# Smart Load Solutions
# Consumer and producer model API documentation

2015

## Contents

## 1 Introduction

Production/consumer model purpose is to optimize the power production/consumption plan for the next day by maximizing revenue for the producer or by minimizing the total cost for the consumer. The producer and consumer consists of source, producer (consumer) and storage. Source consists of raw material to be used in production. Producer uses the raw material to create the final product, which is stored in storage.

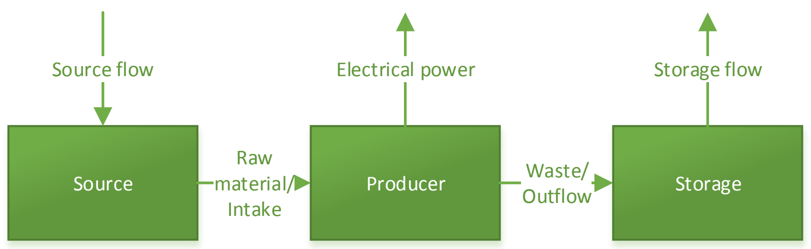

Figure 1: General scheme for producer model

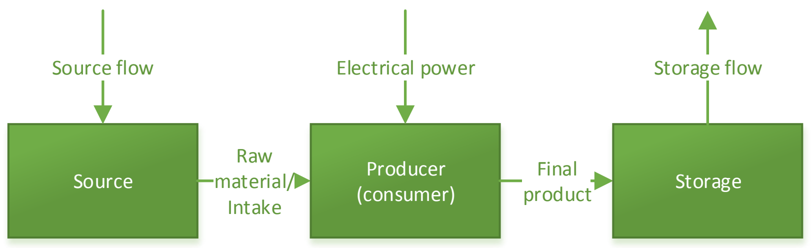

Figure 2: General scheme for consumer model

It can be seen from figures 2 and 1 that consumer or producer actually consists of three components:

- producer or consumer;
- source or input before the producer or consumer;
- storage or output after the producer or consumer.

Source or storage may be omitted.

## 2 Authentication

For authentication one needs to create an user account at SLS registration page and a deviceID for the device, which will be used for authentication.

### 2.1 Creating user account

The user account can be created by using existing Google account by clicking on "Sign in with Google" or by creating a new one by clicking "Sign Up".

### 2.2 Device ID creation

After you have managed to log in to SLS registration page, one can create devices by clicking on "Devices" and "Add New Device". After the device has been created, the deviceID is displayed under "Device Signature". The `deviceID` is used in sensor data sending and optimization request validation.

## 3 Limits and restrictions

Currently there are no restrictions applied to the service.

## 4 Source and storage sensor data

### 4.1 Description

Sensor data is gathered to learn the characteristics of the source and storage. The data should be sent after every 1 to 15 minutes.

### 4.2 Message components

#### 4.2.1 Required components

- `IL` – Level of the source
- `SL` – Level of the storage
- `P` – Power of the producer or consumer in MW

The units of source level and storage level are up to the user to choose. It is suggested to use the same units as the limits. For instance in the case of hydroelectric plant, if the maximum level of the reservoir is measured in meters then the source level should be in meters.

### 4.3 Sample messages

Let us assume that we are optimizing a hydroelectrical plant. One sensor measures the level of the reservoir (source), another measures the level of the river (storage) and one measures the power output of the plant. Let's say that reservoir level reads 10m, river level reads 5m and power output is 1MW. Then following message needs to be sent.

```
POST
https://smartloadsolutions.servicebus.windows.net/sensorinputdata/messages
HTTP//1.1
```

Header:

```
Authorization: SharedAccessSignature sr=smartloadsolutions.servicebus.windows.net
&sig=deviceID
&se=1474525966
&skn=SmartMeters
Host: smartloadsolutions.servicebus.windows.net
```

Body:

```
{"ID":"deviceID", "IL":10, "SL":5, "P":1}
```

The `deviceID` is meant to be replaced by deviceID obtained by registering your device.

### 4.4 Response

If the sending is successful, response with code 201 is sent back.

## 5 Consumer and producer model optimization requests

### 5.1 Description

Optimization request is used to get the recommended power plan for the consumer or producer. The optimization requests consists of the requirements and limitations of the device. The optimization request can be sent once in a day or more frequently i.e. if the limitations change rapidly. One can see different ways to use SLS optimization service on SLS optimization API page.

### 5.2 Request message components

#### 5.2.1 Required components

- `Source.InitialLevel` – Initial level of the source.
- `Source.FinalLevel` – Source level at the end of the optimization period (e.g. end of the day).
- `Source.MinimumLevel` – Minimum level of the source.
- `Source.MaximumLevel` – Maximum level of the source.
- `Producer.MinimumPower` – Minimum electrical power of the consumer or producer in MW-s.
- `Producer.MaximumPower` –Maximum electrical power of the consumer or producer in MW-s.
- `Storage.InitialLevel` – Initial level of the storage.

- `Storage.FinalLevel` – Storage level at the end of the optimization period (e.g. end of the day).
- `Storage.MinimumLevel` – Minimum level of the storage.
- `Storage.MaximumLevel` – Maximum level of the storage.

#### 5.2.2 Optional components

The components below are usually learned by machine learning, but can be overridden by optimization request.

- `Source.Flow` – Input flow to the source (e.g. m/h).
- `Producer.SourceChangeCharacteristic` – Source change characteristic, which tells us how many units of raw material is consumed per one unit of electrical consumption (e.g. m/MWh).
- `Storage.Flow` – Flow out of the storage (e.g. m/h).
- `Producer.StorageChangeCharacteristic` – Storage change characteristic, which tells us how many units of raw material is produced per one unit of electrical consumption (e.g. m/MWh).

### 5.3 Sample messages

Let us assume the same example as before with the hydroelectrical plant. Let's assume some numbers for the for the required components described in section 5.2.1 and let's send a JSON message.

```
POST
http://slsoptimizationservice.cloudapp.net/json/reply/OptimizeProducer
HTTP//1.1
```

Header:

```
Host: slsoptimizationservice.cloudapp.net
Content-Type: application/json
```

Body:

```
{"producer":{"maximumPower":5,"minimumPower":0},
"source":{"initialLevel":10,"maximumLevel":15,"minimumLevel":5,"finalLevel":10},
"storage":{"initialLevel":5,"finalLevel":5,"minimumLevel":2,"maximumLevel":10},
"deviceID":"deviceID"}
```

# 6 Heating and cooling model optimization responses

After the optimization request has been sent, the power consumption of consumer or producer for each hour is sent back.

- `power` – consumer or producer power consumption or production for each hour in MW;

- `timeStamps` – timestamps for power (starting time);
- `report` – detailed report of the optimization;

## 6.1 Sample message

Sample message returned from the service. The report has been cut out for better reading purposes.

```
{"power":[0.0025,0.0006,0.0006,0.0026,0,0,0,0.0006,0.0006,0.0006,
0.0006,0.0006,0.0006,0.0006,0.0076,0,0,0,0,0,0,0,
0,0,0.0006,0.0006,0.0033,0,0,0,0,0.0012,0,0],
"timeStamps":["\/Date(1444302000000-0000)\/","\/Date(1444305600000-0000)\/",
"\/Date(1444309200000-0000)\/","\/Date(1444312800000-0000)\/",
"\/Date(1444316400000-0000)\/","\/Date(1444320000000-0000)\/",
"\/Date(1444323600000-0000)\/","\/Date(1444327200000-0000)\/",
"\/Date(1444330800000-0000)\/","\/Date(1444334400000-0000)\/",
"\/Date(1444338000000-0000)\/","\/Date(1444341600000-0000)\/",
"\/Date(1444345200000-0000)\/","\/Date(1444348800000-0000)\/",
"\/Date(1444352400000-0000)\/","\/Date(1444356000000-0000)\/",
"\/Date(1444359600000-0000)\/","\/Date(1444363200000-0000)\/",
"\/Date(1444366800000-0000)\/","\/Date(1444370400000-0000)\/",
"\/Date(1444374000000-0000)\/","\/Date(1444377600000-0000)\/",
"\/Date(1444381200000-0000)\/","\/Date(1444384800000-0000)\/",
"\/Date(1444388400000-0000)\/","\/Date(1444392000000-0000)\/",
"\/Date(1444395600000-0000)\/","\/Date(1444399200000-0000)\/",
"\/Date(1444402800000-0000)\/","\/Date(1444406400000-0000)\/",
"\/Date(1444410000000-0000)\/","\/Date(1444413600000-0000)\/",
"\/Date(1444417200000-0000)\/","\/Date(1444420800000-0000)\/"],
"report":"===Solver Foundation Service Report===..."}
```