BESC Energy Savings Scheme (ESS) API Specification

Version: 2.0.0

8 January, 2020

Revision Chart

Version	Primary Author(s)	Description of Version	Date Completed
Draft	TANLSF /Clement	Initial draft created for distribution and review comments	15/04/2018
0.9.0	TANLSF /Clement	First complete draft, which is placed under change control.	22/10/2018
1.0.0	Chun Lam	Change authentication method - using API key in header, added API endpoints brief description, added swagger info	02/05/2019
1.0.1	Chun Lam	Change query type of start_date and end_date to integer in epoch seconds, add 1 more response status which is 500. Remove _v as response.	13/05/2019
1.0.2	Chun Lam	Added checksum feature, for data accuracy. Added txHash as response after data push. Added blockchain endpoint for checking data accuracy with txHash.	29/05/2019
1.0.3	TANLSF	Updated Overview and Services Flow section description. In EnergySaving body, the frequency of meter reading has been added.	23/06/2019
2.0.0	Chun Lam	Updated additional API and new format of data	09/01/2020

Contents

Introduction

Overview

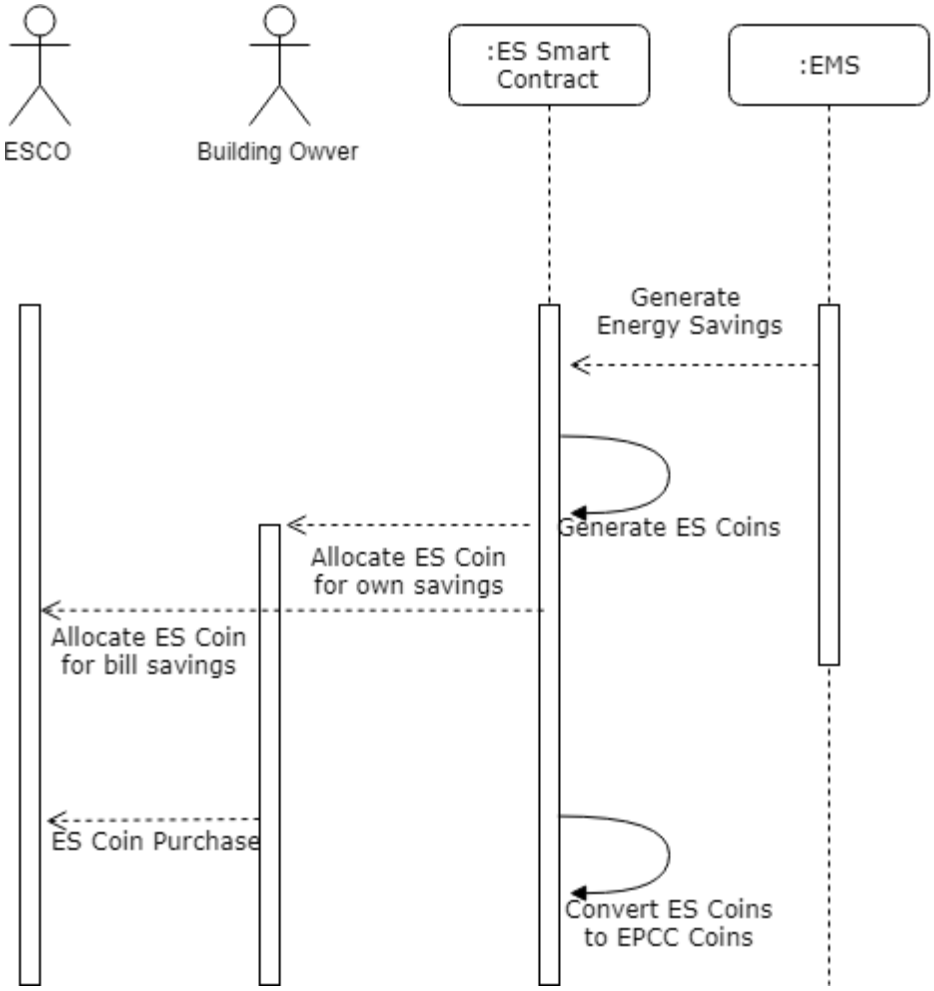
The objective of this document is to facilitate the integration of Blockchain Energy Savings Consortium platform (BESC) with the Energy Management System (EMS) site for (renewable) energy produced or energy usage consumption. The energy data is to be sent for tracking in a cloud based Blockchain network by the EMS using HTTP Requests.

This specification describes the format of the HTTP requests. The energy data send via a RESTFUL HTTP request to BESC (Carbo Server) platform which will then store it in a Blockchain node and calculate the energy savings and CO_2 emission savings by comparison with a baseline. The energy savings and/or CO_2 emission savings transactions that is recorded in the Blockchain network will be tokenize into Energy Savings coin (ES Coin) and Carbon credit coin (EPCC Coin) in order to allocate the savings as per agreement using a Smart Contract.

In order to integrate into BESC Energy Savings Scheme (ESS) HTTP interface, the EMS site will need to implement a HTTP client and server according to the specification. The client will initiate a request using HTTP post or HTTP get to the BESC specific URL and the BESC (Carbo Server) platform will then return indication of success or failure of the call to the EMS via a JSON object.

Service Flows (Sequence Diagram)

This section will describe the common interaction flows between EMS site and BESC using the ESS API. The design concept of BESC flow is based on the EMS site ability to extract the energy data from local power meters and each of the entities (Building Owner/ ESCO) having own a BESC App (with built-in e-Wallet) to store the Energy Savings coins received.



Getting Started

Pre-requisite Information

An access account will be assigned when client signed up the service.

You will need the following from BESC to use the ESS API:

1 Valid account id (unique for each client)

2 Project name (unique for each project under the client)

3 The BESC specific URL(s)

   a. BESC ESS request URL (to read energy savings information)

   b. BESC ESS post URL (for energy savings)

4 A valid API key(s). You will need to include this api key in the header of the http request to make a valid request.

You will need to provide the following to use the ESS API:

1. NEM addresses for the participating entities: -
   a. Building Owner
   b. ESCO (Energy Service Company)
2. Baseline usage for the duration of 24 hours at hourly interval.
3. Savings distribution ratio for all participating entities. This will be used to configure the smart contract.

Please contact your account manager to obtain the above information.

Server URL Provisioning

BESC's Carbo Server: http://carboapi.besc.online/besc-data

API Format

Energy Savings API Request Structure

- HTTP POST request to https://carboapi.besc.online/energysaving
- The content type is application/json
- The body of the POST contains the EnergySaving JSON object
- The header will contain the apikey with value of API Key

EnergySaving header

The header stores the API Key used to access BESC Carbo Server. It also indicates the body's Content-Type

Key	Value	Value Example
apikey	<apikey>	Wsi2jdw12asdjhw4ovmn
Content-Type	application/json	
checksum (only required when passing data to API)	<sha1_string>	ea8ca2ef3e8c92d89dfc

EnergySaving body

An EnergySaving object is used to transfer the energy usage information to BESC Carbo Server for storage and calculation of energy savings. In order to verify the quantity of electrical energy consumed, a half (1/2) hourly reading shall be obtained. Any positive energy savings will then trigger the Smart Contract that will distribute the ES Coins to the participating entities.

The following is an example request body: -

```
{
	"Project": "Hicom",
	"DateTime": "2018-03-16T06:00:00",
	"Devices": [
		{
			"DeviceId": "AC11",
			"EnergyUsage": 130.0,
				“EnergySaved": 50.0,
			"Efficiency”: 10.0,
			“Formulas”: []
		},
		{
			"DeviceId": "AC22",
			"EnergyUsage": 180.0,
			“EnergySaved": 50.0,
			"Efficiency”: 10.0,
			“Formulas”: []
		}
	],
	"TotalEnergyUsage": 310.0,
	"TotalEnergySaved": 100.0,
	"AverageRT": 188.0,
	"Geolocation": "2.91667, 101.7",
}
```

Description of the fields:

Field	Description	Format
Project	The unique Project Name	String length 2-15
DateTime	Javascript ISO DateTime. Since the energy savings is calculated on a per 1/2 hour basis, the seconds should always be zero	YYYY-MM-DDTHH:MM:SS
Devices	Information for the energy usage for each individual device that constitutes the entire project energy saving equipment. This information is not used for calculation but is stored for future reference.	
DeviceId	Unique device id	
EnergyUsage	The energy usage for an individual device. Unit is in kW/h	
EnergySaved	The energy saved for an individual device. Unit is in kW/h	
TotalEnergyUsage	The total energy usage for this data set of the project. This is the sum of all individual device energy usage. Unit is in kW/h	
TotalEnergySaved	The total energy saved for this data set of the project. This is the sum of all individual device energy saved. Unit is in kW/h	
AverageRT (Optional)	The average refrigeration tons. Please put 0 if not applicable	
Geolocation	The location of the project. It can be the GPS location coordinates. At least country level information needed	

The following is an example of response to the energy saving POST:

```
{
    "_id": "5ae31fced165800035575b7e",
    "Project": "SedaRE",
    "EnergySavingHash": "03a384602bde44bde7aaab9693ca713683ef951c",
    "Status": "Recorded",
    "txHash": "<transactionHash of blockchain transaction>"
}
```

Description of the fields:

Field	Description	Format
_id	This is the id of the energy savings record that is stored in MongoDB	
Project	The unique Project Name	
EnergySavingHash	A SHA-3 512 (Secure Hash Algorithm 3) hash of the energy usage JSON object. This hash will be stored as message in the NEM transaction blockchain. It serves as a verification process of the energy saving data either it has been tampered.	
Status	Current status of this record. Possible values are Recorded or Transacted. The energy usage is recorded first before savings are calculated and written to blockchain hence Recorded means it is only stored in database. The recorded usage is then calculating the energy savings and ESCoin distributed according to the Smart Contract via transactions in the NEM Blockchain. At this stage the status is Transacted.	
txHash	Transaction hash of the blockchain transaction that the EnergySavingHash is saved into. It is used to verify the data validity.	

API Endpoints Description

Host : https://carboapi.besc.online (might change according to different environment)

Base path : /energysaving

Final endpoint = host + base path + endpoint below

/project/{project_id}

Method: **GET**

Get project energy saving data. Data response is in descending order.
Query Parameters:

- offset – data return start from offset number
- limit – the number of data will be return back to the requested client, maximum 100
- start_date - the start date of the querying data, in epoch seconds
- end_date – the end date of the querying data, in epoch seconds

Method: **POST**

Receive project energy saving data as structured above (EnergySaving body).

/project/{project_id}/baseline

Method: **GET**

Get baseline of the project.
Headers:

- apikey : The apikey provided to the project
- Content-type : application/json

/project/{project_id}/formulas

Method: **GET**

Get formulas of the project.
Headers:

- apikey : The apikey provided to the project

- Content-type : application/json

/debugs/{project_id}/last

Method: **GET**

This endpoint is for debugging purpose. It returns the last submitted energy saving data that get saved into the API database.
This endpoint will be disabled once go into production.

/blockchain/{project_id}/txHash/{txHash}

Method: **GET**

This endpoint allow the checking of validity of data with txHash. Each txHash is attached to each data, it can uniquely represent the data set that get pushed into the API..

Endpoints Responses

In most cases, all the endpoint will return with a group of similar http response code.
Response code:

- 200 – Success, API request is working fine
- 400 – Invalid content, properly some http request syntax/content-type error.
- 401 – Authentication failed. Unauthorized access.
- 404 – Trying to send request to non-existing endpoint/ not found error
- 409 – Invalid argument, some conflict happened
- 500 – Server encountered error

Swagger Specs File

API is built based on Swagger Codegen. We have swagger v2 and openAPI(swagger v3) specs file available (only for energy).