Nasdaq Trade Surveillance

XML Interface

Specification and process

Revision 1.14
17 May 2021
Approved

While reasonable efforts have been made to ensure that the contents of this document are accurate, the document is provided strictly "as is", and no warranties of accuracy are given concerning the contents of the information contained in this document, including any warranty that the document will be kept up to date. Nasdaq reserves the right to change details in this document without notice. To the extent permitted by law no liability (including liability to any person by reason of negligence) will be accepted by Nasdaq or its employees for any direct or indirect loss or damage caused by omissions from or inaccuracies in this document.

NTS-XML-Interface-specificationV1.13

Document History

Revision	Published	Authors	Description
1.2	20 May 2015	B. Li L. Halaoui	Updated section 3 (3.3-3.5 added). Updated to standard Nasdaq template.
1.3	05 Jun 2015	B. Li	Updated section 2 (2.1-2.2), section 3 (3.1-3.4), added section 4, updated section 5, updated section 6 (6.2.3 added).
1.4	28 Jun 2015	A. Shen	Removed references to API V5 due to a bug
1.5	20 Oct 2015	D. Tennakoon	Updated details of query parameters. New parameter introduced with SMARTS v8 Added new elements available from API v5 and API v6
1.6	15 Jan 2016	D. Berridge	Added information regarding alert deep-linking.
1.7	10 Oct 2016	D. Tennakoon	Included enhancements added with API v7 Added information on the messages element – added with api v5
1.8	28 July 2017	P.Natarajan	Updated data structure for User list XML report introduced as part of SMARTS V9.1
1.9	23 Jan 2018	L. Le, K. Russell, D.Tennakoon	Added apiVersion=8 and 9. Migrated to new template, plus minor edits for readability. Added Structured Data Elements
1.10	15 May 2019	M. Patel	Corrected lookbackdays text
1.11	13 Sep 2019	K. Russell, F. Maisetti	Added a temporary amendment to the optional Alerts XML parameter 'lookBackDays' in Section 3.1 (on page 8). This change temporarily limits lookback to ONE CALENDAR DAY to optimise application performance.
1.12	3 Oct 2019	T. Byford	Updated sections 2.3 and 3.1 around restrictions on API requests
1.13	7 Apr 2020	T. Byford	Rebrand to Nasdaq Trade Surveillance

Revision	Published	Authors	Description
1.14	31 May 2021	M. De Silva	Added API v10 section 3.1.1.8 Updated description of date parameter in 3.1

Table of Contents

1 Overview

Nasdaq Trade Surveillance (NTS) provides a number of interfaces that allow users to export data in XML format. Exported data can then be processed in other applications, or for backup purposes.

The following XML exports are available in Nasdaq Trade Surveillance:

- **Alerts XML:** This interface can be used to export information about alerts, including their statuses and comments.
- **User List XML:** This interface can be used to export the User List report, which provides information about user accounts.

1.1 Audience

This specification can be used by Nasdaq customers who are implementing systems to export data from Nasdaq Trade Surveillance.

2 Export process

2.1 Permissions required

In order to export information to an XML file, the user account must have the following permissions:

- To export alerts only, the **Alerts** permission is required for relevant market(s). To export alerts, alert statuses, and comments, the **Alerts Management** permission is required for relevant market(s).
- To export the User List, the **User List** permission is required.

2.2 XML export service

Authenticated users with sufficient permissions can programmatically export required data in XML format, by entering the location of the service.

The **Alert XML** export service is available at:

```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/exportAlerts?marketCode=<market>&date=<yyyymmdd>
```

The **User List XML** export service is available at:

```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/exportUserList
```

2.3 Using automated exporting systems

The Data Export Interface can produce the export URL that can also be accessed remotely without a browser. This allows clients to write applications that can bulk-export data programmatically (e.g. get all the alerts for a market at the end of every month).

Authentication is performed using HTTP Basic Authentication over SSL/TLS (https) using server authentication. At this point in time, there are no plans to support SSL/TLS client certificate based authentication.

The Nasdaq Trade Surveillance servers will present a server certificate that is issued by the internal Nasdaq Certification Authority. If you need to verify the server certificate presented, Nasdaq can provide you with the root certificate for that Certification Authority.

The requesting user/application must have a user account (with access to the requested data information) and must authenticate themselves before data can be exported.

ⓘ Please note that the number of API calls is limited to **100 requests per 1 minute interval.**

ⓘ Users may utilize “bundle” (see 3.1) to extract alerts and their attachments with a single API call.

2.4 Using manual export

The Data Export Interface can be accessed manually using a web browser in order to test the URLs and data availability. This process includes the following steps:

1. The user enters the URI of the export service, including the applicable parameters required (e.g. host <yourorganisation>, market, date for Alert export).
2. The XML file containing the data is displayed.
3. The user may then save the file to a desired location. If a bundle was requested, the XML file and any other attachments are downloaded automatically.

3 Alert XML

3.1 Optional request parameters

When submitting a request to the Alerts XML export service, the following **optional** parameters can also be specified:

Parameter	Description
`marketCode`	Takes a code identifying the market to export alerts from. If this parameter is not specified, all markets where the user has **Alerts** viewing permissions will be exported.
`date`	Takes a date in the format "`yyyymmdd`" specifying the issuing date of alerts to be exported. The date is interpreted relative to the specified market's (i.e. "`marketCode`") time zone. For example: • Requesting alerts for "`LSE/20080505`" will give you alerts from 20080505 00:00:00 - 20080505 23:59:59 in the UTC time zone (LSE is in UTC time zone). • Requesting alerts for "`ASX/20080505`" will give you alerts from 20080505 00:00:00 - 20080505 23:59:59 in AEST (UTC+10:00) time zone. This is actually 20080504 14:00:00 - 20080505 10:00:00 in UTC time zone. If this parameter is not specified, the exported alerts will be based on the API version: • For API v10, the completed alerts for the most recent date or the last public holiday (whichever occurs later) will be exported. • For API v9 and below, alerts for the current date will be exported.
`bundle`	Takes a "`true`" or "`false`" value specifying whether to export alerts as a zip archive: • If set to "`true`", the XML file and relevant attachments will be downloaded as a single zip file archive. • If set to "`false`" or unspecified, files are individually downloaded. Please note that extraction of non-bundled alerts will be depreciated in a future release and all API responses will be bundled into a zip archive.
`apiVersion`	Takes a number between 6 and 9 specifying the API version of the export format. See section 3.1.1 for more information about each version of export format. If this parameter is not specified, alerts will be exported in the default API version (`version 9`).
`lookbackDays`	Takes up to ONE calendar day from the specified "`date`" for which alerts are to be exported. For example, "`lookbackDays=1`" with no date specified will export the current day's alerts plus one day of alerts 'looking back' from the current day. ⓘ **Note:** This parameter is temporarily limited to ONE CALENDAR DAY to optimise application performance. If this parameter is not specified, only alerts for the specified "`date`" are exported.

Example

```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/exportAlerts?marketCode=<market>&date
=<yyyymmdd>&bundle=<true|false>&apiVersion=<n>&lookbackDays=<n>.
```

As an example, if two days' of alerts need to be exported 'looking back' from 17th September 2015, the following URI can be used:

```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/exportAlerts?marketCode=market&date=2
010917&apiVersion=4&lookbackDays=1
```

As a further example, if customer wishes to query for multiple days across 50 markets the customer could generate 50 requests in parallel within each minute; each request for 1 market and 1 selected day at a time.

3.1.1 Alerts XML structure and the 'apiVersion' parameter

In order to ease the integration into users' internal systems, the Alert XML export service supports the optional parameter '`apiVersion`', which lets you specify the Application Programming Interface (API) version to use to export alerts. Different API versions will result in differently structured Alerts XML files. This feature can be used to customise the exported XML file to meet the requirements of your internal systems.

If no '`apiVersion`' is specified, the exported XML file will use the default API version, that is, API version 9.

The XML file structure for API version 2 is described in Alerts XML structure (on page 12). This following sections describe the differences in the exported Alert XML file for each API version relative to API version 2.

ⓘ Please note that the usage of API version 5 and below is no longer supported.

3.1.1.1 API version 3

- The following child elements have been added to each '`alert`' element:

Element tag	Data type	Description
`alertCount`	Numeric	The alert id for each alert generated.
`startTime`	Time	The time when the triggered alert starts monitoring.

- The following child elements have been added to each '`Sources`' element:

Element tag	Data type	Description
`country`	String	The country where the market is located.
`target`	Sub-element	Details of the target of the Sources.

- The '`target`' element has been broken into the '`primaryTarget`' element and '`secondaryTarget`' element.
- The '`security`' element has been moved from the '`sources`' element into each '`target`' element, and 'securityName' element was added.

Element tag	Data type	Description
securityName	String	The name for the security against which the alert was triggered. There could be more than one security within an alert.

3.1.1.2 API version 4

- Includes all changes added in API version 3.
- The 'participants' element from all sources has been aggregated and moved from each 'source' element to a new element under 'alerts'.
- 'Metahouse', 'Metaaccount' and 'Metatrader' information in cross market alerts has been included under the 'participants' element.

3.1.1.3 API version 5

- Includes all changes added in API versions 3 and 4.
- Child element 'securityCurrency' has been added to 'target' elements.
- Child element 'messages' has been added to 'alert' elements.

Element tag	Data type	Description
messages	Sub-element	Contains details of order and trade IDs captured for the alert
message	Sub-element	Child element of the messages element.
OrderID	Sub-element	Child element of message element. Each element specifies an orderID captured for the alert
TradeID	Sub-element	Child element of message element. Each element specifies a TradeID captured for the alert

3.1.1.4 API version 6

- Includes all changes added in API versions 3 to 5.
- 'securityTypeCode' and 'securityTypeName' have been added to 'target' elements.
- 'alertType' element have been added to 'alerts' element.
- Attachment URLs have been escaped using CDATA escape.

3.1.1.5 API version 7

- Includes all changes present in API versions 3 to 6.
- Property 'filename' has been added to the 'attachmentUrl' element. This contains the file name of the attachment.
- The 'account' element under the 'participants' element has been broken down into two child elements:

Element tag	Data type	Description
`ref`	Sub-element	Account Reference
`type`	Sub-element	Account Type

- The '`messages`' element has been restructured to include all order and trade details under the '`messages`' element, rather than under individual '`orderID`' and '`tradeID`' elements. The message element also includes additional information about the order or trade, for example:

```
<messages>
<message type="order" id="12345" marketCode="asx" oid="12345" securityCode="CIM"/>
<message type="order" id="56789" marketCode="asx" oid="56789" securityCode="CIM"/>
</messages>
```

3.1.1.6 API version 8

- Includes all changes present in API versions 3 to 7.
- Added elements 'attributes' and 'parameters' elements to the 'alert' element. These elements define structured data in alerts[1]. The child elements in these are key/value pairs which are represented in the 'item' tags.

Tag	Parent tag	Child Tag(s)	Attributes
`parameters`	`Alert`	`item`	--
`attributes`	`Alert`	`item`	--
`item`	`Parameters or attributes`	--	• `Key`: Name of field • `Value`: Value of field for alerting scenario

Example:

```
<parameters>
   <item key ="TICK_CHANGE_PERCENT-PAR" value="50%" />
   <item key ="TICK_CHANGE_PERCENT-VAL" value="70%" />
   <item key ="MARKING_CHANGE_MIN-PAR" value="0.4%" />
   <item key ="MARKING_CHANGE_MIN-VAL" value="1.2%" />
</parameters>
<attributes>
   <item key ="TARGET-PARTICIPANT" value="Client Trader TRADER1" />
   <item key ="BUY-OR-SELL" value="Buy" />
   <item key ="BUYSELL-VOLUME-TRADED" value="1200" />
</attributes>
```

Please refer to 'Nasdaq Trade Surveillance - Structured Data Specification - Single Market Alerts' specification for more details. This document is available on request from the client services team.

3.1.1.7 API version 9

- Includes all changes present in API versions 3 to 8.
- Added element 'nontradingdays' to indicate that the requested days are public holidays or on a weekend. The following child element is present in a 'nontradingdays' element:

Element tag	Data type	Description
day	String	Requested day is a public holiday or on a weekend.

The 'day' element has two attributes: 'marketCode' and 'date' for the market's non-trading day.

3.1.1.8 API version 10

- Includes all changes present in API versions 3 to 9.
- Stopped supporting alert XML export using API v5 and below.
- Stopped allowing negative number of days for lookbackDays parameter.
- The default API version is available in the alert export header, even if the apiVersion parameter is not specified in the export request.
- Added element 'supportingFiles' to allow export of custom file attachments in user comments. These custom files can be used to provide supporting evidence and/or extra information during later review of comments. The following table gives a description of each element within each 'supportingFiles' element:

Element tag	Data type	Description
supportingFileUrl	String	The URL location of the custom file containing supporting detail regarding the comment.

3.2 Alerts XML structure

This section describes the structure of an Alert XML file exported from Nasdaq Trade Surveillance.

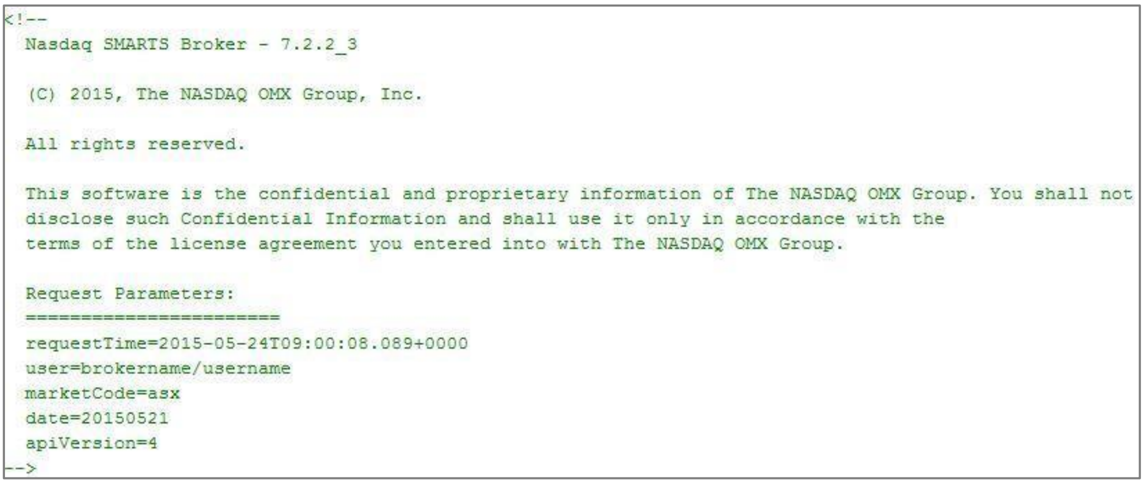
If an API version was specified in the original export request to the server, the structure of the Alerts XML file will vary – see Alerts XML structure and the 'apiVersion' parameter (on page 9) for information about each API version.

3.2.1 File/Document Header

The exported XML files all have a header section that includes:

- The version of Nasdaq Trade Surveillance;
- The Nasdaq Trade Surveillance rights notice;
- The parameters of the request that produced the exported alerts file.

The header looks like the following sample:



```
<!--
  Nasdaq SMARTS Broker - 7.2.2_3

  (C) 2015, The NASDAQ OMX Group, Inc.

  All rights reserved.

  This software is the confidential and proprietary information of The NASDAQ OMX Group. You shall not
  disclose such Confidential Information and shall use it only in accordance with the
  terms of the license agreement you entered into with The NASDAQ OMX Group.

  Request Parameters:
  =======================
  requestTime=2015-05-24T09:00:08.089+0000
  user=brokername/username
  marketCode=asx
  date=20150521
  apiVersion=4
-->
```

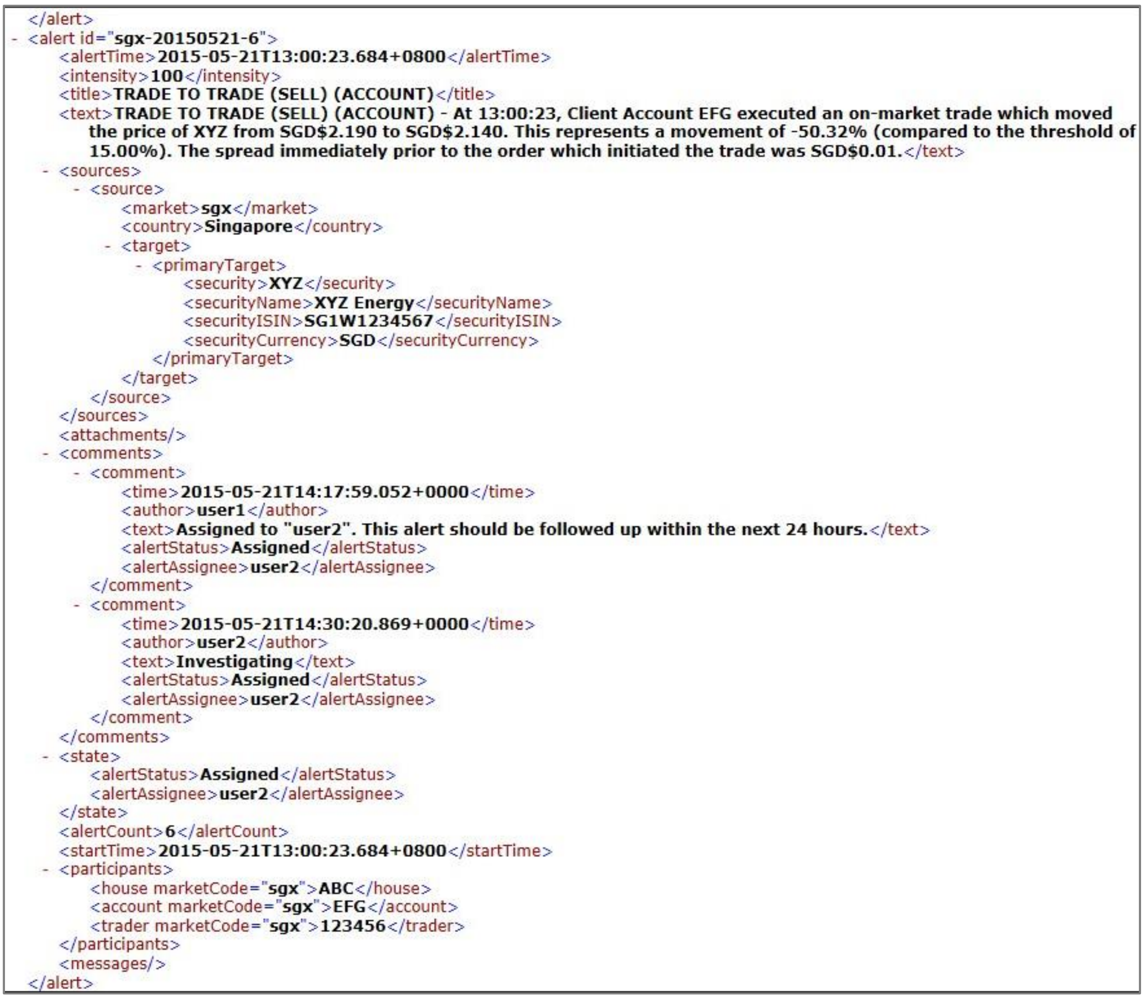
There are a number of parameters described in the document header. The following section gives a description of each parameter within the file header:

- **requestTime**: ISO Date/Time in the format ‘`yyyyMMddTHHmmssmmmm+zzzz`’. This is the time the file export request was executed. Time goes to the millisecond, and includes the timezone qualifier. Note that our time format differs slightly from the ISO 8601 standard, the ISO 8601 Time Zone component has a separator between hours and minutes which we do not use.
- **User**: This is the client from which we request the alerts / the user on the client who is requesting the alerts. The client is usually created with the same name as the broker.

3.2.2 XML Document Body

The body of the XML Alert Export file contains a series of alerts, and within each alert there are a range of data fields relating to that alert.

Below is a sample:



```
   </alert>
 - <alert id="sgx-20150521-6">
      <alertTime>2015-05-21T13:00:23.684+0800</alertTime>
      <intensity>100</intensity>
      <title>TRADE TO TRADE (SELL) (ACCOUNT)</title>
      <text>TRADE TO TRADE (SELL) (ACCOUNT) - At 13:00:23, Client Account EFG executed an on-market trade which moved
         the price of XYZ from SGD$2.190 to SGD$2.140. This represents a movement of -50.32% (compared to the threshold of
         15.00%). The spread immediately prior to the order which initiated the trade was SGD$0.01.</text>
    - <sources>
       - <source>
            <market>sgx</market>
            <country>Singapore</country>
          - <target>
             - <primaryTarget>
                  <security>XYZ</security>
                  <securityName>XYZ Energy</securityName>
                  <securityISIN>SG1W1234567</securityISIN>
                  <securityCurrency>SGD</securityCurrency>
               </primaryTarget>
            </target>
         </source>
      </sources>
      <attachments/>
    - <comments>
       - <comment>
            <time>2015-05-21T14:17:59.052+0000</time>
            <author>user1</author>
            <text>Assigned to "user2". This alert should be followed up within the next 24 hours.</text>
            <alertStatus>Assigned</alertStatus>
            <alertAssignee>user2</alertAssignee>
         </comment>
       - <comment>
            <time>2015-05-21T14:30:20.869+0000</time>
            <author>user2</author>
            <text>Investigating</text>
            <alertStatus>Assigned</alertStatus>
            <alertAssignee>user2</alertAssignee>
         </comment>
      </comments>
    - <state>
         <alertStatus>Assigned</alertStatus>
         <alertAssignee>user2</alertAssignee>
      </state>
      <alertCount>6</alertCount>
      <startTime>2015-05-21T13:00:23.684+0800</startTime>
    - <participants>
         <house marketCode="sgx">ABC</house>
         <account marketCode="sgx">EFG</account>
         <trader marketCode="sgx">123456</trader>
      </participants>
      <messages/>
   </alert>
```

3.2.2.1 ‘alert’ element

Each alert has an identifier, shown as a property of that alert element. This identifier is used for internal purposes only. The following table gives a description of each element within each alert element.

Table 1 - Elements in the 'alert' element

Element tag	Data type	Description
`alertTime`	ISO Date/Time yyyyMMddTHHmmssm mmm+zzzz	Time the file alert was triggered. Time goes to the millisecond, and includes the time zone qualifier. NB: alertTime is interpreted relative to the market's time zone (see above notes on requestTime and date).
`intensity`	Integer	The intensity value assigned to the alert
`title`	String	The title of the alert.
`text`	String	The description of the alert.

Element tag	Data type	Description
		Text is localized to the market's time zone (so times should appear in local market time) and the locale of the server from which it is requested. Alerts for LSE/20080505 will show dates in UTC, alerts for ASX/20080505 will show dates in AEST (with daylight savings offsets added as required). Alert text will differ depending on the locale set on the server. So requesting alerts for MSE (Milan)/20080505 from a server located in Paris will show render one thousand dollars as "1.000". Requesting the same alert from a server in Australia will render "1,000".
`sources`	Sub-element	Details of the source of the alert – see Table 2 on page 16 for data element details
`attachments`	Sub-element	Details of the any attachments to the alert – see Table 4 on page 16 for data element details
`comments`	Sub-element	Details of any comments put on the alert by the broker account that exported the file – see Table 5 on page 17 for data element details.
`state`	Sub-element	Details of the current state of the alert – see Table 6 on page 17 for data element details.
`participants`	Sub-element	Includes details of the participants in the alert, potentially including house, account and/or trader. There could be more than one participant within an alert. The participants sub-element is available under the alert element from API v4. On v2 and 3 this is under the sources sub element. See Table 3 on page 16 for data element details.
`messages`	Sub-element	Contains details of the orders and trades considered for the alert. Available from API v5.
`alertType`	String	Type of Alert. Only available from API v6.
`parameters`	Sub-element	Structured Data parameters. Only available from API V8
`attributes`	Sub-element	Structured Data attributes. Only available from API V8

3.2.2.2 'sources' element

There are potentially multiple sources elements within each alert, and a number of elements within each sources element. The following table gives a description of each element within each sources element.

Table 2 - Elements in the 'sources' element

Element tag	Data type	Description
market	String	The code for the market from which the alert belongs.
security	String	The code for the security against which the alert was triggered. There could be more than one security within an alert.
securityCurrency	String	The currency the security was traded on. Only available from API v5.
securityTypeCode	String	Security type code. Only available on API v6 and above.
securityTypeName	String	Qualified name of security type. Only available on API v6 and above.
participants	Sub-element	Includes details of the participants in the alert, potentially including house, account and/or trader. There could be more than one participant within an alert. The participants sub-element is available under the sources element on API v2 and 3 only. This is under the alert sub element from API v4.

3.2.2.3 'participants' element

There are potentially multiple participants elements within each alert, and a number of elements within each participants element. The following table gives a description of each element within each participants element.

Table 3 - Elements in the 'participants' element

Element tag	Data type	Description
house	String	House code for house participants.
account	String	Account code for account participants. From API version 7, the account element also contains account type, if available. See Section 3.1.1.5**Error! Reference source not found.** for details.
trader	String	Trader code for trading participants.

Note that the participants sub-element is available under the sources element on API v2 and 3 but is under the alert sub element from API v4.1.

3.2.2.4 'attachments' element

There are either no attachments or one attachment element within each alert. These attachments are separate text files that contain detail of interest to the alerts. The following table gives a description of each element within each attachment element.

Table 4 - Elements in the 'attachments' element

Element tag	Data type	Description
`attachmentUrl`	String	The URL location of the file containing more detail regarding the alert.

Note that from API v6, XML escapes attachment URLs using CDATA escape. So if validating XMLs against the XSD please use API v6 or above.

3.2.2.5 'comments' element

There are potentially multiple comment elements within each alert, and a number of elements within each comments element. The comments are details entered by the user within the alerts management area of Nasdaq Trade Surveillance. An "alert comment" is also generated each time `alertStatus` and/or `alertAssignee` changes. These comments will have no alert text. The author for these comments is the user who instigated the status/assignee change. The following table gives a description of each element within each comments element.

Table 5 - Elements in the 'comments' element

Element tag	Data type	Description
`time`	ISO Date/Time yyyyMMddTHHmmssmmmm+zzzz	Time the file comment was created (UTC). Time goes to the millisecond, and includes the time zone qualifier (see above notes on requestTime and date).
`author`	String	The user account that authored the comment.
`text`	String	The detail of the comment.
`alertStatus`	String	The status of the alert that was selected when the comment was made.
`alertAssignee`	String	The user account to which the alert was assigned when the comment was made.

3.2.2.6 'state' element

Every alert must have a state which carries the current status of the alert along with the current assignee. The following table describes each element.

Table 6 - Elements in the 'state' element

Element tag	Data type	Description
`alertStatus`	String	The current status of the alert.
`alertAssignee`	String	The user account to which the alert was currently assigned.

3.2.2.7 ‘parameters’ element

Some alerts[2] will contain structured data. The parameters element will be included in these alerts, and will contain the parameter related structured data for the alert. The following table describes details of the ‘parameters’ element

Table 7 - Elements in the 'parameters' element

Element tag	Data type	Description
`item`	Sub-element	Contains the key value pairs of each structured data element.

3.2.2.8 ‘attributes’ element

Some alerts[2] will contain structured data. The attributes element will be included in these alerts, and will contain the structured data attributes for the alert. The following table describes details of the ‘attributes’ element

Table 8 - Elements in the 'parameters' element

Element tag	Data type	Description
`item`	Sub-element	Contains the key value pairs of each structured data element.

3.2.2.9 ‘item’ element

‘item’ elements will come under both ‘parameters’ and ‘attributes’ elements on alerts publishing structured data. The following table describes details of the ‘attributes’ element

Table 9 - Elements in the 'parameters' element

Element tag	Data type	Description
`Key`	attribute	Name of Structured data field.
`Value`	attribute	Value of structured data field for alerting scenario.

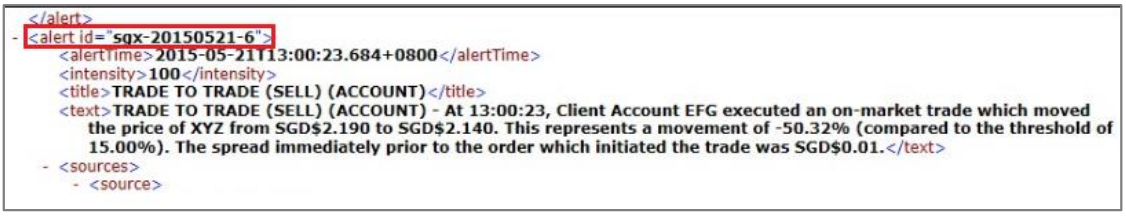
Please refer to section 3.1.1.6 for an example including the ‘parameter’, ‘attributes’ and ‘item’ elements.

3.3 Alert Deep-linking Functionality

The Deep-linking function allows the user to view and load one or a selection of XML exported alerts into the Nasdaq front-end for viewing. Similar to the alert XML export feature used to export the alerts, the deep-linking feature also makes use of an internal web service to function.

The alert deep-linking is facilitated through the ‘`alert id`’ element, as specified within the alert XML output.

[2] Please refer to Nasdaq Trade Surveillance - Structured Data Specification - Single Market Alerts specification for more details. This document is available on request from the client services team.



```
  </alert>
- <alert id="sgx-20150521-6">
    <alertTime>2015-05-21T13:00:23.684+0800</alertTime>
    <intensity>100</intensity>
    <title>TRADE TO TRADE (SELL) (ACCOUNT)</title>
    <text>TRADE TO TRADE (SELL) (ACCOUNT) - At 13:00:23, Client Account EFG executed an on-market trade which moved
      the price of XYZ from SGD$2.190 to SGD$2.140. This represents a movement of -50.32% (compared to the threshold of
      15.00%). The spread immediately prior to the order which initiated the trade was SGD$0.01.</text>
  - <sources>
    - <source>
```

Authenticated users with sufficient permissions can use the deep-linking functionality to load exported alerts back into the NTS front-end by entering the location of the below service with the additional '`alertId`'. Note that before the below linking service can be used, the user must have an already open Nasdaq window via the Java Applet or Java WebStart.

The Alert Deep-linking Functionality can be used via the following link -

```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/selectAlert?alertId=<AlertId>
https://<yourorganisation>.smartsbroker.com/cmss/citadel/selectAlert?alertId=asx-20160114-1
```

Once available alert ID(s) have been entered through the above service, the alert(s) will be opened within the NTS front-end in the Alert History plugin.

The alert deep-linking functionality can also be used to view multiple alerts from various markets through the use of multiple *AlertIds* In order to implement this, each *AlertId* must be entered separated by a comma. As per below -

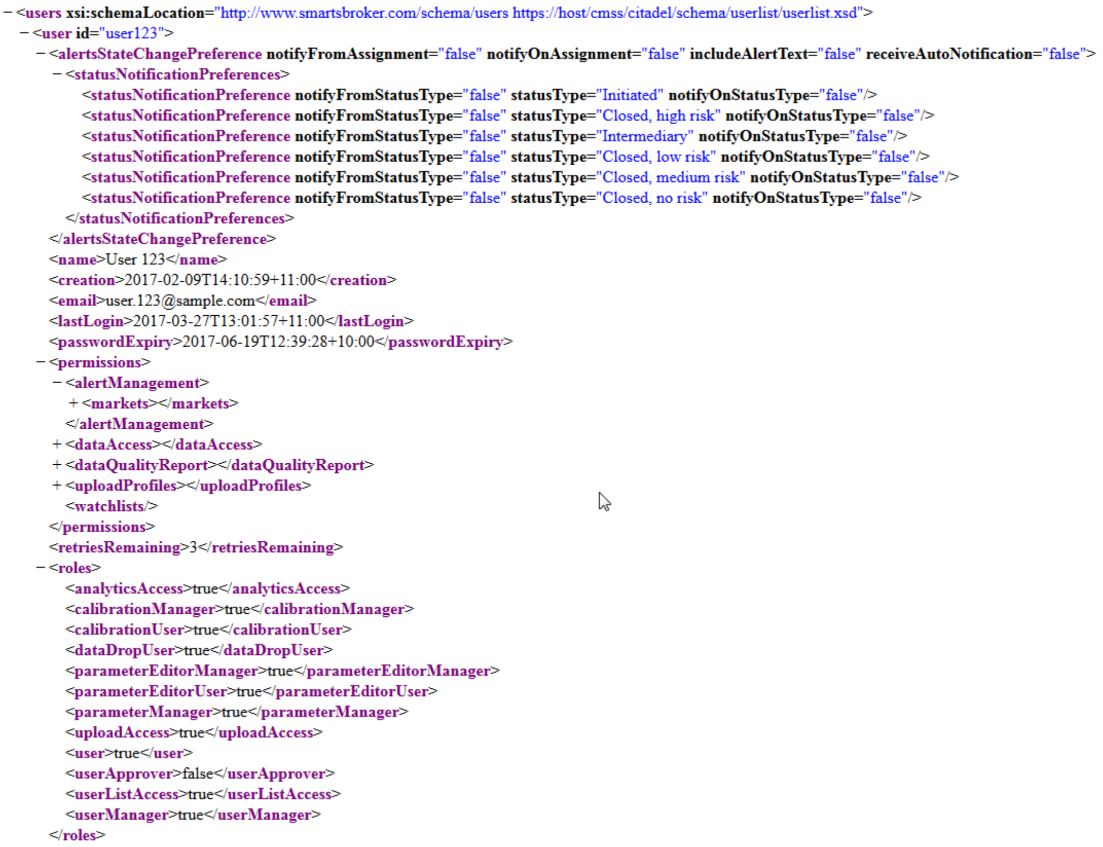
```
https://<yourorganisation>.smartsbroker.com/cmss/citadel/selectAlert?alertId=asx-20160114-
1,nzx-20100302-1
```

4 User List XML

The file exported from Nasdaq Trade Surveillance allows authenticated users with User List permissions to download the User List PDF report in XML format.

The report generated is useful for meeting auditor requirements regularly confirming active users for security reasons and contains a range of data fields relating to the users details including their permissions.

Below is a sample:



```
− <users xsi:schemaLocation="http://www.smartsbroker.com/schema/users https://host/cmss/citadel/schema/userlist/userlist.xsd">
  − <user id="user123">
    − <alertsStateChangePreference notifyFromAssignment="false" notifyOnAssignment="false" includeAlertText="false" receiveAutoNotification="false">
      − <statusNotificationPreferences>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Initiated" notifyOnStatusType="false"/>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Closed, high risk" notifyOnStatusType="false"/>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Intermediary" notifyOnStatusType="false"/>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Closed, low risk" notifyOnStatusType="false"/>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Closed, medium risk" notifyOnStatusType="false"/>
          <statusNotificationPreference notifyFromStatusType="false" statusType="Closed, no risk" notifyOnStatusType="false"/>
        </statusNotificationPreferences>
      </alertsStateChangePreference>
      <name>User 123</name>
      <creation>2017-02-09T14:10:59+11:00</creation>
      <email>user.123@sample.com</email>
      <lastLogin>2017-03-27T13:01:57+11:00</lastLogin>
      <passwordExpiry>2017-06-19T12:39:28+10:00</passwordExpiry>
    − <permissions>
      − <alertManagement>
        + <markets></markets>
        </alertManagement>
      + <dataAccess></dataAccess>
      + <dataQualityReport></dataQualityReport>
      + <uploadProfiles></uploadProfiles>
        <watchlists/>
      </permissions>
      <retriesRemaining>3</retriesRemaining>
    − <roles>
        <analyticsAccess>true</analyticsAccess>
        <calibrationManager>true</calibrationManager>
        <calibrationUser>true</calibrationUser>
        <dataDropUser>true</dataDropUser>
        <parameterEditorManager>true</parameterEditorManager>
        <parameterEditorUser>true</parameterEditorUser>
        <parameterManager>true</parameterManager>
        <uploadAccess>true</uploadAccess>
        <user>true</user>
        <userApprover>false</userApprover>
        <userListAccess>true</userListAccess>
        <userManager>true</userManager>
      </roles>
```

4.1.1 ‘user’ element

The following table gives a description of each element within each user element.

Table 10 - Elements in each user element

Element tag	Data type	Description
`analyticsAccess`	boolean	Permission to access the Analytics module of NTS per the permissioned market. Available values are TRUE or FALSE. TRUE indicates the user has this access, FALSE indicates the user is not permitted to this module.
`name`	String	Name of the user.

Element tag	Data type	Description
creation	ISO Date/Time yyyyMMddTHH mmssmmmm+z zzz	The time the user was created.
email	String	Email address of the user.
lastLogin	ISO Date/Time yyyyMMddTHH mmssmmmm+z zzz	The last time a user logged on to NTS system.
parameterManager	boolean	This is a non-system based authority. An authorized person to approve/submit parameter change request. Available values are TRUE or FALSE.
permissions	Sub-element	Types of permissions the user has – see Table 8 below for permissions details.
roles	Sub-element	Types of additional roles the user has – see Table 12 below for Roles details
retriesRemaining	Integer	The number of password attempts remains.
status	String	The status of the user's account and password
twoFactorAuthenticationEnabled	boolean	Whether Two Factor Authentication is enabled for the user. Available values are TRUE or FALSE.

4.1.2 'permissions' element

There are multiple types of user permissions within permissions element. The following table gives a description of each type of permission.

Table 11 - Types of user permissions

Element tag	Data type	Description
alertManagement	Sub-element	Permission to the change of Alert Status and posting comments on alert –see Table 9 below for data element details.
dataAccess	Sub-element	Permission to access trading data on markets – see Table 9 below for data element details.

Element tag	Data type	Description
`dataQualityReport`	Sub-element	Permission to access data quality report for markets – see Table 9 below for data element details.
`uploadProfiles`	Sub-element	Permission to upload data to the NTS application – see Table 10 below for data element details.
`watchlists`	Sub-element	Permission to edit the Watch List on the NTS application per the permissioned market, this feature is only applicable if the Watch List Alerts are setup and enable on that market –see Table 9 below for data element details.

4.1.3 'roles' elements

There are multiple types of roles within roles elements. The following table gives a description of each type of role.

Table 12 - Types of user roles

Element tag	Data type	Description
`analyticsAccess`	Sub-element	Applicable only for clients with Analytics function set up. Permission to access the Analytics module. Available values are TRUE or FALSE.
`calibrationManager`	Sub-element	Applicable only for clients with Calibration function set up. Permission to access the Calibration module and submit the parameter change request to NTS for deployment. Available values are TRUE or FALSE.
`calibrationUser`	Sub-element	Applicable only for clients with Calibration function set up. Permission to access the Calibration module. Available values are TRUE or FALSE.
`dataDropUser`	Sub-element	Permission to access Data Drop module on the permissioned markets. Available values are TRUE or FALSE.
`parameterEditorManager`	Sub-element	Permission to access Parameter Editor module. The manager can review, create and submit parameter change requests. Available values are TRUE or FALSE.

Element tag	Data type	Description
parameterEditorUser	Sub-element	Permission to access Parameter Editor module. The user can view submitted parameter change requests. Available values are TRUE or FALSE.
parameterManager		Permission to approve the parameter change requests submitted to NTS and to view and download the Parameter Audit Log Available values are TRUE or FALSE.
uploadAccess		Permission to upload input files to NTS and view the list of files uploaded. Available values are TRUE or FALSE.
user		Basic user permissions to access NTS application. Available values are TRUE or FALSE.
userApprover		Enhanced permission where four eyes protocol is applied. Available values are TRUE or FALSE.
userListAccess	boolean	Permission to download the User List in PDF or XML format. Available values are TRUE or FALSE.
userManager	boolean	Permission for the User Management feature (adding, updating or removing of standard Users) on the application front end. Also the authorized person to approve/submit request to update User Managers and Parameter Managers.

4.1.4 ‘markets’ and ‘metamarkets’ elements

alertManagement, dataAccess and watchlists are permitted per market basis and alphabetically categorized into markets (single market) and metamarkets (cross market). The following table gives a description of each element within each markets and metamarkets element.

Table 12 - Elements in each ‘markets’ and ‘metamarkets’ elements

Element tag	Data type	Description
market	String	The code for the single market which the user is permitted access to.
metamarket	String	The code for the cross market which the user is permitted access to.

4.1.5 ‘uploadProfiles’ element

uploadProfiles provides information on the upload configuration allowed for the user.

Table 13 - Elements in each ‘uploadProfiles’ element

Element tag	Data type	Description
`uploadProfile`	String	Upload config name. It is generally named <market_XX> where "XX" starts at "01" and increments for subsequent config names.

5 Glossary and Terminology

All products come with a number of terms that make-up the "language" of the product and they are covered in this section. Note that the below glossary contains terms in particular to this specification. Additional, more global, terms are to be found in the Nasdaq Trade Surveillance Glossary of Terms document.

Term	Description
Alerts	Events fired by the system that represent a pattern of possible irregular trading activity.
User List	The User List provides details on every user in the system including their permissions.
CSV	Comma Separated Values - a file format used by, for example, Microsoft(R) Excel and MS-Access. It is the format used by the Nasdaq Trade Surveillance applet to export alert data.
URL	Abbreviation of Uniform Resource Locator, the global address of documents and other resources on the World Wide Web.
XML	Short for Extensible Markup Language, a specification developed by the W3C. XML is a standard data format which allows designers to create their own customized tags, enabling the definition, transmission, validation, and interpretation of data between applications and between organizations.

Table 14 - Glossary of terms used in this document.

6 Troubleshooting and Support

6.1 Contact Nasdaq

Nasdaq Trade Surveillance users can report incidents to the Nasdaq Trade Surveillance support email at smbc.cs@nasdaq.com.

6.2 FAQs

In this section you will find answers to commonly asked troubleshooting questions. If you are having trouble with Nasdaq Trade Surveillance, please find the answer in this section first. If you cannot find an answer in this section, you should then contact us via the email address above.

6.2.1 Error – Bad Request

If the URL request does not bring back the requested file, and an error message "Your browser (or proxy) sent a request that this server does not understand" is received, you are trying to export a file that does not exist. This may be due to an incorrectly entered URL, an incorrect market name, or possibly a market the user does not have access to.

6.2.2 The XML File Contains a Header but No Alerts

If the XML file is delivered with a header, and the last line is an <alerts> tag, but there are no alerts details below that tag, it is likely that there are no alerts on the date nominated.

6.2.3 Crossmarket Alerts included in Submarket Alerts Export

When a submarket code (market belonging to a crossmarket) is specified in the "marketCode" query parameter, you may also be able to view the corresponding metamarket alerts. This is a default behaviour as all the alerts with primary target of that market will be included. However this behaviour can be switched off upon request.