NSI Data Specification Guide

By PJM Interconnection

Authors

Bhushan Dhuri

Contents

Revision History

Date	Version	Description	Author
09/30/2019	0.1	Initial Draft	Bhushan Dhuri

1. Introduction

1.1. Purpose

The purpose of this document is to describe the architecture and design of the services between two *Balancing Authority* (BA) systems required to exchange Net Scheduled Interchange (NSI) data for the purpose of automating NSI checkout. Every neighboring BA could use this NSI data in their system to compare against NSI calculated by their internal interchange software. BAs have liberty to include additional data elements in their data exchange process for their bilateral purposes.

1.2. Scope

The scope of this document is limited to the proposed architecture and design of the data transfers required for automated checkout of NSI between two BAs. This document is drafted by the PJM Transmission Service Department with the intention to provide specifications for the development and delivery of all required data elements interacting between two BAs in support of the automation. While this document does specify details on the known required data transfers and individual attributes involved, these transfers and attributes may later require alteration during joint design reviews at NAESB and/or throughout the course of development of data specification.

1.3. Acronyms

Term	Meaning
NSI	Net Scheduled Interchange
BA	Balancing Authority
UTC	Coordinated Universal Time
NSI	Net Scheduled Interchange
ATF	After the fact
BTF	Before the fact
RT	Real Time
CCI	Composite Confirmed Interchange

1.4. Regulatory References

NERC INT-009-2.1 Requirement and Measure

R1. *Each Balancing Authority shall agree with each of its Adjacent Balancing Authorities that its Composite Confirmed Interchange with that Adjacent Balancing Authority, at mutually agreed upon time intervals, excluding Dynamic Schedules and Pseudo-Ties and including any Interchange per INT-010-2 not yet captured in the Composite Confirmed Interchange, is:*
1.1. *Identical in magnitude to that of the Adjacent Balancing Authority, and*

1.2. *Opposite in sign or direction to that of the Adjacent Balancing Authority.*

M1. *The Balancing Authority shall have evidence (such as dated logs, voice recordings, electronic records, or other evidence) that its Composite Confirmed Interchange, excluding Dynamic Schedules and Pseudo-Ties and including any Interchange as directed per INT-010-2 not yet captured in the Composite Confirmed*

Interchange, was agreed to by each Adjacent Balancing Authority, identical in magnitude to those of each Adjacent Balancing Authority, and opposite in sign to that of each Adjacent Balancing Authority. (R1)

2. Data Conventions

2.1. Date Format Convention

All interactions use a time interval. The data type of `dateTime` is used to specify a date and a time. The `startDate` and `endDate` fields are specified in ISO format, which is defined as "YYYY-MM-DDThh:mm:ss(Z or ±-hh:mm)" where:

Field	Description
YYYY	Indicates the year
MM	Indicates the month
DD	Indicates the date
T	Indicates the start of the required time section
hh	Indicates the hour
mm	Indicates the minute
ss	Indicates the second
Z or ± hh:mm	UTC 'Zulu' time or timezone offset

Example A:

2019-09-30T16:00:00.000Z

Example B:

2019-09-30T12:00:00-04:00

2.2. Creator BA field

The `creatorBA` field should be filled in with the registered Entity Code of the BA who is responding to the data request and creating a response payload.

2.3. Requestor BA field

The `requestorBA` field is used to specify the list of neighboring BAs for whom the payload is created. This field can be derived from the values from '`area`' parameter.

This is an optional, unbounded field. It is suggested to fill this field with the registered Entity Code of the requestor BA. The creator of payload would determine who the requestor is when request passes authentication. If requestor is authorized to receive NSI for more than one area, then this element will be repeated with multiple values.

2.4. Checkout BA field

The `checkoutBA` field is nested under complexType of `NsiTotals` and `DailyNsiTotals`. Both complexTypes are unbounded so they can be used for sending NSI to multiple areas within a single payload.

The requestor BA may send a comma separated list with multiple areas' Entity Code. When responding to a request with multiple areas, the responder should use **one** `checkoutBA` for each BA `NsiTotal`.

Example:

Example of the response payload to the request with comma separated list of `checkoutBA` field.

```
<NsiTotals>
  <NsiTotal>
    < checkoutBA>CPLE</ checkoutBA>
    <NsiIntervals>...</NsiIntervals>
    <IntegratedIntervals>...</IntegratedIntervals>
  </NsiTotal>
  <NsiTotal>
    <checkoutBA>CPLW</checkoutBA>
    <NsiIntervals>...</NsiIntervals>
    <IntegratedIntervals>...</IntegratedIntervals>
  </NsiTotal>
</NsiTotals>
```

2.5. Sink BA field

The `sinkBA` field is used in combination with the `creatorBA` field to show the direction of net energy flow. As a creator BA, when calculating NSI for an adjacent BA, it is important to isolate the tag's path to only two BAs: creator BA and the adjacent requestor BA.

Example:

#	Scenario	requestorBA	creatorBA	sinkBA	mwNet
1	For net energy flow of 200 MW from BA1 to BA2	BA2	BA1	BA2	200
2	For net energy flow of 500 MW from BA2 to BA1	BA2	BA1	BA1	500

3. Data Transfers Overview

3.1. Request-Response

Description:

A request-response method is recommended for this data exchange, but BAs have the liberty to choose a data transfer method of their choice. Request–response is a message exchange pattern where a requestor sends a request message to a replier system that receives and processes the request, and ultimately returning a message in response. This is a simple, but powerful messaging pattern that allows two applications to have a two-way conversation with each another over a channel. This pattern is especially common in client–server architectures.

For simplicity, this pattern is typically implemented in a purely synchronous fashion, such as in web service calls over HTTP, which holds a connection open and waits until the response is delivered or the timeout period expires.

Example:

A requesting BA initiates the request to retrieve data *from* neighboring BA systems. The requesting BA is making a GET request via the RESTful service passing in the start and stop times in UTC format.

The neighboring BA system has an exposed RESTful service that allows for retrieval of NSI data based on the start and stop times within the service request. The neighbor BA returns NSI data for the interval(s) requested in XML format. This NSI data contains every eTag within the requesting BA's area spanning between the start and stop times defined within the request call. The creator BA does not check if the published data is new, old, or delta. If eTags are not available within the creator BA's system to contribute towards NSI for the requested interval, an empty list is returned to the requestor.

4. Building a Request

4.1. Request Parameters

Description:

Data requests are submitted with inclusive start & stop time windows. The requestor must specify the request type. All other optional data are treated as false and not included if not requested.

Data Elements:

Request Parameter	Required	Sample Value	Description
start	Yes	201706060500	This is beginning of the time inclusive request window, specified with a date and time. This means if Requestor needs scheduled NSI starting from 1300, they will supply 1300 as start time.
stop	Yes	201706060500	This is the end of the inclusive request window, specified with a date and time. It is expected NSI is received for the 15 minute valid scheduling interval before this data-time as long as it's after the Start time parameter.
area	Yes	MISO, CPLE	This lets the requestor to specify the list of areas for which to request NSI totals. (*i.e., If a BA need to request NSI totals for DUK, CPLE & CPLW, then they can request multiple areas within the URL separated by comma.*)
type	Yes	DAY or RT	Provide a type of NSI you need. (e.g., RT, DAY). Details of what you get for each request type are specified under building payload section.
tag	No	t or f	Provides option to include or exclude underlying eTags. If you mark this as true you will get all the underlying tags that build the NSI during start and stop window you provided. More details are in 'Optional eTag Data' section.
integrated	No	t or f	Provides options to include hourly integrated totals along with RT/DAY totals.

URL Operators:

The question mark, ampersand, equals sign and comma are operators used in the syntax of query strings/URL variables.

Operator	Sample Value	Description
?	e.g. `?variable=value` e.g. `?start=201706060500`	The question mark identifies the beginning of the query string and must be placed at the end of the link, before the contents of the query string.
&	e.g. `variable=value&variable2=value2` e.g. `tag=t&integrated=f`	The ampersand is used before each subsequent variable/value pair in the query string.
=	e.g. `variable=value` e.g. `tag=t`	The equals sign separates the variable from the value assigned to that variable.
,	e.g. `variable=value1,value2,value3` e.g. `area=DUK,CPLE,CPLW`	The comma is used to separate multiple values for a single variable.

Date format for start & stop:

The parameters <start> and <end> dateTime must be UTC and follow the format YYYYMMDDhhmm, where:

YYYY = four-digit year
MM = two-digit month (01=January, etc.)
DD = two-digit day of month (01 through 31)
hh = two digits of hour (00 through 23) (am/pm NOT allowed)
mm = two digits of minute (00 through 59)

Example URL:

https://nsi.pjm.com/NSI/rest/getnsi?start=201706060500&stop=201706060500&type=RT&tag=t&integrated=f&area=DUK,CPLE,CPLW

5. Building a Payload

5.1. Request Metadata

Description:

Every response payload shall contain data to indicate the time window requested, type of data and other parameter as requested. Even though this data is not used directly in checkout calculation it helps humans who inspect payload for troubleshooting purposes.

Data Elements:

Field Name	Required	Sample Value	Description
requestStartTime	Yes	2020-01-15T19:00:00.000Z	Fill this field with value that came in un URL **start** request parameter.
requestStopTime	Yes	2020-01-15T21:00:00.000Z	Fill this field with value that came in un URL **stop** request.
responseTimestamp	Yes	2020-01-15T18:22:00.000Z	Fill this field with the time when creator BA created this payload.
requestType	Yes	DAY or RT	Fill this field with value that came in un URL **type** request parameter.
includeIntegrated	Yes	true or false	Fill this field with value that came in un URL **integrated** request parameter
includeTag	Yes	true or false	Fill this field with value that came in un URL **Tag** request parameter
creatorBA	Yes	PJM	Refer to section 2.2.
requestorBA	Yes	MISO	Refer to section 2.3.

5.2. Real-Time NSI

Description:

This is the real-time NSI checkout required by NERC INT-009.

Data Elements:

Field Name	Required	Sample Value	Description
checkoutBA	Yes	MISO	Refer to section 2.4.
intervalStartTime	Yes	2015-05-15T21:00:00.000Z	Fill this field with start time of the interval.
intervalStopTime	Yes	2015-05-15T21:15:00.000Z	Fill this field with stop time of the interval.
sinkBA	Yes	PJM	Refer to section 2.2.
mwNet	Yes	2	Refer to section 2.6
verifiedMatch	Yes	true	Refer to section 5.6.

Field Name	Required	Sample Value	Description
overriddenFlag	No	true	Fill this field with true if you have a manual override to the NSI value. Else, false (default).

5.3. Daily NSI

Description:

There are two times when BAs typically request each other's daily NSI totals.

After-the-Fact (ATF) NSI checkout is performed *for the previous day* by most of the BAs to gather reconciled NSI numbers for previous day for Energy Accounting tasks. BAs check the entire day's NSI total, and in the event the daily totals don't match between BAs, each hourly total can be checked for discrepancies.

Before-the-fact (BTF) NSI is performed *for the next day* to give BAs a general idea of what next day schedule trend looks like; as well as preventing errors that may exist prior to Real Time. Some BAs use this along with their next day weather report to get rough picture of interchange curve for upcoming shifts and next day's peak times. Some use it specifically for various look-ahead assessments to commit generation.

Data Elements:

Field Name	Required	Sample Value	Description
checkoutBA	Yes	MISO	Refer to section 2.4.
intervalStartTime	Yes	2015-05-15T04:00:00.000Z	Fill this field with start time of the interval.
intervalStopTime	Yes	2015-05-16T04:00:00.000Z	Fill this field with stop time of the interval.
sinkBA	Yes	PJM	Refer to section 2.5.
mwDaily	Yes	2	Refer to section 2.6
verifiedMatch	Yes	true	Refer to section 5.6.

5.4. Optional Integrated Interval data

Description:

If a request is made for NSI data with integrated data included, the payload shall include all hours contributing to the NSI total provided in payload.

Data Elements:

Field Name	Required	Sample Value	Description
intervalStartTime	Yes	2015-05-15T21:00:00.000Z	UTC
intervalStopTime	Yes	2015-05-15T21:00:00.000Z	UTC
sinkBA	Yes	PJM	The Entity Code of the BA where net energy is flowing into.
mwNetIntegrated	Yes	2	Absolute value of the hourly integrated NSI MW

Field Name	Required	Sample Value	Description
verifiedMatch	Yes	true	Refer to section 5.6.

5.5. Optional eTag data

Description:

If a request is made for NSI data with underlying eTag data included, the payload shall include all eTags contributing to the NSI total provided in payload.

Data Elements:

Field Name	Required	Sample Value	Description
tagIndex	Yes	123456	Fill Tag index value in this field.
tagName	Yes	MISO_CRGL1ABDD01_PJM	Fill Tag name value in this field.
tagTransactionType	Yes	Normal	Normal or Emergency are the only Tag types that are counted towards NSI. Dynamic and Pseudo-tie types are not counted towards NSI. Refer to section 2.6 for more details.
startTime	Yes	2015-05-15T21:00:00.000Z	Refer to profile building section to see how to fill this value.
endTime	Yes	2015-05-15T21:00:00.000Z	Refer to profile building section to see how to fill this value.
mwEnergy	Yes	2	Refer to profile building section to see how to fill this value.
tagUpdateTimestamp	Yes	2015-05-15T21:00:00.000Z	Fill this field with the last time the tag was updated to its current final state. Which means every time when existing tag gets curtailed/adjusted this timestamp should be updated. This helps counter party to know how latest is other party's data per tag. This also could help in programmatic tag discrepancies solutions.

Profile Building for eTag

Single eTag profiles may be large depending on the number of requests created on each eTag. Users may choose to only include time periods that affect the NSI and compact the matching intervals.

Example

If a request parameter is from 2019-08-11T13:00:00.000Z to 2019-08-11T15:00:00.000Z, then this two hour period has 8 'typical' tagging intervals of 15 minutes each. All eTags contributing to these 8 intervals may have various time profiles. Examples of how Tag profile limits should be applied are provided in the table below.

Field Name	Actual Tag profile	Suggested Tag profile for payload
Tag 1	From : 2019-08-10T04:00:00.000Z To : 2019-08-12T04:00:00.000Z MW: 50	From : 2019-08-10T04:00:00.000Z To : 2019-08-12T04:00:00.000Z MW: 50
Tag 2	From : 2019-08-11T13:30:00.000Z To : 2019-08-11T14:30:00.000Z MW: 50	From : 2019-08-11T13:30:00.000Z To : 2019-08-11T14:30:00.000Z MW: 50
Tag 3	From : 2019-08-11T12:00:00.000Z To : 2019-08-11T13:00:00.000Z MW: 50 From : 2019-08-11T13:00:00.000Z To : 2019-08-11T14:00:00.000Z MW: 60 From : 2019-08-11T14:00:00.000Z To : 2019-08-11T15:00:00.000Z MW: 70 From : 2019-08-11T15:00:00.000Z To : 2019-08-11T16:00:00.000Z MW: 80	From : 2019-08-11T13:00:00.000Z To : 2019-08-11T14:00:00.000Z MW: 60 From : 2019-08-11T14:00:00.000Z To : 2019-08-11T15:00:00.000Z MW: 70
Tag 4	From : 2019-08-11T11:30:00.000Z To : 2019-08-11T12:30:00.000Z MW: 50 From : 2019-08-11T12:30:00.000Z To : 2019-08-11T13:30:00.000Z MW: 60 From : 2019-08-11T13:30:00.000Z To : 2019-08-11T14:30:00.000Z MW: 70 From : 2019-08-11T14:30:00.000Z To : 2019-08-11T15:30:00.000Z MW: 80 From : 2019-08-11T15:30:00.000Z To : 2019-08-11T16:30:00.000Z MW: 90	From : 2019-08-11T12:30:00.000Z To : 2019-08-11T13:30:00.000Z MW: 60 From : 2019-08-11T13:30:00.000Z To : 2019-08-11T14:30:00.000Z MW: 70 From : 2019-08-11T14:30:00.000Z To : 2019-08-11T15:30:00.000Z MW: 80

5.6. Three-part communication

Description:

This document provides general guidance on how one can manage the recording of data. (*Local record management may be done in differently than what is suggested within this document.*) This section is to help users understand general principles and logic for successful three-part communication and effective use of `verifiedMatch` flag.

When BAs exchange data, they need to keep records to ensure the requesting party actually received the creator's payload and all numbers are lining up within their respective systems. Incorporating `<verifiedMatch>false</verifiedMatch>` element ensures that BAs perform three-part communication by ensuring neighbors also have a verified record.

State Management:

Below are important pieces of information a BA will need to record and maintain for each interval in order to be able to successfully checkout NSI with the neighbors.

1. **Own NSI** – each BA needs to maintain its own NSI value.
2. **Neighbor NSI** – each BA needs to record its neighbor's latest NSI they received to compare against their own NSI.
3. **Sink BA** – Sink BA shows the direction of the power/interchange flow. Both BAs need to ensure that they both agree upon the direction of the flow. Direction indication could be managed by
    - additional data columns for their own & their neighbor's indication of interchange flow direction, *or*
    - using positive and negative signs while recording the NSI values for their own & their neighbor's NSI
4. **Own Verified Flag** – BA needs to maintain its own verified flag. BA will change this flag to 'true' when own NSI and the neighbor's NSI are exactly the same and the direction of the net energy flow is also the same.
5. **Neighbor Verified Flag** – BA needs to record and maintain neighbor's `verifiedMatch` flag. This is the final piece that confirms that neighbor has also pulled the NSI data and has verified that NSI sent to them is lining up in their records.

Example:

The following is a sequence of events where everything correctly aligns. This example shows how data would look within a two BAs' systems where BA1 is the requestor BA exchanging data with a neighbor (BA2), who is the creator BA.

1. Within each BA's system, both maintain a Verified Flag for themselves and their neighbor. When there is no Tag data, each NSI column shall have *null* for NSI and each Verified Flag shall be set to 'F'.

   BA1's System

Event	BA1 NSI	Neighbor NSI	BA1 Verified Flag	Neighbor Verified Flag
0	Null	Null	F	F

   BA2's System

Event	BA2 NSI	Neighbor NSI	BA2 Verified Flag	Neighbor Verified Flag
0	Null	Null	F	F

2. When a confirmed Tag comes into BA1's system, and they have not yet requested its neighbor's data, BA1 will record its own NSI value and their own Verified Flag remains set to 'F'.

   BA1's System

Event	BA1 NSI	Neighbor NSI	BA1 Verified Flag	Neighbor Verified Flag
1	100	Null	F	F

   Likewise, the confirmed Tag comes into the BA2's system. They, too, record their own NSI value leaving their own Verified Flag set to 'F'.

   BA2's System

Event	BA2 NSI	Neighbor NSI	BA2 Verified Flag	Neighbor Verified Flag
1	100	Null	F	F

3. At a designated time, BA1 initiates a request (*BA1 is the 'requestor'*) from the neighbor BA (BA2) specified on the Tag. The neighbor then sends a response (*BA2 is the 'creator'*) back to BA1 where it is received.

   - If the NSI data between BA1 and the neighbor match, the BA1 Verified Flag is set to 'T'.
   - If the NSI data does not match, the BA1 Verified Flag remains set to 'F'.

BA1's System

Event	BA1 NSI	Neighbor NSI	BA1 Verified Flag	Neighbor Verified Flag
2	100	100	T	F

At this point the BA2 has not requested BA1's data yet. Therefore, BA2's data remains unchanged within their system.

BA2's System

Event	BA2 NSI	Neighbor NSI	BA2 Verified Flag	Neighbor Verified Flag
2	100	Null	F	F

4. At a designated time, the BA2 initiates a request (*BA2 is the 'requestor'*) to BA1. BA1 then sends a response (*BA1 is the 'creator'*) back to BA2 where it is received. The response payload contains the BA1 Verified Flag value by setting the `verifiedMatch` element as "true".

BA1's System

Event	BA1 NSI	Neighbor NSI	BA1 Verified Flag	Neighbor Verified Flag
3	100	100	T	F

BA2's System

Event	BA2 NSI	Neighbor NSI	BA2 Verified Flag	Neighbor Verified Flag
3	100	100	T	T

5. BA1 initiates another request from the BA2, who then sends a response with the `verifiedMatch` element set to "true". BA1 shall then set the Neighbor Verified Flag to 'T'.

BA1's System

Event	BA1 NSI	Neighbor NSI	BA1 Verified Flag	Neighbor Verified Flag
4	100	100	T	T

BA2's System

Event	BA2 NSI	Neighbor NSI	BA2 Verified Flag	Neighbor Verified Flag
4	100	100	T	T

At this point, each BA shall have both Verified Flags (own & neighbor) set to 'T', that indicates that both parties have independently verified the NSI totals, and this state can be considered as 'checked out'.

6. XSD Definitions

6.1. Schema

```
<?xml version="1.0" encoding="utf-8"?>
<!-- edited with XMLSpy v2020 sp1 (x64) (http://www.altova.com) by PJM (PJM INTERCONNECTION, LLC) -->
<xs:schema xmlns:nsi="http://www.pjm.com/external/schemas/nsi/v1"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.pjm.com/external/schemas/nsi/v1">
  <xs:element name="NsiCheckout" type="nsi:NsiCheckout"/>
  <xs:complexType name="NsiCheckout">
    <xs:sequence>
      <xs:element name="requestStartTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="requestStopTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="responseTimestamp" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="requestType" type="nsi:RequestType" minOccurs="1" maxOccurs="1"/>
      <xs:element name="includeIntegrated" type="xs:boolean" default="false" minOccurs="1"
maxOccurs="1"/>
      <xs:element name="includeTag" type="xs:boolean" minOccurs="1" maxOccurs="1"/>
      <xs:element name="creatorBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="RequestorBAs" type="nsi:RequestorBAs" minOccurs="0" maxOccurs="1"/>
      <xs:choice minOccurs="1" maxOccurs="1">
        <xs:element ref="nsi:NsiTotals" minOccurs="1" maxOccurs="1"/>
        <xs:element ref="nsi:DailyNsiTotals" minOccurs="1" maxOccurs="1"/>
      </xs:choice>
      <xs:element name="RealTimeEnergyTransactions" type="nsi:RealTimeEnergyTransactions" minOccurs="0"
maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:simpleType name="RequestType">
    <xs:restriction base="xs:string">
      <xs:enumeration value="DAY"/>
      <xs:enumeration value="RT"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:element name="NsiTotals" type="nsi:NsiTotals"/>
  <xs:complexType name="NsiTotals">
    <xs:sequence>
      <xs:element name="NsiTotal" type="nsi:NsiTotal" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="RealTimeEnergyTransactions" type="nsi:RealTimeEnergyTransactions"/>
  <xs:complexType name="RealTimeEnergyTransactions">
    <xs:sequence>
      <xs:element name="RealTimeEnergyTransaction" type="nsi:RealTimeEnergyTransaction" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="DailyNsiTotals" type="nsi:DailyNsiTotals"/>
  <xs:complexType name="DailyNsiTotals">
    <xs:sequence>
      <xs:element name="DailyNsiTotal" type="nsi:DailyNsiTotal" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="NsiIntervals" type="nsi:NsiIntervals"/>
  <xs:complexType name="NsiIntervals">
    <xs:sequence>
      <xs:element name="NsiInterval" type="nsi:NsiInterval" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="NsiInterval" type="nsi:NsiInterval"/>
  <xs:complexType name="NsiInterval">
    <xs:sequence>
      <xs:element name="intervalStartTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="intervalStopTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="sinkBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mwNet" type="xs:integer" minOccurs="1" maxOccurs="1"/>
```

```
      <xs:element name="verifiedMatch" type="xs:boolean" default="false" minOccurs="1" maxOccurs="1"/>
      <xs:element name="overriddenFlag" type="xs:boolean" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="DailyNsiTotal" type="nsi:DailyNsiTotal"/>
  <xs:complexType name="DailyNsiTotal">
    <xs:sequence>
      <xs:element name="checkoutBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="nsi:DailyNsiIntervals" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="nsi:IntegratedIntervals" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="DailyNsiIntervals" type="nsi:DailyNsiIntervals"/>
  <xs:complexType name="DailyNsiIntervals">
    <xs:sequence>
      <xs:element name="DailyNsiInterval" type="nsi:DailyNsiInterval" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="RealTimeEnergyTransaction" type="nsi:RealTimeEnergyTransaction"/>
  <xs:complexType name="RealTimeEnergyTransaction">
    <xs:sequence>
      <xs:element name="tagIndex" type="xs:integer" minOccurs="1" maxOccurs="1"/>
      <xs:element name="tagName" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="tagTransactionType" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="tagUpdateTimestamp" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="nsi:Profiles" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Profiles" type="nsi:Profiles"/>
  <xs:complexType name="Profiles">
    <xs:sequence>
      <xs:element name="Profile" type="nsi:Profile" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="DailyNsiInterval" type="nsi:DailyNsiInterval"/>
  <xs:complexType name="DailyNsiInterval">
    <xs:sequence>
      <xs:element name="intervalStartTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="intervalStopTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="sinkBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mwDaily" type="xs:integer" minOccurs="1" maxOccurs="1"/>
      <xs:element name="verifiedMatch" type="xs:boolean" default="false" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Profile" type="nsi:Profile"/>
  <xs:complexType name="Profile">
    <xs:sequence>
      <xs:element name="startTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="endTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mwEnergy" type="xs:integer" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="IntegratedIntervals" type="nsi:IntegratedIntervals"/>
  <xs:complexType name="IntegratedIntervals">
    <xs:sequence>
      <xs:element name="IntegratedInterval" type="nsi:IntegratedInterval" minOccurs="0"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="IntegratedInterval" type="nsi:IntegratedInterval"/>
  <xs:complexType name="IntegratedInterval">
    <xs:sequence>
      <xs:element name="intervalStartTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="intervalStopTime" type="xs:dateTime" minOccurs="1" maxOccurs="1"/>
      <xs:element name="sinkBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element name="mwNetIntegrated" type="xs:integer" minOccurs="1" maxOccurs="1"/>
      <xs:element name="verifiedMatch" type="xs:boolean" default="false" minOccurs="1" maxOccurs="1"/>
    </xs:sequence>
```

```
  </xs:complexType>
  <xs:element name="RequestorBAs" type="nsi:RequestorBAs"/>
  <xs:complexType name="RequestorBAs">
    <xs:sequence>
      <xs:element name="requestorBA" type="xs:string" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="NsiTotal" type="nsi:NsiTotal"/>
  <xs:complexType name="NsiTotal">
    <xs:sequence>
      <xs:element name="checkoutBA" type="xs:string" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="nsi:NsiIntervals" minOccurs="1" maxOccurs="1"/>
      <xs:element ref="nsi:IntegratedIntervals" minOccurs="0" maxOccurs="1"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

6.2. Sample XML

Description:

Below are examples of well-formed xml.

Example 1

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSpy v2020 sp1 (x64) (http://www.altova.com)-->
<nsi:NsiCheckout xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:nsi="http://www.pjm.com/external/schemas/nsi/v1"
xsi:schemaLocation="http://www.pjm.com/external/schemas/nsi/v1 nsi.xsd">
  <requestStartTime>2001-12-17T09:30:47Z</requestStartTime>
  <requestStopTime>2001-12-17T09:30:47Z</requestStopTime>
  <responseTimestamp>2001-12-17T09:30:47Z</responseTimestamp>
  <requestType>RT</requestType>
  <includeIntegrated>true</includeIntegrated>
  <includeTag>true</includeTag>
  <creatorBA>String</creatorBA>
  <RequestorBAs>
    <requestorBA>String</requestorBA>
  </RequestorBAs>
  <nsi:NsiTotals>
    <NsiTotal>
      <checkoutBA>String</checkoutBA>
      <nsi:NsiIntervals>
        <NsiInterval>
          <intervalStartTime>2001-12-17T09:30:47Z</intervalStartTime>
          <intervalStopTime>2001-12-17T09:30:47Z</intervalStopTime>
          <sinkBA>String</sinkBA>
          <mwNet>0</mwNet>
          <verifiedMatch>false</verifiedMatch>
          <overriddenFlag>false</overriddenFlag>
        </NsiInterval>
      </nsi:NsiIntervals>
      <nsi:IntegratedIntervals>
        <IntegratedInterval>
          <intervalStartTime>2001-12-17T09:30:47Z</intervalStartTime>
          <intervalStopTime>2001-12-17T09:30:47Z</intervalStopTime>
          <sinkBA>String</sinkBA>
          <mwNetIntegrated>0</mwNetIntegrated>
          <verifiedMatch>false</verifiedMatch>
        </IntegratedInterval>
      </nsi:IntegratedIntervals>
    </NsiTotal>
  </nsi:NsiTotals>
  <RealTimeEnergyTransactions>
    <RealTimeEnergyTransaction>
      <tagIndex>0</tagIndex>
      <tagName>String</tagName>
```

```
      <tagTransactionType>String</tagTransactionType>
      <tagUpdateTimestamp>2001-12-17T09:30:47Z</tagUpdateTimestamp>
      <nsi:Profiles>
        <Profile>
          <startTime>2001-12-17T09:30:47Z</startTime>
          <endTime>2001-12-17T09:30:47Z</endTime>
          <mwEnergy>0</mwEnergy>
        </Profile>
      </nsi:Profiles>
    </RealTimeEnergyTransaction>
  </RealTimeEnergyTransactions>
</nsi:NsiCheckout>
```

Example 2

```
<?xml version="1.0" encoding="UTF-8"?>
<!--Sample XML file generated by XMLSpy v2020 sp1 (x64) (http://www.altova.com)-->
<nsi:NsiCheckout xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:nsi="http://www.pjm.com/external/schemas/nsi/v1"
xsi:schemaLocation="http://www.pjm.com/external/schemas/nsi/v1 nsi.xsd">
  <requestStartTime>2001-12-17T09:30:47Z</requestStartTime>
  <requestStopTime>2001-12-17T09:30:47Z</requestStopTime>
  <responseTimestamp>2001-12-17T09:30:47Z</responseTimestamp>
  <requestType>DAY</requestType>
  <includeIntegrated>true</includeIntegrated>
  <includeTag>true</includeTag>
  <creatorBA>String</creatorBA>
  <RequestorBAs>
    <requestorBA>String</requestorBA>
  </RequestorBAs>
  <nsi:DailyNsiTotals>
    <DailyNsiTotal>
      <checkoutBA>String</checkoutBA>
      <nsi:DailyNsiIntervals>
        <DailyNsiInterval>
          <intervalStartTime>2001-12-17T09:30:47Z</intervalStartTime>
          <intervalStopTime>2001-12-17T09:30:47Z</intervalStopTime>
          <sinkBA>String</sinkBA>
          <mwDaily>0</mwDaily>
          <verifiedMatch>false</verifiedMatch>
        </DailyNsiInterval>
      </nsi:DailyNsiIntervals>
      <nsi:IntegratedIntervals>
        <IntegratedInterval>
          <intervalStartTime>2001-12-17T09:30:47Z</intervalStartTime>
          <intervalStopTime>2001-12-17T09:30:47Z</intervalStopTime>
          <sinkBA>String</sinkBA>
          <mwNetIntegrated>0</mwNetIntegrated>
          <verifiedMatch>false</verifiedMatch>
        </IntegratedInterval>
      </nsi:IntegratedIntervals>
    </DailyNsiTotal>
  </nsi:DailyNsiTotals>
  <RealTimeEnergyTransactions>
    <RealTimeEnergyTransaction>
      <tagIndex>0</tagIndex>
      <tagName>String</tagName>
      <tagTransactionType>String</tagTransactionType>
      <tagUpdateTimestamp>2001-12-17T09:30:47Z</tagUpdateTimestamp>
      <nsi:Profiles>
        <Profile>
          <startTime>2001-12-17T09:30:47Z</startTime>
          <endTime>2001-12-17T09:30:47Z</endTime>
          <mwEnergy>0</mwEnergy>
        </Profile>
      </nsi:Profiles>
    </RealTimeEnergyTransaction>
  </RealTimeEnergyTransactions>
</nsi:NsiCheckout>
```