            </reasoncontainer>
                                                            <RT value="J"/>
                                                            <REP_TELNO value="866-406-3200"/>
                                                            <ESDD value="10-02-2023"/>
                                                            <REP value="CSD_PORT_OUTS"/>
                                                    </lr_adminsection>
                                    <REMARKS value="The Port has been canceled in LSOA &amp; all
                        Systems. An updated request is required any further Port requests of the TN(s)."/>
                                            </lr>
                                    </jeopardy>
                            <TXID value="WAVE_MOS_WAVE_PROD"/>
                            </lsr_order_response>
                    </Response>
```

6.4 CSR Header

```
<header>
        <Request value="lsr_preorder"/>
        <Subrequest value="csr"/>
        <CustomerIdentifier value="BWC_BRSPD"/>
        <InterfaceVersion value="LSOG6"/>
        <Supplier value="BRSPD_E2E"/>
        <UserIdentifier value="xxxxx"/>
        <UserPassword value="xxxxx"/>
        <ApplyBusinessRules value="Y"/>
        <IsUserPasswordEncoded value="YES"/>
        <Action value="submit"/>
</header>
```

6.1 CSR Request Body

```
<Request>
        <lsr_preorder>
                <RequestHeader>
                        <DTSENT value="02-08-2024-0849AM"/>
                        <CCNA value="CCN"/>
                        <TXTYP value="E"/>
                        <CC value="111X"/>
                        <TOS value="1"/>
                        <ONSP value="111X"/>
                        <TXNUM value="TESTTXNUM001"/>
                </RequestHeader>
                <RequestBody>
                        <csr>
                                <AGAUTH value="Y"/>
                                <service_address>
                                        <WTN value="111-111-1313"/>
                                        <STATE value="CA"/>
                                </service_address>
                        </csr>
                </RequestBody>
        </lsr_preorder>
</Request>
```

6.2 CSR Response Body

Error Response

```
<?xml version="1.0" encoding="UTF-8"?>
<Response>
        <lsr_preorder_response>
                <ResponseHeader>
                        <DTSENT value="02-15-2024-0748PM"/>
                        <TXTYP value="E"/>
                        <TXNUM value="NCSR36386247"/>
                        <TXID value="bandwidth_MCC_PROD"/>
                <presponsecontainer type="container">
                        <presponse>
                                <PRESPD value="TN Not Found"/>
                                <PRESPC value="001"/>
                        </presponse>
                </presponsecontainer>
                </ResponseHeader>
        </lsr_preorder_response>
        <SupplierLSRPreorderResponse>
                <ResponseHeader>
                        <CCNA value="BCJ"/>
                </ResponseHeader>
        </SupplierLSRPreorderResponse>
</Response>
```

Parsed Response

```
<?xml version="1.0" encoding="UTF-8"?><Response>
<lsr_preorder_response>
        <ResponseHeader>
        <DTSENT value="02-15-2024-0639PM"/>
        <TXID value="BNDW_PROD_BRSPD_PROD"/>
        <TXTYP value="E"/>
        <TXNUM value="NCSR36386973"/>
        <TOS value="1"/>
        <AN value="83493005001111"/>
        </ResponseHeader>
<ResponseBody>
<csr_response>
<parsed>
        <huntgroupcontainer type="container">
                <huntgroup>
                <INDEX value="1"/>
                </huntgroup>
        </huntgroupcontainer>
        <service_address>
                <SANO value="480"/>
                <SASD value=""/>
                <SASN value="HELP"/>
                <SATH value="ST"/>
                <SASS value=""/>
                <LD1 value=""/>
                <LV1 value=""/>
                <CITY value="MONROE"/>
                <STATE value="NC"/>
                <ZIP value="28110"/>
        </service_address>
<NAME value="H E L P CRISIS PREGNANCY"/>
<wtncontainer type="container">
        <wtn>
                <INDEX value="1"/>
                <WTN value="704-238-0900"/>
        </wtn>
        <wtn>
                <INDEX value="2"/>
                <WTN value="704-238-0925"/>
        </wtn>
        <wtn>
                <INDEX value="3"/>
                <WTN value="704-635-0936"/>
        </wtn>
        <wtn>
                <INDEX value="4"/>
                <WTN value="704-289-0914"/>
        </wtn>
        </wtncontainer>
</parsed>
</csr_response>
</ResponseBody>
</lsr_preorder_response>
<SupplierLSRPreorderResponse>
        <ResponseHeader>
                <CCNA value="BCJ"/>
        </ResponseHeader>
</SupplierLSRPreorderResponse>
</Response>
```

XML Structure CSR

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
Header				
		Request*	The valid values for Request and Subrequest determine the Basic Service requested.	lsr_preorder
		Subrequest*	The valid values for Request and Subrequest determine the Basic Service requested.	csr
		CustomerIdentifier*	The CustomerIdentifier value corresponds to your Domain. This value is provided by your Neustar account representative.	
		InterfaceVersion*	The InterfaceVersion defines the version of the interface to which a request is submitted.	LSOG6
		Supplier*	The Supplier value defines the trading partner to whom a request is submitted	
		UserIdentifier*	The UserIdentifier and UserPassword values correspond to your Username and Password. These values are provided by your Neustar account representative. Additionally, you may provision your own username/password via the User Admin functionality.	

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
		UserPassword*	Base64 encoded user password. The UserIdentifier and UserPassword values correspond to your Username and Password. These values are provided by your Neustar account representative. Additionally, you may provision your own username/password via the User Admin functionality.	
		IsUserPasswordEncoded	If the value is set to Yes we need to send Base64 encoded password of the user.	Yes or No
		ApplyBusinessRules	The ApplyBusinessRules field defines whether the Clearinghouse should apply business rules prior to submitting the message to the trading partner. This is typically set to “Y” (yes). It is only set to “N” (no) when operational necessities dictate the disabling of Clearinghouse business rules to allow specific transactions to flow to a trading partner. If this field is not included, the default is set to apply business rules.	Y (default) N
		Action	default value is “submit.	save validate submit cancel
Body				
	RequestHeader			
		DTSENT	Identifies the date and time the transaction is sent.	Required
		CCNA	Identifies the COMMON LANGUAGE IAC CODE for the customer submitting the inquiry and receiving the response.	Optional
		TXTYP	Identifies the type of transaction	Required
		CC	Identifies the Exchange Carrier generating the inquiry.	Optional
		ONSP	Identifies the NPAC SPI of the current Network Service Provider.	Optional
		TXNUM	Identifies the customer provided tracking number to link the inquiry with the response.	Required
	RequestBody			
		AGAUTH	Identifies that the customer is acting as an end user's agent and has authorization on file.	Required
		AUTHNM	Identifies the end user who signed the authorization.	Optional
		DATED	Identifies the date appearing on the agency authorization that was previously submitted to the provider.	Optional
		AN	Identifies the main account number assigned by the NSP.	Optional

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
		ATN	Identifies the account telephone number assigned by the NSP.	Optional
		WTN	Identifies the working telephone number at the end user's location.	Required
		SANO	Identifies the number of the service address.	Optional
		SASD	Identifies the street directional prefix for the service address.	Optional
		SASN	Identifies the street name of the service address.	Optional
		SATH	Identifies the thoroughfare portion of the street name of the service address.	Optional
		SASS	Identifies the street directional suffix for the service address.	Optional
		STATE	Identifies the abbreviation for the state or province.	Optional
		CITY	Identifies the city, village, township, etc. of the service location.	Optional

XML Structure LSR

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
Header				
		Request*	The valid values for Request and Subrequest determine the Basic Service requested.	lsr_order
		Subrequest*	The valid values for Request and Subrequest determine the Basic Service requested.	simple_np, np_order
		CustomerIdentifier*	The CustomerIdentifier value corresponds to your Domain. This value is provided by your Neustar account representative.	
		InterfaceVersion*	The InterfaceVersion defines the version of the interface to which a request is submitted.	LSOG6
		Supplier*	The Supplier value defines the trading partner to whom a request is submitted	
		UserIdentifier*	The UserIdentifier and UserPassword values correspond to your Username and Password. These values are provided by your Neustar account representative. Additionally, you may provision your own username/password via the User Admin functionality.	
		UserPassword*	The UserIdentifier and UserPassword values correspond to your Username and Password. These values are provided by your Neustar	

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
			account representative. Additionally, you may provision your own username/password via the User Admin functionality.	
		IsUserPasswordEncoded	If the value is set to Yes we need to send Base64 encoded password of the user.	Yes or No Optional
		ApplyBusinessRules	The ApplyBusinessRules field defines whether the Clearinghouse should apply business rules prior to submitting the message to the trading partner. This is typically set to “Y” (yes). It is only set to “N” (no) when operational necessities dictate the disabling of Clearinghouse business rules to allow specific transactions to flow to a trading partner. If this field is not included, the default is set to apply business rules.	Y (default) N
		Action	default value is “submit.”	save validate submit cancel
Body				
		PON	Identifies the customer's unique purchase-order or requisition number that authorizes the issuance of this request or supplement.	Required
		CCNA	Identifies the COMMON LANGUAGE IAC CODE for the customer submitting the inquiry and receiving the response.	Optional
		VER	Identifies the customer's version number.	Required
		CC	Identifies the Exchange Carrier generating the inquiry.	Required
		DDD	Identifies the customer's desired due date.	Required
		REQTYP	Identifies the type of service being requested and the status of the request.	Required
		ACT	Identifies the activity involved in this service request.	Required
		SUP	A supplement is any new iteration of an LSR. The entry in the SUP field identifies the reason for which the supplement is being issued.	Required, Only for subsequent version and not the initial version.
		PID	Identifies the end users personal identification number	Optional
		NNSP	Identifies the Number Portability Administration Center (NPAC) Service Provider Identifier (SPI) of the new Network Service Provider.	Required

XML Structure			Description	Usage and Valid Values
Element	Contains	Field		
		AGAUTH	Identifies that the customer is acting as an end user's agent and has authorization on file.	Required
		AUTHNM	Identifies the end user who signed the authorization.	Optional
		MI	Migration Indicator	Optional
		NPDI	Identifies the direction of wireless conversion activity on this request.	Required
		AN	Identifies the main account number assigned by the NSP	Required
		ONSP	Identifies the old network service provider	Optional
		INIT	Identifies the customer's representative who originated this request	Optional
		EMAIL	Identifies the electronic mail address of the initiator.	Optional

		ELT	Identifies the listing changes desired by the end user when changing local service providers.	A = Retain End User Listing
				B = Do Not Retain

		ZIP	Identifies the ZIP code, ZIP code + extension or postal code.	Required
		REMARKS	Identifies a free-flowing field which can be used to expand upon and clarify other data on this form.	Optional
		SANO	Identifies the number of the service address.	Optional
		SASD	Identifies the street directional prefix for the service address.	Optional
		SASN	Identifies the street name of the service address.	Optional
		SATH	Identifies the thoroughfare portion of the street name of the service address.	Optional
		SASS	Identifies the street directional suffix for the service address.	Optional
		STATE	Identifies the abbreviation for the state or province.	Optional
		CITY	Identifies the city, village, township, etc. of the service location.	Optional
		NPQTY	Identifies the quantity of ported numbers involved in this service.	Optional

XML Structure			**Description**	**Usage and Valid Values**
Element	Contains	Field		
		LNUM	Indetifies the line number	Required
		LNA	Identifies the activity involved at the line level.	V = conversion of service to new LSP
		TDT	Indicates the request for the activation of a ten digit trigger for local routing number portability.	Optional
		NPI	Identifies the status of the telephone number being ported.	Optional
		NPT	Indicates the type of number portability for this request.	Optional
		PORTEDNBR	Identifies the ported telephone number	Required