



SEMP

Application Note

Electric Vehicle Charger

Revision History

History		Comment
1.0.0	2013-03-13	<ul style="list-style-type: none">• First edition
1.0.1	2014-04-03	<ul style="list-style-type: none">• EarliestStart must not be in past• PlanningRequest adjustments for energy needs described
1.0.2	2014-09-15	<ul style="list-style-type: none">• Timeframes with mandatory and optional parts are now supported. The mandatory part is charged with full power, the optional part with variable power.
1.0.3	2014-11-17	<ul style="list-style-type: none">• Serial-number now mandatory to prevent auto-generated serials in Sunny Portal. Devices without serial-number are still accepted for backward compatibility.• Example section added• Added comment, that power DeviceInfo/Status is sum of all phases

Explanation of Symbols used in this Document

To enable optimal usage of this manual and safe operation of the device during installation, operation and maintenance routines, please note the following description of symbols:



This symbol indicates information that is required for the optimal operation of the product. Read these sections carefully in order to ensure an optimal operation of the product and all its features.



This symbol indicates information that is essential for a trouble-free and safe operation of the product. Please read these sections carefully in order to avoid any damages of the equipment and for optimal personal protection.



This symbol indicates an example.

Content

1	Introduction	5
1.1	Scope	5
1.2	Scenario	6
1.3	Abbreviations	6
2	SEMP webservice	7
2.1	Introduction	7
2.2	Polling behavior of the EM	7
2.3	Information delivered by a Gateway	8
2.3.1	Device information	8
2.3.2	Device status	13
2.3.3	Planning requests	15
2.4	Recommendations from the EM	19
3	Examples:	21

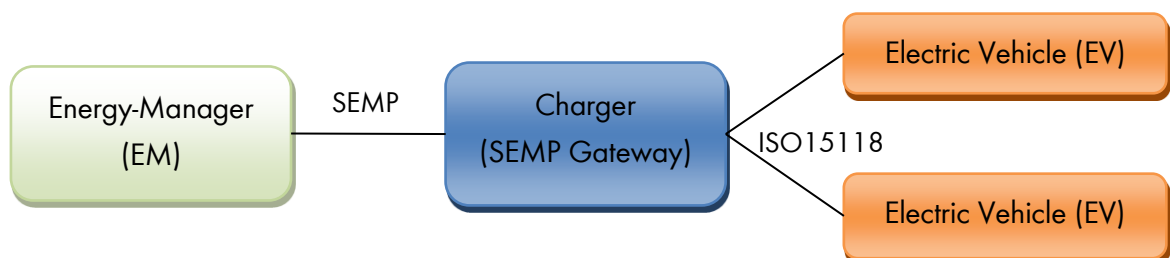
1 Introduction

1.1 Scope

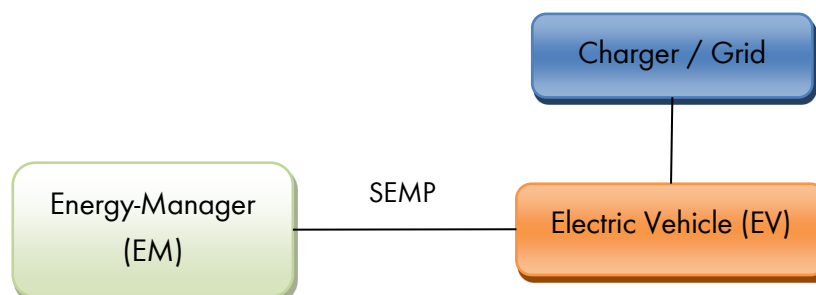
This document describes the steps and data structures in SEMS needed for energy management of an Electric Vehicle (EV) charger or charging station.

Only the SEMS web service is described in this document. The SEMS discovery mechanism, the procedure how to establish a SEMS-connection and the underlying communication mechanisms of the SEMS web service are out of scope. Those topics are described in the main SEMS specification.

The next figure shows a typical Electric Vehicle (EV) charging scenario. The Energy Manager (EM) is connected to an EV charger via an IP-based medium like LAN. The EV is plugged into the charger and may communicate its SOC and/or energy needs to the charger. A protocol typically used for this purpose is ISO15118 over power-line communication. The SEMS-Gateway (here: the charger) communicates the current energy needs to the EM which in turn creates a plan for the device. The EM sends recommendations according to its internal plan to the gateway which either transmits them to the EV or controls the EV charge process directly.



It is also possible that the EM directly communicates with the EV and not with the charger. This scenario is displayed in the next figure:



Both cases are similar in terms of the SEMP data structures used. Following, we assume that the charger will be the SEMP gateway.

1.2 Scenario

The scenario presented in this document considers excess energy charging of an EV only. This means that the EV has an optional energy demand that does not have to be fulfilled by the EM. The EM will only allocate energy to the EV if certain constraints are satisfied. A typical constraint is to use only PV energy for charging or “cheap” energy, which means energy with a price not higher as a given limit. Some of these constraints and limits are provided by the user via the EM configuration interface.

If the EM did not allocate any of the optional energy to the EV during the given time range, the EV charger can switch to a mandatory mode and charge the EV manually.

1.3 Abbreviations

EM	(SEMP) Energy Manager
SHM	SMA Sunny Home Manager. SMA’s EM which acts as a SEMP EM.
(SEMP) Gateway	The entity the SEMP EM communicates with. Either the controllable device itself or a gateway that directly controls the device or passes recommendations to it. A gateway can manage multiple devices.
(SEMP) Device	A device controllable by a SEMP EM. A device does not have to communicate via SEMP directly. It is controlled by a Gateway implementing the SEMP protocol. It is possible that the device itself is a SEMP gateway (for this, it must be able to connect to an IP-based network, e.g. via a LAN interface).
GUI	Graphical User Interface. A device’s configuration or visualization interface. In the case of the SHM the SMA Sunny Portal.
SOC	State of Charge
baseURL	SEMP base URL used as a prefix for requesting SEMP resources. It is specified by the SEMP gateway during discovery (via UPnP SSDP).

2 SEMP webservice

2.1 Introduction

The SEMP webservice is based on HTTP with the SEMP Gateway acting as HTTP server and the EM as HTTP client. The EM can request data from the gateway by sending a HTTP GET request to it. The data (embedded in a Device2EM message) is delivered in the HTTP GET response body. The EM can also send data (embedded in an EM2Device message) to the gateway by sending an HTTP POST request to it.

The basic format of the URLs used in the GET/POST requests is:

```
<baseUrl>/<name of data-structure>
```

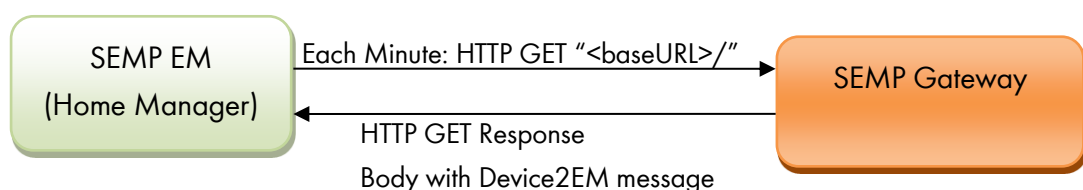
The baseUrl is provided by the gateway in the discovery process (see SEMP specification). The data-structures that can be requested by the gateway are described in the SEMP XML Schema definition (XSD) file. If no data-structure name is given ("`<baseUrl>/`") then all available top-level data-structures should be returned by the gateway.

By default, the gateway returns the requested data for all devices managed by it. With the additional query-parameter "DeviceId", the EM can also request data of only a specific device – instead of the data for all devices.

2.2 Polling behavior of the EM

The EM will periodically poll the SEMP gateway for new device information, device status information and planning requests.

With the current SEMP EM implementation of the Sunny Home Manager (SHM) the behavior is as follows: all of the available information of all SEMP devices connected to the gateway will be polled once every minute by the EM with the "`<baseUrl>/`" URL.



2.3 Information delivered by a Gateway

Whenever the EM requests information from the gateway via "<baseUrl>/" the gateway should return a data-structure in the response body with the following format:

```
<?xml version="1.0" encoding="UTF-8"?>
<Device2EM xmlns=...>
  <DeviceInfo>...</DeviceInfo>
  <!-- further DeviceInfo elements if gateway manages multiple devices -->
  <DeviceStatus>...</DeviceStatus>
  <!-- further DeviceStatus elements if gateway manages multiple devices -->
  <PlanningRequest>...</PlanningRequest>
</Device2EM>
```

The content of each component is explained in the following sections. Note that the SEMP XML namespace has been omitted here (xmlns=...). See the SEMP specification for the correct namespace URL.

As it is possible that a gateway handles multiple devices, each of the three components can occur multiple times to describe one of the managed devices. In this scenario we assume that only one device is present – representing the charger itself.

Note that at least DeviceInfo and DeviceStatus must be returned for all of the managed devices, if the data was requested from the EM with "<baseUrl>". The PlanningRequest element can be omitted for devices with no pending energy demand.

Device List

If an EM requests the DeviceInfo data structure ("<baseUrl>/" or "<baseUrl>/DeviceInfo") for all devices ("DeviceId" parameter not used), the gateway has to list all devices currently managed by it. If a device that was announced by the gateway earlier is missing in the list, the EM assumes that the device has been disconnected and discards all plans for the device.

If a device is temporarily unavailable, the gateway should keep a DeviceInfo element for the device in the list but set the value of the status field in the corresponding DeviceStatus element to "Offline".

2.3.1 Device information

The device information data structure contains static (i.e. non-changing) information about a device. The data should not change during runtime. It is split into three sections: "Identification", "Characteristics" and "Capabilities".


```
<DeviceInfo>
  <Identification>...</Identification>
  <Characteristics>...</Characteristics>
  <Capabilities>...</Capabilities>
</DeviceInfo>
```

The sub-elements Identification, Characteristics and Capabilities are described in the following sections.

Identification

The “Identification” part can be used by the EM in its GUI to help the user to identify a device. Hence ID, name, type and vendor are mandatory.

Providing a vendor specific serial number is optional but might be a good hint to identify a device if the serial number is printed on the device’s case.

The DeviceURL is also optional but (if present) can be used in the GUI of the EM to provide a link to a device’s configuration interface. Note that the EM GUI might be a web portal in the internet (for the SHM the GUI is the Sunny Portal). In this case if a device URL with a local IP-address (e.g. 192.168.x.x) is provided by the gateway, the URL cannot be accessed if the user is using the portal from a network different to the one of the gateway. So if both – a global and a local address – are available, the global address should be provided by the gateway. If an IP-address is used in the URL instead of a DNS name, it is possible that the IP-address changes over time. In this case the DeviceURL field should be updated accordingly. This is the only exception to the DeviceInfo structure being static.

Relevant fields for this scenario:

DeviceId:	SEMP Device-ID of the device to which this information relates.
DeviceName:	Name of the device.
DeviceType:	Type of device. One of the predefined values should be used. In this scenario should be “EVCharger”.
DeviceSerial:	Vendor specific serial number (equal to the one on the device’s case).
DeviceVendor:	Name of the device’s vendor.
DeviceURL (opt.):	URL to configuration interface (global or local address). This can be used in the EM GUI.

Example:

```

<DeviceInfo>
  <Identification>
    <DeviceId>F-11223344-112233445566-00</DeviceId>
    <DeviceName>LS 1</DeviceName>
    <DeviceType>EVCharger</DeviceType>
    <DeviceSerial>JFSOF342432</DeviceSerial>
    <DeviceVendor>EVCharger Ltd</DeviceVendor>
    <DeviceURL>http://evchargerltd.com/config?id=6767</DeviceURL>
  </Identification>
  <Characteristics>...</Characteristics>
  <Capabilities>...</Capabilities>
</DeviceInfo>

```

Characteristics

The “Characteristics” part specifies energy management related characteristics of the device that do not change during operation.

The specification of the maximum power consumption is required. If the device’s power consumption can be regulated by the EM in a range between a minimum and maximum power, the minimum power consumption to be recommended by the EM can be specified. Alternatively, if only fixed power levels are supported, those levels can be provided by the optional “PowerLevels” element. For more information about recommendations, see the sections about planning requests and recommendations.

If a device can be paused during runtime (which is called “interrupted” in SEMP) the device can provide constraints to prevent inefficient use of interruptions or to prevent recommendations that could have a negative impact on the devices lifetime.

- If the device has to be left in “on” state after turning it on for a given time or if the device has to stay in “off” state after pausing it, these constraints can be defined by MinOnTime respectively MinOffTime.

Note that the device or gateway is responsible to check that accepting a recommendation does not cause any damage. Even if a MinOn/OffTime is provided, the device should check that recommendations comply with the device’s constraints before applying it.

- If interrupting the device has the negative side-effect of an additional energy need during the unpausing process this can be defined by AddEnergySwitchOn.
- If unpausing causes additional costs (e.g. by increasing the wear level) this can be defined by the AddCostsSwitchOn elements.

Constraints will influence the decision of the EM whether a device will be paused or not.

Relevant fields for this scenario:

MaxPowerConsumption:

Maximum power consumption (in W), sum of all phases.

If MinPowerConsumption or PowerLevels are not defined, the power cannot be regulated by the EM.

MinPowerConsumption (opt.):	Minimum recommended power consumption if power can be regulated by the EM (in W).
MinOnTime (opt.):	When switched on (or un-paused), the device has to remain on for at least the given amount of seconds.
MinOffTime (opt.):	When switched off (or paused), the device has to remain off for at least the given amount of seconds.
AddEnergySwitchOn (opt.):	Additional energy (in Wh) needed when un-pausing the device.
AddCostsSwitchOn (opt.):	Additional costs (in €) caused by un-pausing the device.
PowerLevels (opt.):	Contains a list of fixed power levels (in W) that can be recommended by the EM.

Example:

```
<DeviceInfo>
  <Identification>...</Identification>
  <Characteristics>
    <MaxPowerConsumption>22000</MaxPowerConsumption><!-- 22kW -->
    <MinPowerConsumption>1380</MinPowerConsumption><!-- 6A at 230V (in W) -->
    <MinOnTime>3600</MinOnTime><!-- 1h -->
    <MinOffTime>1800</MinOffTime><!-- 0.5h -->
    <AddEnergySwitchOn>0.2</AddEnergySwitchOn>
    <AddCostsSwitchOn>0.025</AddCostsSwitchOn>
  </Characteristics>
  <Capabilities>...</Capabilities>
</DeviceInfo>
```

Capabilities

Defines the SEMP specific capabilities of a device.

Relevant fields for this scenario:

CurrentPower.Method:

Specifies how the current power in the device status element is determined.

A device that determines the current power by a power meter should select "Measurement". If it estimates the power consumption by look-up tables or other mechanisms select "Estimation".

This information provides a hint about the quality of the power values provided in the device status section. With it an EM can determine whether the power data can be used for learning device profiles or if it is suitable to be displayed in a GUI.

Timestamps.AbsoluteTimestamps:

SEMP supports two types of timestamps:

- Relative timestamps with the number of seconds since now (might be negative if event was in the past)

- Absolute timestamps according to the UNIX timestamp format (number of seconds elapsed since 01.01.1970 00:00 UTC).

The device has to decide whether it wants to send and receive absolute or relative timestamps. The EM evaluates this option to determine which format to use when sending messages to the device. Because of that the value of this option must not change for a device. Although a default value (see XSD file) is used if not specified, this option should always be provided by a device.

Devices that do not have a synchronized clock (with time server protocols like NTP or radio control like DCF77) or do not have a reliable absolute time source should use relative timestamps.

Should be set to "false" (usage of relative timestamps) if absolute timestamps are not explicitly needed.

Interruptions.InterruptionsAllowed:

Specifies whether the device can be interrupted (paused) during runtime. This allows a more flexible energy management for the device. For instance the EM can interrupt a device in case of unpredictable bad weather conditions or when the user switches on a device with conflicting energy needs and restart the interrupted device afterwards.

Should be set to "true" for EV chargers.

Requests.OptionalEnergy:

Specifies whether the device supports the consumption of optional energy, i.e. energy that can be assigned to the device which is not essentially needed by the device to operate correctly. For example chargeable battery-driven devices which do not always have to be charged to 100% might support this. If this option is set to true, additional configuration options (for constraints, etc.) can be displayed in the EM's GUI.

As EV chargers in this scenario support handling of optional energies this must be set to "true".

Example:

```

<DeviceInfo>
  <Identification>...</Identification>
  <Characteristics>...</Characteristics>
  <Capabilities>
    <CurrentPower>
      <Method>Measurement</Method>
    </CurrentPower>
    <Timestamps>
      <AbsoluteTimestamps>>false</AbsoluteTimestamps>
    </Timestamps>
    <Interruptions>
      <InterruptionsAllowed>>true</InterruptionsAllowed>
    </Interruptions>
    <Requests>
      <OptionalEnergy>>true</OptionalEnergy>
    </Requests>
  </Capabilities>
</DeviceInfo>

```

2.3.2 Device status

The device status data structure contains information about the current status of the devices managed by the gateway.

Relevant fields for this scenario:

DeviceId:

Device ID of the device as specified in the device's DeviceInfo.

EMSignalsAccepted:

Specifies if recommendations from the EM (see DeviceControl) are accepted at the moment.

If "true" recommendations will be accepted. By setting this to "false" this can be used to inform the EM that the device is currently in manual mode and will not be able to follow recommendations. In this case the EM should not send DeviceControl messages to the device.

The value should be set to "false" if there is no timeframe in the planning request section for the device at the moment.

If it is set to "false" and timeframes for the device are given in a planning request section, the EM will not use the information for planning.

Status:

Should be either set to "On", "Off" or "Offline". A device being "On" means that the component controlled by this SEMP device is switched on. It is "Off" if the controlled component is switched off. This is even the case if the device itself is still operating or in standby mode. For instance a charger's status is "Off" when it is not charging - even if it is still consuming (usually only a small amount of) energy as it is still operating (e.g. because it keeps listening for a new charge request).

If the SEMP gateway is not the controllable device itself and the communication to the controllable device is lost, the status can be set to "Offline" to signal a communication timeout.

For an EV charger the status should be "on" if charging is in progress and "off" otherwise.

ErrorCode (opt.):

Can be used to notify the EM of an error state of the device. If this element is not present or a value of "0" is provided, no error is pending. Error codes are vendor specific and not interpreted by the EM.

They might be displayed in the EM GUI to notify the user of error states.

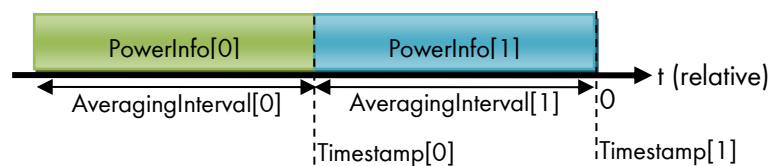
PowerConsumption.PowerInfo:

Whenever the device is switched on - no matter if recommended by the EM or on its own - it should provide the current power consumption (sum of all phases).

At least the average power, the time interval in which the power was measured and the timestamp of the end of the averaging interval have to be provided.

Additionally the range of power consumption during the interval can be given with the Min-/MaxPower elements. If available the standard deviation and skewness of the measured values can be provided by StdDevPower and SkewPower. This is helpful if multiple measurements were merged into one value.

The timestamp marks the time of the end of measurement. If relative timestamps are used (see device capabilities), this will normally be "0".



A device should provide power values gapless for the time it is switched on. If only one power value is transmitted, the averaging interval will be the last polling interval of the EM which is 60s at the moment but might be more due to latency or communication errors. If a communication timeout occurred, the device should provide the power infos determined during the timeout as soon as communication is possible again. This can be done by providing the older PowerInfo elements additionally to the current PowerInfo. Alternatively it is also possible to merge all of the power values into one PowerInfo element. Regardless of how this is done, the EM should be able to track the device's power consumption gaplessly. This is important as the information is used to learn a device's power consumption profile.

Note that an averaging interval of "0" should not be used as this would mean that the power was measured in no time which is not possible.

Power values are interpreted by the EM by taking the CurrentPower.Method capability (see DeviceInfo) into account.

Example:

```

<DeviceStatus>
  <DeviceId>F-11223344-112233445566-00</DeviceId>
  <EMSignalsAccepted>true</EMSignalsAccepted><!-- true: car connected and
ready to be controlled by EM (e.g. charging not forced) -->
  <Status>On</Status><!-- On: charging, Off: otherwise -->
  <PowerConsumption>
    <PowerInfo>
      <AveragePower>7100</AveragePower><!-- in W -->
      <!-- MinPower / MaxPower / StdDevPower / SkewPower if available -->
      <Timestamp>0</Timestamp><!-- end of averaging interval -->
      <AveragingInterval>60</AveragingInterval>
    </PowerInfo>
    <!-- older PowerInfo values if communication error occurred -->
  </PowerConsumption>
</DeviceStatus>

```

2.3.3 Planning requests

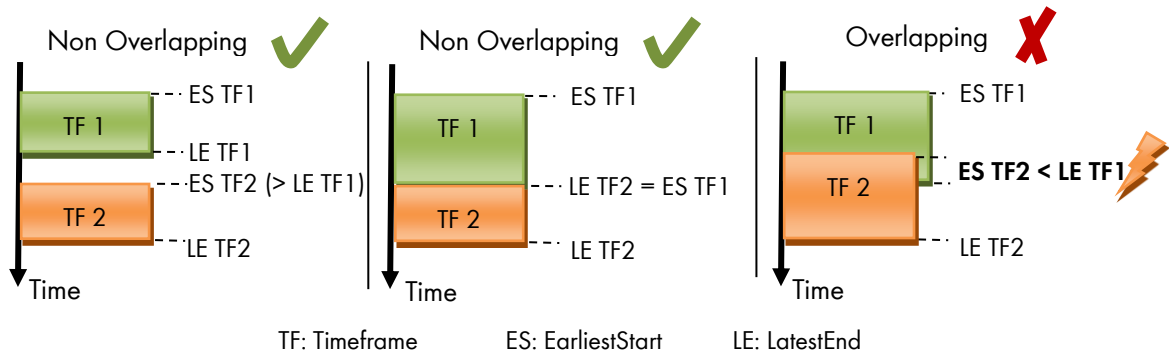
The planning request data structure is the core component of SEMP. It allows the device to specify its current energy needs. The EM uses this information to schedule devices by generating plans.

Timeframes

Each time the EM requests the PlanningRequest data structure of a device (once for each polling interval) the gateway provides a list of pending energy demands for each device managed by it. Each demand is represented by a timeframe in the planning request. A timeframe specifies the range of time in which the demand should be fulfilled. This is done by the combination of EarliestStart and LatestEnd which determines that the device operation must be finished at the latest by the specified time. This means that at the time given by LatestEnd, the minimum required runtime (MinRunningTime) or energy (MinEnergy) must have been allocated to the device by the EM.

Overlapping

Timeframes defined for one device must not overlap. Overlapping occurs if the time ranges defined by EarliestStart and LatestEnd of two timeframes of a device intersect. Overlapping is not allowed as a device cannot be switched on multiple times at one moment.

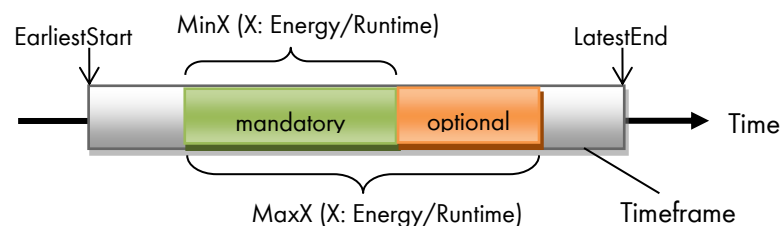


Energy and runtime demands

In addition to the time range of the device operation a timeframe specifies the amount of energy needed by the device. Either by specifying a runtime need (Min-/MaxRuntime) or energy need (Min-/MaxEnergy) - but not both. The respective MinX element (X is used as a placeholder for either "Runtime" or "Energy") defines the minimal amount of energy or runtime needed by the device in the given time range. If this is set to "0" the device does not have to operate, all of the energy defined by MaxX is optional and should only be allocated to the device if certain (user defined) constraints are fulfilled (e.g. cheap PV-energy available). If MinX equals MaxX then all of the specified energy/runtime is required. In this case the EM tries to find the optimal time for the operation of the device but it might result in grid energy use if not enough PV-energy is available. If $\text{MinX} < \text{MaxX}$ and $\text{MinX} > 0$ then MinX specifies the required part and $\text{MaxX} - \text{MinX}$ the optional part.

Notes:

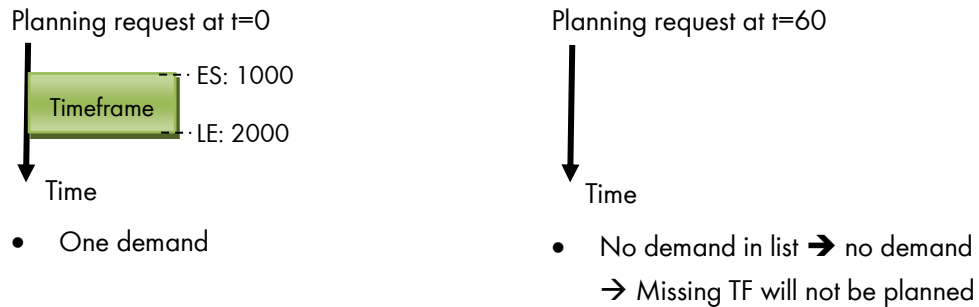
- As this scenario only covers charging with excess energy, MinEnergy and MaxEnergy should be used with MinEnergy set to 0 (optional energy demand) and $\text{MaxEnergy} > 0$. This is because in this scenario the power can be controlled by the EM. As a result the runtime depends on the recommended charging power during operation. Hence Min-/MaxEnergy should be used instead of Min-/MaxRuntime-



Updating planning requests

The planning request **must contain all current energy demands** (timeframes) for devices selected by the EM (with "<baseUrl>/" all devices managed by the gateway are selected). Timeframes defined during previous poll intervals are discarded and only the timeframes defined in the current PlanningRequest are used to generate plans for the devices. If a previously defined timeframe is missing now, it is assumed that the energy need is gone.

Example:



If a device has already received a switch on recommendation for a timeframe, the timeframe must still be listed in the planning request until it is completed (requested runtime or energy consumed). If the device was switched on by the EM and the timeframe is removed while the device is still running, the EM will recommend to switch off if signals are accepted by the device (`EMSignalsAccepted=true` in `DeviceStatus`).

If energy demands of a device change, those changes should be reflected in the planning request. In each poll interval the energy demands can (and in some cases should) be adjusted. This applies to changes in EarliestStart/LatestEnd as well as changes in the running time or energy need. This way the predictions of the expected energy need can be defined more precisely over time. Although a planning request can be provided that totally differs from the previous one, usually the updated requests should only differ slightly.

In case energy is requested (i.e. Min-/MaxEnergy is used), EarliestStart and LatestEnd are not allowed to be in the past (relative time < 0 or absolute timestamp $<$ current timestamp). This implies that Min-/MaxEnergy is always set to the remaining energy need, i.e. Min-/MaxEnergy has to be adjusted by the amount of energy that was already consumed. This in turn allows the EM to do a better tracking of the remaining energy need. There are four cases that have to be distinguished in order to adjust EarliestStart, LatestEnd and Min-/MaxEnergy. In the following examples we assume that the device has an initial energy request of at least E_{min0} and maximum E_{max0} . The amount of the initially requested energy that has already been consumed is E_{cons} . Further we assume that times are given relative to now:

1. Device has not started operation:
 - a. Device is not ready to be started by the EM:
EarliestStart > 0 , MinEnergy = E_{min0} , MaxEnergy = E_{max0}
 - b. Device is ready to be started by the EM now:
EarliestStart = 0, MinEnergy = E_{min0} , MaxEnergy = E_{max0}
2. Device has started operation:
EarliestStart = 0, MinEnergy = $\max(0, E_{min0} - E_{cons})$, MaxEnergy = $\max(0, E_{max0} - E_{cons})$

This is under the assumption that the initial energy need did not change. If there is a change in the energy need, simply set Min-/MaxEnergy according to the remaining need.

Relevant fields for this scenario:

DeviceId:	Device ID of the device as specified in the device's DeviceInfo..
EarliestStart:	Specifies the earliest start time of the device.
	<p>Note that timestamps in the past are not allowed as energy is requested in this scenario.</p> <ul style="list-style-type: none"> - Set EarliestStart=0 (if rel. time used or to current time for abs. time) if the charger is ready to start the charging process now. - Set EarliestStart =0 (if rel. time used or to current time for abs. time) when the charging process has started or still is in progress.
LatestEnd:	Specifies the end of the planning range. Either the latest possible time the device operation has to be finished.
	For EV chargers LatestEnd should be set to the time that is available for excess energy charging. Can be set to 86400 (24h).
MinEnergy:	The minimal amount needed in the specified time range.
	Should be "0" for charging with excess energy as all of the energy is optional and should only be allocated to the device if certain conditions are fulfilled (e.g. cheap PV-energy available)
MaxEnergy:	The maximal amount needed in the specified time range.
	Should be the amount of energy from current SOC to the target (e.g. full) SOC. After the charging process started, the value should be adjusted according to the energy already consumed.

Example:

```
<PlanningRequest>
  <Timeframe>
    <DeviceId>F-11223344-112233445566-00</DeviceId>
    <EarliestStart>0</EarliestStart><!-- start possible now -->
    <LatestEnd>86400</LatestEnd><!-- range: must be finished in 24h -->
    <MinEnergy>0</MinEnergy><!-- charge with excess energy -->
    <MaxEnergy>25000</MaxEnergy><!-- assumption: empty 25kWh battery -->
  </Timeframe>
  <!-- more timeframes if available -->
</PlanningRequest>
```

2.4 Recommendations from the EM

Whenever the gateway provides planning requests to the EM, the EM starts to create new plans with optimal switch-on and switch-off times for the devices. The plan itself will not be provided to the device. Instead the EM will send recommendations via the DeviceControl structure whenever the device should be switched on or off or if the recommended power consumption changes.

Recommendations (switch on/off, power) should be followed by the gateway/device if possible. Recommendations in DeviceControl messages always apply to the current time and should be followed immediately.

As an EM manages multiple devices, ignoring a recommendation might conflict with plans for other devices. If a device does not follow the recommendations and switches on when another device is already running, this might result in high energy prices.

If following a recommendation would lead to damages, inefficient behavior or customer inconvenience, the recommendation can be ignored.

Switch-on recommendation:

```
<?xml version="1.0" encoding="UTF-8"?>
<EM2Device xmlns=...>
  <DeviceControl>
    <DeviceId>F-11223344-112233445566-00</DeviceId>
    <On>true</On>
    <RecommendedPowerConsumption>7000</RecommendedPowerConsumption>
    <Timestamp>0</Timestamp>
  </DeviceControl>
</EM2Device>
```

DeviceId:	SEMP Device-ID of the target device for this control message.
On:	Value is „true“ if the device should switch on or continue
RecommendedPowerConsumption:	The recommended power consumption for the device (in W)
Timestamp:	Timestamp (relative or absolute) of the generation of the message.

When a gateway receives a switch-on recommendation it should make sure that the specified device starts to run or resumes its operation as soon as possible. Note that the timestamp only indicates the time when the message was generated, not the switch-on time.

In the case of an EV charger, the charge process should start or continue.

Switch-off recommendation:

When a gateway receives a switch-off recommendation for a device it should make sure that the specified device stops its operation as soon as possible. Note that the timestamp only indicates the time when the message was generated, not the switch-off time.

If the non-optional energy or runtime requested by an interruptible device is not met when the switch-off recommendation occurs, the switch-off recommendation is an interruption. The EM will later instruct the device to resume with a switch-on recommendation.

In the case of an EV charger, the charge process should be stopped when a switch-off recommendation is received.

```
<?xml version="1.0" encoding="UTF-8"?>
<EM2Device xmlns=...>
  <DeviceControl>
    <DeviceId>F-11223344-112233445566-00</DeviceId>
    <On>false</On>
    <Timestamp>0</Timestamp>
  </DeviceControl>
</EM2Device>
```

Power-change recommendation:

```
<?xml version="1.0" encoding="UTF-8"?>
<EM2Device xmlns=...>
  <DeviceControl>
    <DeviceId>F-11223344-112233445566-00</DeviceId>
    <On>true</On>
    <RecommendedPowerConsumption>6000</RecommendedPowerConsumption>
    <Timestamp>0</Timestamp>
  </DeviceControl>
</EM2Device>
```

In the case of an EV charger, the charge power is regulated by the EM. If the charging power should be changed the EM sends a new recommendation to the device with an adjusted `RecommendedPowerConsumption` value. The device should accept and apply the new value as soon as possible.

3 Examples:

1. SEMP-gateway responds to EM's poll request to the URL "<baseUrl>/\" with all information:

```
<?xml version="1.0" encoding="utf-8"?>
<Device2EM xmlns="http://www.sma.de/DeviceCommunication/SEMP">
  <DeviceInfo>
    <Identification>
      <DeviceId>F-11223344-002233445566-00</DeviceId>
      <DeviceName>Car Charger LS1</DeviceName>
      <DeviceType>EVCharger</DeviceType>
      <DeviceSerial>12345678</DeviceSerial>
      <DeviceVendor>Vendor XYZ</DeviceVendor>
    </Identification>
    <Characteristics>
      <!-- power of all phases -->
      <MaxPowerConsumption>11040</MaxPowerConsumption><!-- 6A, 230V, 3 phases -->
      <MinPowerConsumption>1380</MinPowerConsumption><!-- 6A bei 230V (in W) -->
      <MinOnTime>900</MinOnTime><!-- keep on for 5 minutes -->
      <MinOffTime>900</MinOffTime>
    </Characteristics>
    <Capabilities>
      <CurrentPower>
        <Method>Measurement</Method>
      </CurrentPower>
      <Timestamps>
        <AbsoluteTimestamps>>false</AbsoluteTimestamps>
      </Timestamps>
      <Interruptions>
        <InterruptionsAllowed>>true</InterruptionsAllowed>
      </Interruptions>
      <Requests><OptionalEnergy>>true</OptionalEnergy></Requests>
    </Capabilities>
  </DeviceInfo>
  <DeviceStatus>
    <DeviceId>F-11223344-002233445566-00</DeviceId>
    <EMSignalsAccepted>>true</EMSignalsAccepted>
    <Status>On</Status><!-- On: charges, Off: no charge in progress -->
    <PowerConsumption>
      <PowerInfo>
        <AveragePower>7100</AveragePower><!-- power of all phases (in W) -->
        <Timestamp>0</Timestamp>
        <AveragingInterval>60</AveragingInterval>
      </PowerInfo>
    </PowerConsumption>
  </DeviceStatus>
  <PlanningRequest>
    <Timeframe><!-- charge with excess energy -->
      <DeviceId>F-11223344-002233445566-00</DeviceId>
      <!-- possible now -->
      <EarliestStart>0</EarliestStart>
      <!-- over one day -->
      <LatestEnd>86400</LatestEnd>
      <MinEnergy>0</MinEnergy><!-- no energy mandatory -->
      <!-- max. 100% SoC (here: empty 20kWh battery) -->
      <MaxEnergy>20000</MaxEnergy>
    </Timeframe>
  </PlanningRequest>
</Device2EM>
```

2. EM sends a switch-on recommendation for a device to the SEMP gateway:

```
<?xml version="1.0" encoding="utf-8"?>
<EM2Device xmlns="http://www.sma.de/DeviceCommunication/SEMP">
  <DeviceControl>
    <DeviceId>F-11223344-002233445566-00</DeviceId>
    <On>true</On>
    <!-- recommended power for all phases (in W) -->
    <RecommendedPowerConsumption>7000</RecommendedPowerConsumption>
    <Timestamp>0</Timestamp>
  </DeviceControl>
</EM2Device>
```