WANDSLLinkConfig:1 Service Template Version 1.01

For UPnP™ Version 1.0
Status: Standardized DCP
Date: November 12, 2001

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™ Forum Membership Agreement. UPnP™ Forum Members have rights and licenses defined by Section 3 of the UPnP™ Forum 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™ Forum 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.

Authors	Company
Frédéric Pennerath, Gert Marynissen	Alcatel

Contents

1. OVERVIEW AND SCOPE 4

1.1. CHANGE LOG 4

2. SERVICE MODELING DEFINITIONS 6

2.1. SERVICETYPE 6
2.2. STATE VARIABLES 6
2.2.1. LinkType 8
2.2.2. LinkStatus 8
2.2.3. AutoConfig 8
2.2.4. ModulationType 8
2.2.5. DestinationAddress 8
2.2.6. ATMEncapsulation 9
2.2.7. FCSPreserved 9
2.2.8. Relationships Between State Variables 9
2.3. EVENTING AND MODERATION 9
2.3.1. Event Model 10
2.4. ACTIONS 10
2.4.1. SetDSLLinkType 11
2.4.2. GetDSLLinkInfo 11
2.4.3. GetAutoConfig 12
2.4.4. GetModulationType 12
2.4.5. SetDestinationAddress 13
2.4.6. GetDestinationAddress 13
2.4.7. SetATMEncapsulation 14
2.4.8. GetATMEncapsulation 14
2.4.9. SetFCSPreserved 15
2.4.10. GetFCSPreserved 15
2.4.11. Non-Standard Actions Implemented by a UPnP Vendor 16
2.4.12. Relationships Between Actions 16
2.4.13. Common Error Codes 16
2.5. THEORY OF OPERATION 17

3. XML SERVICE DESCRIPTION 18

4. TEST 22

List of Tables

Table 1: State Variables 6

Table *1.1:* allowedValueList for `LinkType` 7

Table *1.2:* allowedValueList for `LinkStatus` 7

Table *1.3:* allowedValueList for `ModulationType` 7

Table *1.4:* allowedValueList for `ATMEncapsulation` 8

Table 2: Event Moderation 9

Table 3: Actions 10

Table 4: Arguments for `SetDSLLinkType` 11

Table 5: Arguments for `GetDSLLinkInfo` 11

Table 6: Arguments for `GetAutoConfig` 12

Table 7: Arguments for `GetModulationType` 12

Table 8: Arguments for `SetDestinationAddress` 13

Table 9: Arguments for `GetDestinationAddress` 13

Table 10: Arguments for `SetATMEncapsulation` 14

Table 11: Arguments for `GetATMEncapsulation` 14

Table 12: Arguments for `SetFCSPreserved` 15

Table 13: Arguments for `GetFCSPreserved` 15

Table 14: Common Error Codes 16

1. Overview and Scope

This service definition is compliant with the UPnP Device Architecture version *1.0*.

This service-type models physical and link layer properties specific to a single physical connection of a Digital Subscriber Line (DSL) modem used for Internet access on an ***InternetGatewayDevice****. These properties are common across the different instances of ***WANPPPConnection*** and ***WANIPConnection*** services of the same ***WANConnectionDevice***.

The service is OPTIONAL for supporting DSL WAN interfaces. The use of this service is relevant only for DSL WAN interfaces of an ***InternetGatewayDevice***.

It is specified in:
urn:schemas-upnp-org:device:*WANConnectionDevice* in
urn:schemas-upnp-org:device:*WANDevice*,
one or more instances of which are specified under the root device
urn:schemas-upnp-org:device:*InternetGatewayDevice*

NOTE: A ***WANDevice*** also provides a ***WANCommonInterfaceConfig*** service that encapsulates Internet access properties common across all ***WANConnectionDevice*** of the same WAN interface ***WANDevice***.

1.1. Change Log

This service mainly derives from the obsolete *WANDSLInterfaceConfig* and *WANDSLConnection* services. The history of *WANDSLLinkConfig* is given below:

Changes from *WANDSLInterfaceConfig:0.1*

- Added 'Get' actions per Technical Committee recommendation to not use QueryStateVariable for reading state variables.

Changes from *WANDSLInterfaceConfig:0.2*

- ModulationType and ModulationBandwidth have been merged into ModulationType.
- Added Initializing state to LinkStatus
- Added AutoConfig field and recommendations to support auto configuration of PVC as specified by the DSL Forum.

Changes from *WANDSLLinkInterfaceConfig:0.3*

- The service is part of a ***WANConnectionDevice*** and not anymore a ***WANDevice***.

Changes from *WANDSLLinkInterfaceConfig:0.4*

- Replace VirtualChannel and VirtualPath with DestinationAddress in order to support both PVCs and SVCs.
- Change the link type value PPPoEoA in PPPoE and remove the value IpoEoA
- Few bugs fixed

Changes from *WANDSLLinkInterfaceConfig:0.5*

- Scrubbed Optional versus Required SST variables and actions.
- Scrubbed evented variables.
- Few bugs fixed

Changes from *WANDSLLinkInterfaceConfig:0.51*

- Modified names of formal parameters of actions to be different from 'Related State Variable'.
- Added error code 718.
- Removed 'retval' and empty defaultvalue tags from the XML specification.

Changes from *WANDSLLinkConfig:0.52*

---

* Refer to companion documents defined by the UPnP Internet Gateway working committee for more details on specific devices and services referenced in this document.

- Updated to service template v1.01
- Changed action set to make individual actions for ones that deal with optional variables.
- Renamed GetLinkStatus action to GetDSLLinkInfo (includes LinkType).
- Changed default value for LinkType state var from 'None' to 'Unconfigured'
- Deleted Other from allowedValueList of variables

Changes from *WANDSLLinkConfig:0.8*

- Changed error code 718 to 719 in SetDSLLinkType
- Added error code 719 to SetDestinationAddress and SetATMEncapsulation
- Removed references to ConfigureDSLLink action and replaced with appropriate text
- Changed default values to unspecified
- Changed Required versus Optional in allowedValueList tables
- Deleted Vendor Defined rows in allowedValueList tables

Changes from *WANDSLLinkConfig:0.81*

- Added XML comment tags to comments text in XML template

Changes from *WANDSLLinkConfig:0.82*

- Renamed ATMChecksum to FCSPreserved to better convey semantics of the variable
- Added error 719 for SetFCSPreserved
- Added semantic tests
- Textual clarifications in descriptions of actions and theory of operation sections.

Changes from *WANDSLLinkConfig:0.9*

- Specified the service as optional (not required as before) due to lack of a third sample implementation
- Removed all occurrences of <retval /> from the XML template

Changes from *WANDSLLinkConfig:0.99*

- Version updated to reflect 45-day review completion. No other changes to this draft.

Changes from *WANDSLLinkConfig:0.991*

- Copyright messages and document status updated.

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: *WANDSLLinkConfig:1*.

2.2. State Variables

Table 1: State Variables

Variable Name	Req. or Opt.[1]	Data Type	Allowed Value [2]	Default Value [2]	Eng. Units
LinkType	R	string	See Table 1.1	Not specified	N/A
LinkStatus	R	string	See Table 1.2	Not specified	N/A
AutoConfig	R	boolean	0,1	Undefined - Manufacturer/ Operator dependent	N/A
ModulationType	O	string	See Table 1.3	Undefined – Manufacturer/ implementation dependent	N/A
DestinationAddress	O	string	N/A	Undefined - Manufacturer/ Operator dependent	N/A
ATMEncapsulation	O	string	See Table 1.4	Undefined - Manufacturer/ Operator dependent	N/A
FCSPreserved	O	boolean	0,1	Undefined - Manufacturer/ Operator dependent	N/A
Non-standard state variables implemented by a UPnP vendor go here.	*X*	*TBD*	*TBD*	*TBD*	*TBD*

[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.

NOTE: Default values are not specified in the DCP. A vendor may however choose to provide default values for SST variables where appropriate.

Table *1.1:* allowedValueList for `LinkType`

Value	Req. or Opt. [3]
EoA	*O*
IPoA	*O*
PPPoA	*O*
PPPoE	*O*
CIP	*O*
Unconfigured	*O*

[3]

Table *1.2:* allowedValueList for `LinkStatus`

Value	Req. or Opt.
Up	*R*
Down	*R*
Initializing	*O*
Unavailable	*O*

Table *1.3:* allowedValueList for `ModulationType`

Value	Req. or Opt.
ADSL_G.dmt	*O*
ADSL_G.lite	*O*
G.shdsl	*O*
IDSL	*O*
HDSL	*O*
SDSL	*O*

VDSL	*O*

Table *1.4:* allowedValueList for `ATMEncapsulation`

Value	Req. or Opt.
LLC	*O*
VCMUX	*O*

2.2.1. `LinkType`

This variable indicates the type of DSL connection and refers to the complete stack of protocol used for this connection.

- `EoA` corresponds to RFC1483/2684-bridged "Ethernet over ATM".
- `IPoA` corresponds to RFC1483/2684-routed "IP over ATM".
- `PPPoA` corresponds to RFC2364 "PPP over ATM".
- `PPPoE` corresponds to RFC2516 "PPP over Ethernet" on top of RFC1483-bridged "Ethernet over ATM".
- `CIP` corresponds to RFC1577 "Classical IP over ATM".
- `Unconfigured` corresponds to a free, unconfigured link.

2.2.2. `LinkStatus`

This variable indicates the status of the DSL connection. It is a **read-only** variable.

2.2.3. `AutoConfig`

This variable indicates if the modem is currently using some auto configuration mechanisms for this connection. AutoConfig specified by DSL Forum is one such mechanism. This variable is **read-only**. In this case, variables such as `LinkType`, `DestinationAddress`, `ATMEncapsulation` provided by the mechanism will become read-only. Any attempt to change one of these variables should result in a failure and an error should be returned.

If a modem doesn't support such mechanisms, this variable should always be set to false (0).

2.2.4. `ModulationType`

This variable indicates the type of modulation used on the connection

2.2.5. `DestinationAddress`

This variable indicates ATM destination address. This address identifies the other end of the WAN connection. It can define either a Permanent Virtual Circuit (PVC) or a Switched Virtual Circuit (SVC) according to a standard syntax.

For a PVC, syntax is "PVC:VPI/VCI", i.e. "PVC:8/23"

For a SVC, syntax can be either

- "SVC:ATM connection name"
- "SVC:ATM address"

ATM address is a BCD number whose format can be either

- A NSAP format, itself in one of following three formats
  - DCC format
  - ICD format
  - E.164 format
- A CCITT E.164 format

2.2.6. ATMEncapsulation

This variable indicates the method used to de/encapsulate IP or Ethernet packets from/to ATM payloads according to RFC 1483.

2.2.7. FCSPreserved

This flag tells if a checksum should be added in the ATM payload. It does not refer to the checksum of one of the ATM cells or AALX packets. In case of LLC or VCMUX encapsulation, this ATM checksum is the FCS field described in RFC 1483. It is only applicable in the upstream direction. The value of this variable is required for EoA and PPPoE link types.

2.2.8. Relationships Between State Variables

The variables in the SST have no dependencies or relationship other than what is mandated by relevant DSL modem standards and protocols.

2.3. Eventing and Moderation

Table 2: Event Moderation

Variable Name	Evented	Moderated Event	Max Event Rate[1]	Logical Combination	Min Delta per Event[2]
LinkType	No	No	N/A	N/A	N/A
LinkStatus	Yes	No	N/A	N/A	N/A
AutoConfig	Yes	No	N/A	N/A	N/A
ModulationType	No	No	N/A	N/A	N/A
DestinationAddress	No	No	N/A	N/A	N/A
ATMEncapsulation	No	No	N/A	N/A	N/A
FCSPreserved	No	No	N/A	N/A	N/A
Non-standard state variables implemented by an UPnP vendor go here.	*TBD*	*TBD*	*TBD*	*TBD*	*TBD*

[1] Determined by N, where Rate = (Event)/(N secs).
[2] (N) * (allowedValueRange Step).

2.3.1. Event Model

`LinkStatus` is an eventable variable. However, if `LinkStatus` is unsupported in functionality, i.e. only possible value is Unavailable, the variable will not generate events.

The *AutoConfig* variable is evented to provide updates on its value which in turn may be effected via non-UPnP mechanisms.

More precisely, if a ***WANConnectionDevice*** device is initially not auto-configured (`AutoConfig` is set to 0), and an auto-configuration mechanism becomes available for this connection, then the modem should internally:

1. Set `AutoConfig` to 1.
2. Override the link parameters with the new values provided by the auto-configuration mechanism.
3. Set the overridden parameters as read-only while `AutoConfig` is active. That means to return an error if any configuration action such as `SetLinkType, SetDestinationAddress` or `SetATMEncapsulation` is attempted..
4. Generate a UPnP event with the new value of `AutoConfig` for control points that have subscribed to this service.

On the contrary, if the auto-configuration mechanism was initially active and subsequently becomes unavailable or inactive, the modem should internally:

1. Set `AutoConfig` to 0.
2. Set the overridden parameters as read & write variables. That means to accept configuration requests such as `SetLinkType, SetDestinationAddress` and `SetATMEncapsulation`.
3. Generate an event with the new value of `AutoConfig`.

2.4. Actions

Immediately following this table is detailed information about these actions, including short descriptions of the actions, the effects of the actions on state variables, and error codes defined by the actions.

Table 3: Actions

Name	Req. or Opt. [1]
`SetDSLLinkType`	*R*
`GetDSLLinkInfo`	*R*
`GetAutoConfig`	*R*
`GetModulationType`	*O*
`SetDestinationAddress`	*O*
`GetDestinationAddress`	*O*
`SetATMEncapsulation`	*O*
`GetATMEncapsulation`	*O*

SetFCSPreserved	O
GetFCSPreserved	O
Non-standard actions implemented by an UPnP vendor go here.	X

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

2.4.1.SetDSLLinkType

This action configures the type of DSL physical connection of the ***WANConnectionDevice*** device.

2.4.1.1. Arguments

Table 4: Arguments for SetDSLLinkType

Argument	Direction	relatedStateVariable
NewLinkType	*IN*	LinkType

2.4.1.2. Dependency on State (if any)

This action will succeed only if no auto configuration mechanism is available for this connection, that is to say, if AutoConfig is set to false (0).

2.4.1.3. Effect on State (if any)

This action changes the LinkType variable corresponding to the DSL physical connection of the ***WANConnectionDevice*** device. Note that it will also result in a change to the variable PossibleConnectionTypes in ***WANPPPConnection*** service instances in this ***WANConnectionDevice*** device as described in the companion DCP draft for ***WANConnectionDevice***.

2.4.1.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
719	ActionDisallowed WhenAutoConfigEnabled	The specified action is not permitted when auto configuration is enabled on the modem
501	Action Failed	See Table 2.4.13

2.4.2.GetDSLLinkInfo

This action retrieves the type of DSL physical connection and the status of the link of the ***WANConnectionDevice*** device.

2.4.2.1. Arguments

Table 5: Arguments for GetDSLLinkInfo

Argument	Direction	relatedStateVariable
NewLinkType	*OUT*	LinkType

Argument	Direction	relatedStateVariable
NewLinkStatus	*OUT*	LinkStatus

2.4.2.2. Dependency on State (if any)

2.4.2.3. Effect on State (if any)

2.4.2.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13

2.4.3.GetAutoConfig

This action retrieves the variable that indicates if the modem is using an auto configuration mechanism.

2.4.3.1. Arguments

Table 6: Arguments for GetAutoConfig

Argument	Direction	relatedStateVariable
NewAutoConfig	*OUT*	AutoConfig

2.4.3.2. Dependency on State (if any)

2.4.3.3. Effect on State (if any)

2.4.3.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13

2.4.4.GetModulationType

This action retrieves the type of modulation used on the connection.

2.4.4.1. Arguments

Table 7: Arguments for GetModulationType

Argument	Direction	relatedStateVariable
NewModulationType	*OUT*	ModulationType

2.4.4.2. Dependency on State (if any)

2.4.4.3. Effect on State (if any)

2.4.4.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13

2.4.5.SetDestinationAddress

This action updates the ATM destination address.

2.4.5.1. Arguments

Table 8: Arguments for SetDestinationAddress

Argument	Direction	relatedStateVariable
NewDestinationAddress	*IN*	DestinationAddress

2.4.5.2. Dependency on State (if any)

2.4.5.3. Effect on State (if any)

This action updates the ATM destination address.

2.4.5.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13
719	ActionDisallowed WhenAutoConfigEnabled	The specified action is not permitted when auto configuration is enabled on the modem

2.4.6.GetDestinationAddress

This action retrieves the ATM destination address.

2.4.6.1. Arguments

Table 9: Arguments for GetDestinationAddress

Argument	Direction	relatedStateVariable
NewDestinationAddress	*OUT*	DestinationAddress

2.4.6.2. Dependency on State (if any)

2.4.6.3. Effect on State (if any)

2.4.6.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13

2.4.7.SetATMEncapsulation

This action sets the method to de/encapsulate IP or Ethernet packets from/to ATM payloads according to RFC 1483.

2.4.7.1. Arguments

Table 10: Arguments for SetATMEncapsulation

Argument	Direction	relatedStateVariable
NewATMEncapsulation	*IN*	ATMEncapsulation

2.4.7.2. Dependency on State (if any)

2.4.7.3. Effect on State (if any)

This action sets the method to de/encapsulate IP or Ethernet packets from/to ATM payloads according to RFC 1483.

2.4.7.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13
719	ActionDisallowed WhenAutoConfigEnabled	The specified action is not permitted when auto configuration is enabled on the modem

2.4.8.GetATMEncapsulation

This action retrieves the method to de/encapsulate IP or Ethernet packets from/to ATM payloads according to RFC 1483.

2.4.8.1. Arguments

Table 11: Arguments for GetATMEncapsulation

Argument	Direction	relatedStateVariable
NewATMEncapsulation	*OUT*	ATMEncapsulation

2.4.8.2. Dependency on State (if any)

2.4.8.3. Effect on State (if any)

2.4.8.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13

2.4.9.SetFCSPreserved

This action sets/clears a flag to indicate if a checksum in the ATM payload should be added.

2.4.9.1. Arguments

Table 12: Arguments for SetFCSPreserved

Argument	Direction	relatedStateVariable
NewFCSPreserved	*IN*	FCSPreserved

2.4.9.2. Dependency on State (if any)

2.4.9.3. Effect on State (if any)

This action sets/clears the FCSPreserved flag to indicate if a checksum in the ATM payload should be added.

2.4.9.4. Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13
719	ActionDisallowed WhenAutoConfigEnabled	The specified action is not permitted when auto configuration is enabled on the modem

2.4.10. GetFCSPreserved

This action retrieves the flag value that indicates if a checksum in the ATM payload should be added.

2.4.10.1. Arguments

Table 13: Arguments for GetFCSPreserved

Argument	Direction	relatedStateVariable
NewFCSPreserved	*OUT*	FCSPreserved

2.4.10.2.Dependency on State (if any)

2.4.10.3.Effect on State (if any)

2.4.10.4.Errors

errorCode	errorDescription	Description
402	Invalid Args	See Table 2.4.13
501	Action Failed	See Table 2.4.13

2.4.11.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).

2.4.12.Relationships Between Actions

Actions defined have no specific relationship between them.

2.4.13.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 14: Common Error Codes

errorCode	errorDescription	Description
401	Invalid Action	See UPnP Device Architecture section on Control.
402	Invalid Args	One of following: not enough IN arguments, too many IN arguments, no IN argument by that name, one or more IN arguments are of the wrong data type. See UPnP Device Architecture section on Control.
404	Invalid Var	See UPnP Device Architecture section on Control.
501	Action Failed	May be returned in current state if service prevents invoking of that action. See UPnP Device Architecture section on Control.
600-699	TBD	Common action errors. Defined by UPnP Forum Technical Committee.
701-799		Common action errors defined by the UPnP Forum working committees.
800-899	*TBD*	*(Specified by UPnP vendor.)*

2.5. Theory of Operation

A ***WANConnectionDevice*** that is defined in a DSL interface device ***WANDevice*** MUST contain one and only one ***WANDSLLinkConfig***service.

A control point uses the ***WANDSLLinkConfig***service for configuration and to obtain link specific information..

In order to use already configured connections, a control point does not need to interact with this service, unless it is interested in some fields specific to DSL.

For non-automatic configuration, a control point can use configuration actions to set at least the following variables in order to get the connection working:

- `DestinationAddress`: Configuration support for this field is optional. This variable is used to get the physical connectivity with the network access provider.
- `LinkType`: this field is required to use the right stack of protocols in order to get the logical connectivity with the ISP.
- `ATMEncapsulation:` Configuration support for this field is optional. This variable value is needed when using RFC 1483, depending on the value set in the field `LinkType`. See the `LinkType` variable description to see which values correspond to RFC 1483. This value is ignored when it doesn't make sense for the chosen link type.

`LinkStatus` will be set to 'Up' if link connectivity can be established successfully using the above configuration parameters. If any of the parameters change, connectivity will be reestablished. Changes in the `LinkStatus` will be notified to control points through events.

In the case of an automatic configuration of the connection (`AutoConfig` set to 1), these variables will be automatically set. If a control point attempts to modify their values by calling configuration actions, the gateway MUST return an error.

3. XML Service Description

```
<?xml version="1.0"?>
<scpd xmlns="urn:schemas-upnp-org:service-1-0">
  <specVersion>
    <major>1</major>
    <minor>0</minor>
  </specVersion>
  <actionList>
    <action>
    <name>SetDSLLinkType</name>
      <argumentList>
        <argument>
          <name>NewLinkType</name>
          <direction>in</direction>
          <relatedStateVariable>LinkType</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetDSLLinkInfo</name>
      <argumentList>
        <argument>
          <name>NewLinkType</name>
          <direction>out</direction>
          <relatedStateVariable>LinkType</relatedStateVariable>
        </argument>
        <argument>
          <name>NewLinkStatus</name>
          <direction>out</direction>
          <relatedStateVariable>LinkStatus</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetAutoConfig</name>
      <argumentList>
        <argument>
          <name>NewAutoConfig</name>
          <direction>out</direction>
          <relatedStateVariable>AutoConfig</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
    <name>GetModulationType</name>
      <argumentList>
        <argument>
          <name>NewModulationType</name>
          <direction>out</direction>
          <relatedStateVariable>ModulationType</relatedStateVariable>
        </argument>
      </argumentList>
    </action>
    <action>
```

```
<name>SetDestinationAddress</name>
  <argumentList>
    <argument>
      <name>NewDestinationAddress</name>
      <direction>in</direction>
     <relatedStateVariable>DestinationAddress</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
<name>GetDestinationAddress</name>
  <argumentList>
    <argument>
      <name>NewDestinationAddress</name>
      <direction>out</direction>
     <relatedStateVariable>DestinationAddress</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
<name>SetATMEncapsulation</name>
  <argumentList>
    <argument>
      <name>NewATMEncapsulation</name>
      <direction>in</direction>
      <relatedStateVariable>ATMEncapsulation</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
<name>GetATMEncapsulation</name>
  <argumentList>
    <argument>
      <name>NewATMEncapsulation</name>
      <direction>out</direction>
      <relatedStateVariable>ATMEncapsulation</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
<name>SetFCSPreserved</name>
  <argumentList>
    <argument>
      <name>NewFCSPreserved</name>
      <direction>in</direction>
      <relatedStateVariable>FCSPreserved</relatedStateVariable>
    </argument>
  </argumentList>
</action>
<action>
<name>GetFCSPreserved</name>
  <argumentList>
    <argument>
      <name>NewFCSPreserved</name>
      <direction>out</direction>
      <relatedStateVariable>FCSPreserved</relatedStateVariable>
```

```
        </argument>
      </argumentList>
    </action>
    <!-- Declarations for other actions added by UPnP vendor (if any) go
here -->
  </actionList>
    <serviceStateTable>
        <stateVariable sendEvents="no">
            <name>LinkType</name>
            <dataType>string</dataType>
            <allowedValueList>
                <allowedValue>EoA</allowedValue>
                <allowedValue>IPoA</allowedValue>
                <allowedValue>CIP</allowedValue>
                <allowedValue>PPPoA</allowedValue>
                <allowedValue>PPPoE</allowedValue>
                <allowedValue>Unconfigured</allowedValue>
            </allowedValueList>
        </stateVariable>
        <stateVariable sendEvents="yes">
            <name>LinkStatus</name>
            <dataType>string</dataType>
            <allowedValueList>
                <allowedValue>Up</allowedValue>
                <allowedValue>Down</allowedValue>
                <allowedValue>Initializing</allowedValue>
                <allowedValue>Unavailable</allowedValue>
            </allowedValueList>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>ModulationType</name>
            <dataType>string</dataType>
            <allowedValueList>
                <allowedValue>ADSL G.lite</allowedValue>
                <allowedValue>G.shdsl</allowedValue>
                <allowedValue>IDSL</allowedValue>
                <allowedValue>HDSL</allowedValue>
                <allowedValue>SDSL</allowedValue>
                <allowedValue>VDSL</allowedValue>
            </allowedValueList>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>DestinationAddress</name>
            <dataType>string</dataType>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>ATMEncapsulation</name>
            <dataType>string</dataType>
            <allowedValueList>
                <allowedValue>LLC</allowedValue>
                <allowedValue>VCMUX</allowedValue>
            </allowedValueList>
        </stateVariable>
        <stateVariable sendEvents="no">
            <name>FCSPreserved</name>
            <dataType>boolean</dataType>
```

```
            </stateVariable>
            <stateVariable sendEvents="yes">
                  <name>AutoConfig</name>
                  <dataType>boolean</dataType>

            </stateVariable>
    <!-- Declarations for other state variables added by UPnP vendor (if
any) go here -->
  </serviceStateTable>
</scpd>
```

4. Test

SetDSLLinkType / GetDSLLinkInfo

Test Sequence 1: To test success path
Semantic class: 2
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is NOT supported** by the DSL modem.

SetDSLLinkType Success = 200

In-Arg	Values	State Variables	Current State	Expected State
LinkType	A string representing the LinkType as defined in table 1.1 and supported by the modem implementation			
Out-Arg	**Expected Value**			
		Error Code (if any)	NA	NA

GetDSLLinkInfo Success = 200

In-Arg	Values	State Variables	Current State	Expected State
Out-Arg	**Expected Value**			
LinkType	Value set in previous set action			
LinkStatus	NA			
		Error Code (if any)	NA	NA

Test Sequence 2: To test error 719
Semantic class: 3
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is supported** by the DSL modem.

SetDSLLinkType Success = 200

In-Arg	Values	State Variables	Current State	Expected State
LinkType	A string representing the LinkType as defined in table 1.1 and supported by the modem implementation			
Out-Arg	Expected Value			
		Error Code (if any)	719	NA

SetDestinationAddress / GetDestinationAddress

Test Sequence 3: To test success path
Semantic class: 1
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is NOT supported** by the DSL modem. Also, this test may have dependencies on DSLAM provisioning.

SetDestinationAddress Success = 200

In-Arg	Values	State Variables	Current State	Expected State
DestinationAddress	A string representing a valid ATM destination address			
Out-Arg	Expected Value			
		Error Code (if any)	NA	NA

GetDestinationAddress Success = 200

In-Arg	Values	State Variables	Current State	Expected State
Out-Arg	Expected Value			
DestinationAddress	Value set in previous set action			

		Error Code (if any)	NA	NA

Test Sequence 4: To test error 719
Semantic class: 3
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is supported** by the DSL modem.

SetDestinationAddress Success = 200

In-Arg	Values	State Variables	Current State	Expected State
DestinationAddress	A string representing a valid ATM destination address			
Out-Arg	Expected Value			
		Error Code (if any)	719	NA

SetATMEncapsulation / GetATMEncapsulation

Test Sequence 5: To test success path
Semantic class: 1
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is NOT supported** by the DSL modem. Also, this test may have dependencies on DSLAM provisioning.

SetATMEncapsulation Success = 200

In-Arg	Values	State Variables	Current State	Expected State
ATMEncapsulation	A string representing valid ATM encapsulation/decapsulation method			
Out-Arg	Expected Value			
		Error Code (if any)	NA	NA

GetATMEncapsulation Success = 200

In-Arg	Values	State Variables	Current State	Expected State

Out-Arg	Expected Value			
ATMEncapsulation	Value set in previous set action			
		Error Code (if any)	NA	NA

Test Sequence 6: To test error 719
Semantic class: 3
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is supported** by the DSL modem.

SetATMEncapsulation Success = 200

In-Arg	Values	State Variables	Current State	Expected State
ATMEncapsulation	A string representing valid ATM encapsulation/decapsulation method			
Out-Arg	Expected Value			
		Error Code (if any)	719	NA

SetFCSPreserved / GetFCSPreserved

Test Sequence 7: To test success path
Semantic class: 1
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is NOT supported** by the DSL modem. Also, this test may have dependencies on DSLAM provisioning.

SetFCSPreserved Success = 200

In-Arg	Values	State Variables	Current State	Expected State
FCSPreserved	A valid flag indicating if checksum should be added to ATM payload			
Out-Arg	Expected Value			

		Error Code (if any)	NA	NA

GetFCSPreserved Success = 200

In-Arg	Values	State Variables	Current State	Expected State
Out-Arg	Expected Value			
FCSPreserved	Value set in previous set action			
		Error Code (if any)	NA	NA

Test Sequence 8: To test error 719
Semantic class: 3
Pre-conditions: None.
NOTE: Following test is only applicable when `AutoConfig` **is supported** by the DSL modem.

SetFCSPreserved Success = 200

In-Arg	Values	State Variables	Current State	Expected State
FCSPreserved	A valid flag indicating if checksum should be added to ATM payload			
Out-Arg	Expected Value			
		Error Code (if any)	719	NA

Change History

Change Log for Version 1.0 (10-4-00)

- Revised the Title Page to call out V1.0 of the Service Template
- Changed to be consistent with Sample Designs released to the Technical Committee
- Service State Table: Variable Descriptions removed from the table and are listed in specific sections following the table.
- Actions: Reformatted the information contained in the Action Table:
  - Added overview entry point.
  - Added an Action Summary Table to specify Required or Optional
  - Added enumerated sections to specify each actions: Arguments, Effect on State, and Errors.