Part IX

ECHONET Gateway Specifications

Revision History

Note) Version numbers except Ver.3.20 indicate Japanese editions.

• Version 1.0	March 18th	2000	released / Open to consortium members
	July	2000	Open to the public
• Version 1.01	May 23rd	2001	Open to the public
	Version 1.0 addendum & corrigendum		
	Parts 1, 8, 9, and 10 were unchanged from Version 1.0.		
• Version 2.00 Draft	August 7th	2001	Open to consortium members
	Specifications for Java-API were added to Part 4. Other parts were unchanged from Version 1.01.		
• Version 2.01	December 19th	2001	Open to consortium members
• Version 2.10 Preview	December 28th	2001	Open to consortium members
• Version 2.10 Draft	February 15th	2002	Open to consortium members
• Version 2.10	March 7th	2002	Open to consortium members
• Version2.11	April 26th	2002	Open to consortium members
・Version 3.00	June 12th	2002	Open to consortium members
・Version 3.10	December 18th	2002	Open to consortium members
・Version 3.11	March 7th	2003	Open to consortium members
・Version 3.12	May 23rd	2003	Open to consortium members
・Version 3.20 Draft	October 17th	2003	Open to consortium members

	Revised entry	Revision/addition
1	7.3	Delete

・Version 3.20	January 8th	2004	Open to consortium members.
・Version 3.21	May 26th	2004	Open to consortium members.
・Version 3.30	December 2nd	2004	Open to consortium members.
・Version 3.40 Draft	December 28th	2004	Open to consortium members.
・Version 3.40	February 3rd	2005	Open to consortium members.
・Version 3.41	May 11th	2005	Open to consortium members.
・Version 3.2	October 13th	2005	Open to the public.
・Version 3.42	October 27th	2005	Open to consortium members.

・Version 3.50 Draft August 3rd 2006 Open to consortium members.

The following Table-of-Contents entries were revised:

	Revised entry	Revision/addition
1	1.1, 1.2	Explanations about the service middleware were deleted.
2	Former Sections 1.3 and 1.5	Deleted
3	New Section 1.4	Added
4	Chapters 2 to 7	Deleted
5	Entire Part 1	Added

・Version 3.50 September 20th 2006 Open to consortium members.

・Version 3.51 Draft February 2nd 2007 Open to consortium members.

・Version 3.60 March 5th 2007 Open to consortium members.

December 11th 2007 Open to the public.

Contents

Chapter 1ECHONET Gateway Specification Outline

1.1 Basic Concept

The ECHONET principal specifications provide 1) communication protocol specifications for devices connected to equipment systems and 2) intra-device interface and processing specifications for homes, small buildings, and stores. In such environments the equipment system is rarely operated independently (i.e., without connections to other systems). Instead, it is usually connected both to an external system(s) and to an AVCC system within the building. In the ECHONET standard, devices which are located between the ECHONET system and an external system and which supervise the linked action relay are called ECHONET Gateways.

1.2 Concept of ECHONET Connections with External Systems

The basic concept of connections with external systems in ECHONET is presented below.

(1) Access to an ECHONET Device from outside the building shall be allowed only after a security check by the ECHONET Gateway.

(2) The ECHONET communication protocol does not specify a code for identification of the ECHONET Domain. To identify multiple ECHONET Domains from an external system, each application software program shall define its own identifier for each domain.

1.3 ECHONET Gateway Types

ECHONET Gateways are classified into three types based on usage. This classification is based on the information security level of the ECHONET Domain to be secured. This standard does not specify gateway specifications for each type; instead, only examples are presented for reference.

(1) External service vendor gateway type
(2) Outdoor user gateway type
(3) In-house system gateway type

(1) External service vendor gateway type

The external service vendor gateway is a gateway for a system to be used by external service vendors connected to in-home ECHONET Domains. The home dweller shall have control over information within the in-home ECHONET Domain. Public disclosure of information shall be limited to the scope specified by the dweller.

(2) Outdoor user gateway type

The outdoor user gateway is a gateway for external systems connected to in-home ECHONET Domains. However, unlike the “External service vendor gateway”, the user has control over information contained in the in-home ECHONET Domain. That is, this

gateway is tele-controlled based on the assumption that outside users are accessing the system.

(3) In-house system gateway type

The in-home gateway is a gateway for an ECHONET Domain connected to another system in the same house. Here, the user of the other system is presumed to be the same as the person with control over information in the ECHONET Domain. One example might be gateways for connection with AVCC system networks.

1.4 ECHONET Gateways Covered

This ECHONET Specification defines the specifications for ECHONET gateways that fall under the following category:

Part 1: ECHONET-UPnP Gateways

Part 1. Specifications for ECHONET-UPnP Gateways

This part defines the specifications for those ECHONET gateways which are categorized as premises system gateway type ECHONET-UPnP gateways and connect to UPnP, which is a communication protocol for audiovisual devices.

Chapter 1 Overview

1.1 Basic Concept

1.1.1 Background of the development

With the trend in recent years toward more and more networks capable of supporting connections to information devices and audiovisual devices, UPnP™ has been gaining attention as a communication protocol suitable for that purpose and efforts have been progressing to apply UPnP™ in products. Because it is likely that future household-use ECHONET devices will be provided as devices integrated with information and/or audiovisual devices or as part of systems that include information and/or audiovisual devices, means are being called for to allow coexistence with UPnP devices.

1.1.2 Purpose of the Specifications for ECHONET-UPnP Gateways

To allow ECHONET devices and audiovisual devices in a home to operate in coordination with each other, a gateway function to interconnect the ECHONET protocol with the UPnP protocol for audiovisual devices is required. These Specifications for ECHONET-UPnP Gateways were developed with the purpose of providing UPnP-specific gateway function specifications to allow realistic interconnections to be achieved which transcend the boundary between ECHONET device and audiovisual device development vendors.

1.2 Relationship between the ECHONET DCP/ECHONET-UPnP Gateway Software and Other Communication Layers

This ECHONET Specification defines the requirements for the ECHONET DCP (Device Control Protocol), which overlies the UPnP Device Architecture. The ECHONET DCP defines the UPnP commands for the ECHONET.

The ECHONET-UPnP gateway software overlies the ECHONET communication processing section and ECHONET DCP. This ECHONET Specification (Part 9) defines the specifications for the ECHONET-UPnP gateway software. Fig. 1-1 shows the relationship between the ECHONET DCP/ECHONET-UPnP gateway software (shaded parts) and other communication layers.

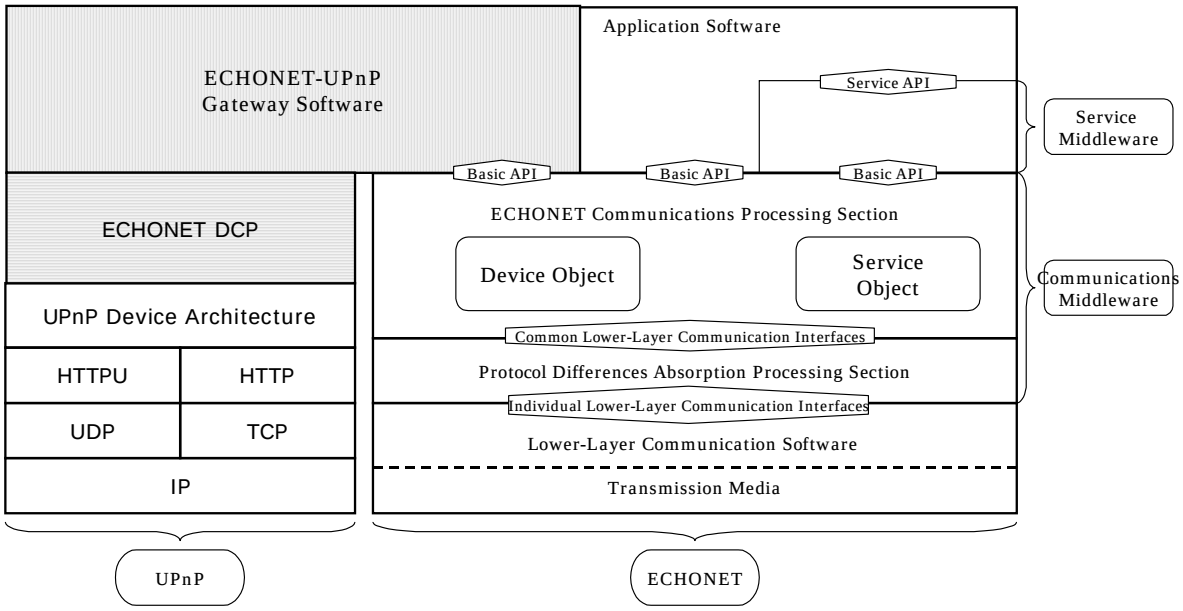

Fig. 1-1 Relationship between the ECHONET-UPnP Gateway Software and Other Communication Layers

1.3 System Structure and the Relationship between the ECHONET-UPnP Gateway and Other System Components

1.3.1 Design principles

The common specifications for the ECHONET-UPnP gateway software were developed based on the following design principles:

－The existing ECHONET Specification (excluding Part 9) should not be altered.

－The existing UPnP specifications (UPnP Device Architecture) should not be altered.

－It should be possible to accommodate existing ECHONET devices without any alteration

－It should be possible, when new ECHONET device object specifications are added, to develop corresponding UPnP specifications easily.

(The term “existing ECHONET Specification” shall refer to Version 3.41.)

Where no UPnP specification is clearly defined in this chapter, the UPnP Device Architecture specifications (including error codes) shall be followed.

1.3.2 System structure

The system structure is as shown in Fig. 1-2. This version of the Specifications for ECHONET-UPnP Gateways defines the specifications for two methods to achieve ECHONET-UPnP gateway-based communications. The first is a method in which device

objects of ECHONET devices are controlled from application software programs (UPnP control points) implemented in audiovisual devices via an ECHONET-UPnP gateway (hereinafter referred to as the “UPnP Device-based Method”), and the second is a method in which functions of audiovisual devices (UPnP devices) are controlled from ECHONET devices via an ECHONET-UPnP gateway (hereinafter referred to as the “ECHONET Object-based Method”).

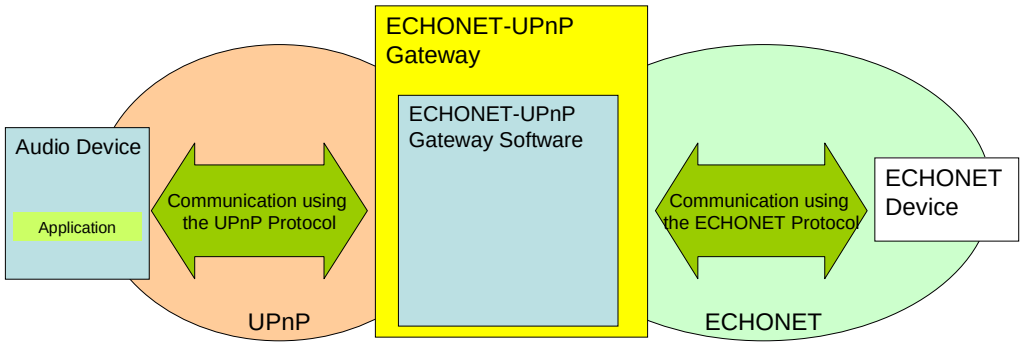

Fig. 1-2 Structure of an ECHONET-UPnP Gateway System

1.3.2.1 UPnP Device-based Method

The UPnP Device-based Method is a method in which functions to control ECHONET nodes and UPnP device functions are implemented in a gateway to publish functions of ECHONET devices to the UPnP side as UPnP devices. In this method, the gateway generates virtual UPnP devices that correspond to the target ECHONET device objects and provides functions to UPnP control points (application software) implemented on the audiovisual device side. ECHONET-UPnP gateways shall be implemented with objects in accordance with Part 2, Section 9.1 so that they can operate as ECHONET nodes. The implementation of the UPnP Device-based Method in ECHONET-UPnP gateways is compulsory.

1.3.2.2 ECHONET Object-based Method

The ECHONET Object-based Method is a method in which functions to allow control from other ECHONET nodes and UPnP control point functions are implemented in a gateway to publish functions of audiovisual devices to the ECHONET side as ECHONET objects. In this method, the gateway generates ECHONET objects corresponding to the target UPnP devices and provides functions to ECHONET nodes (application software) that provide control in the ECHONET system. The implementation of the ECHONET Object-based Method in ECHONET-UPnP gateways is optional.

1.4 Subject Matters and Scope of the Specifications for ECHONET-UPnP Gateways

1.4.1 Subject Matters of the Specifications for ECHONET-UPnP Gateways

The Specifications for ECHONET-UPnP Gateways are the specifications for techniques to allow ECHONET protocol side devices interconnected with UPnP protocol side devices to identify and control the UPnP protocol side devices and vice versa.

In principle, ECHONET-UPnP gateways developed by any vendor must be capable of achieving interconnection with ECHONET devices and with UPnP devices developed based on these specifications. The Specifications for ECHONET-UPnP Gateways provide the minimum requirements and operation sequences to achieve this.

(1) Common specifications for plug & play processing

The device/service search and detection procedures to be performed upon the entry of a device to a network and upon the exit of a device from a network for the ECHONET protocol differ from those for the UPnP protocol. The Specifications for ECHONET-UPnP Gateways are common specifications to allow plug & play processing for ECHONET devices and UPnP devices to be performed taking into account the differences between the ECHONET and UPnP protocols through the use of an ECHONET-UPnP gateway.

Specifically, the Specifications for ECHONET-UPnP Gateways include the specifications for functions to voluntarily notify, upon detection of the entry of a device in a network, the presence of the device to other networks and the specifications for PnP processing functions to respond to incoming device search packets from control terminals in other networks.

(2) Common specifications for the mapping of ECHONET device objects to UPnP DCP

The common specifications for the mapping of ECHONET device objects to UPnP DCP are specifications that apply to the UPnP Device-based Method. These are specifications for the DCP for UPnP corresponding to ECHONET device objects (properties), which are aimed at allowing ECHONET devices to be controlled from UPnP networks as UPnP devices. In the ECHONET Specification, many ECHONET device objects have already been defined. In addition, it is likely that new device objects will be added in the future. In consideration of these facts, the common specifications for the mapping of ECHONET device objects to UPnP DCP define the rules for formulaic mappings of ECHONET device object definitions to UPnP DCP definitions. By defining such mapping rules, it becomes possible to express existing device objects using a formulaic description method.

Specifically, control of ECHONET devices from UPnP control points is achieved by treating ECHONET devices as virtual UPnP devices at an ECHONET-UPnP gateway. To treat ECHONET devices as virtual UPnP devices, device/service description documents for ECHONET devices are required. The common specifications for the mapping of

ECHONET device objects to UPnP DCP define the mapping rules to generate, in a formulaic way, XML descriptions of device/service descriptions of UPnP from ECHONET property codes and service codes and the selection of the ECHONET device data necessary to generate these. The mapping rules were developed after making the necessary organization relating to the classification of the existing ECHONET device objects necessary for the development of such mapping rules. In addition, the specifications for various processing functions to convert device control commands sent from actual UPnP control points into ECHONET commands at an ECHONET-UPnP gateway and send them to the relevant devices are provided.

(3) Common specifications for ECHONET device objects that pass data to UPnP devices
The common specifications for ECHONET device objects that pass data to UPnP devices are specifications that apply to the ECHONET Object-based Method. The common specifications define the requirements for the objects necessary to publish functions of audiovisual devices to the ECHONET side as ECHONET device objects. Provision of DCP requirements for the UPnP device side is outside the scope of the ECHONET Specification.

1.4.2 Scope of the Specifications for ECHONET-UPnP Gateways

The scope of the Specifications for ECHONET-UPnP Gateways is as shown in Fig. 1-3. The core of this version of the Specifications for ECHONET-UPnP Gateways is the rules (mapping rules) for generating the data necessary to publish ECHONET device objects (as defined in the current ECHONET Specification) in UPnP side networks as UPnP devices. The components of the Specifications for ECHONET-UPnP Gateways are as follows:

- Mapping rules (Chapter 3 and succeeding chapters).
- Additional descriptions necessary for device objects and examples of application to current ECHONET device objects (Appendix 2).
- Operation sequences for each method.

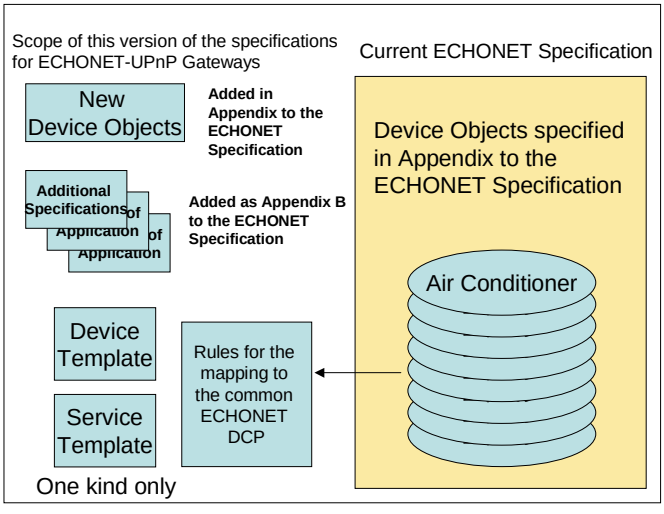

Fig. 1-3 Scope of the Specifications for ECHONET-UPnP Gateways

Chapter 2 Definitions of Terms

- Appendix

An appendix that provides detailed specifications for the properties of the ECHONET device objects.

- Appendix 2

An appendix that provides specifications for the properties of the ECHONET device objects as UPnP devices.

- Appliance name

The name of an ECHONET device object published in a UPnP network.

- ECHONET Object-based Method

A method to publish functions of audiovisual devices to the ECHONET side using an ECHONET-UPnP gateway.

- Virtual UPnP device

An ECHONET object converted through mapping into a UPnP device.

- UPnP Device-based Method

A method to publish functions of ECHONET devices to the UPnP side using an ECHONET-UPnP gateway.

- Property type

The type of the ECHONET property as determined on the basis of a classification based on the property content.

- ECHONET DCP

The UPnP Device Control Protocol, which define the UPnP commands for ECHONET devices.

Chapter 3 UPnP Device-based Method

3.1 Basic Concept

This chapter specifies the requirements for ECHONET-UPnP gateways for the case where they are used as UPnP Device-based Method gateways. Fig. 3-1 shows an example system configuration for the UPnP Device-based Method.

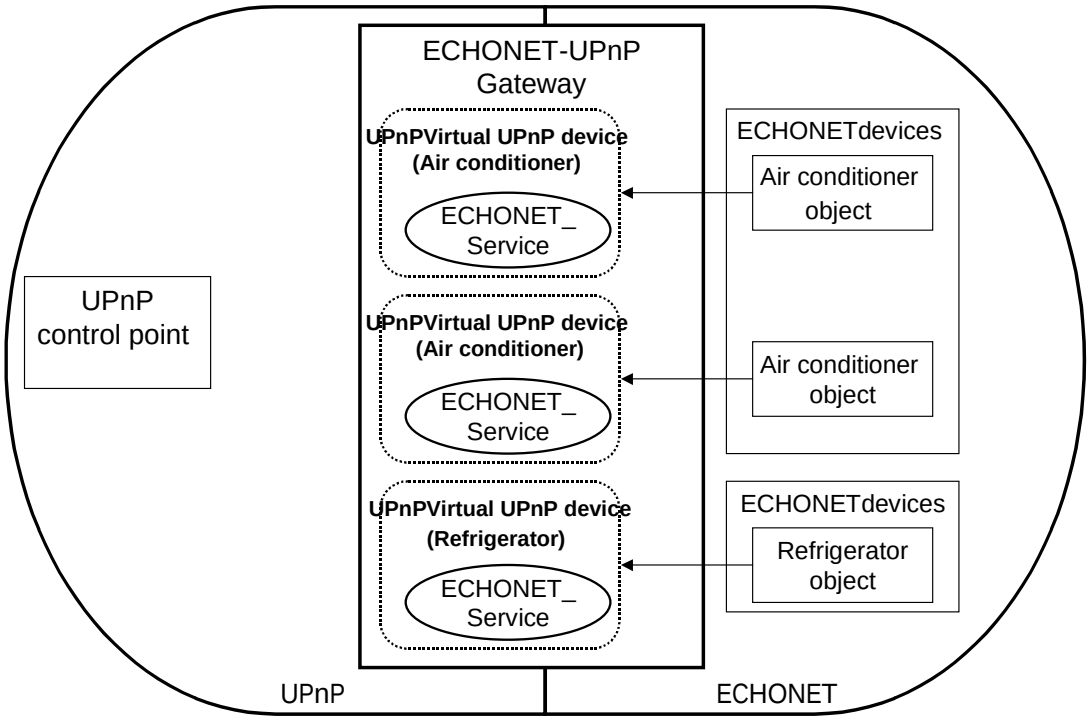

Fig. 3-1 Example System Configuration for the UPnP Device-based Method

The ECHONET-UPnP gateway provides services of ECHONET devices in a UPnP network as virtual UPnP devices. In the case of an ECHONET device equipped with two or more ECHONET objects, the ECHONET-UPnP gateway provides each of the ECHONET objects as a separate virtual UPnP device (See Fig. 3-1).

Each ECHONET device can be identified with the combination of the ECHONET address (EA) and ECHONET object code (EOJ). Each virtual UPnP device can be identified with the unique device UUID value assigned by the ECHONET-UPnP gateway. Therefore, the ECHONET-UPnP gateway can establish a one-to-one association between each ECHONET device and the corresponding virtual UPnP device by means of mappings between combinations of ECHONET addresses and ECHONET object code values used in the ECHONET and UPnP device UUID values.

XML Descriptions to publish device data and service data of virtual UPnP devices are created according to certain rules. These rules are specified for each ECHONET property type. Even when new device specifications are added, ECHONET properties can be mapped into XML Descriptions in a formulaic way simply by classifying the ECHONET properties into types.

The requirements for device object names, ECHONET property names and ECHONET data names used for publication to the UPnP side were newly defined in Appendix B to the ECHONET Specification.

3.2 Classification of ECHONET Properties

This section describes the method to map ECHONET properties into XML Service Descriptions in a formulaic way at the time of publication in a UPnP network of services an ECHONET device can perform. To achieve such mappings, it is necessary to define the following:

・Constituent elements of XML Service Descriptions
・dataType
・“VariableName” naming rules
・“Action” naming rules

With regard to the “VariableName” and “Action” naming rules, mappings can be achieved by classifying ECHONET properties by property type as described in Section 3.2.1. With regard to the “constituent elements” and “dataType,” mappings can be achieved by classifying ECHONET properties by data type as described in Section 3.2.2. Therefore, these Specifications for ECHONET-UPnP Gateways use the following 2 classification methods:

・ECHONET property type-based classification
・Data type-based classification

With regard to arrays, each of the array elements is regarded as an independent property and the classification rules are applied accordingly.

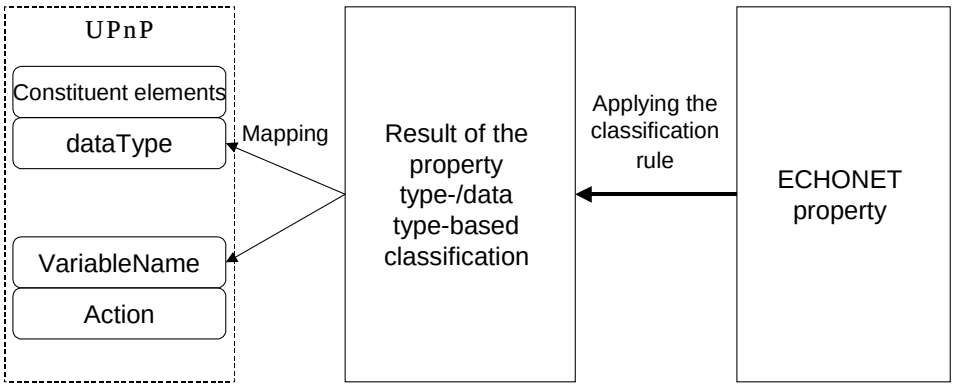

Fig. 3-2 Conceptual Diagram of ECHONET Property Classification

(1) ECHONET property type-based classification

The “ECHONET property type-based classification” method classifies ECHONET properties into property types based on their contents and defines the property type on an ECHONET property by ECHONET property basis. The “VariableName” and “Action” naming rules are specified for each property type. By storing application templates for each

of the defined property types, an application program that controls devices can automatically construct the necessary application program when a new device is added.

(2) Data type-based classification

The “data type-based classification” method classifies ECHONET properties by ECHONET property data type and defines the dataType and constituent elements of the XML Service Description based on the classification result.

When a new ECHONET device object is added, the ECHONET properties must be classified according to the classification rules specified below as well as with the existing ECHONET property definitions.

3.2.1 ECHONET property type-based classification

The “ECHONET property type-based classification” method is a classification method that classifies ECHONET properties based on their contents taking into account the possibility that an application may be constructed at a UPnP control point. In this method, ECHONET properties are classified into the 10 “property types” described below according to the classification rules described below. Explanations about the individual property types follow:

(1) Numerical value type

A “numerical value type” property is a property whose content is a numerical value. The Data Type differs depending on whether or not the value is an integer (whether or not the value has decimal places), whether the value is signed or unsigned and the byte size of the data.

(2) Date type

A “date type” property is a property whose content is data that indicates a date. The Data Type is “Date.” The data format for describing date type properties in XML Descriptions shall be yyyy-mm-dd (year－month－day), which is the format specified by ISO8601.

(3) Time type

A “time type” property is a property whose content is the time of the day or a time length. The Data Type is “Time.” The data format for describing time type properties in XML Descriptions shall be hh-mm-ss (hour－minute－second), which is the format specified by ISO8601.

(4) Level type

A “level type” property is a property which uses “non-numerical value” data that indicates a relative size or relative strength to control an ECHONET device or reference the statuses of ECHONET devices. The Data Type is “String.” An example is the “detection threshold

level" property.

(5) Character type

A "character type" property is a property which acquires character string data from ECHONET devices or controls character string data in ECHONET devices. Examples include the "product code" and "production number" properties. The Data Type is "String."

(6) Reset type

A "reset type" property is a property used to perform resets for a certain state by controlling an ECHONET device using a single defined value. The Data Type is "String."

(7) Switch type

A "switch type" property is a property which controls an ECHONET device by switching between two values or acquires a value out of a pair of values held by an ECHONET device. Examples include "operation status" and "fault status." The Data Type is "String.

(8) Selection type

A "selection type" property is a property which controls an ECHONET device by selecting a value from 3 or more values or acquires a value out of a set of 3 or more values held by an ECHONET device. An example is the "installation location" property. The Data Type is "String."

(9) Composite type

A "composite type" property is a property which has 2 or more settings and/or pieces of data with each of them contained in the specified byte(s) or bit(s). Examples include the "rated power consumption" property, which assigns 2 bytes to each of the property values and the "implemented air purification method" property, which assigns 1 bit to each of the property values. Composite type properties are made by combining types selected out of the types described in "(1) Numerical value type" to "(8) Selection type" above.

(10) 'Others' type

An "'others' type" property is a property that does not fall under any of the types described in "(1) Numerical value type" to "(9) Composite type" above. The ECHONET property value of such a property shall not be converted into a character string and the binary value defined in the ECHONET shall be published in the UPnP network. The Data Type is "bin.hex."

Table 3-1 shows a list of the property types.

Table 3-1 List of the Property Types

Property type	Explanation
Numerical value type	Used for numerical value-related control/referencing.
Date type	Used for date-related control/referencing.
Time type	Used for time-related control/referencing.
Level type	Used for level-related control/referencing.
Character type	Uses character string-based representation. There is no EDT option.
Reset type	There is 1 EDT option. The access rule is Set only.
Switch type	There are 2 EDT options.
Selection type	There are 3 or more EDT options.
Composite type	Has 2 or more settings and/or pieces of data. Each of the settings/pieces of data is contained in the specified byte(s) or bit(s).
'Others' type	Properties that do not fall under any of the types listed above.

ECHONET properties must be classified into property types in a formulaic way according to the classification rules described below referring to the list of ECHONET properties shown in Appendix to the ECHONET Specification. There are 10 classification rules, namely, Rule 1 to Rule 10. These must be applied in ascending order starting with Rule 1.

First, the properties shall be selected which have 2 or more settings and/or pieces of data with each of them contained in the specified byte(s) or bit(s) (Rule 1). The ECHONET properties selected shall be defined as composite type properties. However, properties which have more than 10 settings and/or pieces of data shall be defined as 'others' type properties. Each of the settings and pieces of data contained in a composite type property shall be classified into the applicable one of the "numerical value" to "selection" types described below.

Next, the properties shall be selected for which a unit is specified in the table shown in Appendix (Rule 2). The ECHONET properties selected shall be defined as numerical value type properties.

Next, the properties shall be selected for which a date format is specified in the "Contents of property" column of the table shown in Appendix (Rule 3). The ECHONET properties selected shall be defined as date type properties.

Next, the properties shall be selected for which a time format is specified in the "Contents of property" column of the table shown in Appendix (Rule 4). The ECHONET properties selected shall be defined as time type properties.

Next, the properties shall be selected whose content is "non-numerical value" data that indicates a relative size or relative strength (Rule 5). The ECHONET properties selected shall be defined as level type properties.

Next, the properties shall be selected whose content is treated as a numerical value even

though no unit is specified in the table shown in Appendix (Rule 6). The ECHONET properties selected shall be defined as numerical value type properties.

Next, the properties shall be selected for which only one setting option is specified (Rule 7). The ECHONET properties selected shall be defined as reset type properties.

Next, the properties shall be selected for which two setting options are specified (Rule 8). The ECHONET properties selected shall be defined as switch type properties.

Next, the properties shall be selected for which three or more setting options are specified (Rule 9). The ECHONET properties selected shall be defined as selection type properties.

Lastly, the properties shall be selected for which an explanation is provided on the property content but no setting option is specified (Rule 10). The ECHONET properties selected shall be defined as character type properties.

All properties that remain undefined after applying Rule 10 shall be defined as 'Others' type properties.

Fig. 3-3 shows a flowchart of the classification process described above.

Fig. 3-3 The Rules for Classifying Properties into Property Types

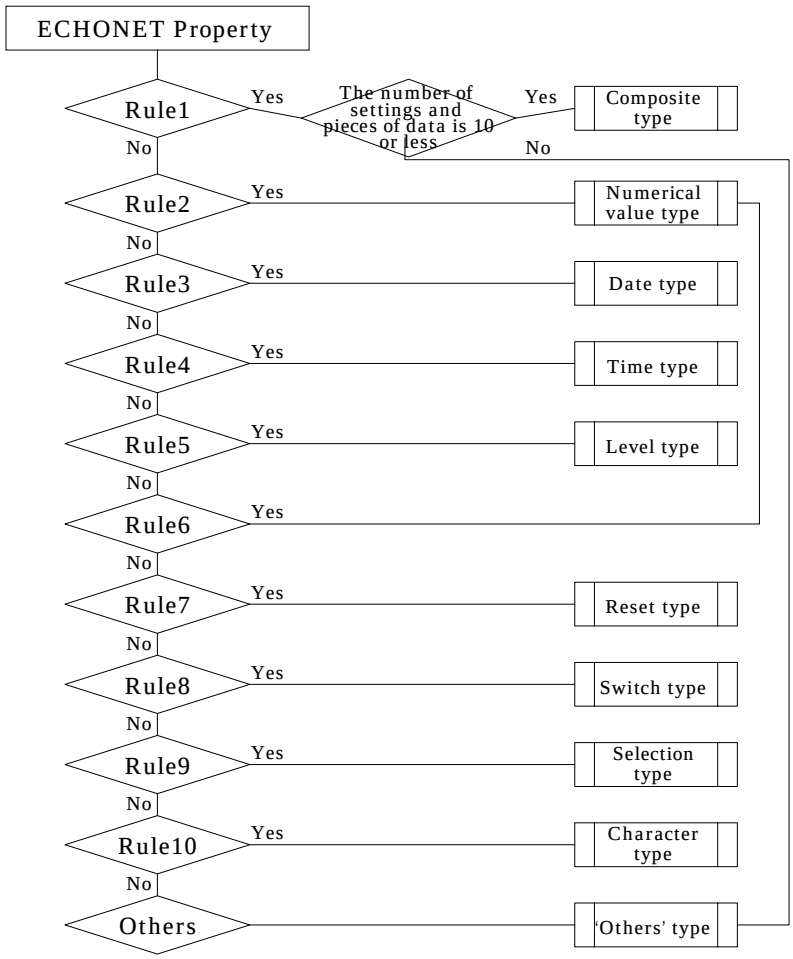

Table 3-2 summarizes the 10 classification rules.

Table 3-2 The Rules for Classifying Properties into Property Types

Rule	Relevant type	Explanation
Rule 1	Composite type	Properties which have 2 or more and 10 or less settings or pieces of data with each of them contained in the specified byte(s) or bit(s) Each of the settings and pieces of data contained in a composite type property shall be classified into the applicable property type according to Rules 2 to 10.
Rule 2	Numerical value type	Properties for which a unit is specified in the table shown in Appendix.
Rule 3	Date type	Properties for which a date format is specified in the “Contents of property” column of the table shown in Appendix.
Rule 4	Time type	Properties for which a time format is specified in the “Contents of property” column of the table shown in Appendix
Rule 5	Level type	Properties whose content is “non-numerical value” data that indicates a relative size or relative strength.
Rule 6	Numerical value type	Properties whose content is treated as a numerical value even though no unit is specified in the table shown in Appendix.
Rule 7	Reset type	Properties for which only one setting option is specified in the “Contents of property” column of the table shown in Appendix.
Rule 8	Switch type	Properties for which two setting options are specified in the “Contents of property” column of the table shown in Appendix.
Rule 9	Selection type	Properties for which three or more setting options are specified in the “Contents of property” column of the table shown in Appendix.
Rule 10	Character type	Properties for which an explanation is provided on the property content but no setting option is specified in the “Contents of property” column of the table shown in Appendix.

3.2.2 Data type-based classification

The “data type-based classification” method is a classification method that classifies ECHONET properties by data type taking into account descriptions in XML Service Descriptions. In this method, ECHONET properties are classified into the 6 “data types” described below according to the classification rules described below. The dataType and constituent elements of the XML Service Description are determined based on the classification result.

(1) AVR type

An “AVR type” property is an ECHONET property whose content as specified in the

"Contents of property" column of the table shown in Appendix is a numerical value and for which a property value range is specified in the "Contents of property" column.

However, note that ECHONET device property value ranges do not necessarily match the ECHONET Specification. In addition, in communications based on this version of the ECHONET Specification, it is not possible to acquire property value range data via network. For this reason, it is permitted to describe ECHONET properties which should ordinarily be classified into the AVR type as value type properties.

When describing an AVR type property in an XML Service Description, the "allowedValueRange" element shall be described in the "serviceStateTable" element.

(2) Value type

A "value type" property is an ECHONET property whose content as specified in the "Contents of property" column of the table shown in Appendix is a numerical value and for which no property value range is specified in the "Contents of property" column.

When describing a value type property in an XML Service Description, the "allowedValueRange" and "allowedValueList" elements shall not be described in the "serviceStateTable" element.

(3) Date type

A "date type" property is an ECHONET property whose content as specified in the "Contents of property" column of the table shown in Appendix is data that indicates a date, time or time length.

When describing a date type property in an XML Service Description, the "allowedValueRange" and "allowedValueList" elements shall not be described in the "serviceStateTable" element.

(4) AVL type

An "AVL type" property is an ECHONET property whose content as specified in the "Contents of property" column of the table shown in Appendix is not a numerical value, date, time or time length and for which property value options are specified in the "Contents of property" column.

When describing an AVL type property in an XML Service Description, the "allowedValueList" element shall be described in the "serviceStateTable" element.

(5) String type

A "string type" property is an ECHONET property whose content as specified in the "Contents of property" column of the table shown in Appendix is not a numerical value, date, time or time length and for which no property value option is specified in the "Contents of property" column.

When describing a string type property in an XML Service Description, the

"allowedValueRange" and "allowedValueList" elements shall not be described in the "serviceStateTable" element.

(6) "Others" type

All ECHONET properties classified into the "others" property type shall be regarded as properties that belong to the "others" data type.

ECHONET properties must be classified in a formulaic way according to the classification rules described below referring to the list of ECHONET properties shown in Appendix.

First, the properties whose property type is "others" shall be defined as the properties that belong to the "others" data type.

Next, the ECHONET properties shall be selected whose content as specified in the "Contents of property" column of the table shown in Appendix is a numerical value. Of the properties selected, the ones for which a numerical value range is specified in the "Contents of property" column shall be defined as AllowedValueRange type (hereinafter referred to as "AVR type") properties and the ones for which no numerical value range is specified shall be defined as value type properties. However, in communications based on the current version of the ECHONET Specification, ECHONET properties for which a numerical value range is defined in the ECHONET Specification may also be published as value type properties to UPnP, because it is not possible to acquire via network value ranges in which ECHONET devices can operate.

Next, the ECHONET properties shall be selected whose content as specified in the "Contents of property" column of the table shown in Appendix is date-, time- or time length-related data. The ECHONET properties selected shall be described as date type properties.

Of the remaining ECHONET properties, the ones for which property options are specified in the "Contents of property" column of the table shown in Appendix shall be defined as AllowedValueList type (hereinafter referred to as "AVL type") properties and the ones for which no property option is specified in the "Contents of property" column shall be defined as string type properties.

Fig. 3-4 shows a flowchart of the classification process described above. For the XML Service Description formats for the individual types, refer to Chapter 6.

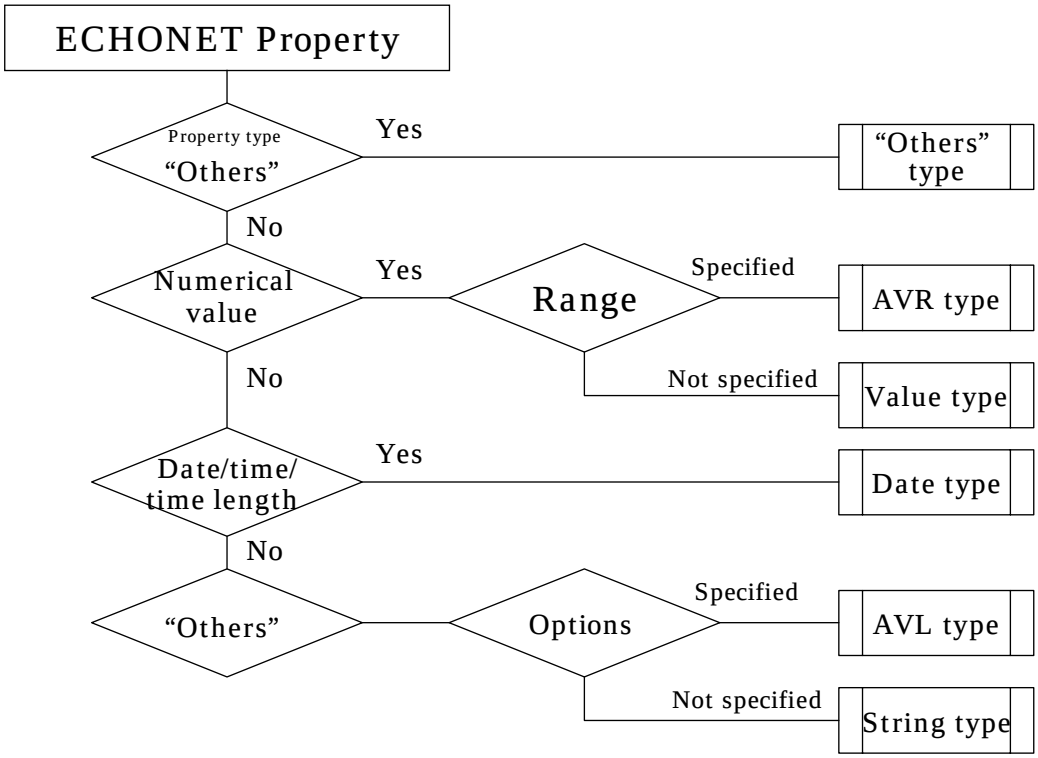

Fig. 3-4 The Rules for Classifying Properties into Data Types

3.3 Naming Rules

To publish device data and service data of an ECHONET device in a UPnP network, it is necessary to describe the ECHONET device object name, ECHONET property name and ECHONET data using a description method that is highly compatible with UPnP.

In addition, to allow easy creation of VariableNames and Action names for existing and newly added ECHONET properties, a method is necessary which achieves formulaic conversions that use names defined by the current ECHONET Specification. For these reasons, the basic rules specified below must be followed when describing data.

Firstly, the ECHONET device object names must be reflected in the Device Type names. In addition, the property type-based classification results and the ECHONET property names must be reflected in the VariableNames and Action names. ECHONET properties classified into property types as described in Section 3.2.1 must be reflected in XML Service Descriptions through the use of combinations of Action name, VariableName and dataType.

The naming rules are as specified below.

3.3.1 Naming rules necessary for XML Device Descriptions

The rules for describing services representing the ECHONET and ECHONET device names when describing data in XML Device Descriptions are as follows:

Each ECHONET device shall be described as "ECHONET_*Appliance*." The ECHONET device object name shall be entered in the *"Appliance"* section. The ECHONET device object names to be stored in the *"Appliance"* section were created based on Appendix. These are shown in Appendix 2.

The Service Type held by each ECHONET_*Appliance* shall be ECHONET_Service.

3.3.2 Naming rules based on ECHONET property type-based classification

The VariableName and Action name are defined based on the ECHONET property type-based classification result. Table 3-3 shows the naming rules for the individual ECHONET property types.

Table 3-3 Naming Rules for VariableName and Action

Property Type	Object	Action prefix
Numerical value type	Type of numerical value	Write / Read
Date type	Date	Set / Get
Time type	Time	Set / Get
Level type	Level	Set / Get
Character type	Code	Set / Get
Reset type	--	Reset
Switch type	Status	Set / Get
Selection type	Status	Set / Get
'Others' type	--	Set / Get

The principles for creating VariableNames are as follows:

- VariableNames must not be unnatural as UPnP Variable Names.
- It must be possible to create VariableNames in a formulaic way.
- VariableNames must be such that their meanings can be inferred in English.
- The number of characters must be 32 or less including the Action prefix.
- With regard to non-array type ECHONET properties, a VariableName must be created for each ECHONET property.
- With regard to array type ECHONET properties, a VariableName must be created for each array element.

On the basis of the principles listed above, the following rules were established:

First, the appropriate "object" indicating the content of the property must be selected from Table 3-3 according to the property type. Then, the VariableName must be created by decomposing the property name into words, selecting the words necessary to construct a VariableName from which the meaning can be inferred in English, combining the words in

the order they appear in the original property name and then suffixing the “object” selected from Table 3-3. The results are shown in Appendix 2.

The naming rule for Action is as follows:

Each Action name shall consists of the Action prefix followed by the VariableName.

3.3.3 Definition of the dataType based on data type-based classification

The DataType is defined based on the data type-based classification result.

Table 3-4 List of dataTypes

Data type-based Classification	dataType
AVR type	ui1, ui2, ui4, i1, i2, i4, float
Value type	ui1, ui2, ui4, i1, i2, i4, float
Date type	Time / Date
AVL type	String
String type	String
‘Others’ type	bin.hex

The dataType for an AVR/value type property shall be determined based on the ECHONET property value range and data size. The dataType (Time or Date) for a date type property shall be determined based on the content of the ECHONET property.

3.3.4 Argument naming rules

In XML Service Descriptions, parameters for actions are defined in ArgumentLists. This section defines the naming rules for Arguments, which are sub-elements of ArgumentLists. The number of characters of each Argument name must be 32 or less.

・Non-composite type ECHONET properties:

Each Argument name must consist of the word “New” (in the case where the purpose is to control devices) or “Current” (in the case where the purpose is to reference the statuses of devices) followed by “*VariableName*,” which is the relevant VariableName. For example, when the purpose is to control an ECHONET property whose VariableName is “OperationStatus,” the Argument name is “NewOperationStatus.”

・Composite type ECHONET properties:

Each Argument name must consist of the word “New” (in the case where the purpose is to control devices) or “Current” (in the case where the purpose is to reference the statuses of devices) followed by the “adjective” followed by “*VariableName*,” which is the relevant VariableName. The adjective is a word that indicates one of the items that comprise the

property. For example, in the case where the purpose is to control an ECHONET property that defines the rated power consumption values for the cooling, heating, dehumidification and air circulation modes using 2 bytes for each mode, the adjectives are “Cool,” “Heat,” “Dehumid” and “Blast,” respectively, and the corresponding Argument names are “NewCoolRatedConsumpPower, ” “NewHeatRatedConsumpPower, ” “NewDehumidRatedConsumpPower ” and “NewBlastRatedConsumpPower,” respectively.

3.4 Summary of the relationships between property types, data types, objects, Action prefixes and dataTypes

Table 3-5 summarizes the relationships between the property types, data type-based classification (explained in “3.2 Classification of ECHONET Properties), objects, Action prefixes and dataTypes (explained in “3.3 Naming Rules”).

Table 3-5 Relationships between Property types, Data types, Objects, Action prefixes and dataTypes

Property Type	Data type-based Classification	Object	Action prefix	dataType
Numerical value type	AVR / Value type	Type of numerical value	Write / Read	ui1, ui2, ui4, i1, i2, i4, float
Date type	Date type	Date	Set / Get	Date
Time type	Date type	Time	Set / Get	Time
Level type	AVL type	Level	Set / Get	String
Character type	String type	Code	Set / Get	String
Reset type	AVL type	--	Reset	String
Switch type	AVL type	Status	Set / Get	String
Selection type	AVL type	Status	Set / Get	String
‘Others’ type	‘Others’ type	--	Set / Get	bin.hex

Chapter 4 Processing for ECHONET-UPnP Gateways for the UPnP Device-based Method

This chapter specifies the requirements regarding the processing to be performed in the case where an ECHONET-UPnP gateway operates as a UPnP device;

- Plug and play processing
- Control of ECHONET devices from UPnP control points
- Notifying the statuses of ECHONET devices to UPnP control points

4.1 Plug and Play Processing

As shown in Fig. 3-1, devices in a UPnP Device-based Method system can be classified into "UPnP control points" (audiovisual devices), "ECHONET-UPnP gateways" and "ECHONET devices." The following subsections specify the requirements regarding the plug and play processing to be performed when an ECHONET-UPnP gateway connects to a network and the plug and play processing to be performed when an ECHONET device connects to a network. This ECHONET Specification does not specify requirements regarding the processing to be performed when a UPnP control point connects to a network, because there is no such processing to be performed on the ECHONET side.

The processing to be performed when an ECHONET device disconnects from a network is beyond the scope of the ECHONET Specification.

4.1.1 Plug and play processing to be performed when an ECHONET-UPnP gateway connects to a network

This subsection specifies the requirements regarding the plug and play processing to be performed when an ECHONET-UPnP gateway connects to an ECHONET network. For details of the processing to be performed when an ECHONET-UPnP gateway connects to an ECHONET network, refer to Part 2, Section 5.3 of the ECHONET Specification.

Each ECHONET-UPnP gateway connects to a UPnP network after connecting to an ECHONET network. The ECHONET-UPnP gateway-related processing to be performed when an ECHONET-UPnP gateway connects to a UPnP network and the sequence are described below. Fig. 4-1 shows the sequence.

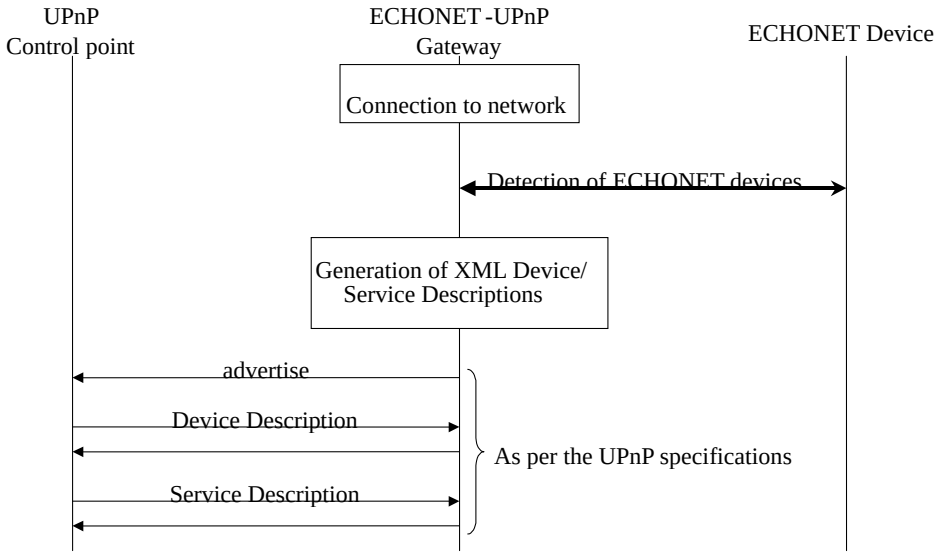

Fig. 4-1 Sequence to Be Followed when an ECHONET-UPnP Gateway Connects to a Network

When an ECHONET-UPnP gateway connects to a UPnP network, it shall detect the ECHONET devices connected to the ECHONET network and generate XML Device Descriptions and XML Service Descriptions based on the result. Or, alternatively, the ECHONET-UPnP gateway may use the XML Device Descriptions and XML Service Descriptions for the ECHONET devices stored at the time of the last startup. One way for the ECHONET-UPnP gateway to detect the connected ECHONET devices is to transmit at the time of the startup a home node instance list S property read request to the node profile class in the form of an intra-domain simultaneous broadcast in the ECHONET network and receive responses from the ECHONET devices.

The ECHONET-UPnP gateway shall multicast an “advertise” after storing the XML Device Descriptions and XML Service Descriptions for the ECHONET devices connected to the ECHONET network. The processing after the transmission of the “advertise” shall be as per the UPnP specifications.

4.1.2 Plug and play processing to be performed when an ECHONET device connects to a network

The ECHONET-UPnP gateway-related processing to be performed when an ECHONET device connects to a network and the sequence are described below. Fig. 4-2 shows the sequence.

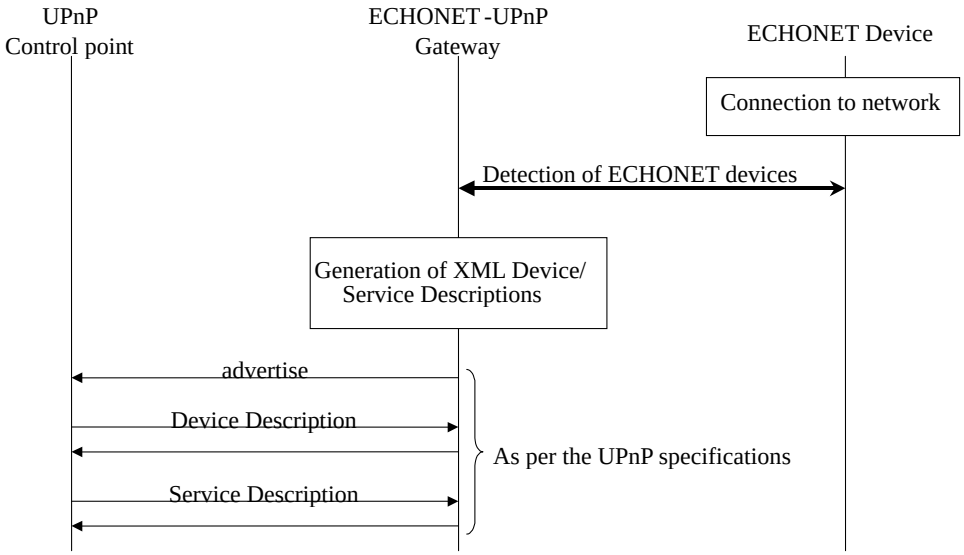

Fig. 4-2 Sequence to Be Followed when an ECHONET Device Connects to a Network

When an ECHONET-UPnP gateway detects that an ECHONET device has connected to a network, it shall generate the XML Device and Service Descriptions for the ECHONET device. Or, alternatively, the ECHONET-UPnP gateway may use the XML Device and Service Descriptions for the ECHONET device it had at the time of the last startup of the ECHONET device. One way for the ECHONET-UPnP gateway to detect that an ECHONET device has connected to a network is to receive an instance change class notification transmitted by the ECHONET device when the ECHONET device starts up and use the notification as the trigger for ECHONET device detection.

The ECHONET-UPnP gateway shall multicast an “advertise” after describing the XML Device and Service Descriptions. The processing after the transmission of the “advertise” shall be as per the UPnP specifications.

4.2 Control of ECHONET Devices from UPnP Control Points

4.2.1 Control of ECHONET devices

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an UPnP control point controls an ECHONET device via an ECHONET-UPnP gateway, as well as the sequence to be followed.

If an ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall control the ECHONET device(s) according to the synchronous or asynchronous type sequence described below

4.2.1.1 Synchronous type sequence-based control of ECHONET devices

Fig. 4-3 shows the synchronous type sequence for controlling ECHONET devices.

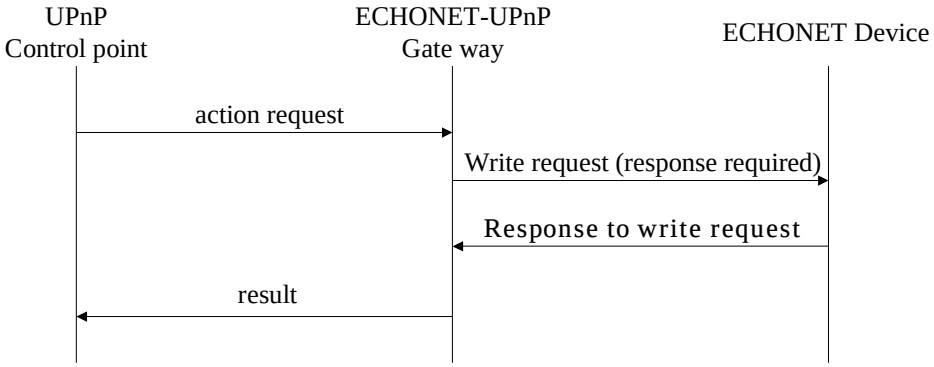

Fig. 4-3 Synchronous Type Sequence

The operation of an ECHONET-UPnP gateway in the case where the synchronous type sequence is used is as follows

If the ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall perform a conversion to the ECHONET protocol data format and transmit to the ECHONET device a write request message that requires a response. If the ECHONET-UPnP gateway subsequently receives from the ECHONET device a response to the write request, the ECHONET-UPnP gateway shall send a “result” to the UPnP control point from which the “action request” was sent.

4.2.1.2 Asynchronous type sequence-based control of ECHONET devices

Fig. 4-4 shows the asynchronous type sequence for controlling ECHONET devices.

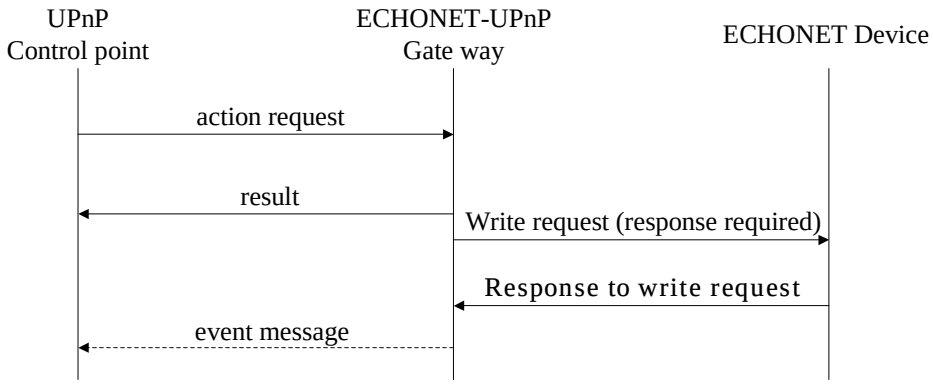

Fig. 4-4 Asynchronous Type Sequence

The operation of an ECHONET-UPnP gateway in the case where the asynchronous type sequence is used is as follows:

If the ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall send a “result” to the UPnP control point from which the “action request” was sent. At the same time, the ECHONET-UPnP gateway shall perform a conversion to the ECHONET protocol data format and transmit to the ECHONET device a write request message that requires a response. If the ECHONET-UPnP gateway subsequently receives a response from the ECHONET device to the write request and the ECHONET-UPnP gateway has accepted a “subscription request” by that point, the ECHONET-UPnP gateway shall notify the UPnP control point of the control action by sending an “event message.”

4.2.2 Referencing the statuses of ECHONET devices

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an UPnP control point references the status of an ECHONET device via an ECHONET-UPnP gateway, as well as the sequence to be followed.

If an ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall reference the status(es) of the ECHONET device(s) according to the synchronous or asynchronous type sequence described below.

4.2.2.1 Synchronous type sequence-based referencing of the statuses of ECHONET devices

Fig. 4-5 shows the synchronous type sequence for referencing the statuses of ECHONET devices.

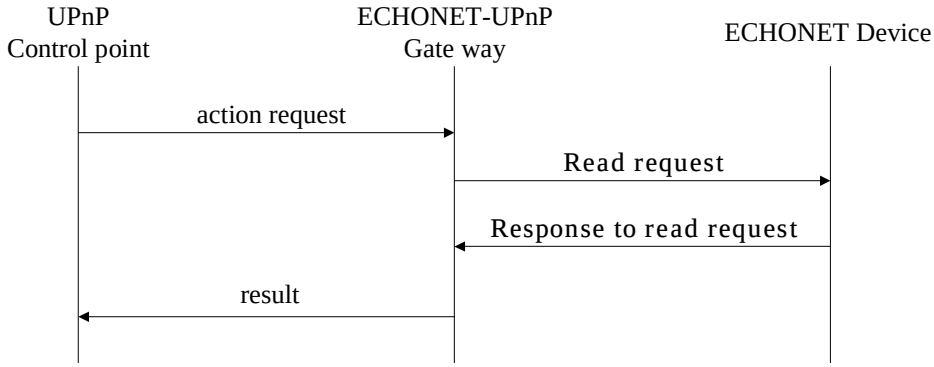

Fig. 4-5 Synchronous Type Sequence

The operation of an ECHONET-UPnP gateway in the case where the synchronous type sequence is used is as follows:

If the ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall perform a conversion to the ECHONET protocol data format and transmit to the ECHONET device a read request message that requires a response. If the ECHONET-UPnP gateway subsequently receives from the ECHONET device a response to the read request, the ECHONET-UPnP gateway shall send a “result” to the UPnP control point from which the “action request” was sent.

4.2.2.2 Asynchronous type sequence-based referencing of the statues of ECHONET devices

Fig. 4-6 shows the asynchronous type sequence for referencing the statuses of ECHONET devices.

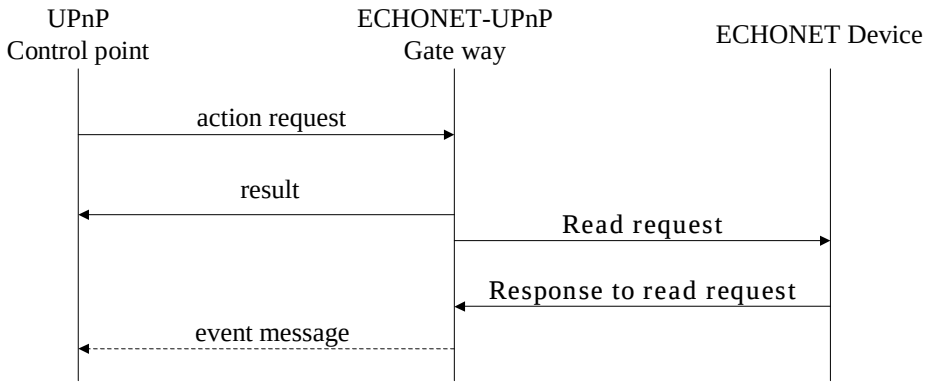

Fig. 4-6 Asynchronous Type Sequence

The operation of an ECHONET-UPnP gateway in the case where the asynchronous type sequence is used is as follows:

If the ECHONET-UPnP gateway receives an “action request” from a UPnP control point, the ECHONET-UPnP gateway shall send a “result” to the UPnP control point from which the “action request” was sent.

At the same time, the ECHONET-UPnP gateway shall perform a conversion to the ECHONET protocol data format and transmit to the ECHONET device a read request message that requires a response. If the ECHONET-UPnP gateway subsequently receives a response from the ECHONET device to the read request and the ECHONET-UPnP gateway has accepted a “subscription request” by that point, the ECHONET-UPnP gateway shall notify the UPnP control point of the device status(es) shown in the response by sending an “event message.”

In the case where an ECHONET-UPnP gateway operates according to the asynchronous type sequence, it shall store the statuses of ECHONET devices. This is because it is necessary for the ECHONET-UPnP gateway to include the status(es) of the relevant device(s) in each “result” response it sends immediately after it receives an “action request.”

4.3 Notifying the statuses of ECHONET devices to UPnP control points

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an ECHONET-UPnP gateway notifies an UPnP control point of the status of an ECHONET device, as well as the sequence to be followed.

No requirement is specified as to when to make a notification (by sending an “event message”), but it is recommended that an “event message” for notification be sent:

- When an status change notification is received from an ECHONET device;
- When an ECHONET-UPnP gateway acquires the status of an ECHONET device and detects that the status of the ECHONET device has changed;
- When an ECHONET-UPnP gateway receives a response from an ECHONET device while the ECHONET-UPnP gateway is operating according to the asynchronous type sequence (See Figs. 4-4 and 4-6).

Fig. 4-7 shows the sequence for the case where the ECHONET-UPnP gateway sends an “event message” to the UPnP control point after receiving a status change notification from an ECHONET device.

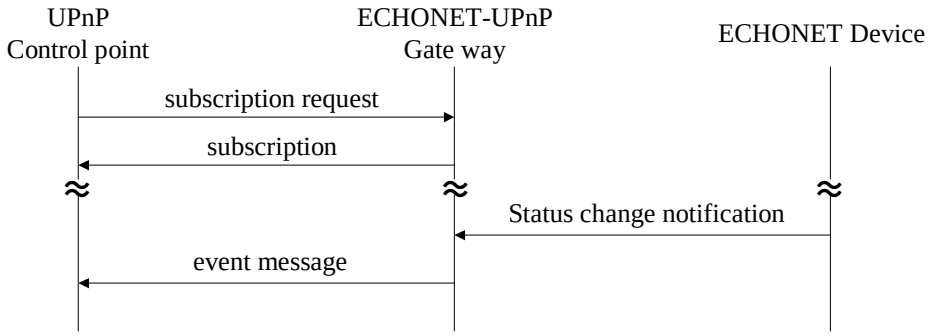

Fig. 4-7 Sequence for the Case where “event message”-based Notifications Are Made

The ECHONET-UPnP gateway shall send a unicast “event message” to the UPnP control points with which a subscription request has been registered.

Chapter 5 Device Template

The design policy for the publication of information on ECHONET devices to the UPnP side is that the corresponding virtual UPnP devices must be specified as UPnP root devices for each ECHONET device object whenever information on ECHONET devices is published to the UPnP side. Each virtual UPnP device shall provide its ECHONET Service.

5.1 Definition of Device

5.1.1 Device Type

Devices of the following device type conform to this Device Template:

urn:echonet-gr-jp:device:ECHONET_*Appliance*:1

The domain name (an element unique to the ECHONET) described in XML Device Descriptions shall be "echonet-gr-jp."

":1" indicates the version of the specifications. The "*Appliance*" section shall be replaced with the Appliance name specified in Appendix 2. For example, in the case of the home air conditioner class, the "*Appliance*" section shall be replaced with the Appliance name "HomeAirConditioner," and the Device Type shall therefore be described as:

"urn:echonet-gr-jp:device:ECHONET_ HomeAirConditioner:1."

5.1.2 Device requirements

Devices that fall under a device type other than "urn:echonet-gr-jp:device:ECHONET_*Appliance*:1" shall be equipped with functions to satisfy the requirements specified below. Table 5-1 shows the device requirements.

Table 5-1

DeviceType	Root	Req. or Opt. [1]	ServiceType	Req. or Opt. [1]	Service ID[2]
ECHONET_***Appliance***:1	Root	R	ECHONET_Service:1	R	ECHONET_***Appliance***

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

2 Prefixed by urn:echonet-gr-jp:serviceId:.

The "Service ID" for "ECHONET_Service" shall be "urn:echonet-gr-jp:serviceId:ECHONET_Appliance." The "Appliance" section shall be replaced with the unique name for the ECHONET device in question which is specified in Appendix 2 to the ECHONET Specification.

For example, in the case of an air conditioner, the "Appliance" section shall be replaced with "HomeAirConditioner," and the Service ID shall therefore be described as:

"urn:echonet-gr-jp:serviceId:ECHONET_ HomeAirConditioner."

5.2 XML Device Descriptions

Table 5-2 show the format for XML Device Descriptions for devices described in 5.1. Each XML Device Description element shall meet the specifications defined in UPnP Device Architecture Ver. 1.0.

Table 5-2 Format for XML Device Descriptions

```
<?xml version="1.0"?>
<root xmlns="urn:echonet-gr-jp:device-1-0">
    <specVersion>
        <major>1</major>
        <minor>0</minor>
    </specVersion>
    <URLBase>base URL for all relative URLs</URLBase>
    <device>
        <deviceType>urn:echonet-gr-jp:device:ECHONET_Appliance:1</deviceType>
        <friendlyName>short user-friendly title</friendlyName>
        <manufacturer>manufacturer name</manufacturer>
        <manufacturerURL>URL to manufacturer site</manufacturerURL>
        <modelDescription>long user-friendly title</modelDescription>
        <modelName>model name</modelName>
        <modelNumber>model number</modelNumber>
        <modelURL>URL to model site</modelURL>
        <serialNumber>manufacturer's serial number</serialNumber>
        <UDN>uuid:UUID</UDN>
        <UPC>Universal Product Code</UPC>
        <iconList>
            <icon>
                <mimetype>image/format</mimetype>
                <width>horizontal pixels</width>
                <height>vertical pixels</height>
                <depth>color depth</depth>
                <url>URL to icon</url>
            </icon>
            XML to declare other icons, if any, go here
        </iconList>
        <serviceList>
            <service>
                <serviceType>urn:echonet-gr-jp:service:ECHONET_Service:1</serviceType>
                <serviceId>urn:echonet-gr-jp:serviceId:ECHONET_Appliance</serviceId>
                <SCPDURL>URL to service description</SCPDURL>
                <controlURL>URL for control</controlURL>
                <eventSubURL>URL for eventing</eventSubURL>
            </service>
```

```
</serviceList>
        <presentationURL>URL for presentation</presentationURL>
    </device>
</root>
```

"UDN" element: When the ECHONET-UPnP gateway detects the connection to a network of an ECHONET device, the ECHONET-UPnP gateway shall generate a unique UUID value for the ECHONET device, store the generated UUID value in the UDN element and describe it in the XML Device Description.

5.3 Example XML Device Description (Air Conditioner)

This section shows an example of an XML Device Description for a virtual UPnP device for an air conditioner (EOJ = 0x013000).

Table 5-3 Example of Actual XML Device Description (Air Conditioner)

```
<root xmlns="urn:echonet-gr-jp:device-1-0">
    <specVersion>
        <major>1</major>
        <minor>0</minor>
    </specVersion>
    <device>
        <deviceType>urn:echonet-gr-jp:device:ECHONET_HomeAirConditioner:1</
deviceType>
        <friendlyName>Home Air Conditioner</friendlyName>
        <manufacturer>manufacturer name of air conditioner</manufacturer>
        <manufacturerURL>URL to manufacturer site</manufacturerURL>
        <modelDescription>Home Air Conditioner</modelDescription>
        <modelName>model name of air conditioner</modelName>
        <UDN>uuid:ad82f4cd-bafd-11da-9d2c-000e7b032792</UDN>
        <serviceList>
            <service>
                <serviceType>urn:echonet-gr-jp:service:ECHONET_Service:1</s
erviceType>
                <serviceId>urn:echonet-gr-jp:serviceId:ECHONET_HomeAirCondi
tioner</serviceId>
                <SCPDURL>/service.xml</SCPDURL>
                <controlURL>/ECHONET/control/ECHONET_HomeAirConditioner1</c
ontrolURL>
                <eventSubURL>/ECHONET/Eventing/ECHONET_HomeAirConditioner1<
/eventSubURL>
  </service>
        </serviceList>
        <presentationURL>/presentation.html</presentationURL>
    </device>
```

```
        </root>
```

Chapter 6 Service Template

The Service Template is specified below for “ECHONET_Service,” which is a ServiceType held by “ECHONET_*Appliance*” in this project.

6.1 Definition of Service Model

6.1.1 Service Type

Services of the following Service Type conform to this Device Template:

urn:echonet-gr-jp:service:ECHONET_Service:1

The domain name (an element unique to the ECHONET) described in XML Device Descriptions shall be “echonet-gr-jp.”

6.1.2 Service Type requirements

Service Type “ECHONET_Service” shall satisfy the following requirements:

The Variable Name shall be a name created according to the naming rules specified in Section 3.3. For the specifications for Data Type, Allowed Value and Eng.Units, refer to Appendix and Appendix 2 to the ECHONET Specification.

Table 6-1 State Variables

Variable Name	Req. or Opt.[1]	Data Type	Allowed Value	Default Value	Eng. Units
VariableName	see ECHONET Specification	see ECHONET Specification	see ECHONET Specification	*None*	see ECHONET Specification

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

- Variable Name

  The character string to be assigned to “***VariableName***” shall be determined based on Appendix 2 to the ECHONET Specification. For example, in the case where the operation status (EPC = 0x80) is described, the VariableName shall be “OperationStatus.”

- Req. or Opt.

  The distinction between “required” and “optional” shall be made based on Appendix to the ECHONET Specification.

- Data Type

  Refer to Appendix 2 to the ECHONET Specification.

- Allowed Value

  Refer to Appendix 2 to the ECHONET Specification.

- Default Value

  No default value is specified for any ECHONET property.
- Eng. Units

  Refer to Appendix to the ECHONET Specification.

6.1.3 Action

Specifies whether the service in question is a service for providing control or for referencing the status in relation to the property specified in the VariableName.

Table 6-2 Description of Action

Name	Req. or Opt.
Set ***VariableName***	Opt.
Get ***VariableName***	Opt.
Write ***VariableName***	Opt.
Read ***VariableName***	Opt.
Reset ***VariableName***	Opt.

- Name

  The Action name shall be “Set” or “Write” followed by the Variable Name in the case of control and “Get” or “Read” followed by the Variable Name in the case of the referencing of the status. In the case of a status reset, the Action name shall be “Reset” followed by the Variable Name. For the rules regarding the prefix to be used, refer to Table 3-3.
- Req. or Opt.

  Optional.

○ Set ***VariableName***/Write ***VariableName***

Used when altering the status of an ECHONET property.

– Arguments

The arguments for the case where the action is Set ***VariableName*** or Write ***VariableName*** are as shown in Table 6-3.

Table 6-3 List of Arguments for Control

Argument	Direction	relatedStateVariable
New ***VariableName***	In	***VariableName***

+ Argument

Indicates the argument. The Argument name shall be “New” followed by the Variable Name (i.e. “New ***VariableName***”). However, in the case of a composite type ECHONET property, the Argument name shall be “New ‘***adjective***’ ***VariableName***.”

For example, in the case of operation status, the Argument name shall be "NewOperationStatus."

+ Direction

The Direction shall be "In" in the case of device control.

+ relatedStateVariable

Indicates the target for reflection after control. This shall be the "***VariableName***." However, in the case of a composite type ECHONET property, this shall be the " '***Adjective' VariableName***."

○ Get ***VariableName***/Read ***VariableName***

Used when referencing the status of an ECHONET property.

– Arguments

The arguments for the case where the action is Get ***VariableName*** or Read ***VariableName*** are as shown in Table 6-4.

Table 6-4 List of Arguments for Referencing Statuses

Argument	Direction	relatedStateVariable
Current ***VariableName***	Out	***VariableName***

+ Argument

Indicates the argument. The Argument name shall be "Current" followed by the ***Variable Name*** (i.e. "Current ***VariableName***"). However, in the case of a composite type ECHONET property, the Argument name shall be "Current '***adjective***' ***VariableName***." For example, in the case of operation status, the Argument name shall be "CurrentOperationStatus."

+ Direction

The Direction shall be "Out" in the case of the referencing of the status of a device.

+ relatedStateVariable

Indicates the target for the referencing of the state. This shall be the "***VariableName***." However, in the case of a composite type ECHONET property, this shall be the " '***Adjective' VariableName***."

○ Reset ***VariableName***

Used for an ECHONET property status reset. There is no Argument.

6.2 XML Service Descriptions

Table 6-5 shows the format for XML Service Descriptions for ECHONET_Service. The XML Service Description format to use can be determined based on the property type or data type. Each element shall meet the specifications defined in UPnP Device Architecture Ver. 1.0.

Table 6-5 Format for XML Service Descriptions

```
<?xml version="1.0"?>
<scpd xmlns="urn:echonet-gr-jp:service-1-0">
    <specVersion>
        <major>1</major>
        <minor>0</minor>
    </specVersion>
    <actionList>
        <!— Date type, Time type, Level type, Character type, Selection type, Switch type,
'Others' type -->
        <action>
            <name>Set VariableName</name>
            <argumentList>
                <argument>
                    <name>New VariableName</name>
                    <direction>in</direction>
                    <relatedStateVariable> VariableName</relatedStateVariable>
                </argument>
            </argumentList>
        </action>
        <!-- Date type, Time type, Level type, Character type, Selection type, Switch type,
'Others' type -->
        <action>
            <name>Get VariableName</name>
            <argumentList>
                <argument>
                    <name>Current VariableName</name>
                    <direction>out</direction>
                    <relatedStateVariable> VariableName</relatedStateVariable>
                </argument>
            </argumentList>
        </action>
        <!-- Numerical value type -->
        <action>
            <name>Write VariableName</name>
            <argumentList>
                <argument>
                    <name>New VariableName</name>
                    <direction>in</direction>
                    <relatedStateVariable> VariableName</relatedStateVariable>
                </argument>
            </argumentList>
        </action>
```

```
        <!-- Numerical value type -->
        <action>
            <name>Read VariableName</name>
            <argumentList>
                <argument>
                    <name>Current VariableName</name>
                    <direction>out</direction>
                    <relatedStateVariable> VariableName</relatedStateVariable>
                </argument>
            </argumentList>
        </action>
        <!-- Reset type -->
        <action>
            <name>Reset VariableName</name>
        </action>
        Declarations for other actions added by UPnP vendor (if any) go here
    </actionList>
    <serviceStateTable>
        <!-- AVR type -->
        <stateVariable sendEvents="yes" or "no">
            <name> VariableName</name>
            <dataType>see ECHONET Specifications</dataType>
            <defaultValue>defaultValue</defaultValue>
            <allowedValueRange>
                <minimum>Minimum Data Value</minimum>
                <maximum>Maximum Data Value</maximum>
                <step>Step Value</step>
            </allowedValueList>
        </stateVariable>
        <!—Value type -->
        <stateVariable sendEvents="yes" or "no">
            <name> VariableName</name>
            <dataType>see ECHONET Specifications</dataType>
            <defaultValue>defaultValue</defaultValue>
        </stateVariable>
        <!—Date type -->
        <stateVariable sendEvents="yes" or "no">
            <name> VariableName</name>
            <dataType>see ECHONET Specifications</dataType>
            <defaultValue>defaultValue</defaultValue>
        </stateVariable>
        <!—AVL type -->
        <stateVariable sendEvents="yes" or "no">
            <name> VariableName</name>
            <dataType>string</dataType>
            <defaultValue>defaultValue</defaultValue>
            <allowedValueList>
                <allowedValue>Property Data1</allowedValue>
                Declarations for other allowed values added by UPnP vendor (if any) go
here
```

```
            </allowedValueList>
        </stateVariable>
        <!—String type -->
        <stateVariable sendEvents="yes" or "no">
            <name>VariableName</name>
            <dataType>string</dataType>
            <defaultValue>defaultValue</defaultValue>
        </stateVariable>
        <!-- 'Others' type -->
        <stateVariable sendEvents="yes" or "no">
            <name>VariableName</name>
            <dataType>bin.hex</dataType>
            <defaultValue>defaultValue</defaultValue>
        </stateVariable>
        Declarations for other state variables added by UPnP vendor (if any) go here
    </serviceStateTable>
</scpd>
```

6.3 Example XML Service Description (Air Conditioner)

This section shows an example of an XML Service Description for a virtual UPnP device for an air conditioner (EOJ = 0x013000).

Table 6-6 Example of Actual XML Service Description (Air Conditioner)

```
        <scpd xmlns="urn:echonet-gr-jp:service-1-0">
            <specVersion>
             <major>1</major>
             <minor>0</minor>
           </specVersion>
            <actionList>
                <action>
                    <name>SetOperationStatus</name>
                    <argumentList>
                        <argument>
                            <name>NewOperationStatus</name>
                            <direction>in</direction>
                            <relatedStateVariable>OperationStatus</relatedStateVariabl
        e>
                        </argument>
          </argumentList>
                </action>
               <action>
                    <name>GetOperationStatus</name>
                    <argumentList>
                        <argument>
                            <name>CurrentOperationStatus</name>
                            <direction>out</direction>
```

```
                        <relatedStateVariable>OperationStatus</relatedStateVaria
ble>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>GetProductCode</name>
                <argumentList>
                    <argument>
                        <name>CurrentProductCode</name>
                        <direction>out</direction>
                        <relatedStateVariable>ProductCode</relatedStateVariable
>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>SetOperationModeStatus</name>
                <argumentList>
                    <argument>
                        <name>NewOperationModeStatus</name>
                        <direction>in</direction>
                        <relatedStateVariable>OperationModeStatus</relatedStat
eVariable>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>GetOperationModeStatus</name>
                <argumentList>
                    <argument>
                        <name>CurrentOperationModeStatus</name>
                        <direction>out</direction>
                        <relatedStateVariable>OperationModeStatus</relatedStat
eVariable>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>WriteDesiredTemp</name>
                <argumentList>
                    <argument>
                        <name>NewDesiredTemp</name>
                        <direction>in</direction>
                        <relatedStateVariable>DesiredTemp</relatedStateVariable
>
                    </argument>
                </argumentList>
            </action>
            <action>
```

```
                <name>ReadDesiredTemp</name>
                <argumentList>
                    <argument>
                        <name>CurrentDesiredTemp</name>
                        <direction>out</direction>
                        <relatedStateVariable>DesiredTemp</relatedStateVariable
>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>SetWindVolumeLevel</name>
                <argumentList>
                    <argument>
                        <name>NewWindVolumeLevel</name>
                        <direction>in</direction>
                        <relatedStateVariable>WindVolumeLevel</relatedStateVar
iable>
                    </argument>
                </argumentList>
            </action>
            <action>
                <name>GetWindVolumeLevel</name>
                <argumentList>
                    <argument>
                        <name>CurrentWindVolumeLevel</name>
                        <direction>out</direction>
                        <relatedStateVariable>WindVolumeLevel</relatedStateVar
iable>
                    </argument>
                </argumentList>
            </action>
        </actionList>
        <serviceStateTable>
            <stateVariable sendEvents="yes">
                <name>OperationStatus</name>
                <dataType>string</dataType>
                <allowedValueList>
                    <allowedValue>ON</allowedValue>
                    <allowedValue>OFF</allowedValue>
                </allowedValueList>
            </stateVariable>
            <stateVariable sendEvents="no">
                <name>ProductCode</name>
                <dataType>string</dataType>
            </stateVariable>
            <stateVariable sendEvents="yes">
                <name>OperationModeStatus</name>
                <dataType>string</dataType>
                <allowedValueList>
```

```
                        <allowedValue>Auto</allowedValue>
                        <allowedValue>Cooling</allowedValue>
                        <allowedValue>Heating</allowedValue>
                        <allowedValue>Dehumidifying</allowedValue>
                        <allowedValue>Blast</allowedValue>
                        <allowedValue>Other</allowedValue>
                    </allowedValueList>
                </stateVariable>
                <stateVariable sendEvents="yes">
                    <name>DesiredTemp</name>
                    <dataType>ui1</dataType>
                    <allowedValueRange>
                        <minimum>16</minimum>
                        <maximum>30</maximum>
                        <step>1</step>
                    </allowedValueRange>
                </stateVariable>
                <stateVariable sendEvents="yes">
                    <name>WindVolumeLevel</name>
                    <dataType>string</dataType>
                    <allowedValueList>
                        <allowedValue>1</allowedValue>
                        <allowedValue>2</allowedValue>
                        <allowedValue>3</allowedValue>
                        <allowedValue>4</allowedValue>
                        <allowedValue>5</allowedValue>
                        <allowedValue>6</allowedValue>
                        <allowedValue>7</allowedValue>
                        <allowedValue>8</allowedValue>
                        <allowedValue>Auto</allowedValue>
                    </allowedValueList>
                </stateVariable>
            </serviceStateTable>
        </scpd>
```

Chapter 7 ECHONET Object-based Method

This chapter specifies the requirements regarding the operation of ECHONET-UPnP gateways for the case where the ECHONET Object-based Method is used.

7.1 Basic Concept

The implementation of the ECHONET Object-based Method is optional, while the implementation of the UPnP Device-based Method is compulsory.

In the ECHONET Object-based Method, ECHONET-UPnP gateways provide services provided by UPnP devices as virtual ECHONET objects in an ECHONET network. Such virtual ECHONET objects are hereinafter referred to as ECHONET objects.

ECHONET objects corresponding to UPnP devices in an ECHONET network shall operate in accordance with the ECHONET Specification and shall be treated in the same way as ordinary ECHONET objects.

Fig. 7-1 shows the system structure for the ECHONET Object-based Method.

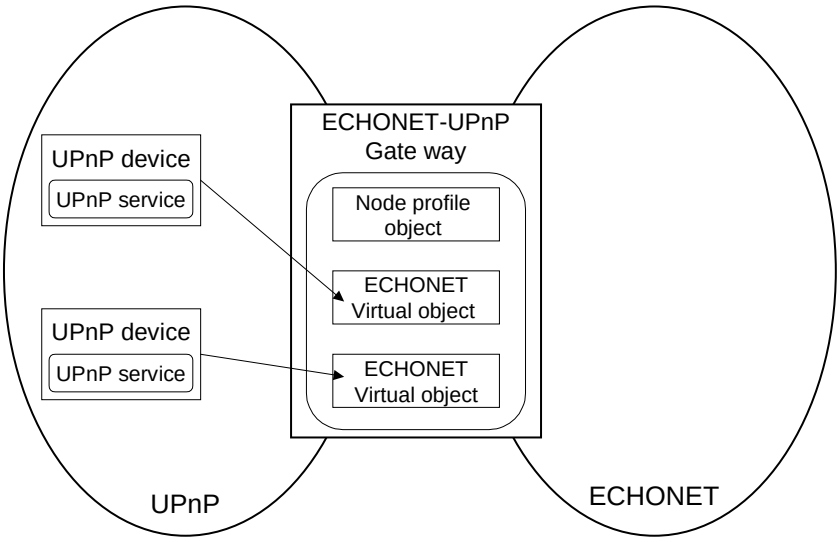

Fig. 7-1 System Structure for the Case where the ECHONET Object-based Method Is Used

Chapter 8 Processing for ECHONET-UPnP Gateways for the ECHONET Object-based Method

This chapter specifies the requirements regarding the processing to be performed in the case where an ECHONET-UPnP gateway operates as an ECHONET device in the ECHONET;

- Plug and play processing
- Control of UPnP devices from ECHONET objects

8.1 Plug and Play Processing

The following subsections specify the requirements regarding the plug and play processing to be performed when an ECHONET-UPnP gateway connects to a network and the plug and play processing to be performed when a UPnP device connects to a network. No requirement is specified regarding the plug and play processing to be performed when an ECHONET object which serves as a controller connects to a network, because the connection does not affect the operation of ECHONET-UPnP gateways.

The ECHONET-UPnP gateway-related processing to be performed when a UPnP device disconnects from a network is beyond the scope of the ECHONET Specification.

8.1.1 Processing to be performed when an ECHONET-UPnP gateway connects to a network

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an ECHONET-UPnP gateway connects to a network, as well as the sequence to be followed. Fig. 8-1 shows the sequence.

When an ECHONET-UPnP gateway starts up, it shall multicast a “search” message in the UPnP network to confirm whether an UPnP device is present. If the ECHONET-UPnP gateway receives a “response” message to the “search” message, it shall acquire the XML device and service descriptions and generate ECHONET device objects based on the acquired XML device description. The ECHONET-UPnP gateway shall then so amend the node profile object properties that the newly generated ECHONET objects are counted in, and shall send a status change notification. The status change notification shall meet the ECHONET Specification.

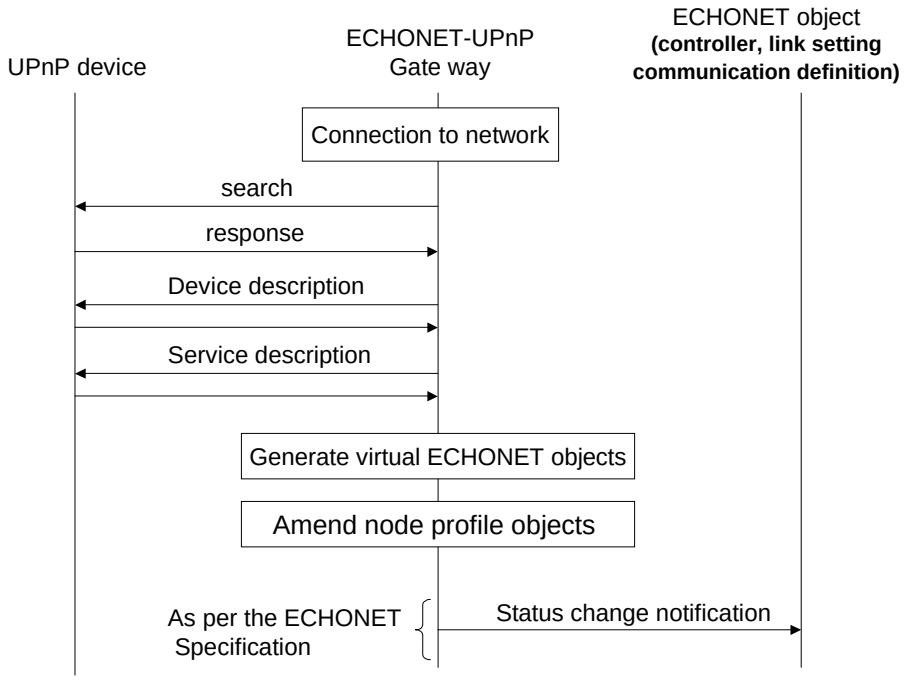

Fig. 8-1 Processing to Be Performed when an ECHONET-UPnP Gateway Connects to a Network

8.1.2 Processing to be performed when a UPnP device connects to a network

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when a UPnP device connects to a network, as well as the sequence to be followed. Fig. 8-2 shows the sequence.

If an ECHONET-UPnP gateway receives an “advertise” message multicast by a UPnP device connecting to a network, the ECHONET-UPnP gateway acquires the XML device and service descriptions and generate ECHONET objects based on the acquired XML device description. The ECHONET-UPnP gateway shall then so amend the node profile object properties that the newly generated ECHONET objects are counted in, and shall send a status change notification. The status change notification shall meet the ECHONET Specification.

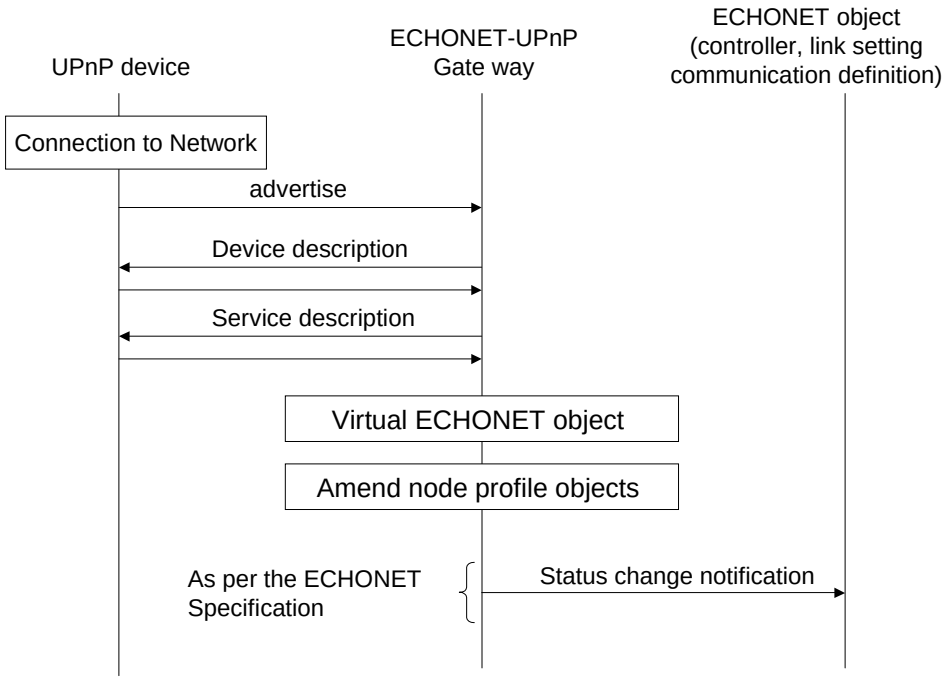

Fig. 8-2 Processing to Be Performed when a UPnP device Connects to a Network

8.2 Control of UPnP Devices from ECHONET Objects

8.2.1 Control of UPnP Devices

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an ECHONET object (controller object, link setting communication definition object, etc.) controls a UPnP device via an ECHONET-UPnP gateway, as well as the sequence to be followed. Fig. 8-3 shows the sequence for the operation of the ECHONET-UPnP gateway.

If an ECHONET-UPnP gateway receives a control request from an ECHONET object, the ECHONET-UPnP gateway shall perform a conversion to the UPnP protocol data format and send an “action request” to the UPnP device. The ECHONET-UPnP gateway shall then receive a “result” from the UPnP device. A control response shall be sent to the ECHONET object which is the sender of the control request only if the control request from the ECHONET object is a write message that requires a response.

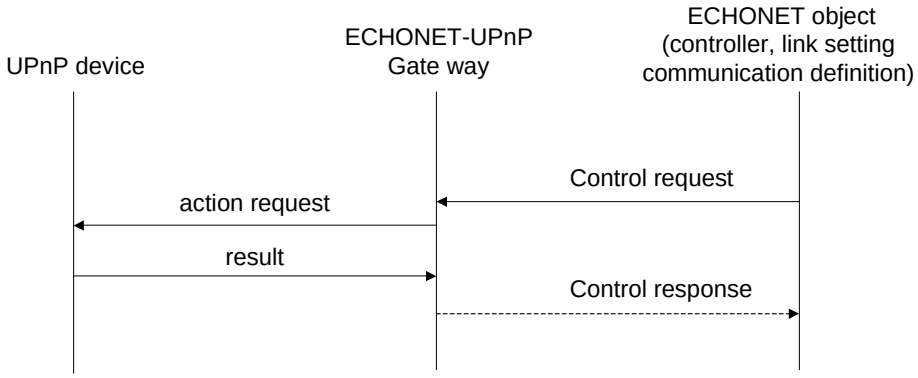

Fig. 8-3 UPnP Device Control Sequence

8.2.2 Referencing the statuses of UPnP devices

This section specifies the requirements regarding the ECHONET-UPnP gateway-related processing to be performed when an ECHONET object (controller object, link setting communication definition object, etc.) references the status of a UPnP device via an ECHONET-UPnP gateway, as well as the sequence to be followed.

Fig. 8-4 shows the sequence for the operation of the ECHONET-UPnP gateway.

If an ECHONET-UPnP gateway receives a status referencing request from an ECHONET object, the ECHONET-UPnP gateway shall perform a conversion to the UPnP protocol data format and send an “action request” to the UPnP device. The ECHONET-UPnP gateway shall then receive a “result” from the UPnP device. The ECHONET-UPnP gateway shall send a status referencing response to the ECHONET object from which the status referencing request was sent.

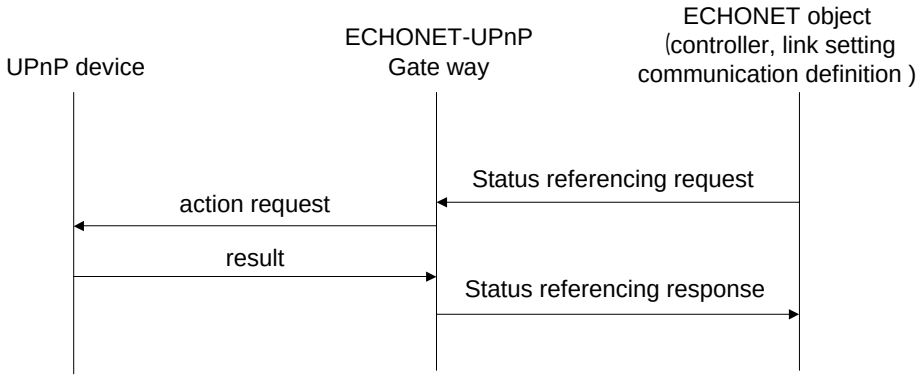

Fig. 8-4 UPnP Device Status Referencing Sequence