RFC 68 - Issues on UPnP Device Service Specification

Confidential, Draft

10 Pages

Abstract

This RFC addresses items that are missing or need clarification in the UPnP Device Service Specification.

0 Document Information

0.1 Table of Contents

0.2 Status

This document specifies a number of improvements in the R3 UPnP recommendation for the OSGi Alliance, and requests discussion and suggestions for improvements. Distribution of this document is unlimited within OSGi.

It has been updated based upon comments from the Munich meeting. 2003.08.05/06

All Page Within This Box

0.3 Acknowledgement

0.4 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in

`Source code is shown in this typeface.`

0.5 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial	Jul 24 2003	Initial creation Jae Shin Lee, jaeleee@samsung.com
Review	July 31 2003	Peter.Kriens@aQute.se
Update	Sep 03 2003	jaeleee@samsung.com
Update	Dec 31.2003	jaeleee@samsung.com

1 Introduction

OSGi UPnP Service Specification was defined in OSGi SPR3. It defines the following:

- How an OSGi bundle can implement a service that is exported to the network via the UPnP protocols.
- How to find and control services that are available on the local network

However, UPnP Device Service in its current form makes it difficult to achieve compatibility with UPnP devices.

Therefore, this RFC addresses the missing links related to UPnP Device Service Specification. These issues are being tracked as Issue #223 and #214, which are UPnPStateVariable change and UPnPException.

All Page Within This Box

2 Problem Description

2.1 UPnPStateVariable Change.

Subject **how to create change events**

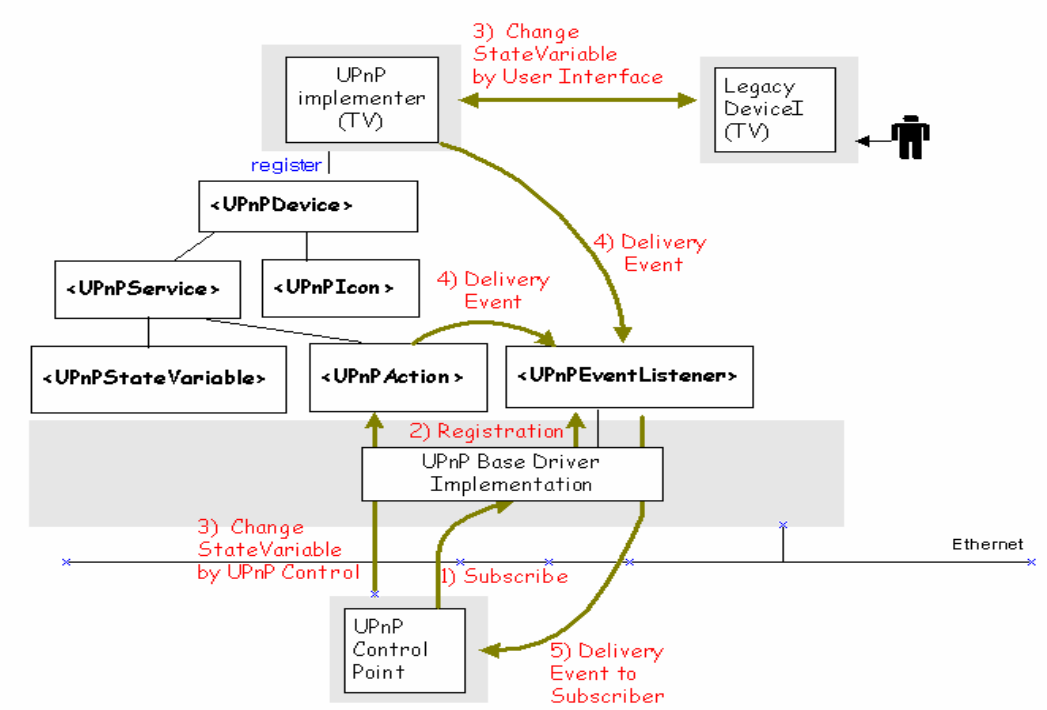
The R3 UPnP recommendation lacks a good description of how changes in the actual physical device are forwarded to the UPnP network.

This issue is about missing point of UPnP event mechanism. The figure below shows detailed behavior of UPnP Base Driver when it receives a subscribe message from a control point.

The scenario below is general situation of event handling that is conducted by UPnP Base Driver.

BACKGROUND ASSUMPTION:

- The UPnP Implementer is an emulator of Legacy TV Device.
- When a user changes a channel of TV device, this event will be forwarded to the UPnP implementer by a vendor specific protocol.
- UPnP Base Driver will register the UPnP Implementer as an OSGi UPnP Device Service and export it as a typical UPnP Device to the local network.



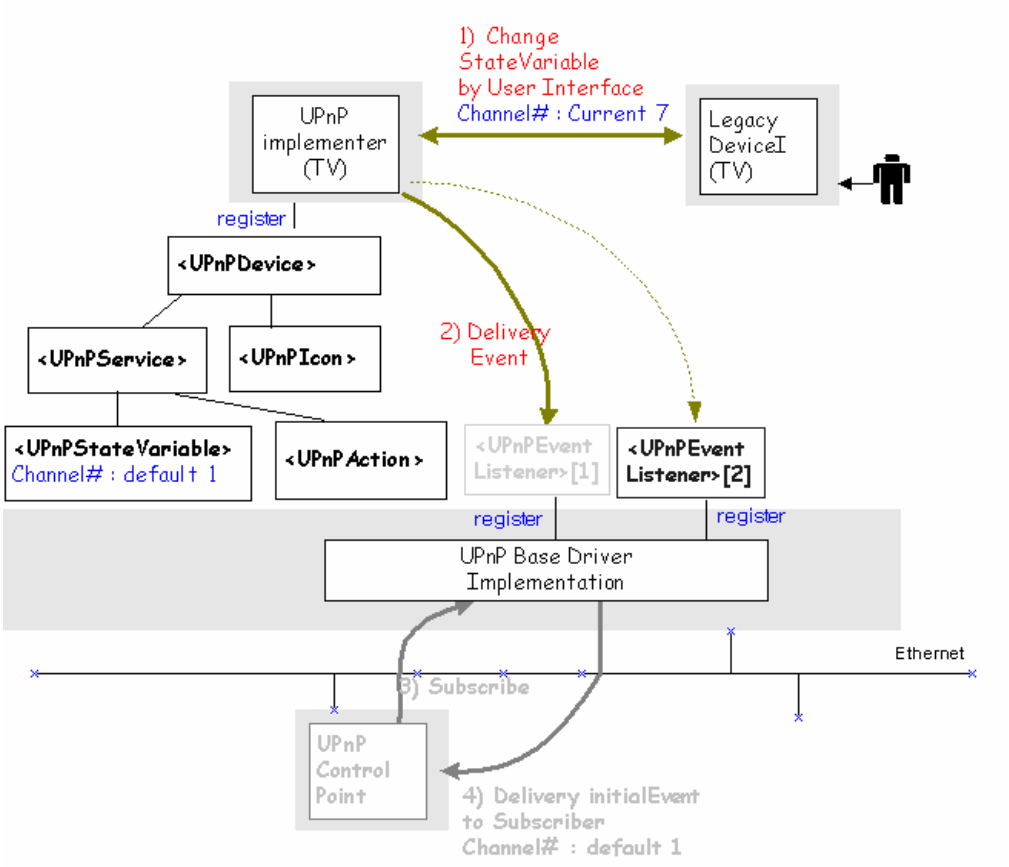
All Page Within This Box

ACTION FLOW :

1) When a control point connects in the UPnP Network, It will collect the information of UPnP devices and send control messages to manage the devices or send subscribe messages to receive events from devices to know value of their statevariable.
2) If UPnP Base Driver receives a subscribe message from the control point, it will register a UPnPEventListener service to receive event messages from the device which the control point is interested in.
3, 4) Whenever the control point or a user changes the value of statevariable of UPnP Devices, UPnPEventListener's callback method, notifyUPnPEvent, will be called.
5) Then UPnP Base Driver will change the local event into a notify message formatted in the UPnP protocol and forward it to the control point.

The OSGi UPnP Service Specification Version 1.0 does not describe detailed explanation about implementation of UPnP Device Implementer. In Section 25.7, "Implementing a UPnP Device", the detailed way of implementation is not fully explained. Also, it does not mention the way of using UPnPEventListener service.

The next scenario shows a problem related to the above explanation. The problem happens because that although the UPnP Implementer has only one way of keeping current values by using UPnP Base Driver, it does not realize that it should use notifyUPnPEvent method of UPnPEventListener service.



All Page Within This Box

ACTION FLOW :

1) Supposed that there is no control point in the network. There is only one UPnP TV device being exported in the network by UPnP Base Driver. In this situation, assume that a user changes TV channel #1 to #7.
2) The legacy TV device will forward this event to TV Implementer. Only when TV Implementer calls it, this event will be transferred to UPnPEventListener service that is registered by UPnP Base Driver without “filter” object. Supposed TV Implementer does not call it.
3) If a control point connects in the network after 1) and 2), and then it sends a subscribe message to exported UPnP Device, the control point will get an initial event from UPnP Base Driver. However, the initial event values will not be current channel values #7 but default value #1 if the TV Implementer does not call the method correctly.

2.2 UPnPException Issue.

Requestors **hargrave@us.ibm.com**
Subject **: Missing support for conveying UPnPError information as a result of a UPnP action in the OSGi R3 UPnP Service Spec**

The Universal Plug and Play Device Architecture specification version 1.0 (http://www.upnp.org/download/UPnPDA10_20000613.htm) specifies in 3.2.2 that "If the service encounters an error while invoking the action sent by a control point, the service must send a response within 30 seconds, including expected transmission time. Out arguments must not be used to convey error information; out arguments must only be used to return data; error responses must be sent in the following format". The following specification requires that a SOAP Fault must be returned containing a required UPnPError element with an errorCode and an errorDescription.

The OSGi UPnP Service Specification Version 1.0 does not provide a way to convey the errorCode and errorDescription for a UPnPError element. In section "25.14.2.6 public Dictionary invoke (Dictionary args) throws Exception" it is specified that a java.lang.Exception is thrown by the implementation of the method to signal an error. Since a java.lang.Exception does not allow an errorCode and errorDescription to be conveyed, the implementor of the UPnP stack must provide a proprietary subclass of java.lang.Exception with an errorCode and errorDescription. The user that implements a UPnP device for exporting to the network must throw an instance of this proprietary class, and therefore his implementation becomes dependent on the vendor's particular UPnP Service implementation.

3 Requirements

- Specify how changes in state variables of the underlying implementation are transferred to the network.
- Provide a mechanism to convey error information received from a UPnP device to flow to the clients of this device.

All Page Within This Box

4 Technical Solution

4.1 UPnP State Variable

4.1.1 overview

We propose that it is necessary to define new interface called UPnPLocalStateVariable. This interface extends UPnPStateVariable interface for backward compatibility with UPnP Service Spec. Version 1.0. Current value of UPnPStateVariable can be accessible by calling getCurrentValue() method. This method will help UPnP Device Implementers to manage current values by themselves.

4.1.2 API

```
package org.osgi.service.upnp;

Public interface UPnPLocalStateVariable extends UPnPStateVariable {
   Public Object getCurrentValue();
}
```

4.2 UPnPException

4.2.1 Overview

The proposed solution is to define a new type of exception, called UPnPException.
According to the UPnP Specification, there are several defined error situations describing UPnP problems. The following table summarizes defined errorCodes and errorDescriptions while a control point invokes the action.

ErrorCode	ErrorDescriptions	Description
401	Invalid Action	No action by that name at this service.
402	Invalid Args	Could be ant of the following: not enough in args, too many in arg by that name, one or more in args are of the wrong data type.
403	Out of Sync	Out of synchronization
501	Action Failed	May be returned in current state of service prevents invoking that action.
600-699	TBD	Common action errors. Defined by UPnP Forum technical Committee.
700-799	TBD	Action-specific errors for standard actions, Defined by UPnP Forum working Committee.
800-899	TBD	Action-specific errors for non-standard actions. Defined by UPnP vendor.

The developers of UPnP Device Implementers should consider error situation while UPnP Device Implementers process an action request. In addition, if an error occurs, the related exception must be thrown so that UPnP Base Driver knows the reason of the error.

4.2.2 API

```
package org.osgi.service.upnp;

public class UPnPException extends Exception {
    int errorCode;

    public UPnPException(int errorCode, String errorDesc) {
        super(errorDesc);
        this.errorCode = errorCode;
    }

    public int getUPnPErrorCode() {
        return errorCode;
    }
}
```

```
package org.osgi.service.upnp;

import java.util.Dictionary;

public interface UPnPAction {
  String getName();
  String getReturnArgumentName();
  String[] getInputArgumentNames();
  String[] getOutputArgumentNames();
  UPnPStateVariable getStateVariable(String argumentName);
  Dictionary invoke(Dictionary args) throws Exception, UPnPException;
}
```

4.3 Implementation Example

The following example illustrates the way of use UPnPException class.

UPnP Device Service must include implementation of UPnPAction API in order to process action request.

```
public UPnPActionImpl implements UPnPAction {
public String[] getInputArgumentsNames() {// … }
public String getName() {// … }
public String[] getOutputArgumentNames() {// … }
public String getReturnArgumentsName() {//  …}
public UPnPStateVariable getStateVariable(String argumentsName) {//  ….}

public Dictionary invoke( Dictionary args ) throws Exception, UPnPException
{
   if (…)
       throw new UPnPException(errorCode, errorDesc);
       // pass exceptions related with UPnP to caller.
```

All Page Within This Box

```
    if (…)
       throw new Exception(errorDesc);
       // pass exceptions to caller.


}
}


// Inside UPnP stack, it will get the reason of Exception that is occurred by UPnP
Implementer as shown in the following example:

try {
           Dictionary result = action.invoke(args);
}
catch (UPnPException upnpe) {
   //
} catch (Exception e) {
  //
}
```

5 Considered Alternatives

None.

6 Security Considerations

None.

All Page Within This Box

7 Document Support

7.1 References

[1] UPnP Device Architecture, available at http://www.upnp.org/download/UPnPDA10_20000613.htm

[2] UPnP Forum http://www.upnp.org

[3] OSGi UPnP Service Specification Version 1.0

[4] Issue #214 - http://membercvs.osgi.org/rt/rt.cgi?serial_num=214&display=History
#223 -http://membercvs.osgi.org/rt/rt.cgi?serial_num=223&display=History

7.2 Author's Address

Name	Jae Shin Lee
Company	SAMSUNG Electronics co., LTD
Address	Apkujong Bd., 599-4, shinsa-dong, kangnam-gu, Seoul, Korea
Voice	82-1-3416-0371
e-mail	jaeleee@samsung.com

7.3 Acronyms and Abbreviations

7.4 End of Document

All Page Within This Box