

**SMART METER**  
**TEXAS™**

SMT Retail Electric Provider (ROR), Competitive Service Provider  
(CSP), Transmission Distribution Service Provider (TDSP), MOU  
Data Access Interface Guide

February 26, 2024

## Table of Contents

1	Executive Summary.....	4
1.1	What Information is Available in SMT? .....	4
1.2	How can Information be retrieved from SMT? .....	5
1.3	Purpose of the SMT Interface Guide .....	6
1.4	SMT Interface Support .....	6
1.5	Quick Reference .....	6
1.5.1	Web Portal .....	7
1.5.2	FTPS .....	9
1.5.3	REST APIs .....	10
1.5.4	SOAP APIs .....	13
2	Security Integration Requirements .....	15
3	API Overview .....	19
3.1	REST Functions.....	19
3.1.1	Ad-Hoc .....	19
3.1.1.1	15-Minute Interval Data .....	19
3.1.1.2	Daily Register Reads .....	26
3.1.1.3	Monthly Billing Reads.....	32
3.1.1.4	Green Button.....	39
3.1.1.5	Meter Attributes.....	46
3.1.1.6	Premise Attributes .....	51
3.1.1.7	On-Demand Read .....	56
3.1.1.8	On-Demand Read Status .....	59
3.1.1.9	Report Status.....	61
3.1.1.10	New Subscription .....	64
3.1.1.11	Unsubscribe / Cancel Subscription.....	68
3.1.1.12	List Subscriptions .....	71
3.1.1.13	New Energy Data Sharing Agreement .....	74
3.1.1.14	List ESIIDs per Energy Data Sharing Agreement.....	78
3.1.1.15	Terminate Energy Data Sharing Agreement .....	82
3.1.1.16	Status of Energy Data Sharing Agreement.....	86

# SMART METER TEXAS™

---

3.1.1.17	List of Energy Data Sharing Agreement .....	88
3.2	SOAP Functions .....	91
3.2.1	Ad-Hoc .....	91
3.2.1.1	Energy Data (15-Minute Interval Data, or Daily Register Reads, or Monthly Billing Reads) 91	
3.2.1.2	Meter Attributes.....	108
3.2.1.3	Premise Attributes .....	121
3.2.1.4	Subscription – Scheduled Reports (New Subscription, or Unsubscribe / Cancel Subscription, or List of Existing Subscriptions).....	133
3.2.1.5	CSP Agreement (New Energy Data Sharing Agreement, or List ESIIDs per Energy Data Sharing Agreement, or Terminate Energy Data Sharing Agreement, or List of Energy Data Sharing Agreements).....	151
3.3	FTPS Overview .....	176
4	Integration.....	177
4.1	Web Portal .....	177
4.2	FTPS .....	177
4.3	API.....	179
5	Glossary of Terms .....	183

## 1 Executive Summary

The Smart Meter Texas (SMT) Interface Guide is the primary interface document for Texas market participants who wish to interface with SMT for the purpose of obtaining energy usage data, meter attributes, premise attributes, establishing subscriptions, or facilitating Competitive Service Provider (CSP) authorizations associated with the service territories covered by American Electric Power (AEP) Texas Central Company, AEP Texas North Company, CenterPoint Energy Houston Electric, LLC (CNP), Texas-New Mexico Power Company (TNMP) and Oncor Electric Delivery.

The Texas market participants associated with the service territories of AEP, CNP, TNMP, and Oncor, who will be able to interface with SMT, will include at a minimum Residential Customers, Small Business Customers, Retail Electric Providers (REPs), Competitive Service Providers (CSPs), ERCOT, Public Utility Commission of Texas (PUCT) and Texas Office of Public Utility Counsel (OPUC)

### 1.1 What Information is Available in SMT?

SMT maintains the following electric metering information and associated energy usage data for the period of two years:

- Meter Attributes
- Premise Attributes
- 15-minute Interval Data
- Daily Register Reads
- Monthly Billing Reads

## 1.2 How can Information be retrieved from SMT?

SMT will only provide information to authorized entities as defined in Section 5 – Glossary of Terms.

SMT provides the following interfaces to obtain Information from SMT:

- Website Portal
- FTPS
- APIs

Depending on the information being requested, it can be retrieved in the following formats:

- Website User Interface (UI)
- CSV
- Green Button (XML)
- LSE
- JSON

Depending on the information being requested, it can be delivered using the following methods:

- Website User Interface (UI Download)
- Email
- FTPS
- API (SOAP and REST)
- Callback API (Only in the JSON format)

Information can be processed for the various functions listed in this document based on the following mechanisms:

- On-Demand Read – Near-real Time Meter Reading
- Ad-Hoc Reporting – On-Demand Information Requests
- Subscription Reporting – Schedule to receive Information on continuing frequency

## 1.3 Purpose of the SMT Interface Guide

The aim of the SMT Interface Guide is to help the Texas market participant's gain better understanding of the functions provided by the various SMT interfaces that will be used to support both individual and business processes, the structure of the Application Programming Interfaces (APIs), and the minimum security requirements that will need to be met prior to using SMTs File Transfer Protocol Secure (FTPS) or API services.

While the intent of the SMT Interface Guide is to provide a complete set of documentation to interface with SMT, we recognize that there may be specific case-by-case examples in which further clarification is required to support SMT integration design and implementation. Please refer to the below section 1.3 SMT Integration Support for further assistance.

## 1.4 SMT Interface Support

The SMT Support team is available to answer specific questions related to integrating with SMT, as well as assist with completing all integration requirements. The SMT Support team can be contacted as follows:

1. Email at [Support@SmartMeterTexas.com](mailto:Support@SmartMeterTexas.com)

(Or)

2. Phone at 1-844-217-8595

## 1.5 Quick Reference

The following quick reference provides a list of SMT Interface functions by SMT's Web Portal, REST API, SOAP API and FTPS services. The below table includes the type of function, a short description of the function, and the page number of this document that will provide details associated with the function.

To use the various functions listed in this document, the user must have either established an SMT Account (e.g. Residential Customer, Small Business Customer, Retail Electric Provider, Competitive Service Provider, TDSP, MOU, ERCOT, PUCT or OPUC), or be an SMT Account Administrator or User of an already establish SMT account. For further clarity on an SMT Account or the role of the SMT Account, please refer to the Security Requirements in section 2 of this document.

### **1.5.1 Web Portal**

The following SMT functions are available to all SMT Account types within the SMT Web Portal located at [www.smartmetertexas.com](http://www.smartmetertexas.com). For those function types that are listed as 'Subscription', the user will have the ability to select whether they wish to receive their information via email or FTPS. For email-based subscriptions, it should be noted that the user's email service may limit or restrict the size of the email that they are able to receive and will need to plan accordingly.

# SMART METER TEXAS™

Function	Type	Function Description
15-Minute Interval Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.
Daily Register Reads	Ad-Hoc	Get Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.
Monthly Billing Reads	Ad-Hoc	Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs for one DUNS.
Premise Attributes	Ad-Hoc	Get Premise Attributes for one or more ESIIDs, or all ESIIDs for one DUNS.
On-Demand Read (ODR)	Ad-Hoc	Get current Meter Read for one ESIID at a time. Note: The daily limit for ODRs is up to 3,000 per Transmission and Distribution Utility Company (TDU).
On-Demand Read Status	Ad-Hoc	Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.
Report Status	Ad-Hoc	Get the current status for Ad-Hoc or Subscription requests. Note: This is a synchronous function.
New Subscription	Ad-Hoc	Create a new subscription for one or more ESIIDs
Unsubscribe / Cancel Subscription	Ad-Hoc	Unsubscribe or Cancel an existing subscription on a one subscription per request basis.
Update Subscription	Ad-Hoc	Update an existing subscription on a one subscription per request basis.
List Subscriptions	Ad-Hoc	Get a list of subscriptions.
New Energy Data Sharing Agreement	Ad-Hoc	Initiate a new Energy Data Sharing Agreement for one or more ESIIDs.
List ESIIDs per Energy Data Sharing Agreement	Ad-Hoc	Get a list of ESIIDs associated with a specific Energy Data Sharing Agreement.



# SMART METER TEXAS™

Terminate Energy Data Sharing Agreements	Ad-Hoc	Terminate one or more Energy Data Sharing Agreements.
Status of Energy Data Sharing Agreements	Ad-Hoc	Get a status of one or more Energy Data Sharing Agreements.
List of Energy Data Sharing Agreements	Ad-Hoc	Get a list of one or more Energy Data Sharing Agreements.
15-Minute Interval Data	Subscription	Request to receive, on a defined frequency, a push of 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.
Daily Register Reads	Subscription	Request to receive, on a defined frequency, a push of Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.
Monthly Billing Reads	Subscription	Request to receive, on a defined frequency, a push of Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Subscription	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.

*Table 1: SMT Functions*

## 1.5.2 FTPS

All SMT Account Administrators and Users may elect to receive the following ad-hoc or subscription-based information delivered to a FTPS folder for retrieval within a period of 10 days:

- 15-minute Interval Data
- Daily Register Reads
- Monthly Billing Reads
- Meter Attributes
- Premise Attributes

# SMART METER TEXAS™

The requested information may be stored in the FTPS folders using either the Comma-Separated Values (CSV) format, LodeStar Enhanced (LSE) format, or eXtensible Markup Language (XML) file format. For Retail Electric Provider of Record (ROR), the daily market defined files format (LSE) will be used for 15 minute interval data.

## 1.5.3 REST APIs

The following SMT functions are available using a specific Representational State Transfer (REST) API to support request and response payload or message in a JavaScript Object Notation (JSON) format, except where the function is based on the Green Button format. For Green Button specified functions, the request is based on the JSON format, and the response payload or message will be an XML format as defined by the Green Button standard.

Function	Type	Function Description	Page #
15-Minute Interval Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.	14
Daily Register Reads	Ad-Hoc	Get Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	21
Monthly Billing Reads	Ad-Hoc	Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	27
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one ESIID for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.	33
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNS.	39
Premise Attributes	Ad-Hoc	Get Premise Attribute data for one or more ESIIDs, or all ESIIDs of one DUNS.	44
On-Demand Read (ODR)	Ad-Hoc	Get current Meter Read for one ESIID at a time. Note: The daily limit for ODRs is up to 3,000 per Transmission and Distribution Utility Company (TDU).	49
On-Demand Read Status	Ad-Hoc	Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.	52
Report Status	Ad-Hoc	Get the current status for an Ad-Hoc or Subscription request. Note: This is a synchronous function.	54
New Subscription	Ad-Hoc	Create a new subscription for one or more ESIIDs	56

# SMART METER TEXAS™

Function	Type	Function Description	Page #
15-Minute Interval Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.	14
Daily Register Reads	Ad-Hoc	Get Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	21
Monthly Billing Reads	Ad-Hoc	Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	27
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one ESIID for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.	33
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNS.	39
Premise Attributes	Ad-Hoc	Get Premise Attribute data for one or more ESIIDs, or all ESIIDs of one DUNS.	44
On-Demand Read (ODR)	Ad-Hoc	Get current Meter Read for one ESIID at a time. Note: The daily limit for ODRs is up to 3,000 per Transmission and Distribution Utility Company (TDU).	49
On-Demand Read Status	Ad-Hoc	Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.	52
Unsubscribe / Cancel Subscription	Ad-Hoc	Unsubscribe or Cancel an existing subscription. Only one subscription per request can be submitted.	60
List Subscriptions	Ad-Hoc	Get a list of subscriptions.	63
New Energy Data Sharing Agreement	Ad-Hoc	Initiate a new Energy Data Sharing Agreement for one or more ESIIDs.	66
List ESIIDs per Energy Data Sharing Agreement	Ad-Hoc	Get a list of ESIIDs associated with a specific Energy Data Sharing Agreement.	69
Terminate Energy Data Sharing Agreements	Ad-Hoc	Terminate one or more Energy Data Sharing Agreements.	73

# SMART METER TEXAS™

Function	Type	Function Description	Page #
15-Minute Interval Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.	14
Daily Register Reads	Ad-Hoc	Get Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	21
Monthly Billing Reads	Ad-Hoc	Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	27
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one ESIID for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.	33
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNS.	39
Premise Attributes	Ad-Hoc	Get Premise Attribute data for one or more ESIIDs, or all ESIIDs of one DUNS.	44
On-Demand Read (ODR)	Ad-Hoc	Get current Meter Read for one ESIID at a time. Note: The daily limit for ODRs is up to 3,000 per Transmission and Distribution Utility Company (TDU).	49
On-Demand Read Status	Ad-Hoc	Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.	52
Status of Energy Data Sharing Agreements	Ad-Hoc	Get a status of one or more Energy Data Sharing Agreements.	76
List of Energy Data Sharing Agreements	Ad-Hoc	Get a list of one or more Energy Data Sharing Agreements.	78
15-Minute Interval Data	Subscription	Request to receive, on a defined frequency, a push of 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.	N/A
Daily Register Reads	Subscription	Request to receive, on a defined frequency, a push of Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	N/A
Monthly Billing Reads	Subscription	Request to receive, on a defined frequency, a push of Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	N/A

# SMART METER TEXAS™

Function	Type	Function Description	Page #
15-Minute Interval Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs for one DUNS.	14
Daily Register Reads	Ad-Hoc	Get Daily Register Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	21
Monthly Billing Reads	Ad-Hoc	Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	27
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one ESIID for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.	33
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNS.	39
Premise Attributes	Ad-Hoc	Get Premise Attribute data for one or more ESIIDs, or all ESIIDs of one DUNS.	44
On-Demand Read (ODR)	Ad-Hoc	Get current Meter Read for one ESIID at a time. Note: The daily limit for ODRs is up to 3,000 per Transmission and Distribution Utility Company (TDU).	49
On-Demand Read Status	Ad-Hoc	Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.	52
<b>Green Button</b> (15-Minute Interval Data, Daily Register Reads, or Monthly Billing Reads)	Subscription	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads or Monthly Billing Reads for one ESIID for one DUNS in the <b>Green Button</b> format. Note: This is a synchronous function.	N/A

*Table 2: SMT REST API Functions*

## 1.5.4 SOAP APIs

The following SMT functions are available using a specific Simple Object Access Protocol (SOAP) API to support request and response payload or message in an XML format, except where the function is based on the Green Button format. For Green Button specified functions, the request is based on the JSON format, and the response payload or message will be an eXtensible Markup Language (XML) format as defined by the Green Button standard.

# SMART METER TEXAS™

Function	Type	Function Description	Page #
Energy Data	Ad-Hoc	Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads, or Monthly Billing Reads for one or more ESIIDs, or all ESIIDs for one DUNS.	81
Meter Attributes	Ad-Hoc	Get Meter Attributes for one or more ESIIDs, or all ESIIDs for one DUNS.	97
Premise Attributes	Ad-Hoc	Get Premise Attributes for one or more ESIIDs, or all ESIIDs for one DUNS.	110
Subscription - Scheduled Reports	Ad-Hoc	Create a new subscription, or Unsubscribe / Cancel an existing subscription, or update an existing subscription, or get a list of existing subscriptions.	121
CSP Agreement	Ad-Hoc	Initiate a new Energy Data Sharing Agreement, or get a list of ESIIDs associated with a specific Energy Data Sharing Agreement, or Terminate one or more Energy Data Sharing Agreements, or get a status of one or more Energy Data Sharing Agreements.	138

*Table 3: SMT SOAP API Functions*

## 2 Security Integration Requirements

The following is a minimum set of security requirements for effectively interfacing with SMT for using REST API, SOAP API, or FTP services.

- Smart Meter Texas (SMT) Account -

A dedicated SMT Account on the SMT portal, belonging to either a Residential Customer, Small Business Customer, REP, CSP, TDU, ERCOT, OPUC, or PUCT, must be established prior to using any of the interface functions for the purpose of obtaining energy usage data, meter attribute data, premise attribute data, establishing subscriptions, or facilitating Competitive Service Provider (CSP) authorizations associated with the service territories covered by AEP, CNP, TNMP and Oncor.

- Smart Meter Texas (SMT) Account Administrator – (Note: this is not applicable to Residential),

An SMT Account Administrator on the SMT portal is capable of defining their account's functional (i.e. operational and security) requirements, and will be responsible for managing the technical aspects of the Company's Account including the DUNS number, the integration services, alerts and/or notifications received from SMT, and generating the appropriate integration service credentials using the SMT portal.

In addition, if an SMT Account owner elects to use an Independent IT Service Provider for integration services, a representative of the IT Service Provider will need to register as an "on-behalf" user and be approved by one of the Company Administrators prior to performing their integration services.

Note: It is the responsibility of the Company Administrator, and ultimately the SMT Account owner, to define and fully understand their overall acceptance of the risk associated with their respective functional requirements.

- Static Internet Protocol (IP) –

A minimum of one static IP will be required to interface with SMT and can be used to support both FTPS and API services. If an SMT Account owner elects to use an independent IT Service Provider (i.e. on-behalf) which supports more than one company, a minimum of one

static IP may be used by the IT Service Provider to support all SMT Accounts as long as each SMT Account owner is aware of and approves this configuration.

- FTPS Secure Socket Layer (SSL) Certificate –

A minimum of one signed registered FTPS SSL certificate for each SMT Account will be required to interface with an SMT Entity Account. No wildcard certificates will be accepted.

An unsigned FTPS SSL certificate may be used for the sole purposes of testing.

The signed certificate for production use must match your Entity domain name. The name format for the signed certificate should be as follows:

- REP: smt.repname.com, or  
repname.com, or
- CSP: smt.cspname.com, or  
cspname.com, or

For IT Vendors providing IT services on behalf of Retail Electric Providers or Competitive Service Providers, integration will require a separate unique SSL certificate for each Entity that you represent. The domain name on the certificate must match your Entity name and the subdomain must match the Retail Electric Provider or Competitive Service Provider Entity name for which you are representing and accessing data. The name format for the signed certificate should be as follows:

- REP: smt.repname.vendorname.com, or  
repname.vendorname.com, or
- CSP: smt.cspname.vendorname.com, or  
cspname.com, or

- API SSL Certificate –

A minimum of one signed registered API SSL certificate to the SMT Entity Account owner will be required to interface with an SMT Entity Account. An unsigned API SSL certificate may only be used for testing.

The signed certificate for production use must match your Entity domain name.



# SMART METER TEXAS™

---

For IT Vendors providing IT services on behalf of Retail Electric Providers or Competitive Service Providers, integration will require a separate unique SSL certificate for each Entity that you represent. The domain name on the certificate must match your Entity name and the subdomain must match the Retail Electric Provider or Competitive Service Provider Entity name for which you representing and accessing data.

- Pretty Good Privacy (PGP) Key –

A minimum of one PGP key per Entity DUNS will be required to interface with an SMT Account for FTPS services.

If you are an IT Vendor providing IT services on behalf of Retail Electric Providers or Competitive Service Providers, you will acquire the PGP Key from the Entity you represent.

- FTPS Credentials –

A minimum of one set of FTPS credentials (e.g. ID and password) will be required for each DUNS associated with an SMT Account. The Smart Meter Texas Support team will provide these credentials.

FTPS credentials will be supplied by the Smart Meter Texas support team.

Note: A residential or small business customer will not be required to acquire a DUNS number to create the FTPS credentials.

- API Credentials –

A minimum of one set of API credentials (e.g. ID and password) will be required for each DUNS associated with an SMT Account. API credentials will be created by your Entity through the Smart Meter Texas website portal. A Company Administrator will have the capability to create these credentials using the SMT portal.

Note: Since residential or small business customer do not associate DUNS numbers to their account, just one set of credentials are required for FTPS or API access

- Transport Layer Security (TLS)
  - Only TLS 1.0 – TLS 1.2 are supported.
  - Mutual TLS authentication will be required.
  
- FTP CIPHER Suites

Only strong CIPHER suites are supported. The following list of CIPHER suites are recommended while integrating with SMT:

  - ECDHE\_RSA\_WITH\_AES\_256\_CBC\_SHA384
  - RSA\_WITH\_AES\_256\_GCM\_SHA384
  - RSA\_WITH\_AES\_256\_CBC\_SHA256
  - RSA\_WITH\_AES\_256\_CBC\_SHA
  - DHE\_RSA\_WITH\_AES\_256\_CBC\_SHA
  - RSA\_WITH\_AES\_128\_CBC\_SHA
  - RSA\_WITH\_3DES\_EDE\_CBC\_SHA
  - ECDHE\_ECDSA\_WITH\_AES\_256\_CBC\_SHA384
  - DHE\_DSS\_WITH\_AES\_256\_CBC\_SHA256
  - DHE\_DSS\_WITH\_AES\_256\_CBC\_SHA
  - DHE\_RSA\_WITH\_AES\_256\_GCM\_SHA384
  - RSA\_WITH\_AES\_128\_CBC\_SHA256
  - RSA\_WITH\_AES\_256\_CBC\_SHA256
  
- FTP Server Public IPs
  - If a firewall is being used to manage outbound traffic, the SMT Support team can provide SMTs FTP server public IPs for whitelisting on the firewall.
  
- FTP Directory Listing or File Transfers
  - The PROTP command must be used before listing any directory or transferring files.

## 3 API Overview

The following information will provide the detailed API or FTPS definitions for each of the SMT functions by REST API, SOAP API, or FTPS service. For each SMT function, this document will provide the function name and description, the associated Uniform Resource Locator (URL), method, data parameters, return status codes, success response, error response, and sample call.

Note: Required fields are identified as a red asterisk (\*).

### 3.1 REST Functions

#### 3.1.1 Ad-Hoc

##### 3.1.1.1 15-Minute Interval Data

- Function Description –

Get 15-Minute Interval Consumption and Generation Data for one or more ESIIDs or all ESIIDs of one DUNS.

Note: This specific service supports a callback feature that can integrate with a customer's API. Additionally,

- Test URL –

<https://uat.services.smartmetertexas.net/15minintervalreads/>

- Production URL -

<https://services.smartmetertexas.net/15minintervalreads/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters –

- **Interval Energy Data Request**

{

# SMART METER TEXAS™

---

trans_id*	<p>string pattern: [a-zA-Z0-9]{1,32}</p> <p>Transaction Id</p>
requestorID*	<p>string minLength: 1 maxLength: 32 example: SMT_PORTAL_ID</p> <p>User ID registered at SMT Portal.</p>
requesterType*	<p>string example: REP</p> <p>The user type of the registered SMT Portal User Id.</p> <p>Enum: Array [ 12 ]</p>
requesterAuthenticationID	<p>string minLength: 1 maxLength: 18 example: 12321313213</p> <p>Requester Company DUNS number</p>
startDate*	<p>string(\$date) pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$ example: 01/20/2018</p> <p>Start date in mm/dd/yyyy format.</p>
endDate*	<p>string(\$date) pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$ example: 02/20/2018</p> <p>End date in mm/dd/yyyy format.</p>
deliveryMode	<p>string example: FTP</p> <p>The Asynchronous delivery mechanism for the requested report. For requesterTypes: REP, CSP, TDSP, REG the default deliveryMode is FTP and for requesterTypes: RES, BUSINESS the default deliveryMode is XML</p>
reportFormat	<p>Enum: Array [ 6 ] string example: CSV</p>

version*	Format of the requested report. The default report format through delivered through FTP and EML is CSV and the only report format delivered through API is JSON. Enum: Array [ 6 ] string <i>example: L</i>
readingType	The version of the energy data required. A-All, L-Latest  Enum: Array [ 2 ] string <i>example: C</i>
esiid*	The type of the reading. C-Consumption, G-Generation, A-Both Consumption and Generation. This field is mandatory field for Interval resource
SMTTermsandConditions*	Enum: Array [ 6 ] [...] string <i>example: Y</i>
	Terms and Conditions. End users needs to accept terms & conditions by sending the value as Y
	Enum: Array [ 4 ]
}	

- **Interval Energy Data Response - Synchronous**

{		
trans_id	string	<i>pattern: [a-zA-Z0-9]{1,32}</i>
		Transaction Id
esiid*	string	

# SMART METER TEXAS™

---

```
energyData* [
  {
    DT* string($date)
        pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}$
        example: 01/20/2018

        Date in mm/dd/yyyy.

    RevTS* string($date-time)
           example: 01/20/2018 11:31:25

           Revision date in mm/dd/yyyy hh:mm:ss

    RT* string
        minLength: 1
        maxLength: 1
        example: C

        Reading Type: C- Consumption, G- Generation.

    RD* string
        example: .085-A,.118-A,.085-A,.153-A,.123-A,.143-A,.133-
        A,.134-A,,,,,.143-A,.125-A,.15-A,.117-A,.159-A,.109-A,.159-
        A,.114-A,.153-A,.121-A,.146-A,.126-A,.141-A,.132-A,.136-
        A,.273-A,.159-A,.175-A,.107-A,.138-A,.132-A,.138-A,.082-
        A,.123-A,.117-A,.147-A,.113-A,.145-A,.121-A,.147-A,.119-
        A,.141-A,.122-A,.138-A,.126-A,.136-A,.127-A,.134-A,.17-
        A,.15-A,.111-A,.142-A,.132-A,.126-A,.141-A,.119-A,.153-
        A,.112-A,.153-A,.105-A,.153-A,.105-A,.136-A,.132-A,.118-
        A,.117-A,.12-A,.118-A,.132-A,.103-A,.147-A,.117-A,.144-
        A,.176-A,.125-A,.162-A,.12-A,.163-A,.125-A,.154-A,.134-
        A,.147-A,.162-A,.111-A,.16-A,.123-A,.148-A,.133-A,.138-
        A,.132-A,.13-A,.151-A,.123-A,.16-A,.114-A,.162-A,.117-
        A,.125-A

        Meter Reading in kWh. A-Actual, E-Estimated.

  }
}
```

- Acknowledgement to the Asynchronous Request

```
{
```

# SMART METER TEXAS™

---

```
trans_id*      string
               pattern: [a-zA-Z0-9]{1,32}
               Transaction Id
correlationId* string
               maxLength: 18
statusCode*    string
statusReason* string
faultESIIDs   [
               {
                   esiid*    string
                           minLength: 9
                           maxLength: 64
                   reason*   string
                           maxLength: 256
               }
               ]
}
```

- **Error Response**

```
{
  errorCode*    string
  errorKey      string
  errorMessage* string
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

# SMART METER TEXAS™

---

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 4: Return Status Codes

- Example(s) –

## Example 1 - Synchronous Response – JSON file through REST API

- **Interval Energy Data Request**

```
{
  "trans_id": "123",
  "requestorID": "smtfirstresidentialuser",
  "requesterType": "RES",
  "startDate": "07/01/2019",
  "endDate": "07/03/2019",
  "reportFormat": "JSON",
  "version": "L",
  "readingType": "C",
  "esiid": [
    "1008901012126195372100"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Interval Energy Data Response - Synchronous**

```
{
  "trans_id": "123",
  "esiid": "1008901012126195372100",
  "energyData": [
    {
      "DT": "07/01/2019",
```



# SMART METER TEXAS™

---

```
"RevTS": "07/02/2019 00:20:05",
"RT": "C",
"RD": ".198-A,.141-A,.183-A,.109-A,.176-A,.12-A,.148-A,.181-A,,,,.156-
A,.132-A,.16-A,.154-A,.132-A,.197-A,.147-A,.148-A,.169-A,.135-A,.139-A,.109-
A,.243-A,.125-A,.093-A,.117-A,.146-A,.103-A,.115-A,.18-A,.133-A,.165-A,.152-
A,.123-A,.126-A,.161-A,.153-A,.183-A,.209-A,.185-A,.136-A,.142-A,.153-A,.151-
A,.198-A,.158-A,.191-A,.18-A,.193-A,.204-A,.241-A,.24-A,.243-A,.226-A,.233-
A,.231-A,.24-A,.275-A,.245-A,.254-A,.235-A,.232-A,.231-A,.297-A,.23-A,.366-
A,.295-A,.313-A,.39-A,.316-A,.317-A,.362-A,.387-A,.383-A,.328-A,.395-A,.396-
A,.395-A,.393-A,.388-A,.382-A,.365-A,.353-A,.289-A,.286-A,.251-A,.245-A,.279-
A,.255-A,.248-A,.294-A,.24-A,.238-A,.275-A,.242-A,.292-A,.242-A,.23-A"
},
{
"DT": "07/02/2019",
"RevTS": "07/03/2019 02:39:13",
"RT": "C",
"RD": ".212-A,.192-A,.172-A,.196-A,.201-A,.151-A,.209-A,.149-A,,,,.151-
A,.173-A,.212-A,.248-A,.295-A,.166-A,.208-A,.184-A,.188-A,.201-A,.196-A,.155-
A,.16-A,.177-A,.156-A,.146-A,.163-A,.135-A,.178-A,.152-A,.165-A,.168-A,.211-
A,.154-A,.184-A,.186-A,.21-A,.188-A,.155-A,.2-A,.166-A,.15-A,.183-A,.18-A,.155-
A,.169-A,.192-A,.175-A,.192-A,.205-A,.247-A,.213-A,.218-A,.244-A,.229-A,.206-
A,.233-A,.239-A,.238-A,.221-A,.202-A,.172-A,.217-A,.213-A,.196-A,.224-A,.201-
A,.178-A,.218-A,.216-A,.209-A,.228-A,.201-A,.407-A,.311-A,.247-A,.246-A,.286-
A,.25-A,.232-A,.233-A,.186-A,.191-A,.21-A,.262-A,.161-A,.198-A,.161-A,.144-
A,.179-A,.174-A,.208-A,.168-A,.167-A,.157-A,.169-A,.184-A,.203-A"
},
{
"DT": "07/03/2019",
"RevTS": "07/04/2019 04:21:07",
"RT": "C",
"RD": ".18-A,.173-A,.17-A,.153-A,.131-A,.137-A,.124-A,.123-A,,,,.13-A,.144-
A,.123-A,.159-A,.113-A,.162-A,.153-A,.114-A,.156-A,.138-A,.155-A,.134-A,.116-
A,.127-A,.122-A,.108-A,.098-A,.123-A,.128-A,.086-A,.132-A,.199-A,.162-A,.171-
A,.175-A,.164-A,.157-A,.189-A,.173-A,.191-A,.186-A,.167-A,.154-A,.169-A,.186-
A,.153-A,.18-A,.19-A,.155-A,.156-A,.179-A,.185-A,.205-A,.18-A,.207-A,.182-A,.221-
A,.18-A,.208-A,.175-A,.192-A,.171-A,.186-A,.187-A,.196-A,.19-A,.207-A,.185-
A,.225-A,.208-A,.26-A,.265-A,.254-A,.27-A,.254-A,.282-A,.228-A,.236-A,.332-
A,.395-A,.4-A,.297-A,.267-A,.289-A,.292-A,.28-A,.234-A,.246-A,.244-A,.306-A,.286-
A,.263-A,.294-A,.244-A,.297-A,.235-A,.224-A,.272-A"
}
]
}
```

## Example 2 - Asynchronous Response – CSV File Sent Through Email

- **Interval Energy Data Request**

```
{
  "trans_id": "1234",
  "requestorID": "intervalbothgc",
  "requesterType": "RES",
  "startDate": "01/20/2018",
  "endDate": "01/25/2018",
  "deliveryMode": "eml",
  "reportFormat": "CSV",
  "version": "L",
  "readingType": "A",
  "esiid": [
    "10204049715823010"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Interval Energy Data Response – Asynchronous Email**

```
{
  "trans_id": "1234",
  "correlationId": "3d4dd55cabf211e9ac0c0a04",
  "statusCode": "0000",
  "statusReason": "Request has been submitted successfully. The CSV report will be delivered through EML"
}
```

### 3.1.1.2 Daily Register Reads

- Function Description –

Get Daily Register Reads for one or more ESIIIDs, or all ESIIIDs of one DUNS.

Note: This specific service supports a callback feature that can integrate with a customer's API.

- Test URL –

# SMART METER TEXAS™

---

<https://uat.services.smartmetertexas.net/dailyreads/>

- Production URL -

<https://services.smartmetertexas.net/dailyreads/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Daily Register Reads

- **Daily Energy Data Request**

{		
trans_id*	string	pattern: [a-zA-Z0-9]{1,32}
		Transaction Id
requestorID*	string	minLength: 1
		maxLength: 32
		example: SMT_PORTAL_ID
		User ID registered at SMT Portal.
requesterType*	string	example: REP
		The user type of the registered SMT Portal User Id.
		Enum:
		Array [ 12 ]
requesterAuthenticationID	string	minLength: 1
		maxLength: 18
		example: 12321313213
		Requester Company DUNS number
startDate*	string(\$date)	pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$
		example: 01/20/2018
		Start date in mm/dd/yyyy format.

# SMART METER TEXAS™

---

endDate*	<p>string(\$date) pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$ example: 02/20/2018</p> <p>End date in mm/dd/yyyy format.</p>
deliveryMode	<p>string example: FTP</p> <p>The Asynchronous delivery mechanism for the requested report. For requesterTypes: REP, CSP, TDSP, REG the default deliveryMode is FTP and for requesterTypes: RES, BUSINESS the default deliveryMode is EML</p>
reportFormat	<p>Enum: Array [ 6 ] string example: CSV</p> <p>Format of the requested report. The default report format through delivered through FTP and EML is CSV and the only report format delivered through API is JSON.</p>
version*	<p>Enum: Array [ 6 ] string example: L</p> <p>The version of the energy data required. A-All, L-Latest</p>
readingType	<p>Enum: Array [ 2 ] string example: C</p> <p>The type of the reading. C-Consumption, G-Generation, A-Both Consumption and Generation. This field is mandatory field for Interval resource</p>
esiid*	<p>Enum: Array [ 6 ] [...]</p>

SMTTermsandConditions\*

string

example: Y

Terms and Conditions. End users needs to accept terms & conditions by sending the value as Y

Enum:

Array [ 4 ]

}

○ **Daily Registered Read Synchronous Response**

{  
trans\_id

string

pattern: [a-zA-Z0-9]{1,32}

Transaction Id

esiid\*

string

registeredReads\* [

{

readDate\*

string(\$date)

pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$

example: 01/20/2018

Date in mm/dd/yyyy.

revisionDate\*

string(\$date-time)

example: 01/20/2018 11:31:25

Revision date in mm/dd/yyyy  
hh:mm:ss

startReading\*

string

example: 43791.955

endReading\*

string

example: 43798.024

energyDataKwh\*

string

example: 6.031

}

]

}

○ **Acknowledgement to the Asynchronous Request**

# SMART METER TEXAS™

---

```
{
  trans_id*      string
                 pattern: [a-zA-Z0-9]{1,32}
                 Transaction Id
  correlationId* string
                 maxLength: 18
  statusCode*    string
  statusReason*  string
  faultESIIDs    [
                  {
                    esiid*    string
                             minLength: 9
                             maxLength: 64
                    reason*    string
                             maxLength: 256
                  }
                ]
}
```

- o **Error**

```
{
  errorCode*    string
  errorKey      string
  errorMessage* string
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error":"Basic authentication header is missing."}

# SMART METER TEXAS™

---

400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 5: Return Status Codes

- Example(s) –

## Example 1 - Synchronous Response – JSON file through REST API

- **Daily Register Read Request**

```
{
  "trans_id": "111",
  "requestorID": "UATRES123",
  "requesterType": "RES",
  "startDate": "07/07/2019",
  "endDate": "07/07/2019",
  "version": "L",
  "readingType": "c",
  "esiid": [
    "1008901005102632580100"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Daily Register Read Response - Synchronous**

```
{
  "trans_id": "111",
  "esiid": "1008901005102632580100",
  "registeredReads": [ {
    "readDate": "07/07/2019",
    "revisionDate": "07/08/2019 04:09:46",
    "startReading": "77357.514",
    "endReading": "77412.583",
    "energyDataKwh": "55.069"
  }
]
```

## Example 2 - Asynchronous Response – CSV File Sent Through Email

- **Daily Register Read Request**

```
{
  "trans_id": "111",
  "requestorID": "UATRES123",
  "requesterType": "RES",
  "startDate": "05/07/2019",
  "endDate": "07/07/2019",
  "deliveryMode": "EML",
  "reportFormat": "CSV",
  "version": "L",
  "readingType": "c",
  "esiid": [
    "1008901005102632580100"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Daily Register Read Response – Asynchronous Email**

```
{
  "trans_id": "111",
  "correlationId": "ac80d2c2ad5411e9ac0c0a04",
  "statusCode": "0000",
  "statusReason": "Request has been submitted successfully. The CSV report will be delivered through EML"
}
```

### ***3.1.1.3 Monthly Billing Reads***

- Function Description –

Get Monthly Billing Reads for one or more ESIIDs, or all ESIIDs of one DUNS.



# SMART METER TEXAS™

---

Note: This specific service supports a callback feature that can integrate with a customer's API.

- Test URL –

<https://uat.services.smartmetertexas.net/monthlybillingInformation/>

- Production URL -

<https://services.smartmetertexas.net/monthlybillingInformation/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Monthly Billing Data

- **Monthly Billing Energy Data Request**

```
{
  trans_id*           string
                      pattern: [a-zA-Z0-9]{1,32}
                      Transaction Id
  requestorID*       string
                      minLength: 1
                      maxLength: 32
                      example: SMT_PORTAL_ID
                      User ID registered at SMT Portal.
  requesterType*     string
                      example: REP
                      The user type of the registered SMT Portal
                      User Id.
                      Enum:
                      Array [ 12 ]
  requesterAuthenticationID string
                      minLength: 1
                      maxLength: 18
                      example: 12321313213
                      Requester Company DUNS number
}
```

# SMART METER TEXAS™

---

startDate*	<p><code>string(\$date)</code> <code>pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$</code> <code>example: 01/20/2018</code></p> <p>Start date in mm/dd/yyyy format.</p>
endDate*	<p><code>string(\$date)</code> <code>pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$</code> <code>example: 02/20/2018</code></p> <p>End date in mm/dd/yyyy format.</p>
deliveryMode	<p><code>string</code> <code>example: FTP</code></p> <p>The Asynchronous delivery mechanism for the requested report. For requesterTypes: REP, CSP, TDSP, REG the default deliveryMode is FTP and for requesterTypes: RES, BUSINESS the default deliveryMode is EML</p>
reportFormat	<p>Enum: Array [ 6 ] <code>string</code> <code>example: CSV</code></p> <p>Format of the requested report. The default report format through delivered through FTP and EML is CSV and the only report format delivered through API is JSON.</p>
version*	<p>Enum: Array [ 6 ] <code>string</code> <code>example: L</code></p> <p>The version of the energy data required. A-All, L-Latest</p>
readingType	<p>Enum: Array [ 2 ] <code>string</code> <code>example: C</code></p> <p>The type of the reading. C-Consumption, G-Generation, A-Both Consumption and</p>

Generation. This field is mandatory field for Interval resource

Enum:  
Array [ 6 ]

esiid\*

[...]

SMTTermsandConditions\*

string

example: Y

Terms and Conditions. End users needs to accept terms & conditions by sending the value as Y

Enum:  
Array [ 4 ]

}

## o Monthly Billing Data Synchronous Response

{

trans\_id string

pattern: [a-zA-Z0-9]{1,32}

Transaction Id

esiid\* string

billingData [

\*

{

startDate\*

string(\$date)

pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$

example: 01/20/2018

Start Date for the Billing month in mm/dd/yyyy.

endDate\*

string(\$date)

pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}\$

example: 02/20/2018

End Date for the Billing month in mm/dd/yyyy.

revisionDate

string(\$date-time)

example: 01/20/2018 11:31:25

Revision date in mm/dd/yyyy hh:mm:ss

# SMART METER TEXAS™

---

actualkWh\* string  
example: 389  
meteredKW\* string  
billedKW\* string  
meteredKVA\* string  
billedKVA\* string

}

]

}

- **Acknowledgement**

{

trans\_id\* string  
pattern: [a-zA-Z0-9]{1,32}  
Transaction Id  
correlationId\* string  
maxLength: 18  
statusCode\* string  
statusReason\* string  
faultESIIDs [  
    {  
        esiid\* string  
            minLength: 9  
            maxLength: 64  
        reason\* string  
            maxLength: 256  
    }]

}

- **Error**

{

errorCode\* string  
errorKey string  
errorMessage\* string

}

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 6: Return Status Codes*

- Example –

### Example 1 - Synchronous Response – JSON file through REST API

- **Monthly Billing Data Request**

```
{
  "trans_id": "111",
  "requestorID": "UATRES123",
  "requesterType": "RES",
  "startDate": "05/07/2019",
  "endDate": "07/07/2019",
  "version": "L",
  "readingType": "c",
  "esiid": [
    "1008901005102632580100"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Monthly Billing Data - Synchronous**

```
{
  "trans_id": "111",
  "esiid": "1008901005102632580100",
  "billingData": [
    {
      "startDate": "04/25/2019",
      "endDate": "05/24/2019",
      "revisionDate": "05/29/2019 01:22:52",
      "actualkWh": "683",
      "meteredKW": "0",
      "billedKW": "0",
      "meteredKVA": "0",
      "billedKVA": "0"
    },
    {
      "startDate": "05/24/2019",
      "endDate": "06/25/2019",
      "revisionDate": "06/27/2019 01:33:05",
      "actualkWh": "1048",
      "meteredKW": "0",
      "billedKW": "0",
      "meteredKVA": "0",
      "billedKVA": "0"
    },
    {
      "startDate": "06/25/2019",
      "endDate": "07/01/2019",
      "revisionDate": "07/02/2019 02:22:47",
      "actualkWh": "211",
      "meteredKW": "0",
      "billedKW": "0",
      "meteredKVA": "0",
      "billedKVA": "0"
    }
  ]
}
```

## Example 2 - Asynchronous Response – CSV File Sent Through Email

- **Monthly Billing Data Request**

```
{
  "trans_id": "111",
  "requestorID": "UATRES123",
  "requesterType": "RES",
  "startDate": "04/07/2019",
  "endDate": "07/07/2019",
  "deliveryMode": "EML",
  "reportFormat": "CSV",
  "version": "L",
  "readingType": "c",
  "esiid": [
    "1008901005102632580100"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Daily Register Read Response – Asynchronous Email**

```
{
  "trans_id": "111",
  "correlationId": "34553b76ad5911e9ac0c0a04",
  "statusCode": "0000",
  "statusReason": "Request has been submitted successfully. The CSV report will
be delivered through EML"
}
```

### 3.1.1.4 Green Button

- Function Description –

Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads, or Monthly Billing Reads for one ESIID.

Note: This is a synchronous function.

- Test URL –

<https://uat-services.smartmetertexas.net/greenbutton/>

- Production URL -

<https://services.smartmetertexas.net/greenbutton/>

- Method –

# SMART METER TEXAS™

---

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Interval Energy Data, or Daily Register Reads, or Monthly Billing Reads

- **GreenButtonRequest**

```
{
  trans_id*           string
                      pattern: [a-zA-Z0-9]{1,32}
                      Id for the green button report
  requestorID*       string
                      minLength: 1
                      maxLength: 100
                      example: KHUDAK
  requesterType*     string
                      example: REP
                      The user type of the registered SMT Portal User Id.
                      Enum:
                      Array [ 12 ]
  requesterAuthenticationID string
                      minLength: 1
                      maxLength: 18
                      Requester Company DUNs number
  ESIID*             string
                      minLength: 9
                      maxLength: 64
  startDate*         string($date)
                      pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}$
                      example: 05/10/2018
                      mm/dd/yyyy
  endDate*          string($date)
                      pattern: ^[0-9]{1,2}/[0-9]{1,2}/[0-9]{4}$
                      example: 05/10/2018
                      mm/dd/yyyy
  reportType*       string
                      Report Type can be interval or daily or monthly
}
```



# SMART METER TEXAS™

---

```

        Enum:
        Array [ 3 ]
        readingType* string
        It can be either C for Consumption or G for Generation
        Enum:
        Array [ 2 ]
        deliveryMode string
        The Asynchronous delivery mechanism for the
        greenbutton report
        Enum:
        Array [ 4 ]
        SMTTermsandConditions* string
        example: Y
        User needs to accept Terms and Conditions by sending
        the value Y or y
        Enum:
        Array [ 4 ]
    }

```

- **Error**

```

{
    errorCode      string
    errorKey       string
    errorMessage   string
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}

500	{"error": "Something went wrong. Please try again later."}
-----	--

Table 7: Return Status Codes

- Example –  
**Example 1 – Interval Green Button -Synchronous in XML format**

- **15-Minutes Interval Reads Request**

- Request

```
{
  "GreenButtonRequest":{
    "trans_id": "A1",
    "requestorID": "CSPAPIUser1",
    "requesterType": "CSP",
    "requesterAuthenticationID": "199999999999",
    "ESIID": "10443720008107413",
    "startDate": "06/20/2018",
    "endDate": "06/20/2018",
    "reportType": "I",
    "readingType": "C",
    "SMTTermsandConditions": "Y"
  }
}
```

- Synchronous Response



GreenButtonReport\_Interval.xml

- **Daily Registered Read Request**

- Request

```
{
  "GreenButtonRequest":{
    "trans_id": "A1",
    "requestorID": "CSPAPIUser1",
    "requesterType": "CSP",
    "requesterAuthenticationID": "199999999999",
    "ESIID": "10443720008107413",
```

```
"startDate": "06/20/2018",  
"endDate": "07/20/2018",  
"reportType": "D",  
"readingType": "C",  
"SMTTermsandConditions": "Y"  
}  
}
```

- Synchronous Response



GreenButtonReport\_D  
aily.xml

- **Monthly Billing Read Request**

- Request

```
{  
  "GreenButtonRequest":{  
    "trans_id": "A1",  
    "requestorID": "CSPAPIUser1",  
    "requesterType": "CSP",  
    "requesterAuthenticationID": "199999999999",  
    "ESIID": "10443720008107413",  
    "startDate": "06/20/2018",  
    "endDate": "07/20/2018",  
    "reportType": "M",  
    "readingType": "C",  
    "SMTTermsandConditions": "Y"  
  }  
}
```

- Synchronous Response



GreenButton\_Monthly  
.xml

## Example 2 – Interval Green Button - Asynchronous Request Acknowledgment

- **15-Minutes Interval Reads Request**

- Request

```
{
  "GreenButtonRequest":{
    "trans_id": "A1",
    "requestorID": "CSPAPIUser1",
    "requesterType": "CSP",
    "requesterAuthenticationID": "199999999999",
    "ESIID": "10443720008107413",
    "startDate": "04/20/2019",
    "endDate": "05/20/2019",
    "reportType": "I",
    "readingType": "C",
    "deliveryMode": "FTP",
    "SMTTermsandConditions": "Y"
  }
}
```

- Acknowledgement Response

```
{
  "trans_id": "A1",
  "correlationId": "40f82f4eb7c411e99e090a04",
  "statusCode": "0000",
  "statusReason": "Request has been submitted successfully. The report will
be delivered through FTP"
}
```

- **Daily Registered Reads Request**

- Request

```
{
  "GreenButtonRequest":{
    "trans_id": "A1",
    "requestorID": "CSPAPIUser1",
    "requesterType": "CSP",
    "requesterAuthenticationID": "199999999999",
    "ESIID": "10443720008107413",
    "startDate": "04/20/2019",
    "endDate": "05/20/2019",
  }
}
```

# SMART METER TEXAS™

---

```
"reportType": "D",  
"readingType": "C",  
"deliveryMode": "FTP",  
"SMTTermsandConditions": "Y"  
}
```

- Acknowledgement Response

```
{  
  "trans_id": "A1",  
  "correlationId": " d7c6ce40b7c811e9a3ad0a04",  
  "statusCode": "0000",  
  "statusReason": "Request has been submitted successfully. The report will  
be delivered through FTP"  
}
```

- **Monthly Billing Data Request**

- Request

```
{  
  "GreenButtonRequest":{  
    "trans_id": "A1",  
    "requestorID": "CSPAPIUser1",  
    "requesterType": "CSP",  
    "requesterAuthenticationID": "199999999999",  
    "ESIID": "10443720008107413",  
    "startDate": "04/20/2019",  
    "endDate": "05/20/2019",  
    "reportType": "M",  
    "readingType": "C",  
    "deliveryMode": "FTP",  
    "SMTTermsandConditions": "Y"  
  }  
}
```

- Acknowledgement Response

```
{  
  "trans_id": "A1",  
  "correlationId": " bcfbc278b7c811e9a3ad0a04",  
  "statusCode": "0000",  
  "statusReason": "Request has been submitted successfully. The report will
```

```
be delivered through FTP"  
}
```

### 3.1.1.5 Meter Attributes

- Function Description –

Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNs.

Note: This specific service supports a callback feature that can integrate with a customer's API.

- Test URL –

<https://uat.services.smartmetertexas.net/meterInfo/>

- Production URL -

<https://services.smartmetertexas.net/meterInfo/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Meter Attributes

- **MeterSearchRequest**

```
{  
  trans_id*           string  
                      minLength: 1  
                      maxLength: 100  
                      example: SMT_PORTAL_ID  
                      User ID registered at SMT Portal.  
  requestorID*       string  
                      example: REP  
                      The user type of the registered SMT Portal User Id.  
                      Enum:  
                      Array [ 12 ]
```

# SMART METER TEXAS™

---

```
requesterType*      string
                    maxLength: 16

                    The DUNS number associated with the
                    REP/TDSP/CSP.

requesterAuthenticationID string
                    example: FTP

                    The Asynchronous delivery mechanism for the
                    requested report. For userTypes: REP, CSP, TDSP,
                    REG the default deliveryMode is FTP and for
                    userTypes: RES, BUSINESS the default deliveryMode is
                    EML

                    Enum:
                    Array [ 6 ]
                    string
                    string
                    example: CSV
                    Format of the requested report. The default report
                    format through delivered through FTP and EML is CSV
                    and the only report format delivered through API is
                    JSON.
                    Enum:
                    Array [ 4 ]
                    string

version*
ESIIDMeterList    [
                  {
                    esiid*      string
                               minLength: 9
                               maxLength: 64
                    meterNumber string
                               minLength: 1
                               maxLength: 30
                  }
                ]

SMTTermsandConditions* string
}

}
```

- **MeterSearchResponse**

# SMART METER TEXAS™

---

```
{
  trans_id      string
  correlationId string
  statusCode    string
  statusReason string
  MeterData    [
    {
      utilityCompanyId string
      ESIID*            string
                        minLength: 9
                        maxLength: 64

      meterSerialNumber* string
                        minLength: 1
                        maxLength: 30

      utilityMeterId*   string
                        minLength: 1
                        maxLength: 30

      KWHMeterMultiplier* string
      configuredChannels* string
      manufacturerName*  string
                        minLength: 1
                        maxLength: 50

      testDate*         string
      meterClass*       string
      installationDate* string
      initialProvisionDate* string
      communicationIndicator* string
                        minLength: 1
                        maxLength: 3

      instrumentRated*  string
                        minLength: 1
                        maxLength: 1 Enum:
                        Array [ 4 ]

      currentTransformersRatio string
                        minLength: 0
                        maxLength: 16

      potentialTransformersRatio string
                        minLength: 0
                        maxLength: 16

      esiFirmwareVersion* string
                        minLength: 1
                        maxLength: 16
    }
  ]
}
```



# SMART METER TEXAS™

---

HANProtocol	string
	<i>minLength: 0</i>
	<i>maxLength: 22</i>
smartEnergyProfile*	string
	<i>minLength: 1</i>
	<i>maxLength: 22</i>
intervalSetting*	string
reverseFlowHandling*	string
	<i>minLength: 1</i>
	<i>maxLength: 1</i> Enum:
	Array [ 4 ]
DGChannel	string
disconnect*	string
	<i>minLength: 1</i>
	<i>maxLength: 1</i> Enum:
	Array [ 4 ]
meterStatus	string
	<i>minLength: 1</i>
	<i>maxLength: 1</i> Enum:
	Array [ 4 ]
meterPhases	string
meterModel	string
	<i>minLength: 0</i>
	<i>maxLength: 30</i>
revisionDate	string
}	
]	
}	
o <b>Error</b>	
{	
errorCode	string
errorKey	string
errorMessage	string
}	

- Return Status Codes –

# SMART METER TEXAS™

---

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 8: Return Status Codes

- Example –

- **Meter Information Request**

```
{
  "trans_id": "123",
  "requestorID": "SUMANTH_CSP",
  "requesterType": "CSP",
  "requesterAuthenticationID": "199999999999",

  "reportFormat": "CSV",
  "version": "L",
  "ESIIDMeterList": [
    {
      "esiid": "10443720008107413"
    },
    {
      "esiid": "10443720007526004"
    }
  ],

  "SMTTermsandConditions": "y"
}
```

- **Meter Information Response**

```
{
  "trans_id": "123",
  "correlationId": "4afcb1e8ad6711e984b60a04",
  "statusCode": "0001",
  "statusReason": "Request submitted successfully for further processing with some
faultESIIDMeters. The CSV report will be delivered through FTP",
  "faultESIIDMeters": [ {
    "esiid": "10443720007526004",
    "REASON": "ESIID NOT VISIBLE TO THIS USER"
  }
]
```

### 3.1.1.6 Premise Attributes

- Function Description –

Get Premise Attributes for one or more ESIIDs, or all ESIIDs of one DUNs.

Note: This specific service supports a callback feature that can integrate with a customer's API.

- Test URL –

<https://uat.services.smartmetertexas.net/premiseInfo/>

- Production URL -

<https://services.smartmetertexas.net/premiseInfo/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Premise Attributes

- **PremiseSearchRequest**

```
{
  trans_id*                               string
                                           pattern: [a-zA-Z0-9]{1,32}
```

# SMART METER TEXAS™

---

requestorID*	Transaction Id string <i>minLength: 1</i> <i>maxLength: 32</i> <i>example: SMT_PORTAL_ID</i>
requesterType*	User ID registered at SMT Portal. string <i>example: REP</i> The user type of the registered SMT Portal User Id. Enum: Array [ 12 ]
requesterAuthenticationID	string <i>minLength: 1</i> <i>maxLength: 18</i> <i>example: 12321313213</i>
deliveryMode	Requester Company DUNs number string <i>example: FTP</i>  The Asynchronous delivery mechanism for the requested report. For userTypes: REP, CSP, TDSP, REG the default deliveryMode is FTP and for userTypes: RES, BUSINESS the default deliveryMode is EML  Enum: Array [ 6 ]
reportFormat	string <i>example: CSV</i> Format of the requested report. The default report format through delivered through FTP and EML is CSV and the only report format delivered through API is JSON. Enum: Array [ 4 ]
version*	string <i>example: L</i>  The version of the energy data required. A-All, L-Latest
esiid	Enum: Array [ 2 ] [...]

# SMART METER TEXAS™

---

SMTTermsandConditions\* **string**  
*example: Y*

Terms and Conditions. End users needs to accept terms & conditions by sending the value as Y

Enum:  
Array [ 4 ]

}

o **PremiseSearchResponse**

{

trans_id	String	
PremiseData	[	
	{	
	utilityCompanyId	<b>string</b>
	ESIID*	<b>string</b> <i>minLength: 9 maxLength: 64</i>
	serviceVoltage*	<b>string</b> <i>minLength: 1 maxLength: 15</i>
	premiseStatus*	<b>string</b> <i>minLength: 1 maxLength: 1</i> Enum: Array [ 4 ]
	timeZone*	<b>string</b> <i>minLength: 3 maxLength: 3</i>
	fractionalHouseAddress	<b>string</b> <i>minLength: 0 maxLength: 10</i>
	leadingDirectional	<b>string</b> <i>minLength: 0 maxLength: 6</i>
	streetName*	<b>string</b> <i>minLength: 1 maxLength: 100</i>
	streetType	<b>string</b> <i>minLength: 0 maxLength: 16</i>

# SMART METER TEXAS™

---

```
trailingDirectional      string
                          minLength: 0
                          maxLength: 6
houseNumber              string
                          minLength: 0
                          maxLength: 50
unitDesignation          string
                          minLength: 0
                          maxLength: 50
locality*                string
                          minLength: 1
                          maxLength: 40
state*                    string
                          minLength: 2
                          maxLength: 2Enum:
zipcode*                  Array [ 1 ]
                          string
                          minLength: 5
                          maxLength: 5
zipcodePlus4*            string
                          minLength: 1
                          maxLength: 4
meterReadCycle*          string
loadProfile*             string
                          minLength: 1
                          maxLength: 30
rateClassCode            string
                          minLength: 0
                          maxLength: 3
AMSPProfileEffectiveDate string
revisionDate             string
}
]
```

○ **Error**

```
{
  errorCode*   string
  errorKey*    string
  errorMessage* string
}
```

- **FaultESIIDs**

```

{
  esiid      string
  reason    string
}

```

- **Acknowledgement**

```

{
  trans_id      string
  correlationId string
  statusCode    string
  statusReason  string
  faultESIIDs  [
    {
      esiid      string
      reason    string
    }
  ]
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}

500	{"error": "Something went wrong. Please try again later."}
-----	--

Table 9: Return Status Codes

- Example –

- **Premise Information Request**

```
{
  "trans_id": "123",
  "requestorID": "SUMANTH_CSP",
  "requesterType": "CSP",
  "requesterAuthenticationID": "199999999999",
  "deliveryMode": "EML",
  "version": "L",
  "esiid": [
    "10443720008107413",
    "10443720007526004"
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Premise Information Response**

```
{
  "trans_id": "123",
  "correlationId": "0bfc96a8ad5c11e9bcc30a04",
  "statusCode": "0001",
  "statusReason": "Request submitted successfully for further processing with some faultESIIDs. The CSV report will be delivered through EML",
  "faultESIIDs": [ {
    "ESIID": "10443720007526004",
    "REASON": "ESIID NOT VISIBLE TO THIS USER"
  } ]
}
```

### 3.1.1.7 On-Demand Read

- Function Description –



# SMART METER TEXAS™

---

Get current Meter Read for one ESIID at a time. Note: The daily limit for On-Demand Reads (ODRs) is up to 3,000 per Transmission and Distribution utility (TDU).

Note: This specific service supports a callback feature that can integrate with a customer's API.

- Test URL –

<https://uat.services.smartmetertexas.net/odr/>

- Production URL -

<https://services.smartmetertexas.net/odr/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – On-Demand Read

- **OnDemandMeterReadRequest**

```
{
  trans_id*           string
  requesterType*     string
  requesterAuthenticationID string
  requestorID*       string
  deliveryMode*      string
  ESIID*             string
  meterNumber        string
  SMTTermsandConditions* string
}
```

- **OnDemandMeterReadResponse**

```
{
  trans_id           string
  correlationId      string
  statusCode         string
  statusReason       string
}
```

- **Error**

```

{
  errorCode      string
  errorKey       string
  errorMessage   string
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 10: Return Status Codes*

- Example –

- **ODR Request**

```

{
  "trans_id": "123",
  "requesterType": "CSP",
  "requesterAuthenticationID": "199999999999",
  "requestorID": "SUMANTH_CSP",
  "deliveryMode": "eml",
  "ESIID": "10443720008107413",
  "SMTTermsandConditions": "Y"
}

```

- **ODR Response**

```

{

```

```
"trans_id": "123",  
"correlationId": "441888",  
"statusCode": "0",  
"statusReason": "Request submitted successfully for further processing"  
}
```

### 3.1.1.8 On-Demand Read Status

- Function Description –

Get the current status of one or more ODR requests using the appropriate Correlation IDs. Note: This is a synchronous function.

- Test URL –

<https://uat.services.smartmetertexas.net/odrstatus/>

- Production URL -

<https://services.smartmetertexas.net/odrstatus/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – On-Demand Read Status

- **ODRStatusRequest**

```
{  
  trans_id*           string  
  requestorID*        string  
  correlationId*      string  
  SMTTermsandConditions* string  
}
```

- **ODRStatusResponse**

```

{
  trans_id      string
  correlationId string
  statusCode    string
  statusReason  string
  odrRead      {
                    registeredRead  string
                    readDate        string
                }
}

```

- **Error**

```

{
  errorCode    string
  errorKey     string
  errorMessage  string
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 11: Return Status Codes*

- Example –

- **ODR Request Status**

```
{
  "trans_id": "ABC123",
  "requestorID": "NEWTDSPUSER1",
  "correlationId": "442256",
  "SMTTermsandConditions": "Y"
}
```

- **ODR Status Response**

```
{
  "trans_id": "ABC123",
  "correlationId": "442256",
  "statusCode": "PEN",
  "statusReason": "Request Submitted by API",
  "odrRead": {
    "registeredRead": ,
    "readDate": "08/05/2019 22:13:03"
  }
}
```

### 3.1.1.9 Report Status

- Function Description –

Get the current status for Ad-Hoc or Subscription requests. Note: This is a synchronous function.

- Test URL –

<https://uat.services.smartmetertexas.net/reportrequeststatus/>

- Production URL -

<https://services.smartmetertexas.net/reportrequeststatus/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

# SMART METER TEXAS™

---

- Body Parameter – Report Status

- **ODRStatusRequest**

```

{
  trans_id*                string
                           pattern: [a-zA-Z0-9]{1,32}
                           Transaction Id for the Meter Information
                           API
  requestorID*            string
                           minLength: 1
                           maxLength: 100
                           example: SMT_PORTAL_ID
                           User ID registered at SMT Portal.
  correlationId*          string
  SMTTermsandConditions* string
                           example: Y
                           Terms and Conditions. Subscription needs
                           to accept terms & conditions by sending
                           the value as Y
                           Enum:
                           Array [ 4 ]
}

```

- Data Parameter Response – Report Status Response

- **StatusReportResponse**

```

{
  trans_id                string
  AdhocStatusResponse    [{
                           correlationId    string
                           reportType       string
                           requestDate      string
                           status           string
                           statusCode        string
                           statusReason     string
                           }
  SubscriptionStatusResponse  [{
                           subscriptionId   string
                           reportType       string
                           subscriptionFrequency string
}
}

```

# SMART METER TEXAS™

---

```

subscriptionType      string
status                string
lastReportGenerationDate string
statusCode            string
statusReason          string

```

```

    }
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 12: Return Status Codes*

- Example –

- **Report Status Request**

```

{
  "trans_id": "A12",
  "requestorID": "NEWTDSPUSER1",
  "serviceType": "ADHOC",
  "correlationId": "6dce368cb7cc11e9a3ad0a04",
  "SMTTermsandConditions": "Y"
}

```

- **Report Status Response**

```
{
  "trans_id": "A12",
  "AdhocStatusResponse" : [ {
    "correlationId": "40f82f4eb7c411e99e090a04",
    "reportType" : "SMTMeterDataSearch",
    "requestDate": "2019-08-05 03:09:08",
    "status": "SUBMITTED"
  } ]
}
```

### 3.1.1.10 *New Subscription*

- Function Description –
  - Create a new subscription for one or more ESIDs
- Test URL –  
<https://uat.services.smartmetertexas.net/NewSubscription/>
- Production URL -  
<https://services.smartmetertexas.net/NewSubscription/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – New Subscription

- **NewSubscription**

```
{
  trans_id*                               string
                                          minLength: 1
                                          maxLength: 32
                                          pattern: [a-zA-Z0-9]{1,32}
                                          Transaction Id for the Subscription
}
```



# SMART METER TEXAS™

---

requestorID*	string <i>minLength: 1</i> <i>maxLength: 32</i> <i>example: SMT_PORTAL_ID</i>
requesterType*	Subscription User ID string <i>example: SMT_PORTAL_ID</i> Enum: Array [ 12 ]
requesterAuthenticationID	string <i>minLength: 1</i> <i>maxLength: 18</i> <i>example: 12321313213</i>
subscriptionType*	Requester Company DUNs number stringEnum: Array [ 4 ]
historicalSubscriptionDuration	integer  ONLY APPLICABLE FOR HISTORICAL BACKFILL SUBSCRIPTION (REPENROLL/ CSPENROLL) DEFINES NUMBER OF MONTHS FOR WHICH HISTORICAL DATA IS REQUIRED.
reportFormat*	Enum: Array [ 5 ] string Format of the requested subscription report. The default report format through delivered through FTP and EML is CSV and the only report format delivered through API is JSON. Enum: Array [ 8 ]
dataType*	stringEnum: Array [ 7 ]
deliveryMode*	string  The Asynchronous delivery mechanism for the requested subscription report. For userTypes: REP, CSP, TDSP, REG the default deliveryMode is FTP and for userTypes: RES, BUSINESS the default deliveryMode is EML  Enum: Array [ 6 ]

reportFrequency **string**  
The frequency for the requested subscription report. A mandatory field for subscriptionType SCHEDULE  
Enum:  
Array [ 4 ]

SMTTermsandConditions\* **string**  
*example: Y*  
  
Terms and Conditions. Subscription needs to accept terms & conditions by sending the value as Y  
  
Enum:  
Array [ 4 ]  
[...]

ESIID  
}

o **NewSubscriptionResponse**

{  
    trans\_id **string**  
            *minLength: 1*  
            *maxLength: 32*  
            *pattern: [a-zA-Z0-9]{1,32}*  
            Id for the Subscription  
    subscriptionNumber **string**  
    statusCode **string**  
    statusReason **string**  
    CustomerESIIDFaultList [...]  
    CustomerDUNSFaultList [...]  
}

o **CustomerESIIDFault**

{  
    esiid **string**  
    reasonCode **string**  
}

- **CustomerDUNSFault**

```
{
  duns          string
  reasonCode    string
}
```

- **Error**

```
{
  errorCode      string
                 maxLength: 4
  errorKey       string
                 maxLength: 25
  errorMessage   string
                 maxLength: 128
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 13: Return Status Codes*

- Example –

- **New Subscription Request**

```
{
  "trans_id": "A1",
```

```
"requestorID": "CSP-ADMIN",  
"requesterType": "CSP",  
"requesterAuthenticationID": "079696342",  
"subscriptionType": "CSPENROLL",  
"historicalSubscriptionDuration": 3,  
"reportFormat": "LSE",  
"dataType": "INTERVAL",  
"deliveryMode": "FTP",  
"SMTTermsandConditions": "Y"  
}
```

- **New Subscription Response**

```
{  
  "trans_id": "A1",  
  "subscriptionNumber": "8E5C4B6451F40096E0530A0403227FE4",  
  "statusCode": "0000",  
  "statusReason": "Success"  
}
```

### 3.1.1.11 *Unsubscribe / Cancel Subscription*

- Function Description –

Unsubscribe or Cancel an existing subscription on a one subscription per request basis.

- Test URL –

<https://uatservices.smartmetertexas.net/UnSubscription/>

- Production URL -

<https://services.smartmetertexas.net/UnSubscription/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Unsubscribe / Cancel Subscription

- **UnsubscriptionRequest**

```
{
  trans_id*                string
                           pattern: [a-zA-Z0-9]{1,32}
                           Transaction Id for the Subscription

  requestorID*             string
                           minLength: 1
                           maxLength: 32
                           example: SMT_PORTAL_ID
                           Subscription User ID

  requesterAuthenticationID string
                           minLength: 1
                           maxLength: 18
                           example: 12321313213
                           Requester Company DUNS number

  subscriptionNumber*      string
  SMTTermsandConditions*   string
                           example: Y
                           Terms and Conditions. Subscription needs to
                           accept terms & conditions by sending the value
                           as Y

  ESIID                    Enum:
                           Array [ 4 ]
                           [...]
}
```

- **UnsubscriptionResponse**

```
{
  trans_id    string
  statusCode  string
  statusReason string
}
```

- **Error**

```
{
  errorCode string
            maxLength: 4
}
```

# SMART METER TEXAS™

---

```
errorKey      string
               maxLength: 25
errorMessage  string
               maxLength: 128
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 14: Return Status Codes

- Example –

- **UnSubscription Request**

```
{
  "trans_id": "A1",
  "requestorID": "CSP-ADMIN",
  "subscriptionNumber": "8E5C4B6451F40096E0530A0403227FE4",
  "SMTTermsandConditions": "Y"
}
```

- **UnSubscription Response**

```
{
  "trans_id": "A1",
```

```
"statusCode": "0000",  
"statusReason": "Success"  
}
```

### 3.1.1.12 *List Subscriptions*

- Function Description –

Get a list of subscriptions.

- Test URL –

<https://uat.services.smartmetertexas.net/Mysubscriptions/>

- Production URL -

<https://services.smartmetertexas.net/Mysubscriptions/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – List Subscriptions

- **SubscriptionList**

```
{  
  subscriptionNumber    string  
  startDate              string  
  reportFormat          string  
  deliveryType          string  
  dataType              string  
  status                 string  
  lastSuccessfulReportDate string  
}
```

- **SubscriptionList**

```
{  
  trans_id              string  
  statusCode            string  
}
```

```

        statusReason    string
        SubscriptionsList [...]
    }

```

- **Error**

```

{
    errorCode    string
                 maxLength: 4
    errorKey     string
                 maxLength: 25
    errorMessage string
                 maxLength: 128
}

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 15: Return Status Codes*

- Example –

- **List My Subscriptions Request**

```

{
    "trans_id": "A123",
    "requestorID": "CSP-ADMIN",
}

```



# SMART METER TEXAS™

---

```
"requesterAuthenticationID": "079696342",  
"SMTTermsandConditions": "y"  
}
```

- **List My Subscriptions Response**

```
{  
  "trans_id": "A123",  
  "statusCode": "0000",  
  "statusReason": "Success",  
  "subscriptionList": [  
    {  
      "subscriptionNumber": "8DBE0A8D6B0D00F8E0530A040322C1D1",  
      "startDate": "2019-07-16 14:12:28",  
      "reportformat": "CSV",  
      "deliveryType": "FTP",  
      "dataType": "DLY",  
      "status": "ACT",  
      "lastSuccessfulReportDate": null  
    },  
    {  
      "subscriptionNumber": "8DB8E9D8C22A0094E0530A04032238C3",  
      "startDate": "2019-07-16 08:05:25",  
      "reportformat": "JSON",  
      "deliveryType": "API",  
      "dataType": "INT",  
      "status": "ACT",  
      "lastSuccessfulReportDate": null  
    },  
    {  
      "subscriptionNumber": "8DCE794681BF00BEE0530A0403223819",  
      "startDate": "2019-07-17 09:48:46",  
      "reportformat": "LSE",  
      "deliveryType": "FTP",  
      "dataType": "HML",  
      "status": "ACT",  
      "lastSuccessfulReportDate": null  
    },  
    {  
      "subscriptionNumber": "8E5C4B6451F40096E0530A0403227FE4",  
      "startDate": "2019-07-24 11:00:41",  
      "reportformat": "LSE",  
      "deliveryType": "FTP",  
    }  
  ]  
}
```

```
"dataType": "INT",
"status": "CLS",
"lastSuccessfulReportDate": null
}
]
}
```

### 3.1.1.13 New Energy Data Sharing Agreement

- Function Description –

Initiate a new Energy Data Sharing Agreement for one or more ESIIDs.

- Test URL –

<https://uat.services.smartmetertexas.net/NewAgreement/>

- Production URL -

<https://services.smartmetertexas.net/NewAgreement/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – New Agreement

- **NewAgreement**

```
{
  trans_id*           string
                      pattern: [a-zA-Z0-9]{1,32}
                      Id for the agreement between CSP and retail
                      customer
  requestorID*       string
                      minLength: 1
                      maxLength: 32
                      example: SMT_CSP_PORTAL_ID
}
```

	CSP User ID registered at SMT Portal, who initiates agreement.
requesterAuthenticationID*	<b>string</b> <i>minLength: 1</i> <i>maxLength: 18</i> <i>example: 12321313213</i>
retailCustomerEmail*	Requester Company DUNs number <b>string</b> <i>minLength: 1</i> <i>maxLength: 64</i> <i>example: XXXXX@gmail.com</i>
agreementDuration*	End customer email address. <b>integer</b> <i>example: 9</i> Duration of agreement in months. Enum: Array [ 6 ]
customerLanguagePreference	<b>string</b> <i>example: ENGLISH</i> Language preference to send agreement notification emails. Enum: Array [ 4 ]
customerMeterList*	[...]
SMTTermsandConditions*	<b>string</b> <i>example: Y</i> Terms and Conditions. CSP needs to accept terms & conditions by sending the value as Y Enum: Array [ 4 ]
}	

- **NewAgreementResponse**

{	
trans_id*	<b>string</b> <i>pattern: [a-zA-Z0-9]{1,32}</i> Id for the agreement between CSP and retail customer

```
agreementNumber* integer
                    example: 100212
statusCode*        string
                    maxLength: 4
                    example: ACK
statusReason*     string
                    maxLength: 256
```

```
}
```

- **NewAgreementFaultResponse**

```
{
  statusCode*       string
                    maxLength: 4
                    example: FLR
  statusReason*    string
                    maxLength: 256
  customerMeterFaultList [...]
```

```
}
```

- **Error**

```
{
  errorCode        string
                    maxLength: 4
  errorKey         string
                    maxLength: 25
  errorMessage     string
                    maxLength: 128
```

```
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

# SMART METER TEXAS™

---

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 16: Return Status Codes

- Example –

- **New Agreement Request**

```
{
  "trans_id": "64576457",
  "requestorID": "UATCSPAPIDee6",
  "requesterAuthenticationID": "9678588888869",
  "retailCustomerEmail": "delluri281@gmail.com",
  "agreementDuration": 9,
  "customerLanguagePreference": "ENGLISH",
  "customerMeterList": [
    {
      "ESIID": "10032789400250865",
      "meterNumber": "144054330",
      "PUCTRORNumber": 10004
    }
  ],
  "SMTTermsandConditions": "Y"
}
```

- **New Agreement Response**

```
{
  "trans_id": "64576457",
  "agreementNumber": "119098",
  "statusCode": "ACK",
}
```

```
"statusReason": "Success"  
}
```

### 3.1.1.14 List ESIDs per Energy Data Sharing Agreement

- Function Description –

Get a list of ESIDs associated with a specific Energy Data Sharing Agreement.

- Test URL –

<https://uat.services.smartmetertexas.net/AgreementESIDs/>

- Production URL -

<https://services.smartmetertexas.net/AgreementESIDs/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Agreement ESIDs

- **AgreementESIDs**

```
{  
  trans_id*                string  
                           pattern: [a-zA-Z0-9]{1,32}  
                           Id for the agreement between CSP and retail customer  
  requestorID*            string  
                           minLength: 1  
                           maxLength: 32  
                           example: SMT_CSP_PORTAL_ID  
                           CSP User ID registered at SMT Portal.  
  requesterAuthenticationID* string  
                           minLength: 1  
                           maxLength: 18  
                           example: SMT_CSP_PORTAL_ID  
                           Requester Company DUNS number  
  agreementNumber*       integer  
                           example: 100212
```

SMTTermsandConditions\* **string**  
*example: Y*

Terms and Conditions. CSP needs to accept terms & conditions by sending the value as Y

Enum:  
Array [ 4 ]

}

○ **TerminateAgreementResponse**

{

trans\_id **string**  
*pattern: [a-zA-Z0-9]{1,32}*  
Id for the agreement between CSP and retail customer

statusCode **string**  
*maxLength: 4*  
*example: 0*

statusMessage **string**  
*maxLength: 128*  
*example: Request submitted for your agreement terminated request*

terminateInitiatedList [...] **string**  
agreementFaultList [...]

}

○ **AgreementTerminateInitiated**

{

agreementNumber\* **integer**  
*example: 100212*

statusCode\* **string**  
*maxLength: 4*  
*example: ACK*

statusReason\* **string**  
*maxLength: 256*

}

○ **AgreementFault**

{

agreementNumber **integer**  
*example: 100212*

retailCustomerEmail **string**  
*minLength: 1*  
*maxLength: 100*  
*example: XXXXX@gmail.com*  
Customer Email Id to send the Agreement Email.

reason **string**  
*maxLength: 256*  
*example: Agreement & CustomerEmailId entered do not belong to CSP.*

Fault Message for the entered input combination of agreementNumber & retailCustomerEmail

}

- **Error**

{

errorCode **string**  
*maxLength: 4*

errorKey **string**  
*maxLength: 25*

errorMessage **string**  
*maxLength: 128*

}

- Return Status Codes –

The following table lists the Status Code and associated Description:



# SMART METER TEXAS™

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 17: Return Status Codes

- Example –

- **Agreement ESIIDs Request**

```
{
  "trans_id": "76876876u8787",
  "requestorID": "UATCSPAPIDee6",
  "requesterAuthenticationID": "9678588888869",
  "agreementNumber": 119098,
  "SMTTermsandConditions": "Y"
}
```

- **Agreement ESIIDs Response**

```
{
  "trans_id": "76876876u8787",
  "statusCode": "0000",
  "statusReason": "Success",
  "AgreementESIIDLList": [ {
    "ESIID": "10032789400250865",
    "meterNumber": "144054330",
    "PUCTORNumber": "10004",
    "status": "Pending - Customer Authorization",
    "startDate": "07/23/2019",
    "endDate": "04/23/2020"
  }
]
```

## 3.1.1.15 Terminate Energy Data Sharing Agreement

- Function Description –

Terminate one or more Energy Data Sharing Agreements.

- Test URL –

<https://uat.services.smartmetertexas.net/Terminateagreement/>

- Production URL –

<https://services.smartmetertexas.net/Terminateagreement/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Terminate Agreement

- **TerminateAgreement**

```
{
  trans_id*           string
                      pattern: [a-zA-Z0-9]{1,32}
  requestorID*       string
                      minLength: 1
                      maxLength: 32
                      example: SMT_CSP_PORTAL_ID
                      CSP User ID registered at SMT Portal.
  requesterAuthenticationID* string
                      minLength: 1
                      maxLength: 18
                      example: SMT_CSP_PORTAL_ID
                      Requester Company DUNs number
  agreementList*    [...]
  SMTTermsandConditions* string
                      example: Y
```

Terms and Conditions. CSP needs to accept terms & conditions by sending the value as Y

Enum:  
Array [ 4 ]

}

○ **TerminateAgreementResponse**

```
{
  trans_id          string
                   pattern: [a-zA-Z0-9]{1,32}
                   Id for the agreement between CSP and retail
                   customer
  statusCode        string
                   maxLength: 4
                   example: 0
  statusMessage    string
                   maxLength: 128
                   example: Request submitted for your agreement
                   terminated request
  terminateInitiatedList [...]
  agreementFaultList  [...]
}
```

○ **AgreementTerminateInitiated**

```
{
  agreementNumber* integer
                   example: 100212
  statusCode*      string
                   maxLength: 4
                   example: ACK
  statusReason*   string
                   maxLength: 256
}
```

○ **AgreementFault**

```
{
  agreementNumber integer
                   example: 100212
  retailCustomerEmail string
                   minLength: 1
                   maxLength: 100
                   example: XXXXX@gmail.com
}
```

reason                      Customer Email Id to send the Agreement Email.  
string  
maxLength: 256  
example: Agreement & CustomerEmailId entered do not belong to CSP.

Fault Message for the entered input combination of agreementNumber & retailCustomerEmail

}

○ **Error**

```
{  
  errorCode            string  
                      maxLength: 4  
  errorKey            string  
                      maxLength: 25  
  errorMessage        string  
                      maxLength: 128  
}
```

• Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 18: Return Status Codes*

- Example –

- **Terminate Agreement Request**

```
{
  "trans_id": "6827364j87987",
  "requestorID": "UATCSPAPIDee6",
  "requesterAuthenticationID": "9678588888869",
  "agreementList": [
    {
      "agreementNumber": 119098,
      "retailCustomerEmail": "delluri281@gmail.com"
    }
  ],
  "SMTTermsandConditions": "Y"
}
```

- **Terminate Agreement Response**

```
{
  "trans_id": "6827364j87987",
  "statusCode": "0000",
  "statusReason": "Agreement termination successful",
  "terminateInitiatedList": [ {
    "agreementNumber": 119098,
    "statusCode": "ACK",
    "statusReason": "Success"
  } ]
}
```

## 3.1.1.16 Status of Energy Data Sharing Agreement

- Function Description –

Get a status of one or more Energy Data Sharing Agreements.

- Test URL –

<https://uat.services.smartmetertexas.net/myagreements/>

- Production URL -

<https://services.smartmetertexas.net/myagreements/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Agreement Status

- **AgreementStatus**

```
{
  agreementNumber*      integer
                        example: 100212
  startDate*            string
                        minLength: 1
                        maxLength: 30
  endDate*              string
                        minLength: 1
                        maxLength: 30
  CSPName*              string
                        minLength: 1
                        maxLength: 100
                        example: cspbrand1
                        The name of the CSP
```

```
    requesterAuthenticationID*  string
                                minLength: 1
                                maxLength: 18
                                example: SMT_CSP_PORTAL_ID
                                Requester Company DUNs number
    retailCustomerEmail*        string
                                minLength: 1
                                maxLength: 100
                                example: XXXXX@gmail.com
                                Customer Email Id to send the Agreement Email.
    status*                      string
                                example: Active, About to expire, Terminated in
                                last 45 days, Expire in last 45 days, Not accepted
                                list
                                The status Message for Agreement
                                Enum:
                                Array [ 11 ]
}
o Error
{
  errorCode    string
               maxLength: 4
  errorKey     string
               maxLength: 25
  errorMessage string
               maxLength: 128
}
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 19: Return Status Codes*

### 3.1.1.17 List of Energy Data Sharing Agreement

- Function Description –

Get a list of one or more Energy Data Sharing Agreements.

- Test URL –

<https://uat.services.smartmetertexas.net/myagreements/>

- Production URL -

<https://services.smartmetertexas.net/myagreements/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – My Agreements

- **MyAgreements**

```

{
  trans_id*           string
                      pattern: [a-zA-Z0-9]{1,32}
                      Id for the agreement between CSP and retail customer

```



# SMART METER TEXAS™

---

requestorID\* *string*  
*minLength: 1*  
*maxLength: 32*  
*example: SMT\_CSP\_PORTAL\_ID*  
CSP User ID registered at SMT Portal.

requesterAuthenticationID\* *string*  
*minLength: 1*  
*maxLength: 18*  
*example: SMT\_CSP\_PORTAL\_ID*  
Requester Company DUNs number

agreementNumber *integer*  
*example: 100212*

statusReason *string*  
*example: ACT*

Status of Agreement for Customer. PEN - Pending, ACT - Active, COM - Completed, NACOM - Not Accepted Completed

Enum:  
Array [ 4 ]

SMTTermsandConditions\* *string*  
*example: Y*

Terms and Conditions. CSP needs to accept terms & conditions by sending the value as Y

Enum:  
Array [ 4 ]

}

## o MyAgreementsResponse

{

trans\_id *string*  
*pattern: [a-zA-Z0-9]{1,32}*  
Id for the agreement between CSP and retail customer

statusCode *string*  
*maxLength: 4*

statusReason *string*  
*maxLength: 256*

agreementStatusList [...]

}

## o Error

{

errorCode      string  
                  maxLength: 4  
errorKey       string  
                  maxLength: 25  
errorMessage   string  
                  maxLength: 128

}

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 20: Return Status Codes*

- Example –

- **My Agreements Request**

```
{  
  "trans_id": "45674657f7376893hjh",  
  "requestorID": "UATCSPAPIDee6",  
  "requesterAuthenticationID": "9678588888869",  
  "agreementNumber": 119098,  
  "statusReason": "PEN",  
  "SMTTermsandConditions": "Y"  
}
```

- **My Agreements Response**

```
{
  "trans_id": "45674657f7376893hjh",
  "statusCode": "0000",
  "statusReason": "Success",
  "agreementStatusList": [ {
    "agreementNumber": 119098,
    "startDate": "07/23/2019",
    "endDate": "04/23/2020",
    "CSPName": "DUNS3",
    "requesterAuthenticationID": "9678588888869",
    "retailCustomerEmail": "delluri281@gmail.com",
    "status": "Pending - Customer Authorization"
  }
]
```

## 3.2 SOAP Functions

### 3.2.1 Ad-Hoc

#### 3.2.1.1 *Energy Data (15-Minute Interval Data, or Daily Register Reads, or Monthly Billing Reads)*

- Function Description –

Get 15-Minute Interval Consumption and Generation Data, or Daily Register Reads, or Monthly Billing Reads for one or more ESIIDs or all ESIIDs of one DUNS.

- Test URL –

<https://uat.services.smartmetertexas.net/energydata/>

- Production URL -

<https://services.smartmetertexas.net/energydata/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameter
  - **Energy Data Request**

```
<xsd:complexType name="EnergyDataRequest">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requestorID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requesterType">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="REP"/>
          <xsd:enumeration value="RES"/>
          <xsd:enumeration value="BUSINESS"/>
          <xsd:enumeration value="CSP"/>
          <xsd:enumeration value="TDSP"/>
          <xsd:enumeration value="REG"/>
          <xsd:enumeration value="rep"/>
          <xsd:enumeration value="res"/>
          <xsd:enumeration value="business"/>
          <xsd:enumeration value="csp"/>
          <xsd:enumeration value="tdsp"/>
          <xsd:enumeration value="reg"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

# SMART METER TEXAS™

---

```
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0" name="requesterAuthenticationID">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="18"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="startDate">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="10"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="endDate">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="10"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0" name="deliveryMode">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="FTP"/>
                <xsd:enumeration value="EML"/>
                <xsd:enumeration value="API"/>
                <xsd:enumeration value="ftp"/>
                <xsd:enumeration value="eml"/>
                <xsd:enumeration value="api"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
```

# SMART METER TEXAS™

---

```
<xsd:element minOccurs="0" name="reportFormat">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="MARS"/>
      <xsd:enumeration value="CSV"/>
      <xsd:enumeration value="JSON"/>
      <xsd:enumeration value="mars"/>
      <xsd:enumeration value="csv"/>
      <xsd:enumeration value="json"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="version">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="A"/>
      <xsd:enumeration value="L"/>
      <xsd:enumeration value="a"/>
      <xsd:enumeration value="l"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="readingType">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="C"/>
      <xsd:enumeration value="G"/>
      <xsd:enumeration value="A"/>
      <xsd:enumeration value="c"/>
      <xsd:enumeration value="g"/>
      <xsd:enumeration value="a"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element maxOccurs="1" minOccurs="1" name="ESIIDList"
type="bons1:ESIIDList">
</xsd:element>
```

```
<xsd:element minOccurs="1" name="SMTTermsandConditions">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Y"/>
      <xsd:enumeration value="N"/>
      <xsd:enumeration value="y"/>
      <xsd:enumeration value="n"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
```

- Body Parameter Response –

### **Asynchronous Response**

```
<xsd:complexType name="Acknowledgement">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="correlationId"
type="xsd:string">
    </xsd:element>
    <xsd:element minOccurs="0" name="statusCode"
type="xsd:integer">
    </xsd:element>
    <xsd:element minOccurs="0" name="statusReason">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

```
        </xsd:element>
        <xsd:element maxOccurs="1" minOccurs="1"
name="faultESIIDList" type="bons1:FaultESIIDList">
        </xsd:element>

    </xsd:sequence>
</xsd:complexType>
```

## Synchronous Response – 15-Minutes Interval Data

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterUsageSOAP"
xmlns:bons1="http://schemas.esb.ams.com/meterusagelib">
    <xsd:import namespace="http://schemas.esb.ams.com/meterusagelib"
schemaLocation="IntervalEnergyDataList.xsd"/>

    <xsd:complexType name="IntervalEnergyDataResponse">
        <xsd:sequence>
            <xsd:element minOccurs="1" name="trans_id">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
            <xsd:element minOccurs="1" name="esiid">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:maxLength value="64"/>
                        <xsd:minLength value="17"/>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>

            <xsd:element maxOccurs="1" minOccurs="1" name="energyDataList"
type="bons1:IntervalEnergyDataList">
```



# SMART METER TEXAS™

---

```
</xsd:element>

</xsd:sequence>
</xsd:complexType>

</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">
  <xsd:include schemaLocation="IntervalEnergyData.xsd"/>

  <xsd:complexType name="IntervalEnergyDataList">
    <xsd:sequence maxOccurs="unbounded" minOccurs="1">
      <xsd:element maxOccurs="unbounded" minOccurs="1"
name="energyData" type="bons0:IntervalEnergyData">
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>

</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">

  <xsd:complexType name="IntervalEnergyData">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="DT">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

```
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="RevTS">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="10"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="RT">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="1"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="RD" type="xsd:string">
    </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

## Synchronous Response – Daily Registered Read

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterUsageSOAP"
xmlns:bons1="http://schemas.esb.ams.com/meterusagelib">
    <xsd:import namespace="http://schemas.esb.ams.com/meterusagelib"
schemaLocation="DailyRegisteredReadList.xsd"/>

    <xsd:complexType name="DailyRegisteredReadResponse">
        <xsd:sequence>
            <xsd:element minOccurs="1" name="trans_id">
```

# SMART METER TEXAS™

---

```
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="esiid">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="64"/>
      <xsd:minLength value="17"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

  <xsd:element maxOccurs="1" minOccurs="1"
name="registeredReadList" type="bons1:DailyRegisteredReadList">
  </xsd:element>

</xsd:sequence>
</xsd:complexType>

</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">
  <xsd:include schemaLocation="DailyRegisteredRead.xsd"/>

  <xsd:complexType name="DailyRegisteredReadList">
    <xsd:sequence maxOccurs="unbounded" minOccurs="1">
      <xsd:element maxOccurs="unbounded" minOccurs="1"
name="registeredRead" type="bons0:DailyRegisteredRead">
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
```

# SMART METER TEXAS™

---

```
</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">

  <xsd:complexType name="DailyRegisteredRead">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="readDate">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="revisionDate">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="startReading" type="xsd:string">

    </xsd:element>
    <xsd:element minOccurs="1" name="endReading" type="xsd:string">

    </xsd:element>
    <xsd:element minOccurs="1" name="energyDataKwh"
type="xsd:string">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

## Synchronous Response – Monthly Billing Information

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterUsageSOAP"
xmlns:bons1="http://schemas.esb.ams.com/meterusagelib">
  <xsd:import namespace="http://schemas.esb.ams.com/meterusagelib"
schemaLocation="MonthlyBillingDataList.xsd"/>

  <xsd:complexType name="MonthlyBillingDataResponse">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="esiid">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="64"/>
            <xsd:minLength value="17"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>

      <xsd:element maxOccurs="1" minOccurs="1" name="billingDataList"
type="bons1:MonthlyBillingDataList">
        </xsd:element>

    </xsd:sequence>
  </xsd:complexType>

</xsd:schema>
```

# SMART METER TEXAS™

---

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">
  <xsd:include schemaLocation="MonthlyBillingData.xsd"/>

  <xsd:complexType name="MonthlyBillingDataList">
    <xsd:sequence maxOccurs="unbounded" minOccurs="1">
      <xsd:element maxOccurs="unbounded" minOccurs="1"
name="billingData" type="bons0:MonthlyBillingData">
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>

</xsd:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://schemas.esb.ams.com/meterusagelib"
xmlns:bons0="http://schemas.esb.ams.com/meterusagelib">

  <xsd:complexType name="MonthlyBillingData">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="startDate">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="endDate">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

```
</xsd:element>
<xsd:element minOccurs="1" name="revisionDate">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="10"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="actualkWh" type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="meteredKW" type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="billedKW" type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="meteredKVA" type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="billedKVA" type="xsd:string">
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

# SMART METER TEXAS™

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 21: Return Status Codes*

- Example –
  - **15-Minutes Interval Data**
    - Request with no delivery mode (Synchronous response)

```

<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:met="http://schemas.esb.ams.com/meterusagesource">
  <soapenv:Header/>
  <soapenv:Body>
    <met:processIntervalEnergyData>
      <EnergyDataRequest>
        <trans_id>A12</trans_id>
        <requestorID>CSPAPIUser1</requestorID>
        <requesterType>CSP</requesterType>
        <!--Optional:-->
        <requesterAuthenticationID>199999999999</requesterAuthenticationID>
        <startDate>04/20/2019</startDate>
        <endDate>04/20/2019</endDate>
        <!--Optional:-->
        <!--<deliveryMode>JSON</deliveryMode>
      :-->
      <!--Optional:-->
      <!-- <reportFormat>?</reportFormat>
    -->
    <version>L</version>
    <!--Optional:-->

```



```
<readingType>C</readingType>
<ESIIDList>
  <!--1 or more repetitions:-->
  <ESIID>10443720008107413</ESIID>
</ESIIDList>
<SMTTermsandConditions>Y</SMTTermsandConditions>
</EnergyDataRequest>
</met:processIntervalEnergyData>
</soapenv:Body>
</soapenv:Envelope>
```

▪ **Response**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:processIntervalEnergyDataResponse
      xmlns:NS1="http://schemas.esb.ams.com/meterusagesource">
      <IntervalEnergyDataSyncResponse>
        <trans_id>A12</trans_id>
        <esiid>10443720008107413</esiid>
        <energyDataList>
          <energyData>
            <DT>04/20/2019</DT>
            <RevTS>04/21/2019 09:49:58</RevTS>
            <RT>C</RT>
            <RD>.2-A,.24-A,.221-A,.232-A,.21-A,.163-A,.192-A,.202-A,,,,,.185-A,.21-
A,.188-A,.174-A,.2-A,.214-A,.199-A,.19-A,.187-A,.18-A,.203-A,.192-A,.164-A,.196-
A,.196-A,.196-A,.212-A,.16-A,1.177-A,1.225-A,1.246-A,.325-A,.211-A,.191-A,.253-
A,.24-A,.197-A,.22-A,.222-A,.211-A,.234-A,.208-A,.18-A,.203-A,.221-A,.229-A,.236-
A,.254-A,.217-A,.275-A,.233-A,.239-A,.253-A,.271-A,.26-A,.286-A,.282-A,.168-A,.218-
A,.209-A,.762-A,.586-A,.222-A,.211-A,.708-A,.275-A,.474-A,.851-A,.475-A,.23-A,1.769-
A,1.566-A,1.492-A,.658-A,.92-A,.821-A,.763-A,.637-A,.668-A,.275-A,.666-A,.731-
A,.307-A,1.17-A,.662-A,.246-A,.232-A,.56-A,.794-A,.353-A,.259-A,.193-A,.2-A,.199-
A,.223-A,.65-A,.714-A,.329-A</RD>
          </energyData>
        </energyDataList>
      </IntervalEnergyDataSyncResponse>
```

```
</NS1:processIntervalEnergyDataResponse>
</soapenv:Body>
</soapenv:Envelope>
```

- **Daily Registered Reads**

- Request with delivery mode email and report format CSV

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:met="http://schemas.esb.ams.com/meterusagesource">
```

```
<soapenv:Header/>
```

```
<soapenv:Body>
```

```
<met:processDailyRegisteredReads>
```

```
<EnergyDataRequest>
```

```
<trans_id>A12</trans_id>
```

```
<requestorID>CSPAPIUser1</requestorID>
```

```
<requesterType>CSP</requesterType>
```

```
<!--Optional:-->
```

```
<requesterAuthenticationID>199999999999</requesterAuthenticationID>
```

```
<startDate>04/20/2019</startDate>
```

```
<endDate>05/20/2019</endDate>
```

```
<!--Optional:-->
```

```
<deliveryMode>EML</deliveryMode>
```

```
<!--Optional:-->
```

```
<reportFormat>CSV</reportFormat>
```

```
<version>L</version>
```

```
<!--Optional:-->
```

```
<readingType>C</readingType>
```

```
<ESIIDList>
```

```
<!--1 or more repetitions:-->
```

```
<ESIID>10443720008107413</ESIID>
```

```
</ESIIDList>
```

```
<SMTTermsandConditions>Y</SMTTermsandConditions>
```

```
</EnergyDataRequest>
```

```
</met:processDailyRegisteredReads>
```

```
</soapenv:Body>
```

```
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:processDailyRegisteredReadsResponse
xmlns:NS1="http://schemas.esb.ams.com/meterusagesource">
      <Acknowledgement>
        <trans_id>A12</trans_id>
        <correlationId>9dd195a0b86b11e9ac0c0a04</correlationId>
        <statusCode>0000</statusCode>
        <statusReason>Request has been submitted successfully. The CSV report will
be delivered through EML</statusReason>
      </Acknowledgement>
    </NS1:processDailyRegisteredReadsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

- **Monthly Billing Information**

- Request with delivery mode FTP and Report format CSV

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:met="http://schemas.esb.ams.com/meterusagesource">
  <soapenv:Header/>
  <soapenv:Body>
    <met:processMonthlyBillingInformation>
      <EnergyDataRequest>
        <trans_id>A12</trans_id>
        <requestorID>CSPAPIUser1</requestorID>
        <requesterType>CSP</requesterType>
        <!--Optional:-->
        <requesterAuthenticationID>19999999999</requesterAuthenticationID>
        <startDate>04/20/2019</startDate>
        <endDate>05/20/2019</endDate>
        <!--Optional:-->
        <deliveryMode>FTP</deliveryMode>
        <!--Optional:-->
        <reportFormat>CSV</reportFormat>
        <version>L</version>
        <!--Optional:-->
        <readingType>C</readingType>
```

```
<ESIIDList>
  <!--1 or more repetitions:-->
  <ESIID>10443720008107413</ESIID>
</ESIIDList>
<SMTTermsandConditions>Y</SMTTermsandConditions>
</EnergyDataRequest>
```

```
</met:processMonthlyBillingInformation>
</soapenv:Body>
</soapenv:Envelope>
```

▪ **Response**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:processMonthlyBillingInformationResponse
xmlns:NS1="http://schemas.esb.ams.com/meterusagesource">
      <Acknowledgement>
        <trans_id>A12</trans_id>
        <correlationId>1fbe0a94b86c11e9ac0c0a04</correlationId>
        <statusCode>0000</statusCode>
        <statusReason>Request has been submitted successfully. The CSV report will
be delivered through FTP</statusReason>
      </Acknowledgement>
    </NS1:processMonthlyBillingInformationResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

### 3.2.1.2 Meter Attributes

- Function Description –  
Get Meter Attributes for one or more ESIIDs, or all ESIIDs of one DUNs.
- Test URL –

<https://uat.services.smartmetertexas.net/MeterInformation/>

# SMART METER TEXAS™

---

- Production URL -

<https://services.smartmetertexas.net/MeterInformation/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameter – Meter Attributes

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterInformationSOAP"
xmlns:bons1="http://com.ibm.smtesb/meter">
  <xsd:element name="MeterSearchRequest">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element minOccurs="1" name="trans_id">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:minLength value="0"/>
              <xsd:maxLength value="32"/>
              <xsd:pattern value="[a-zA-Z0-9]+"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1"
name="requestorID">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:minLength value="1"/>
              <xsd:maxLength value="100"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1" name="requesterType">
          <xsd:simpleType>
```

# SMART METER TEXAS™

---

```
<xsd:restriction base="xsd:string">
  <xsd:enumeration value="REP"/>
  <xsd:enumeration value="RES"/>
  <xsd:enumeration value="BUSINESS"/>
  <xsd:enumeration value="CSP"/>
  <xsd:enumeration value="TDSP"/>
  <xsd:enumeration value="REG"/>
  <xsd:enumeration value="rep"/>
  <xsd:enumeration value="res"/>
  <xsd:enumeration value="business"/>
  <xsd:enumeration value="csp"/>
  <xsd:enumeration value="tdsp"/>
  <xsd:enumeration value="reg"/>
</xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="requesterAuthenticationID">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="16"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

<xsd:element minOccurs="0" name="deliveryMode">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="FTP"/>
      <xsd:enumeration value="EML"/>
      <xsd:enumeration value="API"/>
      <xsd:enumeration value="ftp"/>
      <xsd:enumeration value="eml"/>
      <xsd:enumeration value="api"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="reportFormat">
  <xsd:simpleType>
```







- Body Parameter Response –
  - **Synchronous Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterInformationSOAP"
xmlns:bons1="http://BIM_MeterInformationSOAP">
  <xsd:include schemaLocation="MeterData.xsd"/>
  <xsd:element name="MeterInformationResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element minOccurs="1" name="trans_id">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1" name="esiid">
          <xsd:simpleType>
            <xsd:restriction
              base="xsd:string">
              <xsd:minLength
                value="9"/>
              <xsd:maxLength
                value="64"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>
```

# SMART METER TEXAS™

---

```
</xsd:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterInformationSOAP"
xmlns:bons0="http://com.ibm.smtesb/meter">
  <xsd:complexType name="MeterData">
    <xsd:sequence>
      <xsd:element minOccurs="0" name="utilityCompanyId"
        type="xsd:string">
      </xsd:element>
      <xsd:element minOccurs="1" name="ESIID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="17" />
            <xsd:maxLength value="64" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="meterSerialNumber">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1" />
            <xsd:maxLength value="30" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="utilityMeterId">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1" />
            <xsd:maxLength value="30" />
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="KWHMeterMultiplier"
        type="xsd:string">
```

```
</xsd:element>
<xsd:element minOccurs="1" name="configuredChannels"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="manufacturerName">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1" />
      <xsd:maxLength value="50" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="testDate"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="meterClass"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="installationDate"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="initialProvisionDate"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="communicationIndicator">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1" />
      <xsd:maxLength value="3" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="instrumentRated">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:length value="1" />
      <xsd:enumeration value="Y" />
      <xsd:enumeration value="N" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

# SMART METER TEXAS™

---

```

        <xsd:enumeration value="y" />
        <xsd:enumeration value="n" />
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0"
    name="currentTransformersRatio">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0" />
            <xsd:maxLength value="16" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0"
    name="potentialTransformersRatio">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0" />
            <xsd:maxLength value="16" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="esiFirmwareVersion">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1" />
            <xsd:maxLength value="16" />
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="HANProtocol">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0" />
            <xsd:maxLength value="22" />
        </xsd:restriction>
    </xsd:simpleType>

```

# SMART METER TEXAS™

---

```
</xsd:element>
<xsd:element minOccurs="1" name="smartEnergyProfile">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1" />
      <xsd:maxLength value="22" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="intervalSetting"
  type="xsd:string">

</xsd:element>
<xsd:element minOccurs="1" name="reverseFlowHandling">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:length value="1" />
      <xsd:enumeration value="Y" />
      <xsd:enumeration value="N" />
      <xsd:enumeration value="y" />
      <xsd:enumeration value="n" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="DGChannel"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="disconnect">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:length value="1" />
      <xsd:enumeration value="Y" />
      <xsd:enumeration value="N" />
      <xsd:enumeration value="y" />
      <xsd:enumeration value="n" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
<xsd:element minOccurs="0" name="meterStatus">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="E" />
      <xsd:enumeration value="D" />
      <xsd:enumeration value="e" />
      <xsd:enumeration value="d" />
      <xsd:minLength
value="0"></xsd:minLength>
      <xsd:maxLength
value="1"></xsd:maxLength>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="meterPhases"
  type="xsd:string">
</xsd:element>
<xsd:element minOccurs="0" name="meterModel">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="0" />
      <xsd:maxLength value="30" />
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **Acknowledgement Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_MeterInformationSOAP"
xmlns:bons1="http://com.ibm.smtesb/meter">
  <xsd:import namespace="http://com.ibm.smtesb/meter"
schemaLocation="FaultESIIDLList.xsd"/>
```

```
<xsd:element name="Acknowledgement">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="correlationId"
type="xsd:string">
      </xsd:element>
      <xsd:element minOccurs="0" name="statusCode"
type="xsd:integer">
      </xsd:element>
      <xsd:element minOccurs="0" name="statusReason">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="256"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element maxOccurs="1" minOccurs="1"
name="faultESIIDs" type="bons1:FaultESIIDList">
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
</xsd:schema>
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

# SMART METER TEXAS™

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 22: Return Status Codes

- Example –
  - **Meter Attributes**
    - Request with Delivery mode EML and report format CSV

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_MeterInformationSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:prosessMeterInformation>
      <bim:MeterSearchRequest>
        <trans_id>123</trans_id>
        <requestorID>CSPAPIUser1</requestorID>
        <requesterType>CSP</requesterType>
        <!--Optional:-->
        <requesterAuthenticationID>199999999999</requesterAuthenticationID>
        <!--Optional:-->
        <deliveryMode>EML</deliveryMode>
        <!--Optional:-->
        <reportFormat>csv</reportFormat>
        <version>L</version>
        <!--Optional:-->
        <ESIIDMeterList>
          <!--1 or more repetitions:-->
          <ESIIDMeter>
            <esiid>10443720008107413</esiid>
```



```
<!--Optional:-->
</ESIIDMeter>
</ESIIDMeterList>
<!--Optional:-->
<SMTTermsandConditions>y</SMTTermsandConditions>
</bim:MeterSearchRequest>
</bim:prosessMeterInformation>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:prosessMeterInformationResponse
xmlns:NS1="http://BIM_MeterInformationSOAP">
      <NS1:Acknowledgement>
        <trans_id>123</trans_id>
        <correlationId>2a0d6de6b87611e9bcc30a04</correlationId>
        <statusCode>0000</statusCode>
        <statusReason>Request has been submitted successfully. The CSV report will
be delivered through EML</statusReason>
      </NS1:Acknowledgement>
    </NS1:prosessMeterInformationResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

### 3.2.1.3 Premise Attributes

- Function Description –  
Get Premise Attributes for one or more ESIIDs, or all ESIIDs of one DUNs.
- Test URL –  
<https://uat.services.smartmetertexas.net/PremiseInformation/>
- Production URL -  
<https://services.smartmetertexas.net/PremiseInformation/>

# SMART METER TEXAS™

---

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters – Premise Attributes

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_PremiseInformationSOAP"
xmlns:bons0="http://BIM_PremiseInformationSOAP"
xmlns:bons1="http://com.ibm.smtesb/premise">
  <xsd:import namespace="http://com.ibm.smtesb/premise"
schemaLocation="ESIIDList.xsd">
  </xsd:import>
  <xsd:complexType name="PremiseSearchRequest">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="requestorID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="100"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="requesterType">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="REP"/>
            <xsd:enumeration value="RES"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

```
        <xsd:enumeration value="BUSINESS"/>
        <xsd:enumeration value="NON"/>
        <xsd:enumeration value="CSP"/>
        <xsd:enumeration value="TDSP"/>
        <xsd:enumeration value="REG"/>
        <xsd:enumeration value="rep"/>
        <xsd:enumeration value="res"/>
        <xsd:enumeration value="business"/>
        <xsd:enumeration value="csp"/>
        <xsd:enumeration value="tdsp"/>
        <xsd:enumeration value="reg"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="requesterAuthenticationID">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="16"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="deliveryMode">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="FTP"/>
            <xsd:enumeration value="EML"/>
            <xsd:enumeration value="API"/>
            <xsd:enumeration value="ftp"/>
            <xsd:enumeration value="eml"/>
            <xsd:enumeration value="api"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="reportFormat">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="CSV"/>
            <xsd:enumeration value="JSON"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```

# SMART METER TEXAS™

---

```

        <xsd:enumeration value="csv"/>
        <xsd:enumeration value="json"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="version">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="A"/>
            <xsd:enumeration value="a"/>
            <xsd:enumeration value="L"/>
            <xsd:enumeration value="I"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element maxOccurs="1" minOccurs="0" name="ESIIDList"
type="bons1:ESIIDList">
</xsd:element>
<xsd:element minOccurs="1" name="SMTTermsandConditions">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="Y"/>
            <xsd:enumeration value="N"/>
            <xsd:enumeration value="y"/>
            <xsd:enumeration value="n"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- Body Parameter Response –
  - **Synchronous Response**  
<?xml version="1.0" encoding="UTF-8"?><xsd:schema  
xmlns:xsd="http://www.w3.org/2001/XMLSchema"

# SMART METER TEXAS™

---

```
targetNamespace="http://BIM_PremiseInformationSOAP"
xmlns:bons1="http://com.ibm.smtesb/premise">
  <xsd:import namespace="http://com.ibm.smtesb/premise"
schemaLocation="PremiseData.xsd"/>
  <xsd:element name="PremiseInformationResponse">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element minOccurs="1" name="trans_id">
          <xsd:simpleType>
            <xsd:restriction base="xsd:string">
              <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1" name="esiid">
          <xsd:simpleType>
            <xsd:restriction
              base="xsd:string">
                <xsd:minLength
                  value="9"/>
                <xsd:maxLength
                  value="64"/>
              </xsd:restriction>
            </xsd:simpleType>
          </xsd:element>
        <xsd:element minOccurs="1" maxOccurs="1" minOccurs="1"
          name="PremiseData" type="bons1:PremiseData">
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

# SMART METER TEXAS™

---

```
targetNamespace="http://com.ibm.smtesb/premise"
xmlns:bons0="http://com.ibm.smtesb/premise">
  <xsd:complexType name="PremiseData">
    <xsd:sequence>
      <xsd:element minOccurs="0" name="utilityCompanyId"
type="xsd:string">
      </xsd:element>
      <xsd:element minOccurs="1" name="ESIID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="17"/>
            <xsd:maxLength value="64"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="serviceVoltage">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="15"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="premiseStatus">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:length value="1"/>
            <xsd:enumeration value="A"/>
            <xsd:enumeration value="I"/>
            <xsd:enumeration value="a"/>
            <xsd:enumeration value="i"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="timeZone">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
```

```
        <xsd:length value="3"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element minOccurs="0" name="fractionalHouseAddress">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="0"/>
        <xsd:maxLength value="10"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element minOccurs="0" name="leadingDirectional">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="0"/>
        <xsd:maxLength value="6"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element minOccurs="1" name="streetName">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="1"/>
        <xsd:maxLength value="40"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element minOccurs="0" name="streetType">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:minLength value="0"/>
        <xsd:maxLength value="16"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
  <xsd:element minOccurs="0" name="trailingDirectional">
    <xsd:simpleType>
```

# SMART METER TEXAS™

---

```
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0"/>
            <xsd:maxLength value="6"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="houseNumber">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0"/>
            <xsd:maxLength value="50"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="unitDesignation">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="0"/>
            <xsd:maxLength value="50"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="locality">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="40"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="state">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:length value="2"/>
            <xsd:enumeration value="TX"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```



# SMART METER TEXAS™

---

```
<xsd:element minOccurs="1" name="zipcode">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:length value="5"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="zipcodePlus4">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="4"/>
      <xsd:minLength value="1"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="meterReadCycle"
type="xsd:string">
</xsd:element>
<xsd:element minOccurs="1" name="loadProfile">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="30"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="rateClassCode">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:maxLength value="3"/>
      <xsd:minLength value="0"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="AMSPProfileEffectiveDate"
type="xsd:string">
</xsd:element>
```

```
        </xsd:sequence>
    </xsd:complexType>
</xsd:schema>
```

- **Acknowledgement Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_PremiseInformationSOAP"
xmlns:bons1="http://com.ibm.smtesb/premise">
    <xsd:import namespace="http://com.ibm.smtesb/premise"
schemaLocation="FaultESIIDLList.xsd"/>

    <xsd:element name="Acknowledgement">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element minOccurs="1" name="trans_id">
                    <xsd:simpleType>
                        <xsd:restriction base="xsd:string">
                            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
                        </xsd:restriction>
                    </xsd:simpleType>
                </xsd:element>
                <xsd:element minOccurs="1" name="correlationId"
type="xsd:string">
                    </xsd:element>
                <xsd:element minOccurs="0" name="statusCode"
type="xsd:integer">
                    </xsd:element>
                <xsd:element minOccurs="0" name="statusReason">
                    <xsd:simpleType>
                        <xsd:restriction base="xsd:string">
                            <xsd:maxLength value="256"/>
                        </xsd:restriction>
                    </xsd:simpleType>
                </xsd:element>
                <xsd:element maxOccurs="1" minOccurs="1"
name="faultESIIDs" type="bons1:FaultESIIDLList">
```

```

        </xsd:element>

        </xsd:sequence>
    </xsd:complexType>
</xsd:element>
</xsd:schema>

```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

*Table 23: Return Status Codes*

- Example –

- **Meter Attributes with Delivery mode EML and report format CSV**

- Request

```

<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_PremiseInformationSOAP">
    <soapenv:Header/>
    <soapenv:Body>
        <bim:processPremiseInformation>
            <PremiseSearchRequest>
                <trans_id>1234</trans_id>
                <requestorID>CSPAPIUser1</requestorID>
                <requesterType>CSP</requesterType>
            </PremiseSearchRequest>
        </bim:processPremiseInformation>
    </soapenv:Body>
</soapenv:Envelope>

```

```
<!--Optional:-->
<requesterAuthenticationID>19999999999</requesterAuthenticationID>
  <!--Optional:-->
  <deliveryMode>EML</deliveryMode>
  <!--Optional:-->

  <version>L</version>
  <!--Optional:-->
  <ESIIDList>
    <!--1 or more repetitions:-->
    <ESIID>10443720008107413</ESIID>

  </ESIIDList>
  <SMTTermsandConditions>y</SMTTermsandConditions>
</PremiseSearchRequest>
</bim:processPremiseInformation>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:processPremiseInformationResponse
xmlns:NS1="http://BIM_PremiseInformationSOAP">
      <NS1:Acknowledgement>
        <trans_id>1234</trans_id>
        <correlationId>dba1daccb87811e9bcc30a04</correlationId>
        <statusCode>0000</statusCode>
        <statusReason>Request has been submitted successfully. The CSV
report will be delivered through EML</statusReason>
      </NS1:Acknowledgement>
    </NS1:processPremiseInformationResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

## 3.2.1.4 Subscription – Scheduled Reports (New Subscription, or Unsubscribe / Cancel Subscription, or List of Existing Subscriptions)

- Function Description –

Get the current status for Ad-Hoc or Subscription requests. Note: This is a synchronous function.

- Test URL –

[https://uat.services.smartmetertexas.net/ Subscription/](https://uat.services.smartmetertexas.net/Subscription/)

- Production URL -

[https://services.smartmetertexas.net/ Subscription /](https://services.smartmetertexas.net/Subscription/)

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameter –

- **New Subscription**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_SubscriptionSOAP">
  <xsd:complexType name="NewSubscriptionRequest">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

```
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requestorID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requesterType">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="REP"/>
          <xsd:enumeration value="RES"/>
          <xsd:enumeration value="BUSINESS"/>
          <xsd:enumeration value="CSP"/>
          <xsd:enumeration value="TDSP"/>
          <xsd:enumeration value="REG"/>
          <xsd:enumeration value="rep"/>
          <xsd:enumeration value="res"/>
          <xsd:enumeration value="business"/>
          <xsd:enumeration value="csp"/>
          <xsd:enumeration value="tdsp"/>
          <xsd:enumeration value="reg"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0"
name="requesterAuthenticationID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="18"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
```

```
<xsd:element minOccurs="1" name="subscriptionType">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="REPENROLL"/>
      <xsd:enumeration value="CSPENROLL"/>
      <xsd:enumeration value="SCHEDULE"/>
      <xsd:enumeration value="SCHEDULES"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0"
name="historicalSubscriptionDuration">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:enumeration value="3"/>
      <xsd:enumeration value="6"/>
      <xsd:enumeration value="9"/>
      <xsd:enumeration value="12"/>
      <xsd:enumeration value="24"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="reportFormat">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="LSE"/>
      <xsd:enumeration value="CSV"/>
      <xsd:enumeration value="JSON"/>
      <xsd:enumeration value="XML"/>
      <xsd:enumeration value="lse"/>
      <xsd:enumeration value="csv"/>
      <xsd:enumeration value="json"/>
      <xsd:enumeration value="xml"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="dataType">
  <xsd:simpleType>
```

```
Interval LSE"/>
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="DAILY"/>
        <xsd:enumeration value="INTERVAL"/>
        <xsd:enumeration value="daily"/>
        <xsd:enumeration value="interval"/>
        <xsd:enumeration value="MONTHLY"/>
        <xsd:enumeration value="monthly"/>
        <xsd:enumeration value="Historical 15 min
Interval LSE"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="deliveryMode">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="FTP"/>
            <xsd:enumeration value="EML"/>
            <xsd:enumeration value="API"/>
            <xsd:enumeration value="ftp"/>
            <xsd:enumeration value="eml"/>
            <xsd:enumeration value="api"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="0" name="reportFrequency">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="DAILY"/>
            <xsd:enumeration value="daily"/>
            <xsd:enumeration value="MONTHLY"/>
            <xsd:enumeration value="monthly"/>
            <xsd:enumeration value="WEEKLY"/>
            <xsd:enumeration value="weekly"/>
            <xsd:enumeration value="YEARLY"/>
            <xsd:enumeration value="yearly"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
```



```
<xsd:element minOccurs="0" name="ESIIDs">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element maxOccurs="unbounded"
minOccurs="1" name="esiid">
        <xsd:simpleType>
          <xsd:restriction
base="xsd:string">
            <xsd:maxLength
value="64"/>
            <xsd:minLength
value="9"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="1"
name="SMTTermsandConditions">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:enumeration value="Y"/>
      <xsd:enumeration value="N"/>
      <xsd:enumeration value="y"/>
      <xsd:enumeration value="n"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **UnSubscription**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

# SMART METER TEXAS™

---

```
targetNamespace="http://BIM_SubscriptionSOAP">
  <xsd:complexType name="UnSubscriptionRequest">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="requestorID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="0"
name="requesterAuthenticationID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="18"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="subscriptionNumber">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</pre>
```

```
<xsd:element minOccurs="0" name="ESIIDs">
  <xsd:complexType>
    <xsd:sequence>
      <xsd:element maxOccurs="unbounded"
minOccurs="1" name="esiid">
          <xsd:simpleType>
            <xsd:restriction
base="xsd:string">
              <xsd:maxLength
value="64"/>
              <xsd:minLength
value="9"/>
            </xsd:restriction>
          </xsd:simpleType>
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
  <xsd:element minOccurs="1"
name="SMTTermsandConditions">
    <xsd:simpleType>
      <xsd:restriction base="xsd:string">
        <xsd:enumeration value="Y"/>
        <xsd:enumeration value="N"/>
        <xsd:enumeration value="y"/>
        <xsd:enumeration value="n"/>
      </xsd:restriction>
    </xsd:simpleType>
  </xsd:element>
</xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- **List Subscriptions**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
```

# SMART METER TEXAS™

---

```
targetNamespace="http://BIM_SubscriptionSOAP">
  <xsd:complexType name="SubscriptionRequest">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>

            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="requestorID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="0"
name="requesterAuthenticationID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="18"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1"
name="SMTTermsandConditions">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:enumeration value="Y"/>
            <xsd:enumeration value="N"/>
            <xsd:enumeration value="y"/>
            <xsd:enumeration value="n"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</pre>
```

```
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- Body Parameter Response -

- **New Subscription Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_SubscriptionSOAP"
xmlns:bons1="http://com.smt.csp.subscription/NewSubscription">
```

```
  <xsd:import namespace="http://com.smt.csp.subscription/NewSubscription"
schemaLocation="CustomerESIIDFaultList.xsd">
  </xsd:import>
  <xsd:import namespace="http://com.smt.csp.subscription/NewSubscription"
schemaLocation="CustomerDUNSFaultList.xsd">
  </xsd:import>
```

```
  <xsd:complexType name="NewSubscriptionResponse">
    <xsd:sequence>
      <xsd:element minOccurs="0" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="0" name="subscriptionNumber">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
```

```

                <xsd:minLength value="1"/>
                <xsd:maxLength value="32"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="statusCode"
type="xsd:integer"/>
    <xsd:element minOccurs="1" name="statusReason">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:minLength value="1"/>
                <xsd:maxLength value="256"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>

    <xsd:element maxOccurs="1" minOccurs="1"
name="customerESIIDFaultList" type="bons1:CustomerESIIDFaultList">
    </xsd:element>
    <xsd:element maxOccurs="1" minOccurs="1"
name="customerDUNSFaultList" type="bons1:CustomerDUNSFaultList">
    </xsd:element>

    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **Unsubscription Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_SubscriptionSOAP"
xmlns:bons1="http://com.smt.csp.subscription/UnSubscription">
    <xsd:import namespace="http://com.smt.csp.subscription/UnSubscription"
schemaLocation="UnSubscriptionESIIDsFaultList.xsd">
    </xsd:import>
    <xsd:complexType name="UnSubscriptionResponse">
        <xsd:sequence>
```

```
<xsd:element minOccurs="0" name="trans_id">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="statusCode" type="xsd:integer"/>
<xsd:element minOccurs="1" name="statusReason">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="256"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element maxOccurs="1" minOccurs="1"
name="UnSubscriptionESIIDsFaultList" type="bons1:UnSubscriptionESIIDsFaultList">
  </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **List Subscriptions Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_SubscriptionSOAP"
xmlns:bons1="http://com.smt.csp.subscription/Subscription">

  <xsd:import namespace="http://com.smt.csp.subscription/Subscription"
schemaLocation="SubscriptionList.xsd">
  </xsd:import>

  <xsd:complexType name="SubscriptionResponse">
    <xsd:sequence>
```

# SMART METER TEXAS™

---

```
<xsd:element minOccurs="0" name="trans_id">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="statusCode" type="xsd:string"/>
<xsd:element minOccurs="1" name="statusReason">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="256"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>

  <xsd:element maxOccurs="1" minOccurs="1" name="subscriptionList"
type="bons1:SubscriptionList">
  </xsd:element>

</xsd:sequence>
</xsd:complexType>
</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://com.smt.csp.subscription/Subscription"
xmlns:bons1="http://com.smt.csp.subscription/Subscription">

  <xsd:include schemaLocation="Subscription.xsd"/>
  <xsd:complexType name="SubscriptionList">
    <xsd:sequence>
      <xsd:element maxOccurs="unbounded" minOccurs="1"
name="subscription" type="bons1:Subscription">
      </xsd:element>
    </xsd:sequence>
```



# SMART METER TEXAS™

---

```
</xsd:complexType>  
</xsd:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema  
xmlns:xsd="http://www.w3.org/2001/XMLSchema"  
targetNamespace="http://com.smt.csp.subscription/Subscription">
```

```
<xsd:complexType name="Subscription">  
  <xsd:sequence>  
    <xsd:element minOccurs="0" name="subscriptionNumber">  
      <xsd:simpleType>  
        <xsd:restriction base="xsd:string">  
          <xsd:minLength value="1"/>  
          <xsd:maxLength value="10"/>  
        </xsd:restriction>  
      </xsd:simpleType>  
    </xsd:element>  
    <xsd:element minOccurs="0" name="startDate" type="xsd:string">  
  
  </xsd:element>  
  <xsd:element minOccurs="1" name="reportFormat">  
    <xsd:simpleType>  
      <xsd:restriction base="xsd:string">  
        <xsd:enumeration value="LSE"/>  
        <xsd:enumeration value="CSV"/>  
        <xsd:enumeration value="JSON"/>  
        <xsd:enumeration value="XML"/>  
        <xsd:enumeration value="lse"/>  
        <xsd:enumeration value="csv"/>  
        <xsd:enumeration value="json"/>  
        <xsd:enumeration value="xml"/>  
      </xsd:restriction>  
    </xsd:simpleType>  
  </xsd:element>  
  <xsd:element minOccurs="1" name="deliveryType">  
    <xsd:simpleType>  
      <xsd:restriction base="xsd:string">
```

# SMART METER TEXAS™

---

```
        <xsd:enumeration value="FTP"/>
        <xsd:enumeration value="EML"/>
        <xsd:enumeration value="API"/>
        <xsd:enumeration value="ftp"/>
        <xsd:enumeration value="eml"/>
        <xsd:enumeration value="api"/>
    </xsd:restriction>
</xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="dataType">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="DAILY"/>
            <xsd:enumeration value="INTERVAL"/>
            <xsd:enumeration value="daily"/>
            <xsd:enumeration value="interval"/>
            <xsd:enumeration value="MONTHLY"/>
            <xsd:enumeration value="monthly"/>
            <xsd:enumeration value="Historical 15 min
Interval LSE"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>

    <xsd:element minOccurs="1" name="status" type="xsd:string"/>
    <xsd:element minOccurs="1" name="lastSuccessfulReportDate"
type="xsd:string"/>
    </xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- Return Status Codes –

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 24: Return Status Codes

- Example –

- **New Subscription**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_SubscriptionSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:newSubscription>
      <newSubscriptionRequest>
        <trans_id>7292019324</trans_id>
        <requestorID>bobby@123</requestorID>
        <requesterType>RES</requesterType>

        <subscriptionType>SCHEDULE</subscriptionType>

        <reportFormat>CSV</reportFormat>
        <dataType>INTERVAL</dataType>
        <deliveryMode>EML</deliveryMode>
        <!--Optional:-->
        <reportFrequency>DAILY</reportFrequency>
        <!--Optional:-->
      </newSubscriptionRequest>
    </bim:newSubscription>
  </soapenv:Body>
</soapenv:Envelope>
```

```
<ESIIDs>
  <!--1 or more repetitions:-->
  <esiid>10443720002506579001</esiid>
</ESIIDs>
<SMTTermsandConditions>Y</SMTTermsandConditions>
</newSubscriptionRequest>
</bim:newSubscription>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:newSubscriptionResponse
xmlns:NS1="http://BIM_SubscriptionSOAP">
      <newSubscriptionResponse>
        <trans_id>7292019324</trans_id>
        <statusCode>0001</statusCode>
        <statusReason>Failure</statusReason>
        <customerESIIDFaultList>
          <customerESIIDFault>
            <esiid>10443720002506579001</esiid>
            <reasonCode>Subscription is already
active:10443720002506579001</reasonCode>
          </customerESIIDFault>
        </customerESIIDFaultList>
      </newSubscriptionResponse>
    </NS1:newSubscriptionResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

- **UnSubscription**

- **Request**

```
<soapenv:Envelope
```

# SMART METER TEXAS™

---

```
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_SubscriptionSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:unSubscription>
      <unSubscriptionRequest>
        <trans_id>862019</trans_id>
        <requestorID>bobby@123</requestorID>
        <!--Optional:-->
        <requesterAuthenticationID>?</requesterAuthenticationID>

      <subscriptionNumber>8F77A7E0277400D8E0530A040322F226</subscriptionNumber>
        <!--Optional:-->
        <ESIIDs>
          <!--1 or more repetitions:-->
          <esiid>10443720002506579001</esiid>
        </ESIIDs>
        <SMTTermsandConditions>Y</SMTTermsandConditions>
      </unSubscriptionRequest>
    </bim:unSubscription>
  </soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:unSubscriptionResponse
xmlns:NS1="http://BIM_SubscriptionSOAP">
      <unSubscriptionResponse>
        <trans_id>862019</trans_id>
        <statusCode>0001</statusCode>
        <statusReason>Failure</statusReason>
        <unsubscriptionESIIDsFaultList>
          <unSubscriptionESIIDsFault>
            <esiid>10443720002506579001</esiid>
```

```
<reasonCode>ESIID does not belong to this Subscription
:10443720002506579001:8F77A7E0277400D8E0530A040322F226</reason
Code>
</unSubscriptionESIIDsFault>
</unsubscriptionESIIDsFaultList>
</unSubscriptionResponse>
</NS1:unSubscriptionResponse>
</soapenv:Body>
</soapenv:Envelope>
```

- **List Subscriptions**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_SubscriptionSOAP">
<soapenv:Header/>
<soapenv:Body>
<bim:subscription>
<subscriptionRequest>
<trans_id>862019</trans_id>
<requestorID>bobby@123</requestorID>
<!--Optional:-->
<requesterAuthenticationID>?</requesterAuthenticationID>
<SMTTermsandConditions>Y</SMTTermsandConditions>
</subscriptionRequest>
</bim:subscription>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
<soapenv:Body>
<NS1:subscriptionResponse xmlns:NS1="http://BIM_SubscriptionSOAP">
<subscriptionResponse>
<trans_id>862019</trans_id>
<statusCode>0000</statusCode>
```

```
<statusReason>Success</statusReason>
<subscriptionList>
  <subscription>

    <subscriptionNumber>8F10E261E2E50046E0530A0403228C2E</subscriptionNumber>
      <startDate>2019-08-02T10:27:48</startDate>
      <reportformat>CSV</reportformat>
      <deliveryType>EML</deliveryType>
      <dataType>MON</dataType>
      <status>ACT</status>
      <lastSuccessfulReportDate/>
    </subscription>
  </subscription>

  <subscriptionNumber>8F1B5C285D4D00FCE0530A040322436C</subscriptionNumber>
    <startDate>2019-08-02T22:57:41</startDate>
    <reportformat>CSV</reportformat>
    <deliveryType>EML</deliveryType>
    <dataType>MON</dataType>
    <status>ACT</status>
    <lastSuccessfulReportDate/>
  </subscription>

</subscriptionList>
</subscriptionResponse>
</NS1:subscriptionResponse>
</soapenv:Body>
</soapenv:Envelope>
```

***3.2.1.5 CSP Agreement (New Energy Data Sharing Agreement, or List ESIDs per Energy Data Sharing Agreement, or Terminate Energy Data Sharing Agreement, or List of Energy Data Sharing Agreements)***

- Function Description –

# SMART METER TEXAS™

---

Update an existing subscription on a one subscription per request basis.

- Test URL –

<https://uat.services.smartmetertexas.net/CSPAAuthorization/>

- Production URL -

<https://services.smartmetertexas.net/CSPAAuthorization/>

- Method –

SMT will provide all API services using the POST method. The entire request payload / message needs to be included in the Body Parameter.

- Body Parameters Request –

- **New Agreement**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAAuthorizationSOAP"
xmlns:bons1="http://com.smt.csp.authorization/NewAgreement">
```

```
    <xsd:import namespace="http://com.smt.csp.authorization/NewAgreement"
schemaLocation="ESIIDMeterPUCTNumberList.xsd">
    </xsd:import>
```

```
    <xsd:complexType name="NewAgreement">
        <xsd:sequence>
            <xsd:element minOccurs="1" name="trans_id">
                <xsd:simpleType>
                    <xsd:restriction base="xsd:string">
                        <xsd:minLength value="1"/>
                        <xsd:maxLength value="32"/>
                        <xsd:pattern value="[a-zA-Z0-9]+"\>
                    </xsd:restriction>
                </xsd:simpleType>
            </xsd:element>
```



# SMART METER TEXAS™

---

```
<xsd:element minOccurs="1" name="requestorID">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="32"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1"
name="requesterAuthenticationID">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="18"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="retailCustomerEmail">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
      <xsd:pattern value="[a-z0-9._%+-]+@[a-z0-
9.-]+\.[a-z]{2,4}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="agreementDuration">
  <xsd:simpleType>
    <xsd:restriction base="xsd:integer">
      <xsd:enumeration value="3"/>
      <xsd:enumeration value="6"/>
      <xsd:enumeration value="9"/>
      <xsd:enumeration value="12"/>
      <xsd:enumeration value="24"/>
      <xsd:enumeration value="36"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
```

```
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0"
name="customerLanguagePreference">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="ENGLISH"/>
          <xsd:enumeration value="SPANISH"/>
          <xsd:enumeration value="english"/>
          <xsd:enumeration value="spanish"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element maxOccurs="1" minOccurs="1"
name="customerMeterList" type="bons1:ESIIDMeterPUCTNumberList">
    </xsd:element>
    <xsd:element minOccurs="1"
name="SMTTermsandConditions">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Y"/>
          <xsd:enumeration value="N"/>
          <xsd:enumeration value="y"/>
          <xsd:enumeration value="n"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- **Terminate Agreement**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAAuthorizationSOAP"
xmlns:bons1="http://com.smt.csp.authorization/TerminateAgreement">
```

# SMART METER TEXAS™

---

```
<xsd:import
namespace="http://com.smt.csp.authorization/TerminateAgreement"
schemaLocation="AgreementList.xsd">
</xsd:import>

<xsd:complexType name="TerminateAgreement">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requestorID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0"
name="requesterAuthenticationID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="18"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element maxOccurs="1" minOccurs="1"
name="agreementList" type="bons1:AgreementList">
      </xsd:element>
    <xsd:element minOccurs="1"
name="SMTTermsandConditions">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Y"/>
          <xsd:enumeration value="N"/>
          <xsd:enumeration value="y"/>
          <xsd:enumeration value="n"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

```
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

```
</xsd:schema>
```

- **My Agreements (including Status)**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAauthorizationSOAP"
xmlns:bons1="http://com.smt.csp.authorization/NewAgreement">
```

```
  <xsd:complexType name="MyAgreements">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:pattern value="[a-zA-Z0-9]{1,32}"></xsd:pattern>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="requestorID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="0"
name="requesterAuthenticationID">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="18"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

```
        </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0" name="agreementNumber"
type="xsd:integer">
    </xsd:element>
    <xsd:element minOccurs="0" name="statusReason">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="PEN"/>
                <xsd:enumeration value="ACT"/>
                <xsd:enumeration value="COM"/>
                <xsd:enumeration value="NACOM"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1"
name="SMTTermsandConditions">
        <xsd:simpleType>
            <xsd:restriction base="xsd:string">
                <xsd:enumeration value="Y"/>
                <xsd:enumeration value="N"/>
                <xsd:enumeration value="y"/>
                <xsd:enumeration value="n"/>
            </xsd:restriction>
        </xsd:simpleType>
    </xsd:element>
</xsd:sequence>
</xsd:complexType>
</xsd:schema>
```

- **List ESIIDs per Agreement**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAAuthorizationSOAP">
```

```
<xsd:complexType name="AgreementESIIDs">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:pattern value="[a-zA-Z0-9]{1,32}"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="requestorID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="0" name="requesterAuthenticationID">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="18"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="agreementNumber"
type="xsd:integer"/>
    <xsd:element minOccurs="1" name="SMTTermsandConditions">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:enumeration value="Y"/>
          <xsd:enumeration value="N"/>
          <xsd:enumeration value="y"/>
          <xsd:enumeration value="n"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
  </xsd:sequence>
</xsd:complexType>
```

```
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- Body Parameters Response -

- **New Agreement Response**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAAuthorizationSOAP">

<xsd:complexType name="NewAgreementResponse">
    <xsd:sequence>
        <xsd:element minOccurs="1" name="trans_id">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:minLength value="1"/>
                    <xsd:maxLength value="32"/>
                    <xsd:pattern value="[a-zA-Z0-9]+"\>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1" name="agreementNumber"
type="xsd:integer">
        </xsd:element>
        <xsd:element minOccurs="1" name="statusCode">
            <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                    <xsd:maxLength value="4"/>
                </xsd:restriction>
            </xsd:simpleType>
        </xsd:element>
        <xsd:element minOccurs="1" name="statusReason">
```

```
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:maxLength value="256"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>

</xsd:sequence>
</xsd:complexType>
```

```
</xsd:schema>
```

- **Terminate Agreement**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAAuthorizationSOAP"
xmlns:bons1="http://com.smt.csp.authorization/TerminateAgreement">
```

```
  <xsd:import
namespace="http://com.smt.csp.authorization/TerminateAgreement"
schemaLocation="TerminateInitiatedList.xsd">
  </xsd:import>
```

```
  <xsd:import
namespace="http://com.smt.csp.authorization/TerminateAgreement"
schemaLocation="AgreementFaultList.xsd">
  </xsd:import>
```

```
<xsd:complexType name="TerminateAgreementResponse">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
          <xsd:pattern value="[a-zA-Z0-9]+"\>
```



```

        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="statusCode">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="4"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="statusMessage">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="128"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element maxOccurs="1" minOccurs="1"
name="terminateInitiatedList" type="bons1:TerminateInitiatedList">
</xsd:element>
<xsd:element maxOccurs="1" minOccurs="1"
name="agreementFaultList" type="bons1:AgreementFaultList">
</xsd:element>

</xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- **My Agreements (including Status)**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAauthorizationSOAP"
xmlns:bons0="http://com.smt.csp.authorization/MyAgreements">
```

# SMART METER TEXAS™

---

```
<xsd:import namespace="http://com.smt.csp.authorization/MyAgreements"
schemaLocation="AgreementStatusList.xsd">
</xsd:import>

<xsd:complexType name="MyAgreementsResponse">
  <xsd:sequence>
    <xsd:element minOccurs="1" name="trans_id">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:minLength value="1"/>
          <xsd:maxLength value="32"/>
          <xsd:pattern value="[a-zA-Z0-9]+"\>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="statusCode">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="4"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>
    <xsd:element minOccurs="1" name="statsReason">
      <xsd:simpleType>
        <xsd:restriction base="xsd:string">
          <xsd:maxLength value="256"/>
        </xsd:restriction>
      </xsd:simpleType>
    </xsd:element>

    <xsd:element maxOccurs="1" minOccurs="1"
name="agreementStatusList" type="bons0:AgreementStatusList">
    </xsd:element>

  </xsd:sequence>
</xsd:complexType>
```

# SMART METER TEXAS™

---

```
</xsd:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://com.smt.csp.authorization/MyAgreements"
xmlns:bons0="http://com.smt.csp.authorization/MyAgreements">
  <xsd:include schemaLocation="AgreementStatus.xsd"/>

  <xsd:complexType name="AgreementStatusList">
    <xsd:sequence maxOccurs="unbounded" minOccurs="1">
      <xsd:element maxOccurs="unbounded" minOccurs="1"
name="agreementStatus" type="bons0:AgreementStatus">
        </xsd:element>
      </xsd:sequence>
    </xsd:complexType>

</xsd:schema>
```

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://com.smt.csp.authorization/MyAgreements"
xmlns:bons0="http://com.smt.csp.authorization/MyAgreements">

  <xsd:complexType name="AgreementStatus">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="agreementNumber"
type="xsd:integer">
        </xsd:element>
      <xsd:element minOccurs="1" name="startDate">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="10"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
</xsd:schema>
```

# SMART METER TEXAS™

---

```
<xsd:element minOccurs="1" name="endDate">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="10"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="CSPName">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="100"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1"
name="requesterAuthenticationID">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="18"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="retailCustomerEmail">
  <xsd:simpleType>
    <xsd:restriction base="xsd:string">
      <xsd:minLength value="1"/>
      <xsd:maxLength value="64"/>
      <xsd:pattern value="[a-z0-9._%+~]+@[a-z0-9.-]+\.?[a-z]{2,4}"/>
    </xsd:restriction>
  </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="agreementDuration">
```

```
<xsd:simpleType>
  <xsd:restriction base="xsd:string">
    <xsd:enumeration value="Pending -
Customer Authorization"/>
    <xsd:enumeration value="Active -
Authorization Confirmed"/>
    <xsd:enumeration value="Active - Expiring
in 30 days or less"/>
    <xsd:enumeration value="Active - Expiring
in 15 days or less"/>
    <xsd:enumeration value="Active -
Authorization Confirmed"/>
    <xsd:enumeration value="Non Active -
Rejected"/>
    <xsd:enumeration value="Non Active -
Timed Out"/>
    <xsd:enumeration value="Non Active -
Reported as SPAM"/>
    <xsd:enumeration value="Non Active -
Terminated by Customer"/>
    <xsd:enumeration value="Non Active -
Terminated by CSP"/>
    <xsd:enumeration value="Non Active -
Expired"/>
    <xsd:enumeration value="Non Active -
Moved Out"/>
  </xsd:restriction>
</xsd:simpleType>
</xsd:element>

</xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- **List ESIIIDs per Agreement**

```
<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://BIM_CSPAuthorizationSOAP"
xmlns:bons0="http://com.smt.csp.authorization/AgreementESIIIDs">
```

```
  <xsd:import namespace="http://com.smt.csp.authorization/AgreementESIIIDs"
schemaLocation="AgreementESIIDList.xsd">
  </xsd:import>
```

```
  <xsd:complexType name="AgreementESIIIDsResponse">
    <xsd:sequence>
      <xsd:element minOccurs="1" name="trans_id">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="32"/>
            <xsd:pattern value="[a-zA-Z0-9]+"\>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="statusCode">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="4"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
      <xsd:element minOccurs="1" name="statsReason">
        <xsd:simpleType>
          <xsd:restriction base="xsd:string">
            <xsd:maxLength value="256"/>
          </xsd:restriction>
        </xsd:simpleType>
      </xsd:element>
    </xsd:sequence>
  </xsd:complexType>
```

# SMART METER TEXAS™

---

```
</xsd:element>

    <xsd:element maxOccurs="1" minOccurs="1"
name="agreementESIIDLList" type="bons0:AgreementESIIDLList">
    </xsd:element>

</xsd:sequence>
</xsd:complexType>
</xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://com.smt.csp.authorization/AgreementESIIDs"
xmlns:bons0="http://com.smt.csp.authorization/AgreementESIIDs">
    <xsd:include schemaLocation="AgreementESIID.xsd"/>

    <xsd:complexType name="AgreementESIIDLList">
        <xsd:sequence maxOccurs="unbounded" minOccurs="1">
            <xsd:element maxOccurs="unbounded" minOccurs="1"
name="agreementESIID" type="bons0:AgreementESIID">
                </xsd:element>
            </xsd:sequence>
        </xsd:complexType>

    </xsd:schema>

<?xml version="1.0" encoding="UTF-8"?><xsd:schema
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://com.smt.csp.authorization/AgreementESIIDs"
xmlns:bons0="http://com.smt.csp.authorization/AgreementESIIDs">

    <xsd:complexType name="AgreementESIID">
        <xsd:sequence>

            <xsd:element minOccurs="1" name="ESIID">
                <xsd:simpleType>
```

# SMART METER TEXAS™

---

```
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="9"/>
            <xsd:maxLength value="64"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="meterNumber">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:minLength value="1"/>
            <xsd:maxLength value="18"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="PUCTRORNumber"
type="xsd:integer">
</xsd:element>
<xsd:element minOccurs="1" name="status">
    <xsd:simpleType>
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="10"/>
        </xsd:restriction>
    </xsd:simpleType>
</xsd:element>
<xsd:element minOccurs="1" name="startDate">
</xsd:element>
<xsd:element minOccurs="1" name="endDate">
</xsd:element>
</xsd:sequence>
</xsd:complexType>

</xsd:schema>
```

- Return Status Codes –



# SMART METER TEXAS™

---

The following table lists the Status Code and associated Description:

Status Code	Description
200	OK
400	Bad request
401	Authentication failure
401	Authorization failure
403	{"error": "Basic authentication header is missing."}
400	{"error": "Please provide username."}
400	{"error": "Please provide password."}
401	{"error": "Incorrect username or password."}
500	{"error": "Something went wrong. Please try again later."}

Table 25: Return Status Codes

- Example –

- **New Agreement**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_CSPAAuthorizationSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:newAgreement>
      <newAgreementRequest>
        <trans_id>CSP123</trans_id>
        <requestorID>CSPAPIUser1</requestorID>

        <requesterAuthenticationID>199999999999</requesterAuthenticationID>
        <retailCustomerEmail>rahulp24@gmail.com</retailCustomerEmail>
        <agreementDuration>12</agreementDuration>
        <!--Optional:-->

        <customerLanguagePreference>English</customerLanguagePreference>
        <customerMeterList>
```

```
<!--1 or more repetitions:-->
<customerMeter>
  <ESIID>10204049799602671</ESIID>
  <meterNumber>148239922</meterNumber>
  <PUCTRORNumber>10004</PUCTRORNumber>
</customerMeter>
</customerMeterList>
<SMTTermsandConditions>Y</SMTTermsandConditions>
</newAgreementRequest>
</bim:newAgreement>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:newAgreementResponse
xmlns:NS1="http://BIM_CSPAAuthorizationSOAP">
      <newAgreementResponse>
        <trans_id>CSP123</trans_id>
        <agreementNumber>119720</agreementNumber>
        <statusCode>ACK</statusCode>
        <statusReason>Success</statusReason>
      </newAgreementResponse>
    </NS1:newAgreementResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

- **My Agreements**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_CSPAAuthorizationSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:myAgreements>
      <myAgreementsRequest>
        <trans_id>MyA1234</trans_id>
```

# SMART METER TEXAS™

---

```
<requestorID>CSPAPIUser1</requestorID>
<!--Optional:-->

<requesterAuthenticationID>19999999999</requesterAuthenticationID>
<!--Optional:-->
<!-- <agreementNumber?></agreementNumber>
-->
<!--Optional:-->
<!-- <statusReason?></statusReason>
:-->
<SMTTermsandConditions>Y</SMTTermsandConditions>
</myAgreementsRequest>
</bim:myAgreements>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:myAgreementsResponse
xmlns:NS1="http://BIM_CSPAAuthorizationSOAP">
      <myAgreementsResponse>
        <trans_id>MyA1234</trans_id>
        <statusCode>0000</statusCode>
        <statusReason>Success</statusReason>
        <agreementStatusList>
          <agreementStatus>
            <agreementNumber>118360</agreementNumber>
            <startDate>07/11/2019</startDate>
            <endDate>08/06/2019</endDate>
            <CSPName>cspbrand1</CSPName>

          </agreementStatus>
        </agreementStatusList>
      </myAgreementsResponse>
    </NS1:myAgreementsResponse>
  </soapenv:Body>
</soapenv:Envelope>

<requesterAuthenticationID>19999999999</requesterAuthenticationID>

<retailCustomerEmail>uat1cspagremeent@mailinator.com</retailCustomerE
mail>
```

# SMART METER TEXAS™

---

```
<status>Non Active - Terminated by CSP</status>
</agreementStatus>
<agreementStatus>
  <agreementNumber>118518</agreementNumber>
  <startDate>07/15/2019</startDate>
  <endDate>10/15/2019</endDate>
  <CSPName>cspbrand1</CSPName>
```

```
<requesterAuthenticationID>19999999999</requesterAuthenticationID>
```

```
<retailCustomerEmail>donytest@mailinator.com</retailCustomerEmail>
  <status>Active - Authorization Confirmed</status>
</agreementStatus>
<agreementStatus>
  <agreementNumber>119720</agreementNumber>
  <startDate>08/07/2019</startDate>
  <endDate>08/07/2020</endDate>
  <CSPName>cspbrand1</CSPName>
```

```
<requesterAuthenticationID>19999999999</requesterAuthenticationID>
```

```
<retailCustomerEmail>rahulp24@gmail.com</retailCustomerEmail>
  <status>Pending - Customer Authorization</status>
</agreementStatus>
</agreementStatusList>
</myAgreementsResponse>
</NS1:myAgreementsResponse>
</soapenv:Body>
</soapenv:Envelope>
```

- **List ESIDs of an Agreement**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_CSPAuthorizationSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:agreementESIDs>
```

# SMART METER TEXAS™

---

```
<agreementESIIDsRequest>
  <trans_id>A123</trans_id>
  <requestorID>CSPAPIUser1</requestorID>
  <!--Optional:-->

  <requesterAuthenticationID>19999999999</requesterAuthenticationID>
  <agreementNumber>118518</agreementNumber>
  <SMTTermsandConditions>Y</SMTTermsandConditions>
</agreementESIIDsRequest>
</bim:agreementESIIDs>
</soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:agreementESIIDsResponse
xmlns:NS1="http://BIM_CSPAuthorizationSOAP">
      <agreementESIIDsResponse>
        <trans_id>A123</trans_id>
        <statusCode>0000</statusCode>
        <statusReason>Success</statusReason>
        <AgreementESIIDLList>
          <agreementESIID>
            <ESIID>10204049799602671</ESIID>
            <meterNumber>148239922</meterNumber>
            <PUCTRORNumber>10004</PUCTRORNumber>
            <status>Active</status>
            <startDate>07/15/2019</startDate>
            <endDate>10/15/2019</endDate>
          </agreementESIID>
        </AgreementESIIDLList>
      </agreementESIIDsResponse>
    </NS1:agreementESIIDsResponse>
  </soapenv:Body>
</soapenv:Envelope>
```

- **Terminate Agreement**

- **Request**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:bim="http://BIM_CSPAAuthorizationSOAP">
  <soapenv:Header/>
  <soapenv:Body>
    <bim:terminateAgreement>
      <terminateAgreementRequest>
        <trans_id>T1234</trans_id>
        <requestorID>CSPAPIUser1</requestorID>
        <!--Optional:-->

        <requesterAuthenticationID>19999999999</requesterAuthenticationID>
        <agreementList>
          <!--1 or more repetitions:-->
          <agreement>
            <agreementNumber>119720</agreementNumber>

            <retailCustomerEmail>rahulp24@gmail.com</retailCustomerEmail>
          </agreement>
        </agreementList>
        <SMTTermsandConditions>Y</SMTTermsandConditions>
      </terminateAgreementRequest>
    </bim:terminateAgreement>
  </soapenv:Body>
</soapenv:Envelope>
```

- **Response**

```
<soapenv:Envelope
xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
  <soapenv:Body>
    <NS1:terminateAgreementResponse
xmlns:NS1="http://BIM_CSPAAuthorizationSOAP">
      <terminateAgreementResponse>
        <trans_id>T1234</trans_id>
        <statusCode>0001</statusCode>
        <statusReason>Agreement termination failed</statusReason>
```

# SMART METER TEXAS™

---

```
<agreementFaultList>
  <agreement>
    <agreementNumber>119720</agreementNumber>

<retailCustomerEmail>rahulp24@gmail.com</retailCustomerEmail>
  <statusReason>Agreement cannot be terminated: This Agreement
is not ACTIVE:119720</statusReason>
  </agreement>
</agreementFaultList>
</terminateAgreementResponse>
</NS1:terminateAgreementResponse>
</soapenv:Body>
</soapenv:Envelope>
```

## 3.3 FTPS Overview

The following information provides a general overview of the SMT FTPS services:

- The FTPS services will be based on a per DUNS basis
- For a ROR, two folders will be provided to support the daily LSE file processes and ad-hoc and subscription requests.
  - Example –
    - Folder Name – “intervaldata” Contains the LSE files SMT received from the TDSPs.
      - Example file:  
957877905IntervalData20100122123001980.lse.002.799530915
    - Folder Name – “adhocusage” Contains the SMT generated CSV/LSE/XML files for the ad-hoc usage requests
      - Example files:
        - IntervalMeterUsageefd024ab76ee195c3c326fe9.CSV.8286294561000
        - IntervalMeterUsageefd024ab76ee195c3c326fe9.lse.8286294561000
        - DailyMeterUsage00122c501ff160ca73ad74a7.CSV.799530915
- For all other SMT entity accounts, one folder will be provided to support ad-hoc and subscription requests.
  - Example –
    - Folder Name – “adhocusage” Contains the SMT generated CSV/LSE/XML files for the ad-hoc usage requests
      - Example files:
        - IntervalMeterUsageefd024ab76ee195c3c326fe9.CSV.8286294561000
        - IntervalMeterUsageefd024ab76ee195c3c326fe9.lse.8286294561000
        - DailyMeterUsage00122c501ff160ca73ad74a7.CSV.799530915
- Any FTP client tool which supports FTPS can be used to connect with SMT’s FTPS service
  - Example –
    - Core FTP, CURL etc.



## 4 Integration

### 4.1 Web Portal

The following information provides the requirements to integrate with SMT User Interface (UI):

- Access to the SMT portal UI will require user credentials (e.g. ID and password) associated with an SMT Entity Account
- The SMT user will be able to retrieve information, in which they have been authorized, on an ad-hoc basis
- The SMT user will be able to download information, in which they have been authorized, to their individual computer in either a CSV or Green Button file format
- The SMT user will be able to subscribe information to be sent to them through their individual e-mail on a periodic basis and in a CSV or Green Button file format

### 4.2 FTPS

The following information provides the requirements to integrate with SMT:

- FTPS integration will require the use of the Transport Layer Security (TLS) protocol
- A signed SSL certificate from a certificate authority will be required for mutual authentication
- A set of FTP credentials (e.g. ID and password) will be required for each DUNS
- FTP access will require the FTP credentials and use of mutual TLS authentication
- Port 21, in passive mode, will need to be enabled to access FTP services using <ftp.smartmetertexas.biz> as the hostname of the SMT FTP server.
- The port ranges **35000 to 35100** for ftp data channel, associated with <ftp.smartmetertexas.biz>, need to be enabled through the network.
- CURL Commands to Connect and Download Files:
  - **To List FTP folders**  

```
curl -kG --ftp-method nocwd --ftp-ssl --cert <myclientcert>.cert --key <mykey>.pfx -u <userName>:<password> ftp://ftp.smartmetertexas.biz/
```
  - **To List Files in a Folder**

```
curl -kG --ftp-method nocwd --ftp-ssl --cert <myclientcert>.cert --key <mykey>.pfx -u <userName>:<password> ftp://ftp.smartmetertexas.biz/intervaldata/
```

- **To Download a File**

- `curl -kG --ftp-method nocwd --ftp-ssl --cert <myclientcert>.cert --key <mykey>.pfx -u <userName>:<password> ftp://ftp.smartmetertexas.biz/intervaldata/abc.asc -o abc.asc`

- **SMT CURL Example –**

```
C:\>curl -kG --ftp-method nocwd --ftp-ssl --cert [REDACTED]_w3svc670926.cert --key [REDACTED]_w3svc670926_cert.pfx -u [REDACTED]:[REDACTED] ftp://ftp.smartmetertexas.biz/
drwx----- 1 dir group      1024 Jul 09  2016  adhocusage
drwx----- 1 dir group      1024 Oct 10  04:30  intervaldata

C:\>curl -kG --ftp-method nocwd --ftp-ssl --cert [REDACTED]_w3svc670926.cert --key [REDACTED]_w3svc670926_cert.pfx -u [REDACTED]:[REDACTED] ftp://ftp.smartmetertexas.biz/intervaldata/
-rw----- 1 file group      69716 Jul 21  12:41  007923311IntervalData20190721120710101.lse.001.787476634.asc
-rw----- 1 file group      70906 Jul 22  12:54  007923311IntervalData20190722122732101.lse.001.787476634.asc
-rw----- 1 file group      66839 Jul 23  12:55  007923311IntervalData20190723122616101.lse.001.787476634.asc

C:\>curl -kG --ftp-method nocwd --ftp-ssl --cert [REDACTED]_w3svc670926.cert --key [REDACTED]_w3svc670926_cert.pfx -u [REDACTED]:[REDACTED]
ftp://ftp.smartmetertexas.biz/intervaldata/007923311IntervalData20190721120710101.lse.001.787476634.asc -o 007923311IntervalData20190721120710101.lse.001.787476634.asc
  % Total    % Received % Xferd  Average Speed   Time    Time     Current
                                 Dload  Upload  Total   Spent    Left  Speed
 100 11325    0 11325    0    0  11325    0 --:--:--  0:00:01 --:--:-- 10663
```

*Figure 1: SMT CURL Example*

- **Common Troubleshooting Tips**

- **Directories**
  - If you can login but not able to view your directories, make sure you are using PROTP before issuing a LIST command.
- **Connection Time Out**
  - A connection time out or “waiting for connection” is noted, the two most common problems are:
    - Your firewall public IP is not whitelisted on the SMT firewall, or
    - The SMT Public IPs and ports are blocked on your firewall.
- **DH Key is Too Small**

- Please make sure that you are using one of the CIPHERs listed in Section 2.

## 4.3 API

The following information provides the requirements to integrate with SMT:

- API integration will require the use of the Transport Layer Security (TLS) protocol version 1.0, 1.1, 1.2 and later
  - Note: SMT will no longer support all SSL versions
- A signed SSL certificate from a certificate authority will be required for mutual authentication
- A basic authentication with service ID credentials (e.g. ID and password) will be required
  - Note: The service IDs can be used to support the SMT Entity Account or on a per DUNS basis for that SMT Entity Account
- The service ID user name must be equivalent to the requestor ID in the message payload
- Port 443 will need to be enabled to access API services using <https://services.smartmetertexas.net>
- **Basic Access Authentication** is a method for an HTTP user agent (client) to provide a **user name** and **password** when making a request. In basic HTTP authentication, a request contains a header field of the form `Authorization: Basic <credentials>`, where credentials is the **base64** encoding of id and password joined by a single colon (:).
  - Example –

# SMART METER TEXAS™

```
▼ Request Headers:  
Content-Type: "application/json"  
Authorization: "Basic TkVXVERTUFVTRVIX0nBhc3N3b3Jk"  
User-Agent: "PostmanRuntime/7.15.2"  
Accept: "**/*"  
Cache-Control: "no-cache"  
Postman-Token: "500200a1-115e-4a56-a380-4495d881a168"  
Host: "uat-services.smartmetertexas.net"  
Accept-Encoding: "gzip, deflate"  
Content-Length: 307  
Connection: "keep-alive"
```

Figure 2: Basic Access Authentication Example

- CURL Commands to access the SMT REST API
  - `curl -i -u <apiuser:password> POST -H "Accept:application/json" 'https://uat-services.smartmetertexas.net/odr/' -v --key "<cert.key>" --cert "<cacert.pem:password>" --tlsv1.2 -k -d @payload.json`

- Example for ODR REST API –

### Request

```
curl -u NEWTDSPUSER1:password POST -H "Accept:application/json" 'https://uat-services.smartmetertexas.net/odr/' --key "C:/SMT/API/DataPower/Stage_Skytap/myNaesb.key" --cert "C:/SMT/API/DataPower/Stage_Skytap/NAESB_Stage.pem:password" --tlsv1.2 -k -d @C:/RESTAPI/odrrequest.json
```

### odrrequest.json file

```
{  
  "trans_id": "123",  
  "requesterType": "TDSP",  
  "requesterAuthenticationID": "1039940674000",  
  "requestorID": "NEWTDSPUSER1",  
  "deliveryMode": "API",  
  "ESIID": "10443720009004455",  
  "SMTTermsandConditions": "Y"  
}
```

# SMART METER TEXAS™

---

## Response

% Total	% Received	% Xferd	Average Speed	Time	Time	Time	Current
		Dload	Upload	Total	Spent	Left	Speed
0	0	0	0	0	0	0	0
curl: (6) Could not resolve host: POST							
100	329	0	131	100	198	96	145
241{"trans_id":"123","correlationId":"442178","statusCode":"0","statusReason":"Request submitted successfully for further processing"}							

**SMART METER  
TEXAS™**

---

## 5 Glossary of Terms

# SMART METER TEXAS™

Term	Description
Competitive Service Provider (CSP)	Provides energy usage services to retail customers, excluding the sale of electric energy to the retail customers in the areas of Texas that is open to retail competition. To support the CSP's service offerings, the CSP may obtain access to a retail customer's energy data from SMT after SMT has received confirmation from the customer that the CSP data sharing agreement has been approved
DUNS Number	A proprietary system developed and regulated by Dun & Bradstreet (D&B) that assigns a unique numeric identifier, referred to as a "DUNS number" to a single business entity.
Electric Reliability Council of Texas (ERCOT)	Manages the flow of electric power, as the independent system operator (ISO) for Texas, to more than 25 million Texas customers.
Electric Service Identifier ID (ESI ID)	A 17-digit number found on an electric bill that is unique to a property address in the State of Texas.
Information Technology (IT) Service Provider	A company which provides business and technical expertise to enable organizations in the creation, management and optimization of or access to information and business processes. The IT service provider may directly support an organization based on the organizations governance and standards, or indirectly support an organization "on behalf of" the organization utilizing its own governance and standards.
Public Utility Commission of Texas (PUCT)	Regulates the state's electric utilities, implements legislation related to those utilities, and offers customer assistance in resolving customer complaints.
REP of Record (ROR)	A REP who has an active agreement with a retail customer for the purpose of buying electricity at retail. With this active agreement, the ROR has access to the retail customer's energy usage.
Retail Electric Provider (REP)	Sells electric energy to retail customers in the areas of Texas where the sale of electricity is open to retail competition. A REP buys wholesale electricity, delivery service, and related services, prices electricity for customers, and seeks customers to buy electricity at retail.



# SMART METER TEXAS™

Smart Meter Texas (SMT) Entity Account	A dedicated account on the SMT portal belonging to either a Residential Customer, Small Commercial Business, ROR, CSP, TDU, ERCOT, OPUC or the PUCT. A single DUNS for a ROR, CSP or TDU is required for validation during the SMT portal registration process. The entity account will have a minimum of one DUNS that will be associated with the customer ESI IDs based on the services being provided by the entity, but may elect to have many DUNS associated with the entity account capable of performing customer services based on its unique identity. If an entity wants to further segment their organization, SMT will support the creation of multiple entity accounts representing one or more DUNS for each entity account.
Smart Meter Texas (SMT) Entity User Types	Each entity account will support up to 100 administrators and an unlimited number of end users. The <b>SMT Entity Administrator</b> , using a unique set of credentials (e.g. ID and Password) for that entity account, will be responsible for managing the entity account information and profile, DUNS information and profile, manage integration services with SMT, manage alerts and/or notifications received from SMT related to cyber security and operational events, generating the appropriate integration service credentials using the SMT portal, and manage the end users requesting access on a per DUNS basis. The <b>SMT entity end user</b> , using a unique set of credentials for that entity account, will be responsible for requesting access to one or more DUNS from the SMT entity administrator and managing customer-related services associated on a per DUNS basis. The SMT entity users can be those individuals directly associated with the entity, or those individuals providing services on their behalf.
Smart Meter Texas (SMT) Portal	A website operated and managed by AEP Texas Company, CenterPoint Energy Houston Electric LLC., Oncor Electric Company LLC., and Texas-New Mexico Power Company to store energy usage data, meter and premise attribution, and provide data access services (e.g. FTP, web service, download) to Residential Customers, Small Business Customers, RORs, CSPs, TDUs, ERCOT, OPUC and PUCT.
Texas Office of Public Utility Counsel (OPUC)	Represents residential and small commercial customers in electric utility matters.
Transmission and Distribution Utility (TDU)	Delivers energy and provides energy usage metering services through their Transmission and Distribution facilities to a specific premise (e.g. home or business).

*Table 26: Glossary of Terms*