Contact : deyraud@gimelec.fr

Flex Ready® API – CEMS / Aggregator

This document defines the APIs used to interface CEMS (Customer Energy Management Systems) with flexibility operators. Throughout this document, these operators are referred to as "Service Providers," in accordance with the terminology in 62746-4. When a building has a BACS (Building Automation and Control System), particularly when required by the EPBD Directive, the CEMS is a function of the BACS. In this case, the terms CEMS and BACS may be used interchangeably.
These APIs are defined according to the functionality levels identified for the BEMS function.
Each level of functionality corresponds to a contractual commitment.
This document is a first specification of the CEMS-Aggregator interface. It incorporates the elements of the IEC 62746-4 standard that are relevant to the use case. This specification may evolve to enhance the flexibility services taken into account, and to keep pace with changes in the standard. Whenever possible, upward compatibility will be ensured.

1. What is the API.

An API (Application Programming Interface) is the interface provided by a server to access or receive information from external client software.
An API is made up of
- a: software on the server (php script type) which will execute a request and return information
- b: a definition of the format of the request sent (e.g. POST or GET https)
- c: the way in which the client calls this functionality: this is the visible part of the API
- d: an implementation of this function call (in the form of a supplied library, for example).

For example, an API for requesting a flexibility operation from the Service Provider to a CEMS according to a given profile:
c: **func askFlex(requestID: UUID, CEMS: CEMSID, asset: AssetID, flexProduct: FlexProductType, power: PowerTimeSeries) → Ack**

- The service provider sends the request reference, the CEMSID, the relevant asset, the flexibility product and the profile.
- In return, the CEMS responds by accepting or rejecting the request.

a: The CEMS analyzes the request to determine if it can accept it.
b and d: This is the implementation of the request, which may be available in libraries.

2. Reminder: CEMS characteristic parameters.

Two main parameters are taken into account:
- the ability of the CEMS to forecast its future consumption (horizon H + 2 to D + 2), potentially usage by usage.
- the building manager's choice to commit to a level of flexibility.

Three levels of CEMS considered
Level 1: CEMS with no forecasting capacity - no contractual flexibility commitment
Level 2: CEMS without forecasting capability, with contractual flexibility commitment
Level 3: CEMS with forecasting capacity and contractual flexibility commitment – Not in V1 spec.

APIs are cumulative (level 2: level 1 plus level 2 specific).
APIs are secured, according to cyber security requirements defined in a separate document.

Appendix A of the IEC 62746-4 standard defines the sequence of exchanges between actors for the Use case: Incentive-based building energy management. The Flex Ready® API is based on this sequence (attached at the end of the document) for the exchanges it handles.

3. Data model

The data model is based on IEC 62746-4 data model.
Generally speaking, the CEMS has an mRID identifier (CEMSID), assigned at the time of Flex Ready® registration. This identifier must be unique (as defined by UUID standard) compliant with IEC 62746-4 §5.1 Master resource identifiers.
The controls assets, which may correspond to a functional perimeter (heating, IRVE, etc.) or to zones of the building, or even the entire building. The asset name “WholeBuilding” is reserved for the case where the only asset taken into account is the entire building. In this case, no other assetID is defined.

AssetIDs for Assets are defined by the building administrator. They are provided to the Service Provider in the contract. It is recommended to use a UUID for AssetIDs.
Note: If the Asset is a generator, this AssetID is a UUID, registered by the DSO. Different Assets can be managed by different Service Providers: the API supports this capability.

Time is GMT time. Dates format are as per ISO 8601 standard:
So, for weekdays : 1: Monday, 2 : Tuesday, … 7 : Sunday

In accordance with IEC 62746-4, FloatQuantity is a value accompanied by a unit (eg., W, kW, ...).
A power PowerTimeSeries is a sequence [power: FloatQuantity, date: Date, start: Time, end: Time]. This sequence may consist of a single point.
A flexibility request corresponds to a FlexProductType. In this version of the API, 2 types are used, depending on the time horizon of the request:
- Hour ahead market (HAM)
- Day ahead market (DAM)

Some API return a code:

200 OK: Success
202 Accepted: Request successfully accepted
400 Bad Request: Format error in request
422 Unprocessable Entity: Business validation error

At this stage, the APIs do not allow additional data; a cyber security impact analysis will be carried out to define the conditions under which additional variables could be introduced by the user.

4. Level 1 API

At level 1, CEMS cannot estimate future consumption. The building can therefore not enter into a contractual result commitment with the Service provider *but simply make its best efforts.*

• Potential of the building's assets: the Service provider queries the building's server; the API provides the Service provider with static information on the theoretical potential of the assets that can be used for flexibility (which can be evaluated using GOFlex methods).

func getCEMSAssets(CEMS: CEMSID) → [MEPID: String, [(asset: AssetID, flexProduct: FlexProductType, potentiel: [PotentielFlex])]]

MEPID: Market Evaluation Product ID (usually the grid connection point ID)

AssetID is a UUID String.
The asset type (a first list is defined in IEC 62746-4, Table 25) will be included in a future version of the specification.
If the asset is the entire building, the UUID is that of the CEMS.

Protocol: Timeout: 60s. In the event of no response, an error message is generated and logged. If the CEMS ID is wrong, CEMS returns an empty list.
The API provides a table giving a flex potential for each asset, indicating the type of flex service.

potentiel is an array of PotentielFlex (several elements if differentiated by seasonality, for example). In version 1, a single potential value, at the time of the query. Confidence level is not used in V1.
PotentielFlex is a data structure containing: power, mobilization delay (default: 1 h), max duration, number of activations per day (default: 1, the only value accepted in V1), time periods, optionally a confidence level on implementation (excluded in V1). The potential is that valid at the date of request.
PotentielFlex :

yearPeriod: startDay, endDay
activationPeriods: (months : [1..12], weekDay : [1..7], hours: [Hour]) inside the yearPeriod
notification : Int A duration expressed in minutes
maxDuration : Int A duration expressed in minutes, since the start of an activation period
maxActivationsPerDay : Int
confidence level : 1..5 1 minimum - 5 total confidence. This parameter is optional.

For example:
yearPeriod: (2025-03-25, 2025-06-30)
ActivationPeriods: (months: [3, 4, 6], weekDay: [6,7], hours: [10, 17]): flex can be activated (for this period of the year, every month except May, on Saturdays and Sundays, at 10am GMT and 5pm GMT, for a maxDuration.

The various yearPeriod must be consistent. In the event of inconsistency (e.g., 2 PotentielFlex entries for a same time), the result is not guaranteed (in general, the first one to match the time period will be taken into account).

An assetID is predefined for the entire building. If this asset exists, the list of assets in response must contain only this one asset.

- Flexibility Request sent to CEMS: The Service provider sends a request to CEMS (and generates an identifier for the request) for a given period, by asset (the asset may be the entire building); but with no guarantee of compliance. No response expected, except for an acknowledgement of receipt (or notification of data error) which allows the Service provider to restart (1 time in V1) his request in the event of no response or error.

  func askFlex(requestID: UUID, CEMS: CEMSID, asset: AssetID, flexProduct : FlexProductType, puissance: PowerTimeSeries) → Ack.

  Ack indicates successful receipt and the status "Received" or "Rejected" (Yes/No); the CEMS does not commit (at Level 1): ReceivedWithoutCommitment.

requestID is a UUID that must be retained by the Service Provider and the CEMS as potential evidence.
FlexProductType is one of the types defined in standard 62746-4.

The types used in this version of the API: RPD (Ramp Down: downward flexibility), optionally RPU (Ramp Up: upward flexibility), if the asset is a generator.

Protocol: Timeout: 5s. CEMS has no evaluation to do, it just indicates that the request has been received or rejected, so that the Service provider can integrate it into his flex potential calculations.
CEMS has been able to check that the request is consistent with the flex potential. If not, it does not respond, which cancels the request.

• Request activation. The service provider informs the CEMS that the request has been included in its activation plan. The activation is sent, taking into account the mobilization time. The CEMS has no execution commitment (Level 1). Expected response: YES/NO. The requestID (UUID generated by the issuer) allows the request to be tracked.

func activateFlexRequest(requestID: UUID, CEMS: CEMSID, asset: AssetID, flexProduct: FlexProductType, puissance: PowerTimeSeries) → Ack

Protocol: Timeout: 5s. The CEMS does not have to make any evaluation, it only indicates that the request has been registered and that it will make its best efforts to satisfy it (the incentive being the remuneration stipulated in the contract). If there is no response, the Service Provider excludes this request from its flex potential.

- Request cancellation: not present in V1

func cancelFlexRequest(requestID: String) → Ack

Note: The service provider may cancel the request by returning the request (with the same requestID) with power = 0, provided the mobilization deadline is met.

- Execution reporting: After the activation period, the Service Provider requests from the CEMS the power demand curve (at a time step specified in the PowerTimeSeries—fixed at 15 minutes = 900 seconds in this version of the API) over the flexibility request period, for the specified Asset. The CEMS responds with a table providing, at the selected sampling interval (15 minutes), the called power per Asset (note: CEMS: CEMSID, asset: AssetID—included for security, but normally redundant with requestID. The definition of the actual power calculation must be specified in the contract. In V1: average power value over the period).

func realiseFlexRequest(requestID: UUID, CEMS: CEMSID, asset: AssetID) → [(requestID: UUID, reported: PowerTimeSeries)]

Protocol: Timeout: 15’ (900 s). CEMS needs to collect data (possibly read the counter), which justifies the long timeout. In the event of no response, the Service provider repeats its request a maximum of 1 time, then generates an error message for logging. If the Service provider detects a power > flex potential of the asset: generate an error in the log.

5. Level 2 API

At level 2, CEMS does not have the capacity to estimate its consumption, and therefore its future flexibility. The building may delegate this forecast to the Service provider *in order to enter into a contract with the* Service provider*.*

At levels 2, a flexibility proposal may come with a price proposal. CEMS will either accept or reject the proposal, taking into account any price proposals.

- Building asset potential: same as level 1.

- Instantaneous consumption by asset (less than one hour old) of assets that can enter flexibility, at the time of the request, for each MEPID involved.

func getCEMSAssetsConsos(CEMS: CEMSID) → [pdl: MEPID, [(asset: AssetID, date: Date, heure: Heure, conso: FloatQuantity)]]

Protocol: Timeout: 15’ (900 s). CEMS needs to collect data (possibly query assets), which justifies the long timeout. In the event of no response, the Service provider repeats its request a maximum of 1 time, then generates an error message for logging.

- Historical consumption of assets.

The CEMS has archived its consumption over a period of time to be defined (in the contract, minimum 1 month), as an average value (or pMax later) over a time step (1 hour) defined in the contract. The Service provider can retrieve this history (by asset), which gives him very important information to build his forecast:

func getCEMSAssetsHisto(CEMS: CEMSID, asset: AssetID) → [(date: Date, heure: Heure, conso: FloatQuantity)]

Protocol: Timeout: 60s. If there is no response, an error message is generated and logged. If the CEMS ID or AssetID is incorrect, CEMS returns an empty list.
A 1-month history corresponds to 720 values per asset.

- The Service provider sends a proposal with a price offer to CEMS, which accepts or rejects it, or asks for renegotiation in its Reply.

  func askFlexWithPrice(requestID: UUID, CEMS: CEMSID, asset: AssetID, flexProduct: FlexProductType, puissance: PowerTimeSeries, askingPrice: PriceTimeSeries) → Reponse

  The answer is explicit (accepted, refused, to be modified).

PriceTimeSeries: TimeSeries, with values in €. In V1, only one data point. Its time window is the same as for power.
Note: Incorporates the time window, amount, and price (BidPriceSchedule) parameters from the IEC 62746-4 standard.

Protocol: Timeout: 15'. CEMS must assess its ability to execute the flex request, which justifies the timeout duration. If the CEMS is unable to evaluate, it must refuse the request for this asset. If an identifier is incorrect, CEMS responds with an error code (identifier unknown).
If there is no response, the request can be repeated 1 time.
If the CEMS refuses, the request is not retransmitted.

In version 1 of the protocol, only the power can be modified. If the price is unsuitable, a refusal response is sent.

Note: 62746-4 defines ProductBid as “a container which has two associations: the top being the MarketProduct (Energy, Reserve, Regulation, etc.) and the bottom association being the BidPriceSchedule”.
The BidPriceSchedule class captures data on three axes signifying:

1) The time window, e.g. from 6:00 to 7:00
2) The amount of service, e.g. 1 MWh
3) The asking price, e.g. 20 €/MWh

In the case of API Flex, the MarketProduct is a request for flexibility (Regulation up or down).
This askingPrice data is optional, depending on the nature of the contract (which may define a default value). CEMS needs to know whether or not to receive this information.

Note: from Table A.3 of IEC 62746-4:

Table A.3 – Information exchanged in price notification and energy consumption plan notification

Name of information	Description of information exchanged	Requirements to information data
Energy Price for Building Group	Time series of price for energy for Building Group	Time series of price for energy consumption, (price/Wh) per unit time period (e.g. 30 min, 15 min, etc.)
Energy Price for Building	Time series of price for energy for Building	Time series of price for energy consumption, (price/Wh) per unit time period (e.g. 30 min, 15 min, etc.)
Intended Energy consumption	Time series of energy consumption and in the future (e.g. day ahead, week ahead, etc.) which is set by Building Group Manager as the control target under the given incentive / price.	Time series of Energy (Wh) per unit time period (e.g. 30 min, 15 min, etc.)
Energy consumption plan for Building	Time series of energy in the future (e.g. day ahead, week ahead, etc.) for the building under the given incentive / price.	Time series of Energy (Wh), Power (W) and emission quantity of CO2 (gram) per unit time period (e.g. 30 min, 15 min, etc.)
Energy consumption plan for Building Group	Time series of energy consumption in the future (e.g. day ahead, week ahead, etc.) for the Building Group, which is made from Energy consumption plans of all Buildings within the Building Group.	Time series of Energy (Wh) per unit time period (e.g. 30 min, 15 min, etc.)

- The Service provider sends the request confirmation (according to mobilization time), identified by its requestID, to CEMS if it has accepted the initial request, for activation.

func activateFlexRequest(requestID: UUID, CEMS: CEMSID, asset: AssetID, flexProduct: FlexProductType, puissance: PowerTimeSeries, askingPrice: PriceTimeSeries) → Ack

Protocol: Timeout: 5s. To acknowledge receipt.
Expected response: YES/NO. No details on the reason for rejection (the standard is currently being developed in this regard).
Failure to send an **activateFlexRequest** within the activation window (for askFlexWithPrice) results in the cancellation of the initial request.

- Cancel request confirmation: not present in V1

func cancelConfirmation(requestID: String) → Ack

6. Level 3 API

At level 3, the CEMS will be able to estimate its consumption, and therefore its future flexibility, in order to make proposals to the Service provider *and enter into a contract with the* Service provider*.*

Level 3 API will be specified in a future version of Flex Ready® API.

7. API Validation

To obtain the Flex Ready label, with level, the CEMS supplier must have verified that it complies with the defined interfaces.

Initially, this will be done by self-declaration, after running test scenarios on the emulator.

For this purpose, Flex Ready provides:

- a Flex Ready Service provider server emulator.
- a CEMS emulator with the different levels.
- definition of the test sequence to be carried out (including cases of erroneous data).

The emulator provides a test report at the end of the sequence, to be attached to the self-declaration.

8. Test scenarios

The exchange of Identifiers (e.g., CEMSID) is predefined at the time of contracting and is not part of the API. They are “hard-coded” into the emulator.

For each API:

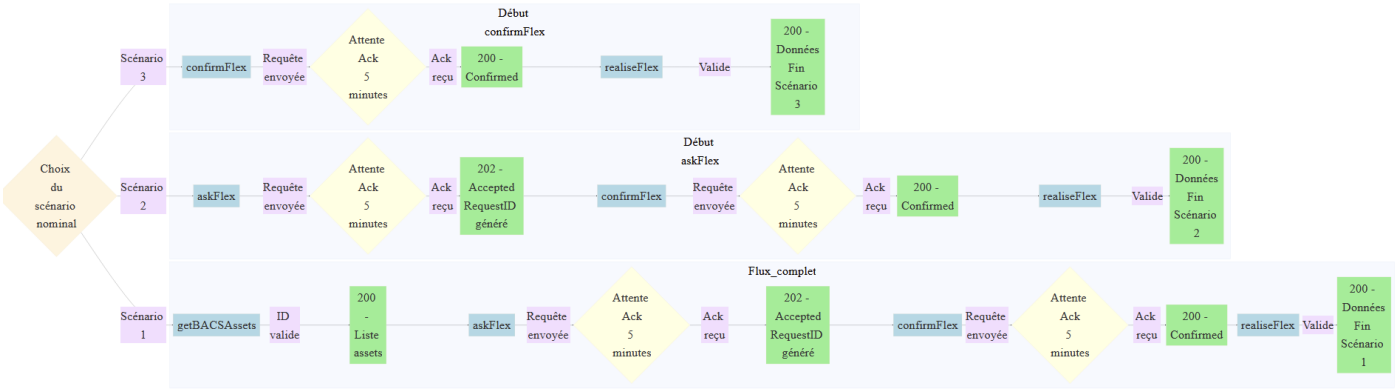
- Authentication exchange, according to the chosen protocol (Oauth 2., to be confirmed)

- If the identifier (e.g., CEMSID) is not recognized, CEMS ignores the request and replies “Error”. After a repeat (5) of an invalid identifier, CEMS warns of a possible intrusion or denial-of-service attempt.

- Sending a valid request
- Response: once in acceptance, once in refusal

- Test invalid requests (on dates, AssetID, volume, etc.),

Test scenarios are presented in the diagrams below:



Appendix 1 - General exchange process (from IEC 62746-4). The Flex Ready API field is outlined in red.

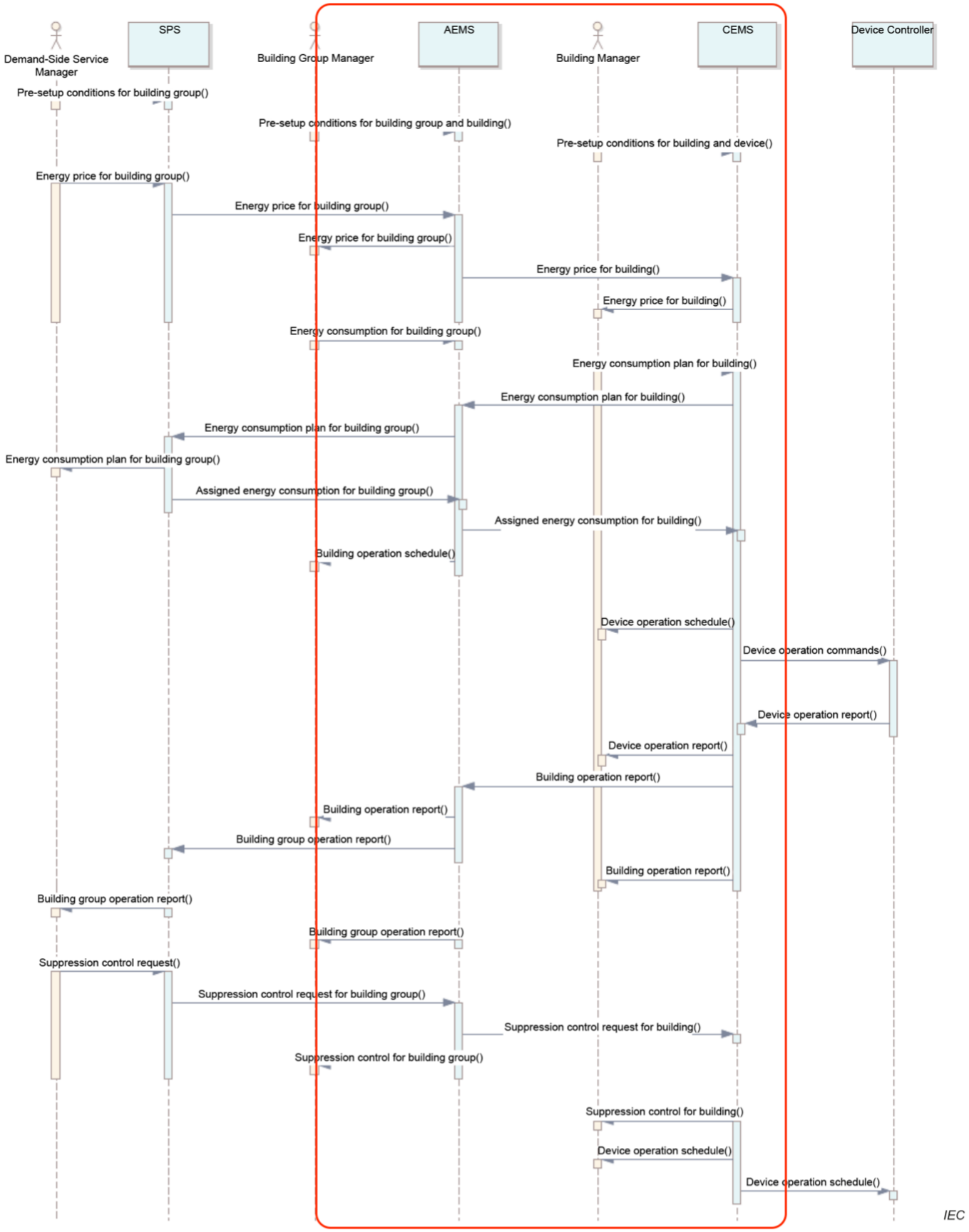


Figure A.2 – The whole view of this use case

Appendix 2. Definition of acronyms

Acronym	Phrase
AEMS	Aggregator Energy Management System
CEMS	Customer Energy Management System
DAM	Day-Ahead Market
DER	Distributed Energy Resource
EPS	Electrical Power System
PV	Photovoltaic
RTM	Real-Time Market
SPS	Service Procurement System
UML	Unified Modeling Language
VPP	Virtual Power Plant
XML	Extensible Markup Language
XSD	XML Schema Definition