xLink Reference Manual

Version 1.2

15 Nov 2005

xlink-support@covad.com

1. API Overview

Covad's xLink API provides an electronic interface for business requests such as service pre-qualification, order entry and status inquiries, trouble ticketing, order summary, and so forth, through a well defined and standard data format. This lets you integrate your order management system with Covad and develop your own interfaces to your customers.

The Covad xLink API is a Web/HTTP-based interface. You send your requests to Covad's xlink API (https://xapi.covad.com/servlet/MainVCAServlet) through an HTTPS URL connection (non-secure http connections are also supported). Your request XML must conform to the requirements specified in the request DTD (http://xapi.covad.com/dtd/request.dtd) and related DTDs as appropriate. Within seconds, the Covad xLink API server returns a response XML that conforms to the format specified in the response DTD (http://xapi.covad.com/dtd/response.dtd).

REQUEST TYPE AND FUNCTIONS

The Covad API can handle requests for a variety of actions:

- Service availability
- Order entry
- Order status
- Order summary
- Ticket entry
- Ticket update
- Ticket status
- CPE lookup
- Kit shipment information
- Network weather
- DSLAM circuit statistics

Service Availability Request

Service Availability module is used to pre-qualify an end user for DSL Service. That is, this module checks if Covad could provide the DSL Service for the address and telephone number provided. You send an XML request of type

serviceavailabilityrequest to the API server, with specific details concerning the end user for whom the service is desired. The API server responds with an XML of type serviceavailabilityresponse. The response includes the available services and their availability dates, which may be in the future. If the server encounters any errors, it returns them as part of the XML response.

Order Entry Request

An Order Entry request places an order for an end user. You provide specific details concerning the order, such as the physical address where the DSL service is desired, phone number, the user's hardware configuration etc., in an orderentryrequest. In response, Covad's API server returns an XML stream of type orderentryresponse which contains the order number and Covad circuit number which you will later use to track the order. If the server encounters any errors, it returns them as part of the XML response.

Order Status Request

You will use an Order Status request to determine the current status of an order. When you post an *orderstatusrequest* to the Covad API server, you will specify which order you wish to learn about by specifying an order number, or, for example, a particular milestone or date range. Covad responds with XML of type *orderstatusresponse* containing information on the order's current status of the order, including any problems encountered by Covad during the process.

Order Summary Request

The Order Summary Request can provide you the status of a whole range of orders. You provide a date range for the *ordersummaryrequest* and the XML response provides the status of each order along with the current state and any problems encountered.

The following table summarizes the modules you will use to qualify a prospect, place an order, and then check its status.

	PROCESS STEP	SUB PROCESS STEP	ACTION
1	Pre-Qualification	Pre-Qualification Request	You POST an XML data stream, which has a request type “ServiceAvailabilityRequest.” This data stream contains relevant information like the NPA-NXX and address.
		Pre-Qualification Response	Covad sends back an XML data stream of type “ServiceAvailabilityResponse”. This contains a list of services that the above request qualifies for.
2	Order Entry	Order Entry Request	You post an XML data stream of type “OrderEntryRequest.”
		Order Entry Response	Covad sends back an XML data stream of type “OrderEntryResponse”. This contains the Order ID and, if you request it, will also include the Covad circuit number and your billing code.
3	Order Status	Order Status Request	You post an XML data stream of type “OrderStatusRequest.”
		Order Status Response	Covad sends back an XML data stream of type “OrderStatusResponse.” This contains order status and work log information.
4	Order Summary	Order Summary Request	You post an XML data stream of type “OrderSummaryRequest.” This specifies a date range for which you want a summary.
		Order Summary Response	Covad sends back an XML data stream of type “OrderSummaryResponse.” This contains a summary of service requests for your date range and their statuses.

XML HEADER

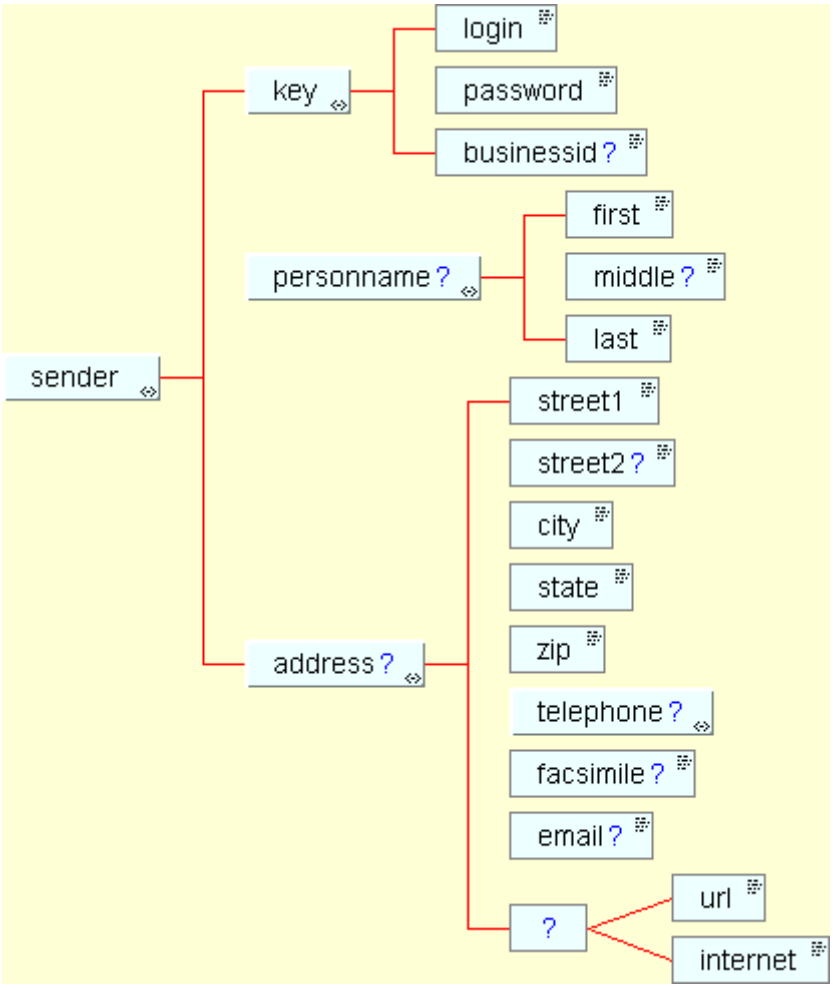
Each XML Request you post to Covad must have a well-formed header. The header contains information regarding authentication and may include your contact personnel and contact person at Covad.

The header has three elements:

- *sender* – Authentication and Sender Information
- *recipient* – This is deprecated
- *datetime* – An optional date and time this request is being made

Sender

The Sender element is primarily for authentication and identification of you as the requesting entity. Covad's xLink API does not use the basic authentication scheme laid out in the HTTP RFC, because it is not very amenable for machine to machine authentication. xLink authentication occurs in the Request XML under the key element. The figure below illustrates the sender schema:



An example Sender element looks like this:

```
<sender>
  <key>
    <login>testcustomer</login>
    <password>testcustomer</password>
    <businessid>38</businessid>
  </key>
  <personname>
    <first>Mike</first>
    <middle></middle>
```

```
      <last>Stitch</last>
    </personname>
    <address>
      <street1>1st Street</street1>
      <street2></street2>
      <city>San Francisco</city>
      <state>CA</state>
      <zip>94444</zip>
      <telephone>
        <npa>415</npa>
        <nxx>999</nxx>
        <lastfour>0007</lastfour>
      </telephone>
      <facsimile></facsimile>
      <email></email>
      <url></url>
    </address>
  </sender>
```

The first element under the sender element is the key, which contains the following elements – login, password, and businessid. After you have been authorized to use the API, Covad will provide you the login, password, and businessid you will use in your requests. This provides the authentication required to submit requests to the server. Values for login and password are required. businessid is optional. If your login or password are invalid, you will receive an error response (see http://xapi.covad.com/dtd/transactioncode.mod for details).

The personname key specifies a point contact at your company. The last element in the sender is the address element. Here, you should provide the actual physical address of your company.

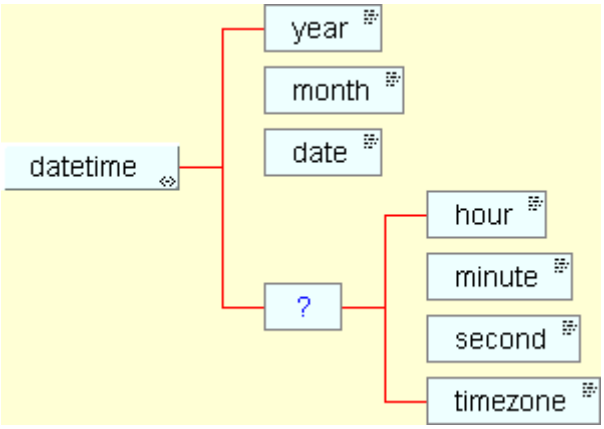
Recipient

The recipient element is now deprecated for requests, but at one time specified your contact at Covad.

datetime

Datetime element specifies the date and time at which you are issuing your request. Though optional, it can help you track errors and identify the system state when you issued the request. Every effort should be made to synchronize the time on the computer issuing the request with standard time servers. Covad strongly recommends using third party time synchronization tools to automate the process.

The figure below illustrates the datetime schema:



An example datetime element is presented here

```
<datetime>
  <year>2000</year>
  <month>3</month>
  <date>10</date>
  <hour>2</hour>
  <minute>50</minute>
  <second>24</second>
  <timezone>-8</timezone>
</datetime>
```

2. Pre-Qualification Module

The pre-qualification or *ServiceAvailability* Module qualifies an end user for service. When you wish to place an order for DSL service, you will submit an XML request document of type *serviceavailabilityrequest* to COVAD's xLink API, with specific details concerning the end user such as their address, phone number, and ZIP code. Covad's xLink API processes the request information and sends an XML response document of type *serviceavailabilityresponse*. The response document contains the speeds and availability dates (for speeds not available currently) for the requested location. Any errors encountered in this process are also returned as part of the XML in the response document. You can then parse the response document and provide the information to the customer in a format of your choice.

SERVICEAVAILBILITYREQUEST REQUEST DOCUMENT

To access the service availability information, a valid request document should be sent to xLink API (http://xapi.covad.com/servlet/MainVCAServlet) through a URL connection. The request document should have a subrequest type '**serviceavailability'.** If the **cpeid** for serviceavailability is specified, then the response will contain services that are compatible with that **cpe**.

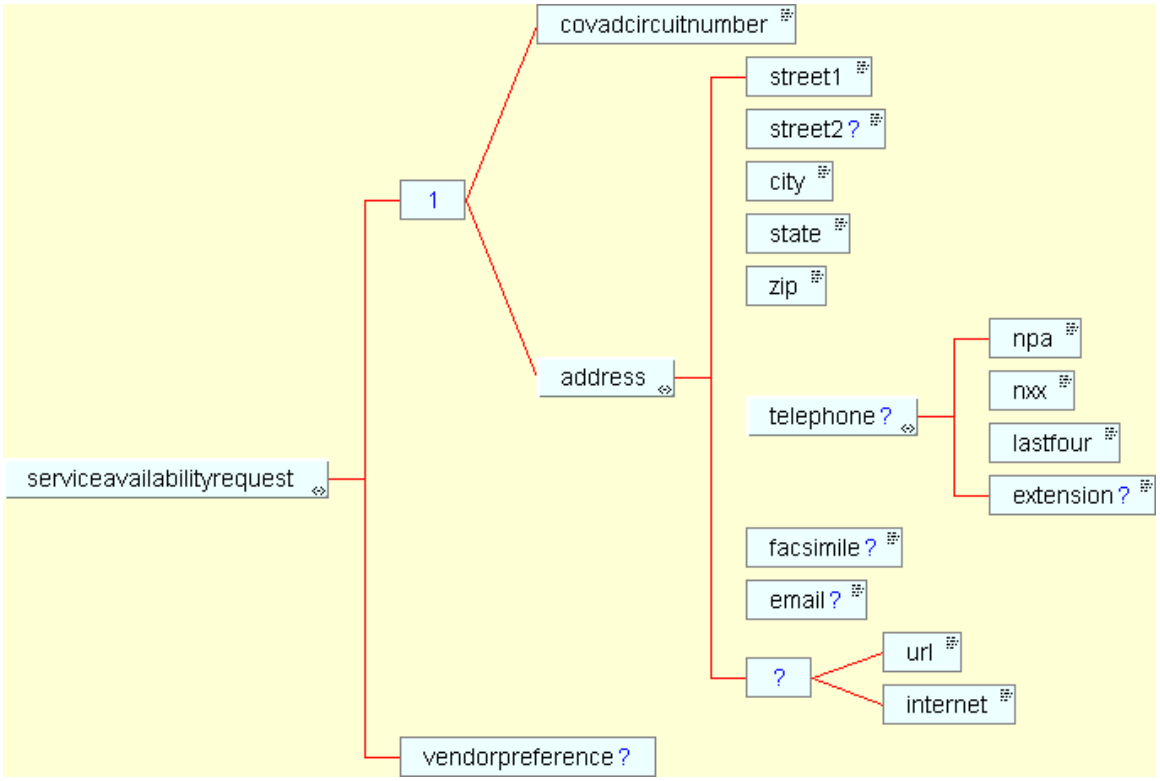
You can alternatively provide a covadcircuitnumber to check for availability of service.

covadcircuitnumber: If you provide a covadcircuitnumber instead of an address, the response will contain a list of compatible services for that circuit. You will find this particularly useful when you wish to determine what upgrades you can offer the customer.

address: The address requesting the DSL service should be provided here. The street1, city, state and ZIP are required to provide the available speeds and services for that location. The response will contain an error if any of these values is missing from your request. If system recognizes an apartment number in street1, the system will remove it from street1 and prepend it to the street2 element. Also, be aware that if street2 exceeds 30 characters in length, it will cause an exception.

vendorpreference: This information is deprecated and ignored.

The figure below shows the serviceavailabilityrequest schema:



An example of *serviceavailabilityrequest* element within a body and subrequest is shown below:

```
<body>
	<subrequest>
		<serviceavailabilityrequest>
			<address>
				<street1>2330 Central Exp</street1>
				<street2>Suite #A</street2>
				<city>Santa Clara</city>
				<state>CA</state>
				<zip>95050</zip>
				<telephone>
					<npa>408</npa>
					<nxx>844</nxx>
					<lastfour>1111</lastfour>
					<extension></extension>
				</telephone>
				<facsimile></facsimile>
				<email></email>
				<url></url>
			</address>
		</serviceavailabilityrequest>
	</subrequest>
</body>
```

SERVICEAVAILABILITYREQUEST RESPONSE DOCUMENT

When you submit a valid *serviceavailabilityrequest*, Covad's xLink API server returns a *serviceavailabilityresponse* document in XML format. You can then parse the response document and present the information in an appropriate format to the end customer.

The body of the response document contains the address for which you want DSL service information and a list of services available there. The response may be formatted in several different ways according to *orderabilityresult* or *availabilityresult* based on the type of service requested. The response document also contains the *transactioncodeid*, which specifies the status of service availability. For instance, if the *transactioncodeid* is 1000, then the service availability is OK. For a full listing of transaction codes and their description, please refer to http://xapi.covad.com/dtd/transactioncode.mod.

orderabilityresult: If you specify a "type" attribute to your *serviceavailabilityrequest,* then the response will organize the services available according to service family. This is how to specify "type":

```
<serviceavailabilityrequest type="orderable">
```

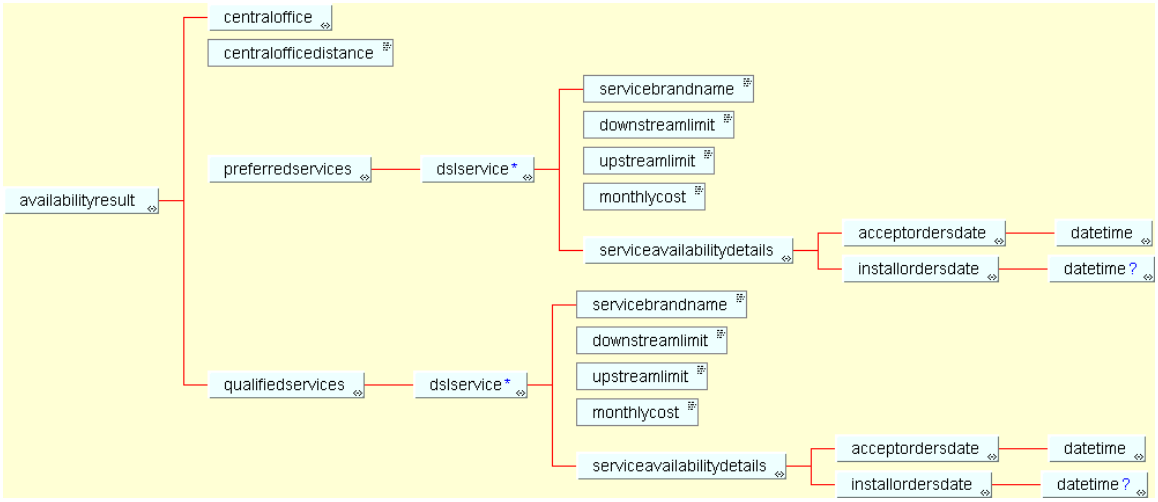
The response will contain a series of four <servicefamily> elements, one each for local, remote, ip, and prip services:

```
<servicefamily type="remote">
<service id="9">TeleSpeed Remote 144</service>
<service id="13">TeleSpeed Remote 192</service>
<service id="10">TeleSpeed Remote 384</service>
<service id="12">TeleSpeed Remote 768</service>
<service id="11">TeleSpeed Remote 1.1</service>
<service id="23">TeleSpeed Remote 1.5S</service>
<service id="440">SOHO 3.0/768 Remote (Access Only)</service>
<service id="441">SOHO 3.0/768 Remote (Self-Install)</service>
<backhaulcircuit id="99585" capacity="DS-3" regionid="5"
status="1" national="0" coverage="5"/>
<backhaulcircuit id="99844" capacity="DS-3" regionid="7"
status="1" national="0" coverage="7"/>
</servicefamily>
```

The response document will also return the CLLI code and location of the central office and the estimated physical distance from CO. The distance from the CO will determine the second-line speeds that will be available for the requested address. If you also include the optional additionaladslinfo attribute, the response

will include the telephone number's actual distance from the CO according to the ILEC's records, if we have that value.

availabilityresult: If you do not specify "type," then the response will contain all available services. The *availabilityresult* returns the CO information such as code, location, and distance between the CO and the service address. It also returns *installordersdate* and *acceptorders* dates which you can use to determine if you wish to offer upgrades to the user now or at some point in the future.



You will find some example *serviceavailability* requests and their responses on the xLink site: http://xlink.covad.com/Library/usecasefiles/index.html

3. Order Entry Module

The Order Entry module primarily deals with entering an order. You post a request of type *orderentryrequest* to the Covad API Server and it responds with a document of type *orderentryresponse*.

The *orderentryerquest* element can have one of the following action types:

- submit
- submitloop
- save
- test

The *submit* action is the default case and submits the order to Covad. For backward compatibility, no Covad circuit number is returned if you make an implicit *submit*. If you explicity specify `action="submit"` then Covad's response will contain the order's Covad circuit number.

submitloop is only applicable for frame orders, and is the first part of a two-part ordering process.

The *save* action stores the order for later completion or processing. *save* is supported only for initial install orders. To submit or update a previously saved order, you must specify an id attribute in *orderentryinformation* which refers to the saved order number. To remove a previously saved order, use the *savedorderremoval* request. To view a previously saved order, use the *savedorderretrieval* request.

test checks your order for consistency without saving or submitting.

ORDERENTRYINFORMATION ELEMENT

The orderentryinformation element is the main element within orderentryrequest and contains the following elements in it.

serviceaddress

The service address specifies the installation address of the end. It represents the physical address where Covad will install the service. It is very important that this element be populated accurately.

clientbusinessname

This optional element specifies the user's business name.

billingcode

This is an optional element you may use for your own billing or department code. If not applicable, you may leave this field blank or simply not send it in your request.

clientcontactinformation

Provide the contact name, address, and phone number for the end-user within this element. This element is mandatory. If the contact information is same as the service address, repeat that information here.

service

Enter the id corresponding to the service you wish to order for the end user, according to the results of your pre-qualification response. This field is mandatory and should be accurately entered. The contractid attribute specifies the term of contract. A contractid of 1 means one year and 2 means two years, and if no contract is specified, use a contractid of 0.

customercircuit

The customercircuit element refers to the backhaul you wish to use when you provide the Layer2 service. You specify the value for the circuit you wish to use as an attribute of the customercircuit element:

```
<customercircuit id='73581'></customercircuit>
```

The serviceavailability results will return usable backhauls if you specify type="orderable" in your serviceavaiability request. If you wish to order a “local” service, then the backhaul you specify in customercircuit must be in the same region as the user’s service address. However, if you choose a Remote service, you may choose a circuit in a region other than the one corresponding to the service address for the end-user. If you specify an invalid backhaul, Covad’s response will give you a list of valid backhauls.

cpe

cpe element specifies the hardware to be installed at the end-user’s premises. It is applicable to second-line services. The cpe element has the following attributes.

id

This represents the numerical id of the CPE. See http://xapi.covad.com/dtd/cpemodelid-enum.pen for a list of supported and

available CPEs. You must select a CPE according to the service you are ordering, because not all CPEs can be used with every type of second-line service.

cpeprovider

Though this attribute is mandatory, the default value is "1" meaning that Covad will provide the CPE. Use a value of "2" if you will be providing the CPE.

cpeconfigurer

This represents who will be configuring the CPE, and its default value and options are the same as cpeprovider.

clientsitedetails

This field should specify all known data about the client's site. If an element is unknown, you can provide an element with no spaces in between tags should be entered. See http://xapi.covad.com/dtd/clientsitedetails.mod for more information on which are required and which are deprecated.

networkconfiguration

The networkconfiguration element is optional for SOHO services but mandatory for TeleXtend and TeleSpeed services. It specifies the network configuration for the order. The type attribute indicates the type of network to be installed, either IP routing, IP routing with NAT, or Bridging. Depending on the type of network you are requesting, you may need to provide additional elements.

assignedpvc

This element is only applicable in Covad responses, where it will specify the value of the PVC when you provide Layer2 service.

requestedpvc

If you are providing Layer2 service, you may request a particular PVC on your backhaul. This is a numeric representation (maximum of 7 digits) of the PVC and its format depends on the type of backhaul you are using for the installation. The format is defined as follows:

a) For a Frame Relay circuit, you must specify a DLCI. A legal value for a DLCI is an integer greater than 31. If a DLCI greater than 31 is unavailable, another number should be entered or the field should be left blank to submit the order.

b) For an ATM circuit, you must specify a VPI.VCI combination. A legal value of VPI.VCI has the following format: X.Y where X is an integer >= 0 and <= 31, and Y is an integer >= 32 and <=65535. You may find that a legal value of VPI.VCI is

sometimes unavailable, in which case you should specify another number or leave it unspecified blank, in which case we will pick the next available PVC.

preferredpvc

This element is deprecated. Use *requestedpvc* instead.

cpe

This element is optional and should be omitted. Covad uses this element to return CPE information in responses to you.

wanconfiguration

This field is mandatory. All the subelements must be entered if the type of network configuration is either 'routing' or 'routing with NAT'. For 'bridging' this field may be left blank.

lanconfiguration

This field is mandatory. All the subelements must be entered if the type of network configuration is either 'routing' or 'routing with NAT'. For 'bridging' this field may be left blank.

dhcpinformation

This field is mandatory. If the value of 'usedhcp' attribute is 'yes', then you must specify a dhcp server ip. Otherwise the ip may be left blank and it is ignored.

notes

This fields may be left blank, but you will find it helpful for sending additional notes or information to Covad for proper order processing.

ORDERENTRYRESPONSE DOCUMENT

The orderentryresponse element is the main element that Covad returns in our response to your orderentryrequest post. An orderentryresponse element can contain the following

- covadcircuitnumber
- vendorordernumber

We will return a covadcircuitnumber only if you explicitly used the action=”submit” attribute in your request. We do this to maintain backward

compatibility. If you do not specify action=”submit” then we do not return the circuit number.

The *vendorordernumber* is the order number itself.

Any errors are noted in the header and in the body depending on the severity.

A successful *orderentryresponse* XML looks like this:

```
<subresponse id="" type="">
  <transactioncodelist>
    <transactioncode id="2000">
      <phrase>OK</phrase>
      <message>Order Entry - OK</message>
    </transactioncode>
  </transactioncodelist>
  <orderentryresponse>
    <vendorordernumber>5048903</vendorordernumber>
    <covadcircuitnumber>100-362-801</covadcircuitnumber>
    <billingcode>ACK050518-01</billingcode>
    <telephone>
      <npa>415</npa>
      <nxx>242</nxx>
      <lastfour>3365</lastfour>
    </telephone>
  </orderentryresponse>
</subresponse>
```

The subresponse id (which has no value in the example above) is the optional identifier ‘id’ you could have specified as an attribute of your subrequest. There is one subresponse for each subrequest submitted, and although you can submit multiple subrequests within a single request, we recommend that you submit only one subrequest in each request.

The transactioncodelist entity contains a numeric status code and a human readable text message indicating the status of the order placement. The *orderentryresponse* entity contains the *vendorordernumber* which specifies the order number belonging to the subrequest and, as in the example above, may contain the circuit number, too. It will also include your *billingcode* if you provided one in your request.

4. Order Status Module

The Order Status module reveals the status of an order placed with Covad. An XML request of type '*OrderStatusRequest*' returns status information within an '*OrderStatusResponse*'.

ORDERSTATUSREQUEST REQUEST DOCUMENT

You can specify the level of detail using the "details" attribute, either basic, full, fulllog, xfull or xfulllog, for example:

```
<orderstatusrequest details='xfulllog'>
```

xfull and xfulllog provide intricate details such as insidewiring, cpeprovider etc. To refine the orders returned, you can specify the following parameters:

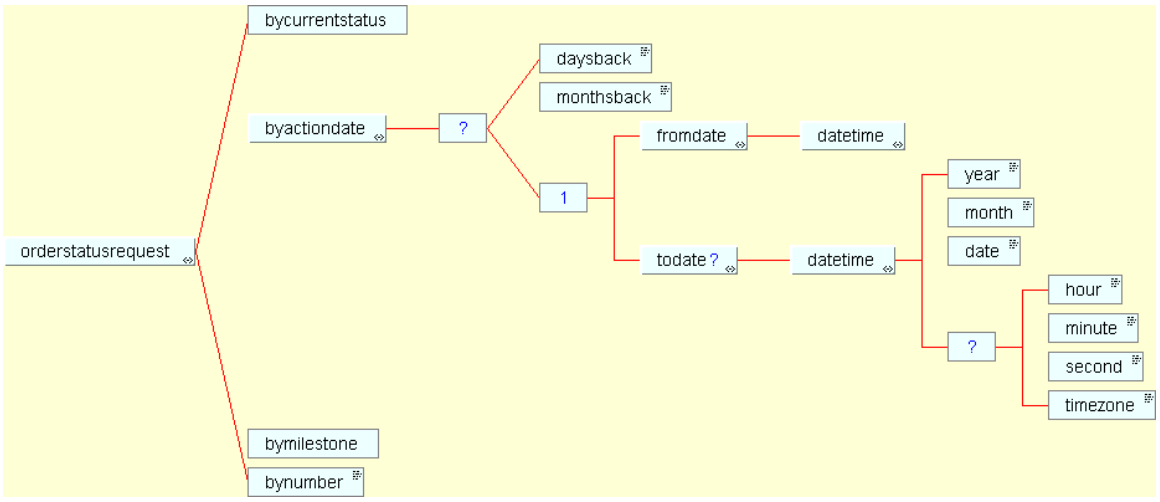
bycurrentstatus: This is a deprecated synonym for *bymilestone*.

bymilestone: You can request orders within the following milestones: orderreceived, lineordered, linecommitted, lineverified, appointmentscheduled, appointmentcompleted, lineorderissue, linecommitissue, lineverifyissue, appointmentscheduleissue, appointmentcompleteissue, ordercloseissue.

byactiondate: You can request the status of a submitted or closed order by providing either to and from dates, or the number of months or days in the past the order was placed.

bynumber: You can request the status of a particular order by simply providing its order number.

The figure below illustrates the *orderservicerequest* schema.



Here are three examples of an *orderstatusrequest*:

```
<subrequest type='orderstatus' id='basic details'>
  <orderstatusrequest details='basic'>
<bynumber id='301444'/>
  </orderstatusrequest>
 </subrequest>

 <subrequest type='orderstatus' id='full details'>
  <orderstatusrequest details='full'>
   <bynumber id='301444'/>
  </orderstatusrequest>
 </subrequest>

 <subrequest type='orderstatus' id='full w/worklog'>
  <orderstatusrequest details='fulllog'>
   <bynumber id='301444'/>
  </orderstatusrequest>
 </subrequest>
```

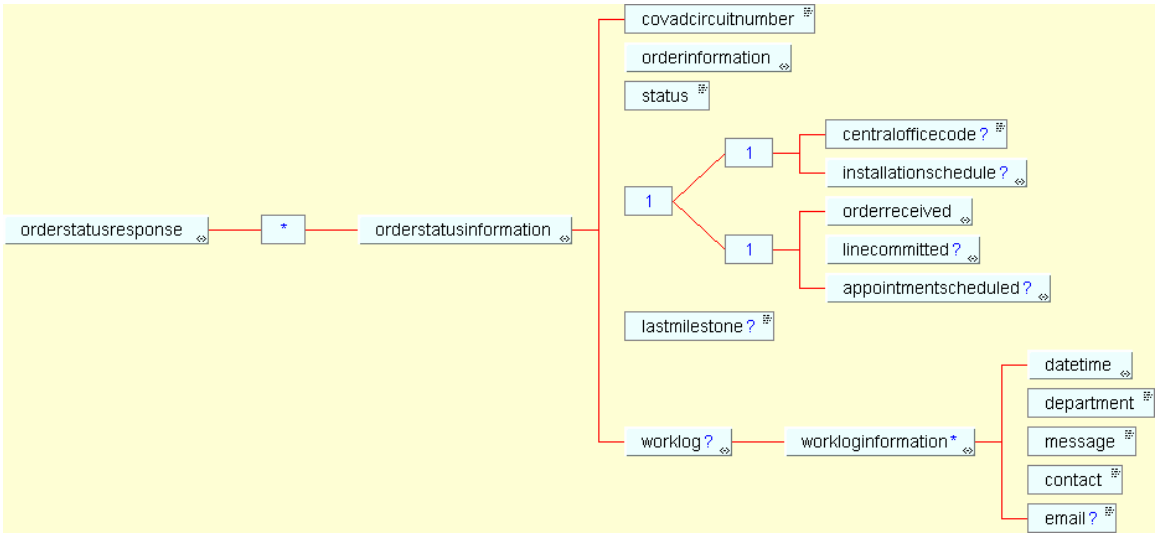
Level of detail

Note that type='orderstatus' in the subrequest element is not required. The type of request is specified by the <orderstatusrequest> element itself.

ORDERSTATUSRESPONSE RESPONSE DOCUMENT

Based on the level of detail you have requested, the orderstatusresponse document provides the information on the status of the orders requested in the *orderstatusrequest* request document. The transactioncodeid for a successful *orderstatusresponse* document is **3000**. The response document provides

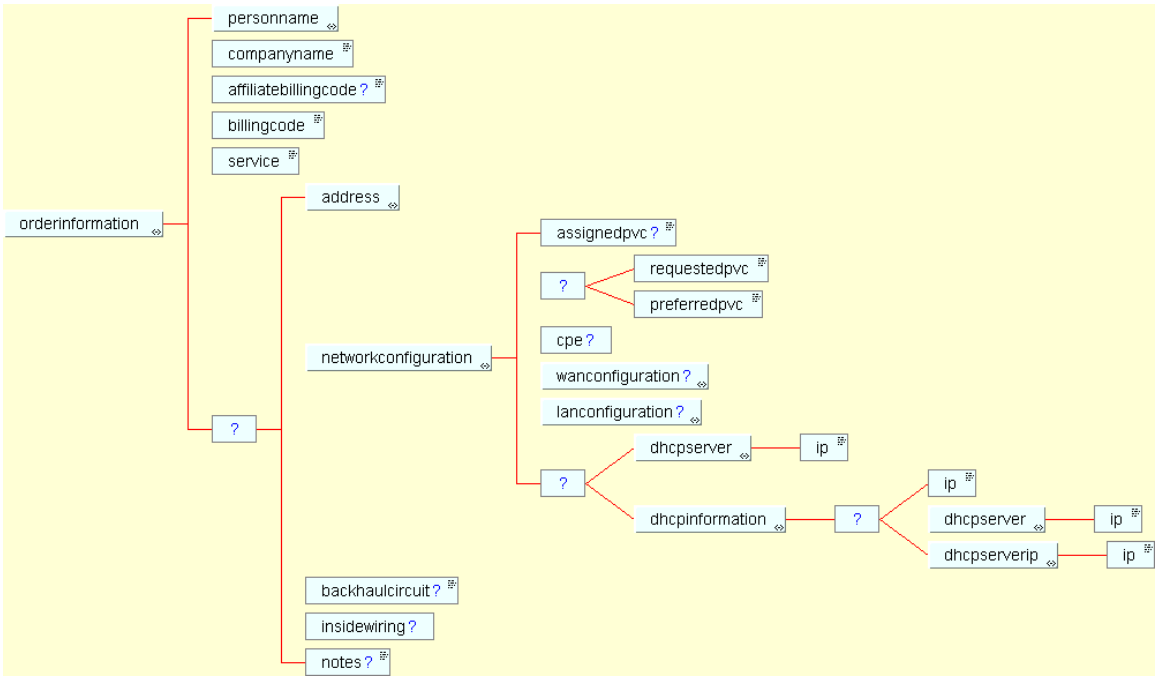
order information for each of the orders that satisfy the criteria you specified in your orderstatusrequest. Shown below is the schema for orderstatusresponse object. The * indicates that the response document can contain multiple orderstatusinformation elements. In other words, if your request the orderstatus returns information on more than one order, the response document will send *orderstatusinformation* for each of the orders.



orderstatusinformation contains the following elements

Covadcircuitnumber: Covad provides this number. You can use the *covadcircuitnumber* to check for service availability at a later time.

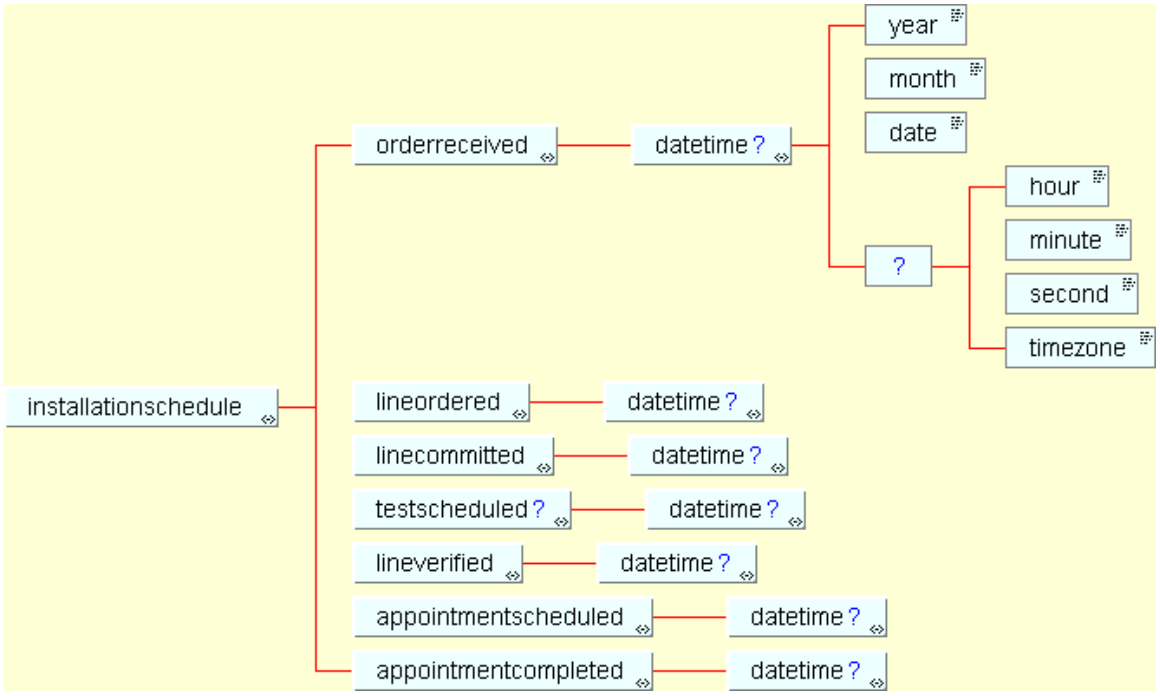
Orderinformation: This element contains the information of the customer requesting the DSL service such as *personname*, *billing code*, and *service*. Details such as *address*, *networkconfiguation, backhaulcircuit* are optional and are provided if *full* or *xfull* details are requested. The figure below displays the *orderinformation* element schema, and contains IP information.



status: This indicates the current status of the order. This is free form text and has a message such as *Install order received*.

centralofficecode: This element contains Covad's central office code.

installationschedule: This element contains information about the dsl installation process. You will receive this in the response if you specify *basic* details in your request. installationschedule will contain the date and time of *orderreceived*, *lineordered, linecommitted, testscheduled, lineverified, appointmentscheduled,* and *appointmentcompleted*.



orderreceived: This element returns the date and time that Covad received the order. The response contains this information if you request either *full, xfull,* or *xfulllog* details in your request document

linecommitted: This element returns the date committed by the ILEC or ISP to deliver a DSL line. The response contains this information if you request either *full, xfull,* or *xfulllog* details in your request document

lineOrdered: This element returns the date and time that Covad placed the order with the ILEC.

appointmentscheduled: This element returns the date and time of the scheduled appointment by Covad. The response contains this information if you request either *full, xfull,* or *xfulllog* details in your request document.

lastmilestone: This element contains the description of the last milestone reached. The response contains this information only if you request *basic* details in your request document.

worklog: This element is a log of the work information such as the *message* of work, *date*, *department*, *contact* and *email*. The response contains this information if you request *fulllog* details in your request document.

Shown below is a sample *worklog* element:

```
<worklog>
      <workloginformation title="Install Order Accepted">
              <datetime>
                <year>2005</year>
                <month>5</month>
                <date>23</date>
                <hour>10</hour>
                <minute>30</minute>
                <second>35</second>
                <timezone>-8</timezone>
              </datetime>
              <department/>
              <message>We received your DSL installation
Service Request on May 23, 2005  10:30:35 AM and have begun
processing the request to your specifications. Your request
is currently in an unconfirmed state, please allow up to 72
hours for Covad to receive confirmation from the phone
company.  Please monitor the work logs for the request
confirmation note, whereby the order will be referred to as
an installation order. Should there be any delay in
fulfilling your order, we will promptly notify
you.</message>
              <contact/>
              <email/>
      </workloginformation>
</worklog>
```

5. Order Summary Module

The Order Summary module provides you a summary of all the events that have been logged against an order. You will find this module useful when you want to query a whole range of orders currently pending with Covad. When you submit and ordersummaryrequest, Covad's xLink server returns an *ordersummaryresponse* document.

OrderSummaryRequest Request Document

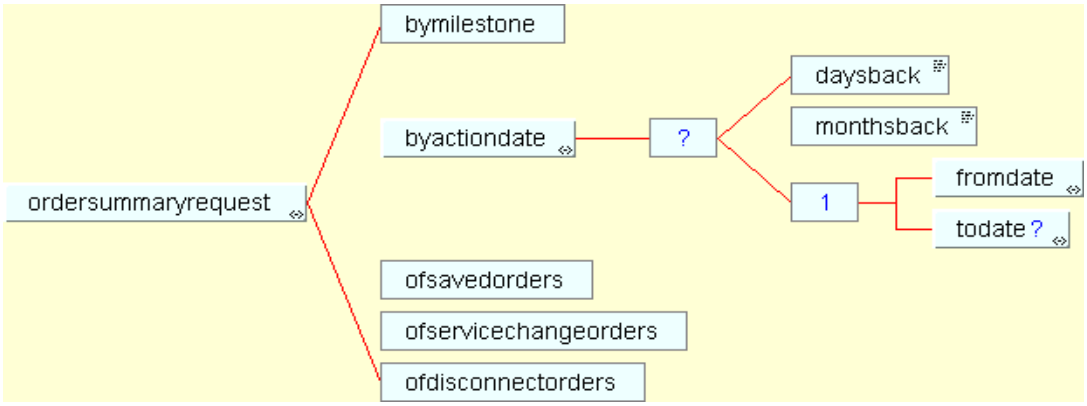
You can retrieve a summary of orders according to these parameters:

bymilestone: The user can request a summary of orders according to the following milestones: orderreceived, lineordered, linecommitted, lineverified, appointmentscheduled, appointmentcompleted, lineorderissue, linecommitissue, lineverifyissue, appointmentscheduleissue, appointmentcompleteissue, ordercloseissue.

byactiondate: You can request a summary of orders where the action date is either orderopen, ordersubmitted, ordercanceled, orderputonhold, orderclosed, or orderupdated. You can provide to and from dates, or the number of months or days in the past that you placed the order.

ofsavedorders: You can have affiliates save orders on your behalf, and you can a see a summary of saved orders for each affiliate using the *ofsavedorders* element.

The figure below depicts the schema for ordersummaryrequest request element:



.*ORDERSUMMARYRESPONSE* RESPONSE DOCUMENT

For a given milestone or action date in the request document, the order summary response document provides a count of orders in various categories. The transaction code id for this response document is 4000.

category: The response document provides the different phases of the orders such as *Direct Orders, Order Received by Covad, New Line Ordered, Line Delivery Commitment, Line Verified, Covad Appointment Scheduled, Covad Appointment Completed, Problem With Address/Number, Problem Obtaining Line, Problem With Line Delivery, Problem With Line, Problem With Scheduling, Problem After Appointment* for the requested milestone or datetype.

count: The count of the order in each of the above-mentioned categories.

The figure below depicts the schema for *ordersummaryresponse*.

6. Trouble Ticket Module

The Trouble Ticket module enables you to submit concerns or issues about DSL service electronically. Like any call center, when you submit a ticket entry request to the Covad xLink sever, the server logs the issue and returns a trouble ticket number to you. You can then check the status of the issue by sending a ticket status request XML. The server processes this information and returns the status of the problem.

TICKETENTRYREQUEST REQUEST OBJECT

ticketentryrequest contains two elements, *ticketcontact* and *ticketentrydetails*.

ticketcontact contains contact information of the person submitting the problem. It has the three sub-elements:

personname: name of the person submitting the problem

email: Email Id of the contact

telephone: Telephone number of the contact

ticketentrydetails is the element that contains the details of the problem. It has two attributes: *typecode* and *serviceaffecting* and has the following sub-elements:

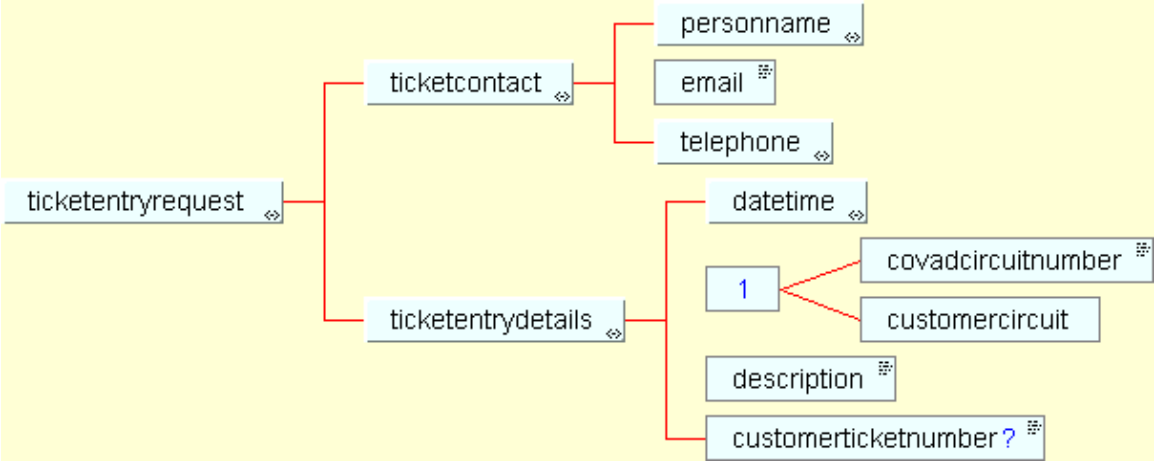
datetime: date and time you send the xml request to Covad.

covadcircuitnumber: You can provide a *covadcircuitnumber* or *customercircut* that is having the problem.

description: This is a textual description of the problem.

customerticketnumber: This element is optional. You should send this value if you have already logged the issue with Covad and you are submitting additional information.

Shown below is the schema of *ticketentryrequest* element:



TICKETENTRYRESPONSE OBJECT

When you submit a ticketentryrequest request to the Covad xLink server, the server logs the problem and returns a ticketentryresponse document containing a ticketnumber.

TICKETSTATUSREQUEST OBJECT

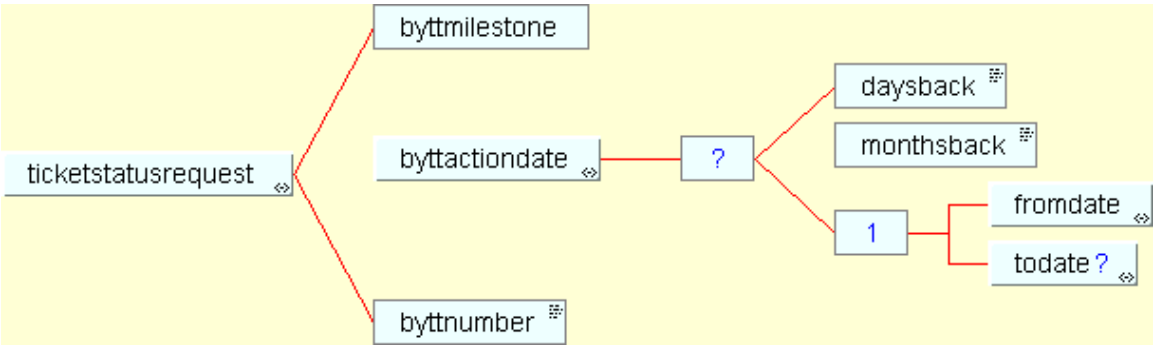
You can check the status of the trouble ticket by submitting a ticketstatusrequest XML containing either the milestone, actiondate, or ticket number. You can request a basic, full or fulllog report using the details attribute. The ticketstatusrequest element contains the following sub-elements:

byttmilestone: This is an optional element. If you provide this value, you should also specify the *statustype*. The statustype can have values from 1 through 25 where statustype = 1 is ***assigned*** and statustype=25 is ***OPEN-Pending ILEC Cable Maintenance***. See http://xapi.covad.com/dtd/ticketstatus-enum.pen for more information on statuses.

byttactiondate: You can check tickets within date ranges by specifying monthback or dayback or todate and fromdate.

byttnumber: You can specify a particular trouble ticket by providing its number.

The *ticketstatusrequest* object schema is shown below:



TICKETSTATUSRESPONSE RESPONSE OBJECT

If you request basic details in your ticketstatusrequest, then the response will contain *ticketcontact* and *basicticketinfo* information. If you request fulllog details, then the response will include additional information such as *extraticketinfo*, *worklog* and *dispatchlog*.

ticketcontact : This element contains the contact information of the person who submitted the problem/issue. It contains three subelements: *personname*, *email*, and *telephone*.

basicticketinfo: This element contains basic information about the ticket in these subelements:

ticketnumber: The unique identifier for the case.

ticketstatus: The current status of the problem.

opendate: The date and time the case was opened.

lastupdate: The date and time the ticket was last updated.

personname: The contact person's name.

telephone: The telephone number of the contact person.

company_name: This is an optional element.

extraticketinfo: This information is provided to you if you specify "full" details. Information about the problem, priority, assignedto, resolutioncode etc. appears in this element.

worklog: This element contains a log of all the tasks that have been carried out to resolve the problem.

dispatchlog: This element contains a taskID, the name of the engineer handling the problem, and the dispatch status such as TT-Urgent (the status of the trouble ticket based on priority).

Shown below is the schema for ticketstatusresponse: