



SDT extension proposal

Publication Date: 14/5/2015

Notice

Copyright © 2015 Energy@home Association. All rights reserved.

Use of the technologies described in this specification may infringe patents, copyrights or other intellectual property rights of Energy@home Members and non-members. Nothing in this specification should be construed as granting permissions to use any of the technologies described. Anyone planning to make use of technology covered by the intellectual property rights of others should first obtain permissions from the holder(s) of the rights.

Energy@home strongly encourages anyone implementing any part of these specifications to determine first whether part(s) sought to be implemented are covered by the intellectual property of others and, if so, to obtain appropriate licenses or other permission from the holder(s) of such intellectual property prior to implementation.

These specifications are subject to change without notice. Neither Energy@home nor any of its Members accept any responsibility whatsoever for damages or liability, direct or consequential, which may result from the use of this specifications.

This document is property of Energy@home. It is forbidden to reproduce this document even partially, unless unanimously authorized by the owners. Similarly, it is forbidden to divulge the information which it contains, even partially, unless unanimously authorized by the owners.

Version: 1.0

Change history

The following table shows the change history for this specification.

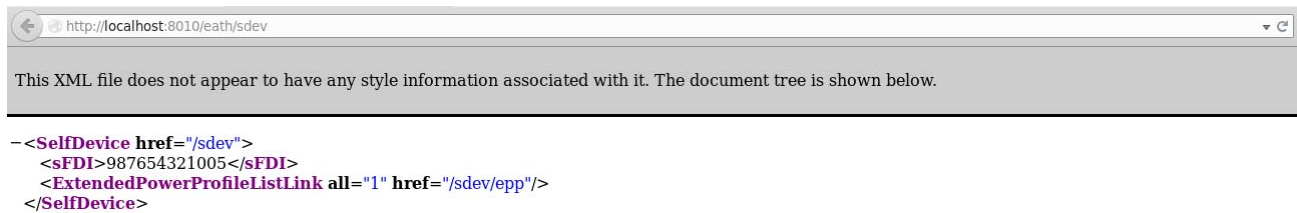
Revision	Description
1.0	First Version (Daniele Pala), Revision by Fabio Bellifemine
0.2	
0.3	
0.4	
0.5	

Table of contents

SDT EXTENSION PROPOSAL	2
ADDITIONAL DETAILS.....	4
EXAMPLE: APPLIANCE.....	5
EXAMPLE: POWER PROFILE	5

SDT extension proposal

Many of the gaps found with SDT are related to the difficulties in representing structured data. As an example of data exchange related to Power Profile, consider a client trying to get information about power profiles available on a smart device: first the client GETs the *SelfDevice* resource representation of the smart device, obtaining a response like the one shown in the following figure.



```

- <SelfDevice href="/sdev">
  <sFDI>987654321005</sFDI>
  <ExtendedPowerProfileListLink all="1" href="/sdev/epp"/>
</SelfDevice>

```

From this, the client knows that the device holds one power profile resource, located at address */sdev/pp*. If it makes an HTTP GET request to that address it gets a response like the following:

false</energyRemote> <ExtendedPowerProfile all=\"1\" results=\"1\" href=\"/sdev/epp/1\"> <PowerProfile href=\"/sdev/epp/1/1\"> <alternativeModesNumber>0</alternativeModesNumber> <duration>0</duration> <minPowerProfileDelay>0</minPowerProfileDelay> <mixEnable>false</mixEnable> <Mode all=\"1\" results=\"1\" href=\"/sdev/epp/1/1/mod\"> <EnergyPhase> <energy> <multiplier>2</multiplier> <value>5</value> </energy> <expectedDuration>100</expectedDuration> <macroPhaseID>1</macroPhaseID> <maxActivationDelay>10</maxActivationDelay> <maxOverloadPause>0</maxOverloadPause> <peakPower> <multiplier>2</multiplier> <value>1</value> </peakPower> </EnergyPhase> <repetitionNumber>0</repetitionNumber> </Mode> </PowerProfile> </ExtendedPowerProfile> <multipleScheduling>false</multipleScheduling> <totalProfileNum>20</totalProfileNum> </ExtendedPowerProfileList>" data-bbox="91 405 908 732"/>

```

- <ExtendedPowerProfileList all="1" results="1" href="/sdev/epp">
  <energyRemote>>false</energyRemote>
- <ExtendedPowerProfile all="1" results="1" href="/sdev/epp/1">
  - <PowerProfile href="/sdev/epp/1/1">
    <alternativeModesNumber>0</alternativeModesNumber>
    <duration>0</duration>
    <minPowerProfileDelay>0</minPowerProfileDelay>
    <mixEnable>false</mixEnable>
  - <Mode all="1" results="1" href="/sdev/epp/1/1/mod">
    - <EnergyPhase>
      - <energy>
        <multiplier>2</multiplier>
        <value>5</value>
      </energy>
      <expectedDuration>100</expectedDuration>
      <macroPhaseID>1</macroPhaseID>
      <maxActivationDelay>10</maxActivationDelay>
      <maxOverloadPause>0</maxOverloadPause>
    - <peakPower>
      <multiplier>2</multiplier>
      <value>1</value>
    </peakPower>
    </EnergyPhase>
    <repetitionNumber>0</repetitionNumber>
  </Mode>
</PowerProfile>
</ExtendedPowerProfile>
<multipleScheduling>false</multipleScheduling>
<totalProfileNum>20</totalProfileNum>
</ExtendedPowerProfileList>

```

This XML payload is returned by a smart device in response to an HTTP GET request. In SDT, this could be encapsulated into an Action called (for example), *GetExtendedPowerProfileList*. However, right now it is not possible to describe the complex structure that should be returned by this Action, since only simple data types are allowed.

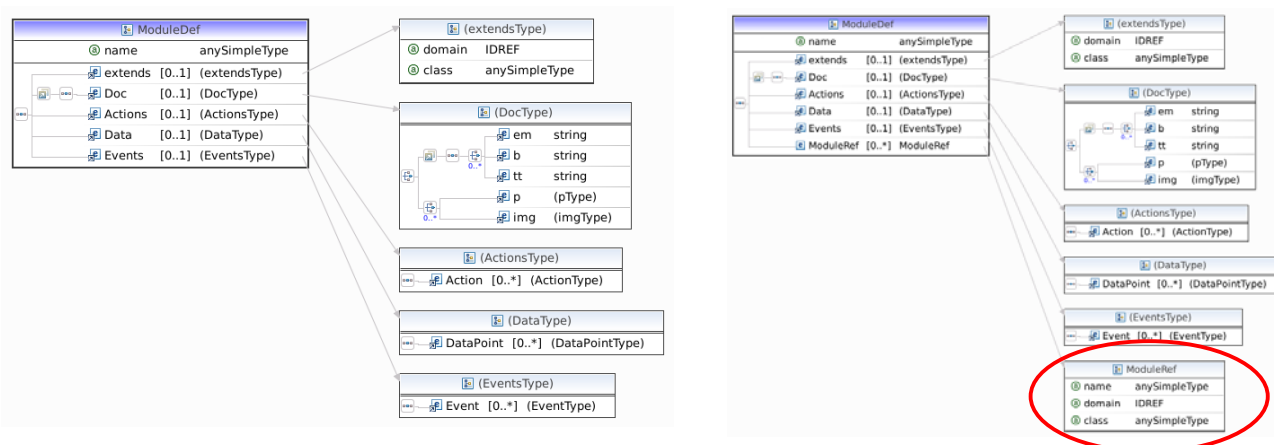
In order to overcome this limitations, the following modification of the SDT XSD are proposed:

- Allow a Module to represent more complex data structures, by having references to other modules
- Allow an Action to take complex data structures in input and return them in output.

These points are better illustrated in the following paragraphs.

Allow a Module to represent more complex data structures

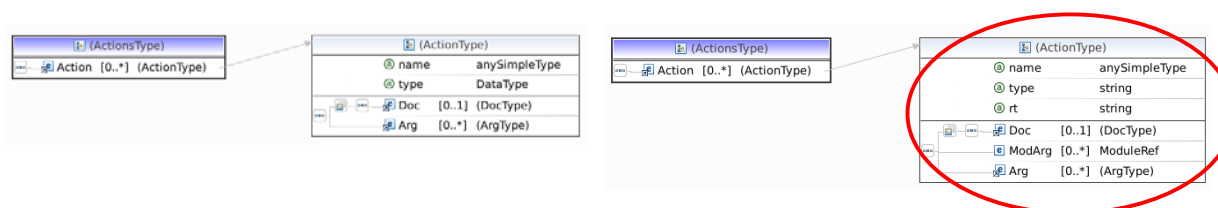
This is implemented by adding a “ModuleRef” field to ModuleDef, as illustrated in the following: on the left there is the actual SDT model, on the right the proposed extension. A “ModuleRef” represents a reference to another module.



Allow an Action to take complex data structures in input and return them in output

This is implemented by adding a “ModArg” field to an action description, of type ModuleRef. In this way an action can take both simple and complex arguments as inputs. Regarding the output of the action, we add an “rt” attribute, which is a string describing the return type of the action. Regarding the “type” attribute, we turn it into a string and we use it to describe the type of action, which is mainly useful to map HTTP methods (but may be useful in general):

- type=”safe”: a control that triggers a safe, idempotent state transition (e.g. HTTP.GET or HTTP.HEAD).
- type=”idempotent”: a control that triggers an unsafe, idempotent state transition (e.g. HTTP.PUT or HTTP.DELETE).
- type=”unsafe”: a control that triggers an unsafe, non-idempotent state transition (e.g. HTTP.POST).



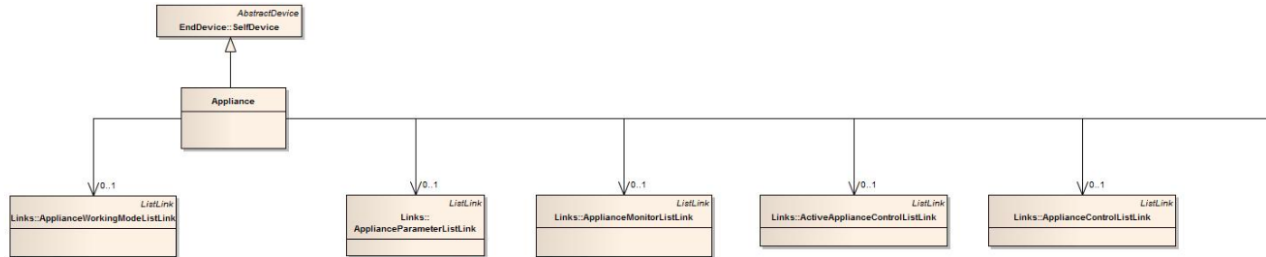
Additional details

Some concepts (like the “type” attribute) have been somewhat inspired by the ALPS format, defined at

<http://tools.ietf.org/html/draft-amundsen-richardson-foster-alps-01>

Example: Appliance

Consider the “Appliance” class defined by Energy@home:



In this (partial) diagram, it can be seen that an Appliance is a subclass of “SelfDevice”, and it has “links” to other related classes (links are used in the data model to represent state transitions, i.e. control actions). This is represented in SDT as follows:

```

-<Module name="Appliance">
  <extends class="SelfDevice" domain="domain1"/>
  -<Actions>
    <Action name="GetActiveApplianceControlList" rt="ApplianceControlList" type="safe"/>
    <Action name="GetApplianceControlList" rt="ApplianceControlList" type="safe"/>
    <Action name="GetApplianceLogEventList" rt="ApplianceLogEventList" type="safe"/>
    <Action name="GetApplianceMonitorList" rt="ApplianceMonitorList" type="safe"/>
    <Action name="GetApplianceParameterCompatibilityActionList" rt="ApplianceParameterCompatibilityActionList" type="safe"/>
    <Action name="GetApplianceParameterList" rt="ApplianceParameterList" type="safe"/>
    <Action name="GetApplianceWorkingModeList" rt="ApplianceWorkingModeList" type="safe"/>
  </Actions>
</Module>
  
```

In the above, only “GET” operations are described, but in fact also other operations would be present, like “AddApplianceControl” to add a new control. The allowed operations for each link are described in the Energy@home WADL file, which can be used to automatically generate SDT representations.

Example: Power Profile

Consider the “ExtendedPowerProfileList” and “ExtendedPowerProfile” shown at the beginning of the document. They are represented in SDT as follows:

```

-<Module name="ExtendedPowerProfile">
  <extends class="List" domain="domain1"/>
  <ModuleRef class="PowerProfile" domain="domain1" name="PowerProfile"/>
</Module>
-<Module name="ExtendedPowerProfileList">
  <extends class="List" domain="domain1"/>
  -<Data>
    <DataPoint name="energyRemote" type="boolean"/>
    <DataPoint name="multipleScheduling" type="boolean"/>
    <DataPoint name="totalProfileNum" type="integer"/>
  </Data>
  <ModuleRef class="ExtendedPowerProfile" domain="domain1" name="ExtendedPowerProfile"/>
</Module>
  
```