TemperatureSetpoint:1 Service Template
For UPnP™ Device Architecture V 1.0

Status: Standardized DCP
Date: May 13th, 2003

This Standardized DCP has been adopted as a Standardized DCP by the Steering Committee of the UPnP Forum, pursuant to Section 2.1(c)(ii) of the UPnP Membership Agreement. UPnP Forum Members have rights and licenses defined by Section 3 of the UPnP Membership Agreement to use and reproduce the Standardized DCP in UPnP Compliant Devices. All such use is subject to all of the provisions of the UPnP Membership Agreement.

THE UPnP FORUM TAKES NO POSITION AS TO WHETHER ANY INTELLECTUAL PROPERTY RIGHTS EXIST IN THE STANDARDIZED DCPS. THE STANDARDIZED DCPS ARE PROVIDED "AS IS" AND "WITH ALL FAULTS". THE UPnP FORUM MAKES NO WARRANTIES, EXPRESS, IMPLIED, STATUTORY, OR OTHERWISE WITH RESPECT TO THE STANDARDIZED DCPS INCLUDING BUT NOT LIMITED TO ALL IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT AND FITNESS FOR A PARTICULAR PURPOSE, OF REASONABLE CARE OR WORKMANLIKE EFFORT, OR RESULTS OR OF LACK OF NEGLIGENCE.

© 2001-2003 Contributing Members of the UPnP™ Forum. All Rights Reserved

<table>
<thead>
<tr>
<th>Authors</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>Larry Stickler</td>
<td>Honeywell</td>
</tr>
<tr>
<td>Pete Bergstrom</td>
<td>Honeywell</td>
</tr>
<tr>
<td>Andrew Fiddian-Green</td>
<td>Siemens Building Technologies</td>
</tr>
</tbody>
</table>
Contents

1. OVERVIEW AND SCOPE ............................................................................................................. 4
   1.1. CHANGE LOG FOR: TEMPERATURESETPOINT:1 ............................................................... 4

2. SERVICE MODELING DEFINITIONS .................................................................................. 5
   2.1. SERVICEType ................................................................................................................... 5
   2.2. STATE VARIABLES ........................................................................................................... 5
      2.2.1. Application .................................................................................................................. 6
      2.2.2. CurrentSetpoint ......................................................................................................... 6
      2.2.3. SetpointAchieved ....................................................................................................... 6
      2.2.4. Name ......................................................................................................................... 6
      2.2.5. Relationships Between State Variables ..................................................................... 6
   2.3. EVENTING AND MODERATION ................................................................................... 7
      2.3.1. Event Model .............................................................................................................. 7
   2.4. ACTIONS .......................................................................................................................... 7
      2.4.1. GetApplication ......................................................................................................... 8
      2.4.2. SetApplication ......................................................................................................... 8
      2.4.3. SetCurrentSetpoint .................................................................................................. 9
      2.4.4. GetCurrentSetpoint ................................................................................................. 10
      2.4.5. GetSetPointAchieved ............................................................................................. 10
      2.4.6. GetName .................................................................................................................. 11
      2.4.7. SetName .................................................................................................................. 11
      2.4.8. Non-Standard Actions Implemented by a UPnP Vendor ........................................ 12
      2.4.9. Relationships Between Actions ............................................................................. 12
      2.4.10. Common Error Codes ............................................................................................ 12
   2.5. THEORY OF OPERATION ............................................................................................. 12

3. XML SERVICE DESCRIPTION .......................................................................................... 14

4. TEST ....................................................................................................................................... 17

List of Tables

Table 1 State Variables ........................................................................................................... 5
Table 2 AllowedValueList for Application ............................................................................... 5
Table 3 AllowedValueRange for CurrentSetpoint ................................................................... 6
Table 4 Eventing & Moderation ............................................................................................. 7
Table 5 Event Model ............................................................................................................... 7
Table 6 Action list ..................................................................................................................... 7
Table 7 Arguments for GetApplication .................................................................................. 8
Table 8 Arguments for SetApplication ................................................................................ 8
Table 9 Arguments for SetCurrentSetpoint .......................................................................... 9
Table 10 Arguments for GetCurrentSetpoint ........................................................................ 10
Table 11 Arguments for GetSetPointAchieved .................................................................... 10

Table 12 Arguments for GetName .................................................................11
Table 13 Arguments for SetName..............................................................11
1. Overview and Scope
This service definition is compliant with the UPnP Device Architecture version 1.0.

- This service-type enables the following functions:
- The means to set and get a temperature set point for use by a temperature controller.
- The highest and lowest valid temperature values for the set point are defined by the vendor
- An interface is provided that allows notification when a controlled region has reached the temperature control band per this temperature controller’s design.
- A vendor fined application type. This allows re-use of this service for multiple applications

This service does not include:
- The closed-loop control interface that is part of a temperature controller

1.1. Change Log for: TemperatureSetpoint:1

7/26 Changes per 7/17 meeting of Home Automation and Security Working Group; conversion to 0.996 template, rename from TemperatureController

8/24/00 Clean up, renamed IsStable variable to SetpointAchieved, rewrote and expanded theory of operation section.

8/29/00 Added XML, Removed HighestValid and LowestValid.

9/28/00 Change from Centigrade to Celsius units, moved reserved application names to data type column, indicated that allowed value is not used for Application string, added Name

11/27/00 Added DualHeatingCooling application, added step to xml, added error 701

2/14/01 Updates to meet TemplateDesign Complete. Changed NewCurrent to NewCurrentSetpoint, corrected capitalization. Improved service description

2/15/01 Moved to Template 1.1, added optional SetApplication action

2/26/01 Proof read

3/2/01 Changed Application variable to allow writing. Change to 0.85

4/2/01 Fixed minor errors – improved description on 2.4.3.2; moved to 0.87

5/31/02 Revision marks deleted; Moved to 0.9; Test chapter added.

[13 May 2003] v1.0 Converted to Approved Standard.
2. **Service Modeling Definitions**

2.1. **ServiceType**

The following service type identifies a service that is compliant with this template:

`urn:schemas-upnp-org:service:TemperatureSetpoint:1`

2.2. **State Variables**

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Req. or Opt.</th>
<th>Data Type</th>
<th>Allowed Value</th>
<th>Default Value</th>
<th>Eng. Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>R</td>
<td>string</td>
<td>see table</td>
<td>(none)</td>
<td>n/a</td>
</tr>
<tr>
<td>CurrentSetpoint</td>
<td>R</td>
<td>i4</td>
<td>see table</td>
<td>(none)</td>
<td>.01 degrees Celsius</td>
</tr>
<tr>
<td>SetpointAchieved</td>
<td>O</td>
<td>boolean</td>
<td>1, 0</td>
<td>0</td>
<td>none</td>
</tr>
<tr>
<td>Name</td>
<td>O</td>
<td>string</td>
<td>Zero length string</td>
<td>N/a</td>
<td></td>
</tr>
</tbody>
</table>

Non-standard state variables implemented by an UPnP vendor go here.

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vendor defined as “none”</td>
<td>O</td>
</tr>
<tr>
<td>R/W -This allows a control point to establish the application type</td>
<td>O</td>
</tr>
<tr>
<td>Vendor defined – one value only</td>
<td>O</td>
</tr>
<tr>
<td>Reserved names are:</td>
<td></td>
</tr>
<tr>
<td>Heating,</td>
<td></td>
</tr>
<tr>
<td>Cooling,</td>
<td></td>
</tr>
<tr>
<td>DualHeatingCooling,</td>
<td></td>
</tr>
<tr>
<td>Dryer,</td>
<td></td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional, X = Non-standard.

2 Values listed in this column are required. To specify standard optional values or to delegate assignment of values to the vendor, you must reference a specific instance of an appropriate table below.
Table 3 AllowedValueRange for CurrentSetpoint

<table>
<thead>
<tr>
<th>Value</th>
<th>Req. or Opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>minimum</td>
<td>Vendor-defined</td>
</tr>
<tr>
<td>maximum</td>
<td>Vendor-defined</td>
</tr>
<tr>
<td>step</td>
<td>Step=1 (i.e. 0.01 Celsius)</td>
</tr>
</tbody>
</table>

1 R = Required, O = Optional, X = Non-standard.

2.2.1. Application
This variable states the intended application of this service.

2.2.2. CurrentSetpoint
This variable exposes the setpoint of a service that is controlling temperature to that setpoint.

2.2.3. SetpointAchieved
This variable changes from false to true when the temperature in the controlled region is within the control band. This variable changes to false when a new setpoint is set or when the temperature is no longer in the control band. The value of this variable is determined from information provided by a temperature controller—typically PID.

2.2.4. Name
This optional variable may be used to capture a friendly name or location for this sensor.

2.2.5. Relationships Between State Variables
SetpointAchieved changes from false to true when the temperature in the controlled region is within the control band determined by the CurrentSetpoint. This variable changes to false when a new CurrentSetpoint is set or when the temperature is no longer in the control band.
2.3. Eventing and Moderation

Table 4 Eventing & Moderation

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>Evented</th>
<th>Moderated Event</th>
<th>Max Event Rate&lt;sup&gt;1&lt;/sup&gt;</th>
<th>Logical Combination</th>
<th>Min Delta per Event&lt;sup&gt;2&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Yes</td>
<td>No</td>
<td>none</td>
<td>none</td>
<td>On-change</td>
</tr>
<tr>
<td>Application</td>
<td>Yes</td>
<td>No</td>
<td>none</td>
<td>none</td>
<td>n/a</td>
</tr>
<tr>
<td>CurrentSetpoint</td>
<td>Yes</td>
<td>Yes</td>
<td>none</td>
<td>none</td>
<td>On-change</td>
</tr>
<tr>
<td>SetpointAchieved</td>
<td>Yes</td>
<td>Yes</td>
<td>none</td>
<td>none</td>
<td>On-change</td>
</tr>
<tr>
<td><strong>Non-standard state variables implemented by an UPnP vendor go here.</strong></td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
<td>TBD</td>
</tr>
</tbody>
</table>

<sup>1</sup> Determined by N, where Rate = (Event)/(N secs).

<sup>2</sup> (N) * (allowedValueRange Step).

2.3.1. Event Model

Table 5 Event Model

<table>
<thead>
<tr>
<th>Variable Name</th>
<th>UI requirements</th>
<th>Async Requirements</th>
<th>Func. Vs max rate tradeoffs</th>
<th>Est of Max rate</th>
<th>Reason not evented</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>Needed for UI</td>
<td></td>
<td></td>
<td></td>
<td>N/a</td>
</tr>
<tr>
<td>CurrentTemperature</td>
<td>Needed for UI</td>
<td></td>
<td></td>
<td></td>
<td>N/a</td>
</tr>
<tr>
<td>Name</td>
<td>Needed for UI</td>
<td></td>
<td></td>
<td></td>
<td>N/a</td>
</tr>
<tr>
<td>SetpointAchieved</td>
<td>Needed for UI</td>
<td></td>
<td></td>
<td></td>
<td>N/a</td>
</tr>
</tbody>
</table>

2.4. Actions

Table 6 Action list

<table>
<thead>
<tr>
<th>Name</th>
<th>Req. or Opt.</th>
</tr>
</thead>
<tbody>
<tr>
<td>GetApplication</td>
<td>R</td>
</tr>
<tr>
<td>SetApplication</td>
<td>O</td>
</tr>
<tr>
<td>SetCurrentSetpoint</td>
<td>R</td>
</tr>
<tr>
<td>GetCurrentSetpoint</td>
<td>R</td>
</tr>
<tr>
<td>GetSetpointAchieved</td>
<td>O</td>
</tr>
</tbody>
</table>
### 2.4.1. GetApplication

Provides the Application value to a control point or other devices

#### 2.4.1.1. Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentApplication</td>
<td>Out(^R)</td>
<td>Application</td>
</tr>
</tbody>
</table>

\(^R\) Return Value

#### 2.4.1.2. Dependency on State (if any)

Depends on Application

#### 2.4.1.3. Effect on State (if any)

None

#### 2.4.1.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td>none</td>
</tr>
</tbody>
</table>

### 2.4.2. SetApplication

If the allowed value for Application is not set to a fixed value this action allows a control point to establish the value for Application

#### 2.4.2.1. Dependency on State (if any)

None
2.4.2.2. Effect on State (if any)

Changes the value of Application

2.4.2.3. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>701</td>
<td></td>
<td>This implementation of this service does not permit writing of this variable.</td>
</tr>
</tbody>
</table>

2.4.3. SetCurrentSetpoint

This action establishes a new setpoint for this service. This directs a temperature controller associated with this service to control to a new temperature

2.4.3.1. Arguments

Table 9 Arguments for SetCurrentSetpoint

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewCurrentSetpoint</td>
<td><strong>In</strong></td>
<td>CurrentSetpoint</td>
</tr>
</tbody>
</table>

2.4.3.2. Dependency on State (if any)

If this service is for cooling and a heating setpoint service is also provided, the cooling value must be less than the heating setpoint. If this service is for heating and a cooling setpoint service is also provided, the heating value must be greater than the cooling setpoint

2.4.3.3. Effect on State

Changes CurrentSetpoint to = NewCurrentSetpoint

2.4.3.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>700</td>
<td>Invalid Temperature</td>
<td>NewCurrentSetpoint is outside of the specified range</td>
</tr>
<tr>
<td>701</td>
<td>Rejected – value inconsistent with other setpoint values</td>
<td>If cooling, the value is less than the heating setpoint. If heating the value is more than the cooling setpoint</td>
</tr>
</tbody>
</table>

2.4.4. GetCurrentSetpoint

The action retrieves the current setpoint value from this service.
2.4.4.1. Arguments

Table 10 Arguments for GetCurrentSetpoint

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentSP</td>
<td>Out⁶</td>
<td>CurrentSetpoint</td>
</tr>
</tbody>
</table>

² Return Value

2.4.4.2. Dependency on State (if any)
Depends on CurrentSetpoint

2.4.4.3. Effect on State
None

2.4.4.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.4.5. GetSetPointAchieved

Provides the SetpointAchieved Value to a control point or other devices

2.4.5.1. Arguments

Table 11 Arguments for GetSetPointAchieved

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentSPA</td>
<td>Out⁶</td>
<td>SetpointAchieved</td>
</tr>
</tbody>
</table>

² Return Value

2.4.5.2. Dependency on State (if any)
Depends on whether controller has reached the setpoint and is inside the control proportional band.

2.4.5.3. Effect on State
None

2.4.5.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
2.4.6. GetName
Provides the Name value to a control point or other UPnP device

2.4.6.1. Arguments

Table 12 Arguments for GetName

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>CurrentName</td>
<td>Out</td>
<td>Name</td>
</tr>
</tbody>
</table>

\(^R\) Return Value

2.4.6.2. Dependency on State (if any)
Depends on Name

2.4.6.3. Effect on State
None

2.4.6.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.4.7. SetName
Provides a new Name value for the Name variable.

2.4.7.1. Arguments

Table 13 Arguments for SetName

<table>
<thead>
<tr>
<th>Argument</th>
<th>Direction</th>
<th>relatedStateVariable</th>
</tr>
</thead>
<tbody>
<tr>
<td>NewName</td>
<td>In</td>
<td>Name</td>
</tr>
</tbody>
</table>

2.4.7.2. Dependency on State (if any)
None

2.4.7.3. Effect on State
Changes Name
2.4.7.4. Errors

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>none</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

2.4.8. Non-Standard Actions Implemented by a UPnP Vendor

To facilitate certification, non-standard actions implemented by UPnP vendors should be included in this service template. The UPnP Device Architecture lists naming requirements for non-standard actions (see the section on Description).

2.4.9. Relationships Between Actions

None.

2.4.10. Common Error Codes

The following table lists error codes common to actions for this service type. If an action results in multiple errors, the most specific error should be returned.

Table 6: Common Error Codes

<table>
<thead>
<tr>
<th>errorCode</th>
<th>errorDescription</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>401</td>
<td>Invalid Action</td>
<td>See UPnP Device Architecture section on Control.</td>
</tr>
<tr>
<td>402</td>
<td>Invalid Args</td>
<td>See UPnP Device Architecture section on Control.</td>
</tr>
<tr>
<td>404</td>
<td>Invalid Var</td>
<td>See UPnP Device Architecture section on Control.</td>
</tr>
<tr>
<td>501</td>
<td>Action Failed</td>
<td>See UPnP Device Architecture section on Control.</td>
</tr>
<tr>
<td>701-799</td>
<td></td>
<td>Common action errors defined by the UPnP Forum working committees.</td>
</tr>
<tr>
<td>800-899</td>
<td>TBD</td>
<td><em>(Specified by UPnP vendor.)</em></td>
</tr>
</tbody>
</table>

2.5. Theory of Operation

This service exposes the variables to control and observe a temperature controller that controls the heating or cooling of a region or a space. Examples of these mechanisms are an oven control, a water heater control, a room heater control, or a thermostat for a central heating or cooling system.

To achieve closed-loop control of the temperature, these mechanisms frequently sense the current temperature of the region and compare it with a temperature setpoint and then take the appropriate action to have the current temperature be equal (within a control band) to the setpoint.

To control and observe a temperature controller this service uses the following variables:

- Application
- CurrentTemperature
- SetpointAchieved
TemperatureSetpoint allows a ControlPoint or other device to establish a new temperature setpoint.

Manufacturers shall establish the allowable range of temperatures using Maximum and Minimum allowed values. These values are observable via the XML description.

Application allows a manufacturer to designate the intended application for this service. The value is observable by a Get action. The following applications are defined:

- **Heating** – for use in a HVAC heating system. Both heating and cooling setpoints are used in some systems that both heat and cool.
- **Cooling** – for use in a HVAC cooling system. Both heating and cooling setpoints are used in some systems that both heat and cool.
- **DualHeatingCooling** – used in systems that use a single setpoint for both heating and cooling
- **Dryer** – used for a clothes dryer
- **WaterHeater** – used for a domestic water heater
- **Refrigerator** – used for the main fresh food compartment of a refrigerator
- **Freezer** – used for a standalone freezer or the frozen food compartment of a refrigerator.

A vendor also has the ability to define the allowed value of Application to be “none” if the vendor wishes to make it writable.

The optional variable, SetpointAchieved, is provided for situations where an event is to be generated when the temperature controller first observes the current temperature has entered the control band. This variable is set to “False” when a new CurrentSetpoint is set or when the temperature is outside the control band.
3. XML Service Description

```xml
<?xml version="1.0`?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <actionList>
    <action name="GetApplication">
      <argumentList>
        <argument name="CurrentApplication" direction="out" />
        <relatedStateVariable name="Application" />
      </argumentList>
    </action>
    <action name="SetApplication">
      <argumentList>
        <argument name="NewApplication" direction="in" />
        <relatedStateVariable name="Application" />
      </argumentList>
    </action>
    <action name="SetCurrentSetpoint">
      <argumentList>
        <argument name="NewCurrentSetpoint" direction="in" />
        <relatedStateVariable name="CurrentSetpoint" />
      </argumentList>
    </action>
    <action name="GetCurrentSetpoint">
      <argumentList>
        <argument name="CurrentSP" direction="out" />
        <relatedStateVariable name="CurrentSetpoint" />
      </argumentList>
    </action>
  </actionList>
</scpd>
```

The following action is optional
<action>
  <name>GetSetpointAchieved</name>
  <argumentList>
    <argument>
      <name>CurrentSPA</name>
      <direction>out</direction>
      <retval />
      <relatedStateVariable>SetpointAchieved</relatedStateVariable>
    </argument>
  </argumentList>
</action>
The following action is optional

<action>
  <name>GetName</name>
  <argumentList>
    <argument>
      <name>CurrentName</name>
      <direction>out</direction>
      <retval />
      <relatedStateVariable>Name</relatedStateVariable>
    </argument>
  </argumentList>
</action>

The following action is optional

<action>
  <name>SetName</name>
  <argumentList>
    <argument>
      <name>NewName</name>
      <direction>in</direction>
      <relatedStateVariable>Name</relatedStateVariable>
    </argument>
  </argumentList>
</action>

Declarations for other actions added by UPnP vendor (if any) go here

<serviceStateTable>
  <stateVariable sendEvents="yes">
    <name>Application</name>
    <dataType>string</dataType>
    <allowedValueList>
      <allowedValue>vendor defined</allowedValue>
      Other allowed values defined by UPnP Forum working committee (if any) go here
    </allowedValueList>
  </stateVariable>

  <stateVariable sendEvents="yes">
    <name>CurrentSetpoint</name>
    <dataType>i4</dataType>
    <allowedValueList>
      <minimum>manufacturer defined</minimum>
      <maximum>manufacturer defined</maximum>
      <step>1</step>
    </allowedValueList>
  </stateVariable>
</serviceStateTable>
<stateVariable sendEvents="yes">
  <name>SetpointAchieved</name>
  <dataType>boolean</dataType>
  <defaultValue>0</defaultValue>
  <allowedValueList>
    <allowedValue>0</allowedValue>
    <allowedValue>1</allowedValue>
  </allowedValueList>
</stateVariable>

The following state variable is optional
<stateVariable sendEvents="yes">
  <name>Name</name>
  <dataType>string</dataType>
</stateVariable>

Declarations for other state variables defined by UPnP Forum working committee (if any) go here
Declarations for other state variables added by UPnP vendor (if any) go here
4. Test

Testing of the UPnP functions Addressing, Discovery, Description, Control (Syntax) and Eventing are performed by the UPnP Test Tool v1.1 based on the following documents:

- UPnP Device Architecture v1.0
- The Service Definitions in chapter 2 of this document
- The XML Service Description in chapter 3 of this document
- The UPnP Test Tool service template test file: TemperatureSetpoint1.xml
- The UPnP Test Tool service template test file: TemperatureSetpoint1.SyntaxTests.xml

The test suite does not include tests for Control Semantics, since it is felt that such tests would not provide a higher level of interoperability.