Part VIII

ECHONET Service Middleware 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 consortium members
	Version 1.0 addendum & corrigendum		
	Parts 1, 8, 9, and 10 were unchanged from Version 1.0.		
• Version 2.00	August 7th	2001	Open to consortium members
	Specifications for secure communications were added to Part 2. 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
	Typographical errors in Version 2.10 Draft were also corrected.		
•Version 2.11	April 26th	2002	Open to consortium members
•Version 3.00	August 29th	2002	Open to consortium members
•Version 3.10 Draft	November 8th	2002	Open to consortium members

Revised Parts
* “Chapter 4 EMS Service Middleware for Housing (Suggested Practical Applications)” was renumbered as Chapter 5.
* “Chapter 5 EMS Service Middleware for Stores and Small Buildings (Suggested Practical Applications)” was renumbered as Chapter 6.
* The file transfer service middleware requirements were added as Chapter 4.

•Version 3.10	December 18th	2002	Open to consortium members
•Version 3.11	March 7th	2003	Open to consortium members
•Version 3.12	May 22th	2003	Open to consortium members
•Version 3.20 Draft	October 17th	2003	Open to consortium members

Revised Parts
* “Chapter 5 EMS Service Middleware for Housing (Suggested Practical Applications)” was renumbered as Chapter 6.
* “Chapter 6 EMS Service Middleware for Stores and Small Buildings (Suggested Practical Applications)” was renumbered as Chapter 7.
* The link setting service middleware requirements were added as Chapter 5.

•Version 3.20	January 8th	2004	Open to consortium members

Revised Parts

* 1.1 Description on Basic Service Middleware was deleted * 5.2.1 "When the action conditions for different instances are satisfied simultaneously and the action messages to be transmitted are identical, it is not required to transmit the same message twice (or more)." was added.

•Version 3.21 May 26th 2004 Open to consortium members

Revised Parts
* 5.3.3 The value of b3:b2 of message format was corrected

•Version 3.20 January 8th 2004 Open to consortium members

Revised Parts
* 1.1 Description on Basic Service Middleware was deleted * 5.2.1 "When the action conditions for different instances are satisfied simultaneously and the action messages to be transmitted are identical, it is not required to transmit the same message twice (or more)." was added.

・Version 3.30 December 2nd 2004 Open to consortium members.

Revised Parts
・The group broadcast management service was added in Table 1.1 of Section 1.3. ・ The values "0x91" to "0x9F" for IEEE802.11/11b were added in the " 'Node identification number' property" section of "Table 2.1 List of Service Object Super Class Properties" of Chapter 2. ・The group broadcast number management service middleware was added in Chapter 6.

・Version 3.40 Draft December 28th 2004 Open to consortium members.

Revised Parts
・The layout of Fig. 1.1 was changed.

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

•Version 3.20 December 12th2006 Open to the public

Revised Parts
* Description of the basic service middleware (1.1) was deleted. * The following explanation was added in 5.2.1: "In the case where the action conditions for different instances are met simultaneously and the action messages to be transmitted are the same, it is not required to send the same message twice (or more)."

・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.

The specifications published by the ECHONET Consortium are established without regard to industrial property rights (e.g., patent and utility model rights). In no event will the ECHONET Consortium be responsible for industrial property rights to the contents of its specifications.

The publisher of this specification is not authorized to license and/or exempt any third party from responsibility for JAVA, IrDA, Bluetooth or HBS.
A party who intends to use JAVA, IrDA, Bluetooth or HBS should take action in being licensed for above-mentioned specifications.

In no event will the publisher of this specification be liable to you for any damages arising out of use of this specification.

The original language of The ECHONET Specification is Japanese. The English version of the Specification was translated the Japanese version. Queries in the English version should be refereed to the Japanese version.

Contents

Chapter 1 Overview

1.1 Basic Concept

In complex systems with advanced applications, the development load of application software can be reduced by using software that can provide such processing in the form of shared libraries, etc. When specific functions or applications are desired, there are many more expert but common types of processing. The ECHONET service middleware provides an API so that defined common processing functions can be accessed from application software. The service middleware also opens part of its functions to the public to allow vendors to build a system efficiently. Such functions are called ECHONET service objects.

The service middleware may include software handling common functions such as simple device linked action services, scheduled operation services, and gateway services for connections outside ECHONET. Software handling specific applications may also be included, such as energy management service (EMS) applications for efficient energy use in houses, smaller buildings, and stores; applications for automatic metering of power meters and gas meters; and applications for device maintenance. This standard defines as basic service middleware software that handles common functions regardless of the application, while software dedicated to specific applications is defined as individual service middleware. Thus, common functions and application types will be gradually expanded and standardized. In particular, the gateway service is a special service that handles connections between ECHONET and external systems, and is therefore separately specified in Part 9.

This definition of service middleware is intended not only to relieve the load of developing application software through the use of service middleware but also to allow vendors to concentrate on system/device development based on essential functions and performance, thereby creating an environment that encourages vendors to develop useful products for users.

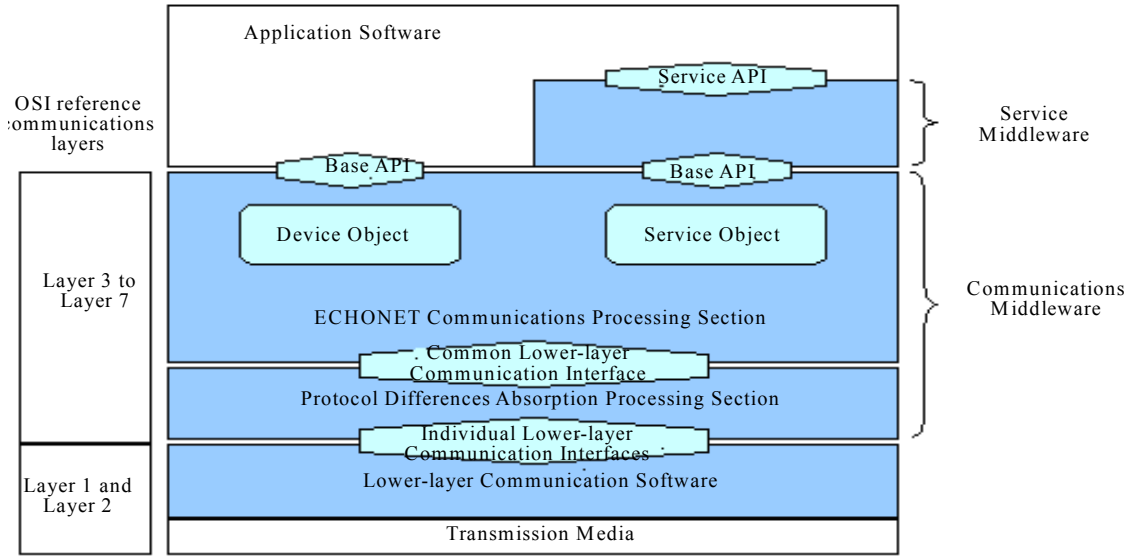
An object model is defined to enable the efficient construction of a system by accessing service middleware from the network. Called the ECHONET service object, this model permits access from the network using an ECHONET protocol.

Also defined is an API to access service middleware from application software. This is called a service API.

Application software developers may use the service middleware and service API (for accessing it) to facilitate the development of home system applications. They can also access service objects from the network to construct a system easily and efficiently.

1.2 Positioning on Communication Layers

The service middleware is positioned in the high-order layer of the ECHONET communications processing block in the communications layer configuration and provides common functions for implementing a certain application service for application software. Figure 1.1 shows the positioning of the service middleware and service objects in the communications layer. The service middleware is positioned within specific application software. Accordingly, service middleware uses the basic API to access ECHONET communication middleware functions in its internal processing. ECHONET communication middleware treats the service middleware as application software.



Lower-layer Communication Software Supported by the Current Version

Simbol	Name of Lower-layer Communication Software	Transmission Medium
A	Power Line Communication Protocol A System Power Line Communication Protocol D System	Power distribution lines
B	Low-power Radio	Low-power radio
C	Extended HBS	Twisted-pair cables
D	IrDA Control	Infrared
E	LonTalk®	Low-power radio
F	Bluetooth® (UDP/IP)	Low-power radio (BT)
G	Ethernet IEEE802.3 (UDP/IP)	Ethernet
H	IEEE802.11 IEEE802.11b (UDP/IP)	Low-power radio (WLAN)
I	Power Line Communication Protocol C System	Power distribution lines

Fig. 1.1 Relationship between the Service Middleware and Communication Layers

1.3 ECHONET Service Middleware and Service Object Defining Method

The concept of the service middleware and service object definitions is described below.

(1) Discussion regarding the system model

The scope of specific service application is defined, and the assumed system configuration is classified.

(2) Definition of service middleware functions

Standard and common functions are extracted and defined.

(3) Definition of service object

The network functions to be open to the public are provided as models on the basis of service middleware functions and defined as class specifications (function, service, property).

(4) Definition of sequence

The information exchange sequence between objects is defined.

(5) Definition of service API

APIs to access the service middleware are designed in accordance with the API design level.

The class group code of service objects is specified as 0x0D in Part 2, Chapter 4, Table 4.1. Table 1.1 shows a list of specified class codes in the service class group code 0x0D concerning the class codes of the service object that will be specified sequentially in detail in and after Chapter 2 of Part 8. 0x00 to 0xDF are assigned as basic and common service objects.

Table 1.1 List of Class Codes in Class Group Code (X1 = 0x0D service class)

Class code	Object name	Detail specification	Remarks
0x00	Reserved for future use		
0x01	Linked action service	○	
0x02	Reserved for future use	○	
0x03	Gateway service	○	
0x04	Action link service	○	
0x05	Trigger link service	○	
0x06	Group broadcast number management service	○	
0x07 to 0xDE	Reserved for future use		

0xDF	Gateway service	
0xE0 to 0xFF	Reserved for future use	

[Note] ○: The detailed requirements including those for the property composition are specified in Part 8.

1.4 Service API

The service API is specified as an API for application software to access the service middleware described in Chapter 1. API function items are shown as functional outlines, and the two levels mentioned in the specified levels of the basic API are assumed for the service API as levels of the detailed specifications.

(1) Level 1: Specification of function items and input/output data taking mounting into consideration
(2) Level 2: Detailed interface specifications intended for a specific language

Chapter 2 Service Object Super Class

This chapter defines the property composition that applies commonly to all classes belonging to the class group (class group code = 0x0D) that corresponds to service objects, as the service object super class.

2.1 Overview of Service Object Super Class Requirements

The service object super class properties are inherited by and implemented in all service objects. Table 2.1 shows the properties implemented in the service object super class together with brief explanations thereof. Detailed requirements for the individual properties are specified in Section 2.2 and succeeding sections.

Table 2.1 List of Service Object Super Class Properties (1/2)

Property name	EPC	Content of property Value range (decimal notation)	Data type	Size	Access rule	Required	Announcement on status change	Remarks
Operation status	0x80	Indicates the operation status of the service middleware. ON = 0x30, OFF = 0x31	Unsigned char	1 byte	Set Get	 ○	○	
Installation Location	0x81	Indicates the installation location of the ECHONET instance. See "2.3 Installation Location Property."	Unsigned char	1 byte	Set/Get		○	
Information on the version of specification	0x82	Indicates the version number of the corresponding standard. First byte: Indicates the major version number (above the decimal point) in binary notation. Second byte: Indicates the minor version number (below the decimal point) in binary notation. Third byte: Fixed at 0x00 (reserved for future use). Fourth byte: Fixed at 0x00 (reserved for future use).	Unsigned char	4 bytes	Get	○		
Node identification number	0x83	Used to uniquely specify within the domain the node in which the functions represented by this class are implemented. First byte: Lower-layer communication software ID field The assignment ranges for the individual media are as follows: * 0x11 to 0x1F: household power distribution line lower-layer communication software * 0x31 to 0x3F: Designated low power radio lower-layer communication software * 0x41 to 0x4F: Extended HBS lower-layer communication software * 0x51 to 0x5F: IrDA-dependent lower-layer communication software * 0x61 to 0x6F: LonTalk®-dependent lower-layer communication software * 0x71 to 0x7F: IP/Bluetooth™-dependent lower-layer communication software * 0x81 to 0x8F: IP/Ethernet/IEEE802.3-dependent lower-layer communication software * 0xFF: Random number-based generation * 0x00: Node identification number not specified. Second through ninth bytes: Unique number fields	Unsigned char	9 bytes	Get	○	○	

Table 2.1 List of Service Object Super Class Properties (2/2)

Property name	EPC	Content of property Value range (decimal notation)	Data type	Size	Access rule	Required	Announcement on status change	Remarks
Presence/absence of fault	0x88	Indicates that a fault (a sensor problem, etc.) has occurred. A fault has occurred = 0x41 No fault = 0x42	Unsigned char	1 byte	Get	○	○	
Description of fault	0x89	Description of fault 0x0000 to 0x03E8 (0 to 1000)	Unsigned short	2 bytes	Get			
Manufacturer code	0x8A	Specified in 3 bytes. (Specified by the ECHONET Consortium)	Unsigned char	3 bytes	Get	○		
Factory code	0x8B	Three-byte factory code. (Specified by each manufacturer)	Unsigned char	3 bytes	Get			
Product code	0x8C	Specified using ASCII code. (Specified by each manufacturer)	Unsigned char	12 bytes	Get			
Production number	0x8D	Specified using ASCII code. (Specified by each manufacturer)	Unsigned char	12 bytes	Get			
Date of manufacture	0x8E	Specified in 4 bytes. Indicated in the YYMD format (one byte for each letter): YY: dominical year (e.g. 1999 = 0x07CF) M: month (e.g. December = 0x0C) D: day (e.g. 20th = 0x14)	Unsigned char	4 bytes	Get			
SetM property map	0x9B	See Supplement 2.	Unsigned char	Max. 17 bytes	Get	○		
GetM property map	0x9C	See Supplement 2.	Unsigned char	Max. 17 bytes	Get	○		
Status change announcement property map	0x9D	See Supplement 2.	Unsigned char	Max. 17 bytes	Get	○		
Set property map	0x9E	See Supplement 2.	Unsigned char	Max. 17 bytes	Get	○		
Get property map	0x9F	See Supplement 2.	Unsigned char	Max. 17 bytes	Get	○		

[Note] The “o” for status change announcement indicates that the processing is required when the property is implemented.

2.2 Operation Status Property

The “Operation status” property of the service object super class indicates the operation status (ON/OFF) of the service in question. For cases in which the service object starts operating as soon as the node starts operating, this property may be implemented with the value fixed at 0x30 (The operation status of the communication function of the node shall be indicated by the operation status property of the node profile object).

2.3 Installation Location Property

When used in relation to service objects, the term “installation location” shall mean a location where a node is installed in which a service is in operation. The installation location property indicates the location where a node is installed in which the service middleware is in operation using 1-byte bit map data. When a value change occurs, the value after change must be broadcast over the domain (intra-domain general broadcast).

A free definition designation bit, installation location code and location number are assigned to the 8 bits of the installation location property. When all bits are zeros, it becomes a special code indicating that the installation location has not been specified. When all bits are ones, it becomes a special code indicating that the installation location is indefinite.

The content of each bit is explained below. Table 2.2 shows the relationship between the installation location, free definition designation bit, installation location code and location number.

* Free definition designation bit (b7)

Comprises the b7 bit. When b7 is 1, the installation location code and location number can be freely defined. When b7 is 0, the installation location code and location number indicate the service installation location in accordance with Table 9.3.

* Installation location code (b3 to b6)

Comprises the b3, b4, b5 and b6 bits. When the free definition designation bit is 1b, this code can be freely defined. When the free definition designation bit is 0b, this code indicates the type of service installation location in accordance with Table 9.3.

* Location number (b0 to b2)

Comprises the b0, b1 and b2 bits. When the free definition designation field is 1b, this number can be freely defined. When the free definition designation field is 0b, this is used as the number to distinguish the same type of multiple spaces from each other. For example, when there is a bathroom on the first floor and another on the second floor, they can be distinguished from each other by assigning 001b to the first-floor bathroom and 010b to the second-floor bathroom.

If the free definition designation field is 0b and the location number field is 000b, it shall mean that the installation location property has been initialized on the assumption that a piece of equipment will be installed in the installation location indicated by the installation location code. This state shall be called the “location number not specified” state.

When the installation location property has been initialized without assuming a service installation location type, the value must be the “installation location not specified” code (0x00). When it is inappropriate to specify a specific service installation location type, the installation location property value must be the “installation location indefinite” code (0xFF).

The values 0x01 through 0x07 shall be reserved for future use.

Table 2.2 Relationship Between Installation Location and Assigned Values

MSB LSB

Type of installation location	Free definition designation bit	Installation location code				Location number		
	b7	b6	b5	b4	b3	b2	b1	b0
Living room	0	0	0	0	1	“000b” to “111b” (“000b” indicates that the location number has not been specified.)		
Dining room	0	0	0	1	0			
Kitchen	0	0	0	1	1			
Bathroom	0	0	1	0	0			
Lavatory	0	0	1	0	1			
Washroom	0	0	1	1	0			
Hallway	0	0	1	1	1			
Room	0	1	0	0	0			
Stairway	0	1	0	0	1			
Front door	0	1	0	1	0			
Storeroom	0	1	0	1	1			
Garden, perimeter	0	1	1	0	0			
Garage	0	1	1	0	1			
Balcony	0	1	1	1	0			
Others	0	1	1	1	1			
Free definition*	1	“0000000b” to “1111110b”						
Installation location not specified	0	0	0	0	0	0	0	0
Installation location indefinite	1	1	1	1	1	1	1	1
Reserved for future use	“00000001b” to “00000111b”							

* Free definition means that the value can be freely defined for anticipated use in a store, small building, etc.

2.4 Information on the Specification Version

Indicates the version number of the main document of the corresponding standard using a 2-byte binary value and the release order of the supplements using a 1-byte ASCII code.

The first byte indicates the major version number (above the decimal point) and the second byte indicates the minor version number (below the decimal point). For example, the first and second bytes for “Version 2.10” will contain 0x02 (2) and 0x0A (10), respectively.

The third and fourth bytes are reserved for future expansions. These are fixed at 0x00 in this version of the Specification.

2.5 Node Identification Number

“Node Identification Number” is used to uniquely specify a node within the domain. This property consists of a lower-layer communication software ID field that stores an ID defined for each kind of lower-layer communication software, and a unique number field that stores an identification number uniquely given for each product by its own method of lower-layer communication software. This unique number is defined in the lower-layer communication software of Part 3. (However, it is only defined in the Bluetooth™-dependant lower-layer communication software or the Ethernet-dependant lower-layer communication software for the ECHONET Specification Version 3.00)

The ECHONET communication middleware can obtain this unique number from the lower-layer communication software by using the node identification number request service of the common lower-layer communication interface. (Here, the lower-layer communication software shall be Version 3.00 or later.) When a unique number is not be given by the lower-layer communication software, one will be generated as a random number. The random number generation algorithm should be implemented to minimize the possibility of duplicating the unique number of another node.

2.6 “Presence/Absence of Fault” Property

The “Presence/Absence of Fault” property of the service object super class indicates that a fault has occurred in an actual piece of equipment. Property codes used as property values are 0x41 for a fault occurrence and 0x42 for no fault.

2.7 "Description of Fault" Property

For the "Description of Fault" property, the fault description codes shown in Table 2.3 shall be used.

Table 2.3 Fault Description Code Assignment

"Description of Fault" property value (decimal notation)	Description of fault	
0x0000 (0)	No fault	
0x0001 (1)	Restorable faults	Turn off the Operation/Power switch or pull out the plug and turn on the switch again or reinsert the plug to restart operation.
0x0002 (2)		Press the reset button to restart operation.
0x0003 (3)		Improper setting
0x0004 (4)		Replenishment
0x0005 (5)		Cleaning (filter, etc.)
0x0006 (6)		Battery replacement
0x0007 to 0x0009 (7 to 9)		Reserved for future use
Fault	Faults requiring repair	Fault / safety mechanism activation
0x0014 to 0x001D (20 to 29)		Switch fault
0x001E to 0x003B (30 to 59)		Sensor fault
0x003C to 0x0059 (60 to 89)		Function parts fault
0x005A to 0x006E (90 to 110)		Control substrate fault
0x006F to 0x 03E8		Open to users
0x03E9 to 0xFFFF	Reserved for future use	

2.8 Manufacturer Code Property

The manufacturer code property identifies a manufacturer using a 3-byte code. The ECHONET Consortium assigns a unique property value to each of its members.

2.9 Factory Code Property

The factory code property identifies the factory of a manufacturer using a 3-byte code. The factory code property value is specified by each manufacturer, not by the ECHONET Consortium.

2.10 Product Code Property

The product code property identifies the product of a manufacturer using a 12-byte ASCII

code. The product code property value is specified by each manufacturer, not by the ECHONET Consortium.

2.11 Production Number Property

The production number property identifies the production number of a product of a manufacturer using a 12-byte ASCII code. The production number property value is specified by each manufacturer, not by the ECHONET Consortium.

2.12 Date of Manufacture Property

The date of manufacture property identifies the date of manufacture of a product of a manufacturer in 4 bytes (2 bytes for the year of manufacture and 1 byte each for the month and day of manufacture).

2.13 Property Map Property

The service object super class provides five property maps, each of which defines the information for describing the services that can be offered by the properties disclosed by the objects.

Four of the five property maps, namely the “Set property map,” “Get property map,” “SetM property map” and “GetM property map,” provide the information that indicates the relationship between the properties disclosed by the implemented objects and access rules (see Part 2, Section 4.2.8; hereinafter referred to as ARs) defined in product specifications.

The “status change announcement property map” indicates that an intra-domain general broadcast must be performed when the property value changes.

The map description formats are as shown in Supplement 2.

The property maps are defined as follows:

(1) Set property map
Indicates the properties relating to the “Set” AR.

(2) Get property map
Indicates the properties relating to the “Get” AR.

(3) SetM property map
Indicates the properties relating to the “SetM” AR.

(4) GetM property map

Indicates the properties relating to the “GetM” AR.

(5) Status change announcement property map

Lists the properties that are set for a general broadcast of changes in their values. In addition to the intra-domain general broadcast specified in the “Status Change Announcement” columns of the ECHONET Specification for various object properties supported by product specifications, properties for making an independent “status change announcement” in compliance with product specifications are included as well. This property map does not include a status notification that is set, for system operation purposes, by the “communication definition object for specifying the status notification method,” which is stated later.

No property maps are specified for the “AddM,” “DelM,” “AddMS,” “Anno,” “AnnoM” and “CheckM” ARs.

Chapter 3 Address Resolution Service Middleware

This chapter defines the specifications for the address resolution service class and the address resolution service middleware used to manage the ECHONET addresses of functions implemented at the other end of communication. The use of the address resolution service middleware allows access by the application software to functions that are not dependent on an ECHONET address.

3.1 Functions of the Address Resolution Service Middleware

The required functions for the address resolution service middleware are listed below together with brief explanations thereof.

① Address resolution database management function

Manages a database that has fields for the ECHONET object code, ECHONET address and node identification number of each of the functions implemented on the other end of communication.

② Address change notification transmission function

Broadcasts a node identification number over the domain (address change notification) upon a node startup or upon a node ECHONET address change.

③ Address change notification reception function

Receives address change notifications and updates the address DB.

④ Presence confirmation function

Periodically confirms whether or not the required function is present on the other end of communication. When the presence of the function is not confirmed, a search is performed throughout the domain.

⑤ Search function

Searches for a node with the specified node identification number or the specified function.

⑥ Non-home node object management function

Generates objects for other nodes.

⑦ Timeout period management function

Sets a communication timeout period for each subnet.

Detailed specifications for the individual functions follow:

3.2 Address Resolution Database

Table 3.1 lists the fields defined for the address resolution database (hereinafter referred to as "address resolution DB"). The address resolution service middleware assigns one record for each function implemented on the other end of communication.

Table 3.1 Address Resolution DB Fields

Field name	Range	Type	Remarks
Management ID	0 to 65535	Integer	Required
Identification information	0 to 2^{72}	Integer	Required
Identification information type	0 to 2	Integer	Required
Presence information	0 to 1	Integer	Required
NetID	0 to 255	Integer	Required
NodeID	0 to 255	Integer	Required
Class designation code	0 to 65535	Integer	Required
Instance designation code	0 to 255	Integer	Required
Presence confirmation interval	0 to 65535	Integer	Optional
Presence confirmation interval type	0 to 1	Integer	Optional
Installation location	0 to 255	Integer	Optional
Timeout period (in minutes)	0 to 65535	Integer	Optional
Number of retransmissions	0 to 3	Integer	Optional
Nickname	16 two-byte characters	String	Optional

3.2.1 Management ID

Management ID is used by the application software to specify the record for a function implemented on the other end of communication that is registered in the address resolution DB. The address resolution service middleware must uniquely specify this value within the address resolution DB.

3.2.2 Identification information

Identification information is used by the address resolution service middleware to uniquely identify within the domain a function implemented on the other end of communication. One of the following 3 value types is used in this field.

① Node identification number

For a function defined in the class specifications of Version 3.0 and later versions, the (9-byte) value of the node identification number property of the object that represents that function shall be used.

② Combination of unit identification information and manufacturer code

For a function defined in the class specifications of Version 2.1* and older versions, the (2-byte) unit identification information property value of the node profile object of the node in which that function is implemented shall be used in combination with the (3-byte) manufacturer code property value of the object that represents the function. The first four

bytes, second 2 bytes and last 3 bytes of this field shall contain the value 0x00, unit identification information and manufacturer code, respectively.

③ Null

When all the bytes of this field contain zero, it indicates that the identification information for the function of the record has not been specified.

3.2.3 Identification information type

Identification information type is the information specified to indicate the type of information specified in the identification information field. Zero, 1 or 2 must be specified when the type of information specified in the identification information field is the node identification number, a combination of unit identification information and manufacturer code, or user-defined identification information, respectively.

3.2.4 Presence information

When the value of this field is 1, it indicates that the presence in the domain of the function that corresponds to the record has been confirmed. When the value of this field is 0, it indicates that the presence in the domain of the function that corresponds to the record has not been confirmed. In the case where no confirmation of presence is to be made, this field always contains 1.

3.2.5 NetID

Indicates the NetID of the node that contains the function implemented on the other end of communication that is represented by that record.

3.2.6 NodeID

Indicates the NodeID of the node that contains the function implemented on the other end of communication that is represented by that record.

3.2.7 Class designation code

This code indicates the type of function implemented on the other end of communication that is represented by that record. The first byte shall be the class group code (X1) and the second byte shall be the class code (X2).

3.2.8 Instance designation code

This code is used to distinguish a function from another identical function in the same node. The instance code (X3) shall be specified.

3.2.9 Installation location

Indicates the location where the function has been installed. This field shall contain a value that satisfies the requirements specified in Part 2, Section 9.3.3 (Installation location property).

3.2.10 Presence confirmation interval

This field shall contain a value indicating the presence confirmation interval in minutes. When the presence confirmation interval type is “periodic,” a read request (presence confirmation request) is made every time the presence confirmation interval time period elapses, to the function described in the record (the function whose presence is to be confirmed), for the node identification number property (EPC = 0x83) or to the node profile object of the node in which the function whose presence is to be confirmed is implemented for both the unit identification information property (EPC = 0xBF) and manufacturer code property (EPC = 0x8A). When the presence confirmation interval type is “trigger,” a read request (presence confirmation request) is made to the function described in the record in the same manner as in the case of the periodic type when the occurrence interval of a notification or request from the function described in the record exceeds the presence confirmation interval.

If no response (presence confirmation response) is returned for a presence confirmation request, a function search shall be performed. If the value of this field is 0x0000, no presence confirmation request shall be made. The detailed requirements for the function search are stated later.

3.2.11 Presence confirmation interval type

“0” is written when the presence confirmation interval type is “periodic” and “1” is written when the presence confirmation interval type is “trigger.”

3.2.12 Timeout period

The timeout period (specified in seconds) allowed to elapse before it is determined whether or not there has been a response to a request sent to a function described in the specified record. When there are two or more functions for which NetID field values are the same, the same timeout period shall be used for all such functions.

3.2.13 Number of retransmissions

This field is used to specify whether or not to retransmit a read request, write request requiring a response or notification requiring a response when no read response, write response or notification response, respectively, is received during the timeout period immediately following the first transmission, to the object, of the read request, the write request requiring a response or the notification requiring a response. When the value in this field is 0, no retransmission shall be made to the function described in the record. When the

value in this field is 1, 2 or 3, a retransmission shall be made upon expiration of the timeout period up to once, twice or three times, respectively.

3.2.14 Nickname

This field shall contain a string of up to 16 two-byte characters specified by the application software to specify a record.

3.3 Address Resolution Service Class

This section specifies the requirements for the address resolution service class, so that relationships between functions provided in nodes that have been implemented, ECHONET addresses and node identification numbers, and operation status of the address resolution service middleware can be published within the domain. The object codes for the address resolution service class are as follows:

Class group code: 0x0D
Class code: 0x01
Instance code: 0x01

Table 3.2 lists the properties of the address resolution service class.

Table 3.2 List of Properties of the Address Resolution Service Class

Property name	EPC	Content of property Range (decimal notation)	Data type	Size	Access rule	Required	Announcement on status change	Remarks
Operation status	0x80	Indicates the operation status of the service. In operation: 0x30 Not in operation: 0x31	Unsigned char	1 byte	Get	○		
Address data	0xC1	Indicates the address data. First to ninth bytes: node identification number Tenth byte: Number of implemented functions (instances) Eleventh and succeeding bytes: Lists the EOJs of the implemented functions (instances).	Unsigned char	Max. 255 bytes	Get	○	○	

[Note]
The “o” for status change announcement indicates that the processing is required when the property is implemented.

3.3.1 Operation status

Indicates the operation status of the address resolution service middleware. The value is “0x31” during the “stop” state, cold start processing and warm start processing and “0x30” during normal operation.

3.3.2 Address data

Indicates all functions present in the node in which the address resolution service middleware is implemented by listing the ECHONET object codes. The first to ninth bytes indicate the node identification number and the tenth byte indicates the number of functions implemented. The eleventh and succeeding bytes list the 3-byte ECHONET object codes indicating the implemented functions. The number of object codes listed is equal to the number specified in the tenth byte. All functions represented by profile classes, services and pieces of equipment are listed here.

3.4 Service Boot Process

The address resolution service middleware boots up after receiving a service start request from the application software. There are 2 kinds of requests: cold start request and warm start request.

3.4.1 Boot by cold start

The address resolution service middleware begins a boot process after receiving a cold start request from the application software. The address resolution service middleware executes the following processes sequentially after which the status changes to normal operation. Here, the status during the boot processing is set to cold start status.

(1) Operation status confirmation of ECHONET communications

After the operation status of ECHONET communications is acquired and normal operation is confirmed, the next process is executed.

(2) Generation of address resolution service object

Confirmation is made that an address resolution object exists. If it does not exist, generation of an object is requested. If the generation fails, the process is terminated and the status changes to a quitted state.

(3) Binning off address resolution DB information

When records exist in the address resolution DB, all of the records are binned off.

(4) Generation of address information

The node identification number, number of instances, and EOJ are acquired from ECHONET communications, after which address information and its size are generated

(5) Writing values into address resolution service object

After a value of 0x31 is written into the operation status property, the status changes to a normal operation state. Then, the address information and its size generated in (3) are written into the property.

3.4.2 Boot by warm start

The address resolution service middleware begins a boot process after receiving a warm start request from the application software. The address resolution service middleware executes the following processes sequentially after which the status changes to a normal operation status. Here, the status during the boot processing is set to warm start status.

(1) Operation status confirmation of ECHONET communications

After the operation status of ECHONET communications is acquired and normal operation is confirmed, the next process is executed.

(2) Generation of address resolution service object

Confirmation is made that an address resolution object exists. If it does not exist, generation of an object is requested. If the generation fails, the process is terminated and

the status changes to a quitted state.

(3) Generation of address information

The node identification number, number of instances, and EOJ are acquired from ECHONET communications, after which address information and its size are generated

(4) Writing the values into address resolution service object

After a value of 0x31 is written into the operation status property, the status changes to a normal operation state. Then, the address information and its size generated in (3) are written to the property.

3.5 Processing During Normal Operation

3.5.1 Processing to register a function implemented on the other end of communication upon request

Registering a function implemented on the other end of communication upon request means registering a function located in a domain in the address resolution DB in response to a registration request from the application software. The application software must make a request to register the necessary function(s) immediately after a cold start, because the address resolution DB is cleared every time a cold start is made. There are two types of registration:

① Request-based registration of a single function
② Request-based registration of all functions

Functions that can be registered upon request are as follows:

① Functions based on Version 3.00 and succeeding versions
② Functions based on Version 2.1* and older versions

The address resolution service middleware must be capable of registering, in the address resolution DB, functions based on Version 3.00 in at least one of the two ways specified above. The registration of functions based on Version 2.1* and older versions shall be optional.

(1) Request-based registration of a single function

A search for the function specified by the application software is made within the domain and the function is written into the address resolution DB. Functions are specified by means of the highest 2 bytes (class group code X1 and class code X2) of the ECHONET class code bytes.

In the case of a function implemented in a node that supports Version 3.00 or a later version, the address resolution service middleware shall, upon receipt of a request for registration of a single function, read out the node identification number (EPC = 0x83) by means of a class-specific general broadcast with the source ECHONET object (SEOJ) specified as the address resolution service class. The highest 2 bytes of the ECHONET class code bytes that are specified by the application software shall be used to specify the class. The source ECHONET address (SEA), source ECHONET object code (SEOJ) and ECHONET data (EDT) are extracted from each of the response messages. The higher byte of the SEA, the lower byte of the SEA, the highest 2 bytes of the SEOJ, the lowest byte of the SEOJ and the EDT are written into the NetID field, NodeID field, class designation code field, instance designation field and identification information field of the address resolution DB,

respectively, to create the record for the function. A unique control ID is generated in the database and written. After this, both the number of fields generated and the control ID are returned to the application software.

No requirement is specified for the registration of a function implemented in a node that supports Version 2.1* or an older version.

In cases where the presence confirmation interval is specified together with the function, that value shall be specified in the presence confirmation interval field. If no presence confirmation interval is specified, the default interval is written. There is no requirement for the default value.

(2) Request-based registration of all functions

In response to a request by the application software, a search for all functions located within the domain is made and the functions are written into the address resolution DB.

For functions implemented in nodes that support Version 3.00 or a later version, the address resolution service middleware shall, upon receipt of a request for registration of all functions, read out the address data (EPC = 0xC0) by means of a general class broadcast with the specified address resolution service object that specifies the source ECHONET object (SEOJ) as the address resolution service class. The source ECHONET address (SEA), source ECHONET object code (SEOJ) and ECHONET data (EDT) are extracted from each of the response messages. The number of functions is read out from the extracted EDT and records are created in the address resolution DB until the number of records created equals the number of functions. The higher byte of the SEA and the lower byte of the SEA are written into the NetID and NodeID fields, respectively, of each of the records created. The node identification number is extracted from the EDT and written into the identification information field of each of the records created. The highest 2 bytes and lowest byte of the EOJ of the implemented functions listed in the EDT are written into the class designation code field and instance designation field, respectively, of each of the records created. A unique control ID is generated in the database and written. After this, both the number of fields generated and the control ID are returned to the application software.

No requirement is specified for the registration of a function implemented in a node that supports Version 2.1* or an older version.

In cases where the presence confirmation interval is specified together with the functions, that value shall be specified in the presence confirmation interval field. If no presence confirmation interval is specified, the default interval is written. There is no requirement for the default value.

3.5.2 Processing to automatically register a function implemented on the other end of communication

The processing to automatically register a function implemented on the other end of communication shall be activated upon receipt of a notification of the address data property (EPC = 0xC1) from the address resolution service object of another node by means of a broadcast. The address resolution service middleware extracts the following from the notification message:

* Source NetID
* Source NodeID
* Node identification information
* EOJ pair of the implemented function

If no record exists that matches the extracted data, a new record shall be created in the address resolution DB and the required fields shall be filled out.

The following information is registered in advance in the address resolution service middleware, to allow the application of a filter to selectively register functions in the address resolution DB.

① Highest 2 bytes of EOJ (X1, X2)
② Source NetID

That is, it is possible to register functions only when a match is achieved (or not achieved) with regard to ① or ②.

[Note] The instance code shall not be changed by a warm start.

3.5.3 Processing to register a function implemented on the other end of communication based on data held by the lower-layer communication software

When the lower-layer communication software has pairs of identification information about the nodes located in the subnet and the ECHONET addresses, records may be added to the address resolution DB by receiving a notification thereof or by reading the data. This processing is optional and no requirement is specified for the method to send a notification of the necessary data from the lower-layer communication software or for the method to read the necessary data.

Because it is not possible to acquire the ECHONET object code of a function implemented on the other end of communication with this processing, it is necessary to acquire it separately and fill out the fields of the record created.

3.5.4 Processing to confirm the presence of a function implemented on the other end of communication

Confirmation of the presence of a function implemented on the other end of communication means confirming whether or not the specified function is present in the domain. This processing must be activated upon request from the application software, upon expiration of the presence confirmation interval (address resolution DB) or immediately after a warm start.

(1) Activation in response to a request from the application software

The function implemented on the other end of communication whose presence is to be confirmed is specified by means of a control ID or identification information.

The presence of functions implemented in a node that supports Version 3.00 or a later version shall be confirmed by reading out the node identification number by means of a class-specific broadcast to the class specified in the “class designation code” field of the record specified by the control ID or identification information (destination ECHONET object (DEOJ), address resolution service object – source ECHONET address (SEOJ)). If there is a response message that has identification information matching the ECHONET data (EDT), “True” will be reported to the application software and “1” will be written into the presence data field. If there is no response message that has identification information matching the ECHONET data (EDT), “False” will be reported to the application software and “0” will be written into the presence data field.

If there is a response message that has identification information matching the ECHONET data (EDT) and the source NetID and source NodeID are different from those registered in the address resolution DB, the fields will be rewritten.

(2) Activation upon expiration of the presence confirmation interval

The function, implemented on the other end of communication, whose presence is to be confirmed, is specified by means of the identification information field in the record containing the elapsed interval field.

The presence of functions implemented in a node that supports Version 3.00 or a later version shall be confirmed by reading out the node identification number by means of a class-specific broadcast to the class specified in the class designation code field of the record specified by the control ID or identification information (destination ECHONET object (DEOJ), address resolution service object – source ECHONET address (SEOJ)). If

there is a response message that has identification information matching the ECHONET data (EDT), “1” will be written into the presence data field. If there is no response message that has identification information matching the ECHONET data (EDT), a “presence not confirmed” notification will be sent to the application software together with the control IDs for the functions that no longer exist and “0” will be written into the presence data fields.

If there is a response message that has identification information matching the ECHONET data (EDT) and the source NetID and source NodeID are different from those registered in the address resolution DB, the fields will be rewritten.

(3) Activation immediately after a warm start
The presence of all functions registered in the address resolution DB is confirmed.

The presence of functions implemented in a node that supports Version 3.00 or a later version shall be confirmed by reading out the node identification number by means of a class-specific broadcast to the class specified in the class designation code field of the record specified by the control ID or identification information (destination ECHONET object (DEOJ), address resolution service object – source ECHONET address (SEOJ)). If there is a response message that has identification information matching the ECHONET data (EDT), “1” will be written into the presence data field. If there is no response message that has identification information matching the ECHONET data (EDT), a “presence not confirmed” notification will be sent to the application software together with the control IDs for the functions that no longer exist and “0” will be written into the presence data fields.

If there is a response message that has identification information matching the ECHONET data (EDT) and the source NetID and source NodeID are different from those registered in the address resolution DB, the fields will be rewritten.

3.5.5 Processing to use the address resolution DB

The address resolution service middleware must be capable of providing the application software with services to change and reference a specified field of a record specified by a control ID of the address resolution DB and a service to delete such a record. The address resolution service middleware must also be capable of generating a record upon request from the application software and returning an appropriate control ID.

3.5.6 Processing to manage other node objects

The address resolution service middleware shall be capable of generating, in the communication middleware, other node objects for functions registered in the address resolution DB. This may be achieved using either of the following 2 methods:

① Automatic generation of other node objects
② Property designation-based generation of other node objects

(1) Automatic generation of other node objects
Generates other node objects for the functions specified by control IDs. An inquiry is made to the functions in question and objects are generated for other nodes that can handle all properties of the functions.

(2) Property designation-based generation of other node objects
Generates other node objects for the function specified by a control ID. Other node objects generated with this method can handle the specified properties only.

The address resolution service middleware generates other node objects for the following purposes:

① To determine the broadcast notification messages to receive. That is, if a broadcast notification from a function to which an object of another node belongs is a message that identifies a property present in the object, the value will be reflected in the relevant property of another node's object or reported to the application software (address resolution service middleware). When the instance designation code of the record specified by the control ID is 0x00, the value will be reported to the application software (address resolution service middleware) regardless of the source instance code in the message, as long as the value is one for the relevant property.

② The application software (address resolution service middleware) acquires values of functions that are not in synchronization with broadcast notifications. This becomes possible when it is in possession of values reported through broadcast notifications as values of properties of other node objects, and is achieved by referencing the properties of another node's object for the value in question.

3.5.7 Control ID-based object processing

Control ID designation-based services to reference and change object properties and make object property notifications shall be provided to the application software. The address resolution service middleware shall convert control IDs into appropriate ECHONET addresses and ECHONET object codes by referencing the address resolution DB and make service requests to the ECHONET communications processing section.

Response messages and reception notification messages shall be handled in the same manner. That is, the ECHONET addresses and ECHONET object codes are converted into

appropriate control IDs and the contents are presented to the application software.

3.5.8 Processing to manage timeouts

Retransmits a read request, write request requiring a response or notification requiring a response when no read response, write response or notification response, respectively, is received during the timeout period immediately following the first transmission, to a function that corresponds to a record for which the “number of retransmissions” and “timeout period” fields in the address management DB are not zero, of the read request, the write request requiring a response or the notification requiring a response. Retransmissions can be made up to the number of times specified in the “number of retransmissions” field. If no response is received by the end of the last timeout period, a “communication failed” notification will be sent to the application software.

[Note] This processing can be achieved only when the ECHONET communications processing section is capable of handling frame numbers.

3.6 Capability Requirements

The address resolution service middleware shall:

Have address resolution service objects;

Be capable of broadcasting the contents of the address data property to the address resolution service objects after a warm/cold start; and

Be capable of receiving address data property broadcasts and incorporating their contents.

3.7 Address Resolution Service Interface

This section specifies the type of services that the address resolution service API must provide and the data communication that must be performed between the application software and address resolution service middleware.

3.7.1 Services to start up and stop the address resolution service middleware

(1) Cold start request service

① Overview

The application software requests the address resolution service middleware to initiate a cold start. The processing to be performed by the address resolution service middleware is specified in Section 2.4.1.

② Direction

Application software → address resolution service middleware

③ Input data

None

④Response data

Data name	Explanation
Request result	0: Cold start successfully completed 1: Cold start failed

(2) Warm start request service

① Overview

The application software requests the address resolution service middleware to initiate a warm start. The processing to be performed by the address resolution service middleware is specified in Section 2.4.2.

② Direction

Application software → address resolution service middleware

③ Input data

None

④Response data

Data name	Explanation
Request result	0: Warm start successfully completed 1: Warm start failed

(3) “Request for stop” service

① Overview

The application software requests the address resolution service middleware to stop the processing currently being performed. The address resolution service middleware stops the processing and sets the "operation status" property value of the address resolution service object to 0x31.

② Direction

Application software → address resolution service middleware

③ Input data

None

④Response data

Data name	Explanation
Request result	0: Processing successfully stopped 1: Attempt to stop processing failed

3.7.2 Services to register functions implemented on the other end of communication

(1) Selective registration request service

① Overview

Requests the address resolution service middleware to start the processing for an intra-domain search for and registration of the specified function. While this processing is being performed, no other processing requiring communication can be performed.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Class designation code	X1, X2

④ Response data

Data name	Explanation
Processing result	0: Processing successfully started 1: Failed (processing for registration request in progress

(2) Selective registration result notification service

① Overview

After completion of the processing for the intra-domain search for and registration of the specified function(s), the registration result is returned to the application software.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Registration result	0: Registration successfully completed 1: Registration failed (the function in question does not exist)
Number of registered functions	The number of functions registered in the address resolution DB.
List of the control IDs of the registered functions	A list of the control IDs of the functions registered in the address resolution DB.

④ Response data

None

(3) Automatic aggregate registration service

① Overview

Requests the address resolution service middleware to start the processing for an intra-domain search for and registration of all functions. While this processing is being performed, no other processing requiring communication can be performed.

② Direction

Application software → address resolution service middleware

③ Input data

None

④ Response data

Data name	Explanation
Processing result	0: Processing successfully started 1: Failed (processing for registration request in progress

(4) Automatic aggregate registration result notification service

① Overview

After completion of the processing for the intra-domain search for and registration of all functions, the registration result is returned to the application software.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Registration result	0: Registration successfully completed 1: Registration failed (the function(s) in question do not exist)
Number of registered functions	The number of functions registered in the address resolution DB.
List of the control IDs of the registered functions	A list of the control IDs of the functions registered in the address resolution DB.
List of the class codes of the registered functions	A list of the X1 and X2 values.

④ Response data

None

(5) "Reporting of functions registered by the lower-layer communication software" service

① Overview

Each time the lower-layer communication software registers data about a function implemented on the other end of communication in the address resolution DB, the data is reported to the application software.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Number of registered functions	The number of functions registered in the address resolution DB.
List of the control IDs for the registered functions	A list of the control IDs for the functions registered in the address resolution DB.

④ Response data

None

(6) Registration request cancellation service

① Overview

Requests the cancellation of the registration request processing currently being performed. If this service is accepted, no registration result notification service will be issued to the application software. The registration of a function in the address resolution DB that was searched for and found before the acceptance of this service will not be terminated.

② Direction

Application software → address resolution service middleware

③ Input data

None

④ Response data

Data name	Explanation
Processing result	0: Cancellation successfully completed 1: Cancellation failed (there is no registration request currently being processed)

3.7.3 Confirmation of the presence of a function implemented on the other end of communication

(1) Control ID designation-based presence confirmation service

① Overview

Requests the address resolution service middleware to start the processing for an intra-domain search for and registration of functions having the specified control IDs.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Class designation code	X1, X2

④ Response data

Data name	Explanation
Processing result	0: Processing successfully started 1: Failed (registration request processing in progress)

3.7.4 Services to use the address resolution DB

(1) Record creation request service

① Overview

The application software requests the address resolution service middleware to create a new record.

The address resolution service middleware creates a record, writes a value that is unique within the DB into the control ID field of the record created, and returns the control ID to the application software.

② Input data

None

③ Response data

Data name	Explanation
Control ID	The value of the control ID generated by the address resolution service middleware. If the control ID generation process fails, NULL will be returned.

(2) Record deletion request service

① Overview

The application software requests the address resolution service middleware to delete the specified record.

The address resolution service middleware deletes the record that has a field containing a value or string matching the specified control ID value or nickname string.

② Input data

Data name	Explanation
Record designation information	Control ID value or nickname string.

③ Response data

Data name	Explanation
Deletion result	Deletion successfully completed, deletion failed, or the specified record does not exist.

(3) Field value referencing request service

① Overview

The application software requests the address resolution service middleware to provide the value of the specified field of the specified record.

The address resolution service middleware returns to the application software the value of the specified field of the record having a field containing a value or string matching the specified control ID value or nickname string.

② Input data

Data name	Explanation
Record designation information	Control ID value or nickname string
Field designation information	Control ID, identification information, identification information type, presence information, NetID, NodeID, class designation code, instance designation code, presence confirmation interval, presence confirmation interval type, installation location, timeout period, number of retransmissions, or nickname

③ Response data

Data name	Explanation
Referencing result	Referencing successfully completed, or the specified record or field does not exist.
Referenced value	The value of the referenced field.

(4) Field value alteration request service

① Overview

The application software requests the address resolution service middleware to write the specified value into the specified field of the specified record.

The address resolution service middleware writes the specified value into the specified field of the record having a field containing a value or string matching the specified control ID value or nickname string.

② Input data

Data name	Explanation
Record designation information	Control ID value or nickname string
Field designation information	NetID, NodeID, class designation code, instance designation code, presence confirmation interval, presence confirmation interval type, installation location, timeout period, number of retransmissions, or nickname

③ Response data

Data name	Explanation
Alteration result	Alteration successfully completed, or the specified record or field does not exist.

(5) Installation location designation request service

① Overview

The application software requests the address resolution service middleware to write a value into the installation location field of the specified record.

The address resolution service middleware requests the ECHONET communications processing section to send a message that reads out the installation location code from the NetID, NodeID, class designation code and instance designation code fields of the record having a field containing a value or string matching the specified control ID value or nickname string. If the installation location code is read out successfully, the value will be written into the installation location code field.

② Input data

Data name	Explanation
Record designation information	Control ID value or nickname string

③ Response data

Data name	Explanation
Request result	Request made successfully, the specified record or field does not exist, or reading failed.

3.7.5 Home node object management services

(1) Object generation request service

① Overview

Generates home node objects. After completion of object generation, the address data of the address resolution service object shall be changed.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Class designation code	X1 and X2 of the home node object to be generated.
Number of properties	The number of properties contained in the home node object to be generated.
EPC code list	A list of the EPC values of the properties.
EPC list (size)	A list of the sizes (EPC).
EPC list (type)	A list of the types (EPC).
ReadOnly flag list	A list of the EPC-specific flags indicating whether or not Get can be performed. 0: Get can be performed 1: Get cannot be performed
Status change notification flag list	A list of the EPC-specific flags indicating whether or not to enable the status change notification function. 0: Status change notification function disabled 1: Status change notification function enabled
Status notification method designation flag list	A list of the EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for status notification method designation). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
Set acceptance method designation flag list	A list of the EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for Set control acceptance method designation). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
Action link flag list	A list of the EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for link setting (action setting)). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
Trigger link flag list	A list of the EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for link setting (trigger setting)). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network

	2: A corresponding property exists and has been published in the network
EPC initial value list	A list of the values to initialize the EPC codes on the EPC code list.
Status change notification property initial value list	A list of the values to initialize the properties identified in the status notification method designation flag list as existing properties.
Set acceptance method property initial value list	A list of the values to initialize the properties identified in the Set acceptance method designation flag list as existing properties.
Action link property initial value list	A list of the values to initialize the properties identified in the action link flag list as existing properties.
Trigger link property initial value list	A list of the values to initialize the properties identified in the trigger link flag list as existing properties.
Array EPC code list	A list of the EPC values of the array properties.
Size list (array, EPC)	A list of the sizes (array, EPC).
Type list (array, EPC)	A list of the types (array, EPC).
"Number of array elements" list	A list of the maximum numbers of elements (array, EPC).
List (array, ReadOnly flag)	A list of the array EPC-specific flags indicating whether or not Get can be performed. 0: Get can be performed 1: Get cannot be performed
List (array, status change notification flag)	A list of the array EPC-specific flags indicating whether or not to enable the status change notification function. 0: Status change notification function disabled 1: Status change notification function enabled
List (array, status notification method designation flag)	A list of the array EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for status notification method designation). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
List (array, Set acceptance method designation flag)	A list of the array EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for Set control acceptance method designation). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
List (array, action link flag)	A list of the array EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for link setting (action setting)). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
List (array, trigger link flag)	A list of the array EPC-specific flags indicating whether or not a corresponding property exists (communication definition object for link setting (trigger setting)). 0: No corresponding property exists 1: A corresponding property exists but has not been published in the network 2: A corresponding property exists and has been published in the network
List (array, EPC initial value)	A list of the values to initialize the EPC codes on the array EPC code list.
List (array, status change notification property, initial value)	A list of the values to initialize the properties identified in the array status notification method designation flag list as existing properties.
List (array, Set acceptance method property, initial value)	A list of the values to initialize the properties identified in the array Set acceptance method designation flag list as existing properties.
List (array, action link property, initial value)	A list of the values to initialize the properties identified in the array action link flag list as existing properties.
List (array, trigger link property, initial value)	A list of the values to initialize the properties identified in the array trigger link flag list as existing properties.

④ Response data

Data name	Explanation
Processing result	0: Generation successfully completed 1: Generation failed
Control ID	The control ID value generated by the address resolution

	service middleware. If the control ID generation process fails, NULL will be returned.

(2) Object deletion service

① Overview

Deletes the objects specified by control IDs. After completion of object deletion, the address data of the address resolution service object shall be changed.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control IDs of the home node objects to be deleted.

④ Response data

Data name	Explanation
Processing result	0: Deletion successfully completed 1: Deletion failed

(3) Property alteration notification service

① Overview

Notifies the application software that alterations to property values have been made from another node.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the altered properties belong.
Number (alteration, EPC)	The number of properties altered.
List (alteration, EPC)	A list of the EPC values of the properties altered.
List (alteration, EPC, size)	A list of the sizes of the properties altered.
List (alteration, EPC, type)	A list of the types of the properties altered.
List (alteration, EPC, value)	A list of the altered values.

④ Response data

Data name	Explanation
Processing result	0: Alteration accepted 1: Alteration rejected

(4) Array property alteration notification service

① Overview

Notifies the application software that array property values have been altered by another node.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the altered array properties belong.
Alteration (EPC)	The EPC values of the array properties altered.
Size (alteration, EPC)	The sizes of the array properties altered.
Type (alteration, EPC)	The types of the array properties altered.
Value (alteration, EPC)	A list of the altered values.

④ Response data

Data name	Explanation
Processing result	0: Alteration accepted 1: Alteration rejected

(5) Property update request service

① Overview

The address resolution service middleware requests the application software to update properties using the latest data.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties to be updated belong.
Number of EPC values to be updated	The number of properties to be updated.
Update list (EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, size)	A list of the sizes of the properties to be updated.
List (update, EPC, type)	A list of the types of the properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Control ID	The control ID of the home node object to which the properties to be updated belong.

Number of updated EPC values	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, size)	A list of the sizes of the properties to be updated.
List (update, EPC, type)	A list of the types of the properties to be updated.
List (update, EPC, data)	A list of the values to be changed.

(6) Array property update request service

① Overview

The address resolution service middleware requests the application software to update array properties using the latest data.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the array properties to be updated belong.
Update (EPC)	The EPC values of the array properties to be updated.
Size (update, EPC)	The sizes of the array properties to be updated.
Type (update, EPC)	The types of the array properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Control ID	The control ID of the home node object to which the array properties to be updated belong.
Update (EPC)	The EPC values of the array properties to be updated.
Size (update, EPC)	The sizes of the array properties to be updated.
Type (update, EPC)	The types of the array properties to be updated.
Data (update, EPC)	The values to be changed.

(7) Property notification request service

① Overview

Requests the address resolution service middleware to report the values of the specified properties. In the case of a notification requiring a response, no other notification of the values of the properties in question can be made until a response is received or the timeout period expires.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties selected for notification belong.
Number of EPC values (notification)	The number of properties selected for notification.
EPC list (notification)	A list of the EPC values of the properties selected for notification.
Broadcast request flag	A list of the flags for the individual properties selected for notification that specify whether or not to perform a broadcast: 0: Unicast 1: Broadcast (over the domain) 2: Broadcast (over the home subnet)
Control ID of the recipient of the notification	The control ID of the function to which the notification is to be made. This will be ignored when a broadcast is specified.
Response request flag	A flag specifying whether or not a response is required. This will be ignored when a broadcast is specified. 0: Response not required 1: Response required

④ Response data

Data name	Explanation
Processing result	0: Notification accepted 1: Notification rejected

(8) Array property notification request service

① Overview

Requests the address resolution service middleware to report the values of the specified array properties. In the case of a notification requiring a response, no other notification of the values of the properties in question can be made until a response is received or the timeout period expires.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties selected for notification belong.
EPC value (notification)	The EPC values of the array properties selected for notification.
Array number (notification)	The array element numbers of the array elements selected for notification.
Broadcast request flag	A list of the property-specific flags for the properties selected for notification that specify whether or not to perform a broadcast: 0: Unicast 1: Broadcast (over the domain) 2: Broadcast (over the home subnet)

Control ID of the recipient of the notification	The control ID of the function to which the notification is to be made. This will be ignored when a broadcast is specified.
Response request flag	A flag specifying whether or not a response is required. This will be ignored when a broadcast is specified. 0: Response not required 1: Response required

④ Response data

Data name	Explanation
Processing result	0: Notification accepted 1: Notification rejected

(9) Timeout notification service

① Overview

Notifies the application software that no notification response was received during the timeout period.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties determined to require a timeout notification belong.
Number (notification, EPC)	The number of properties requiring a timeout notification.
List (notification, EPC)	A list of the properties requiring a timeout notification.

④ Response data

Data name	Explanation
Processing result	0: Notification accepted 1: Notification rejected

(10) Property update request service

① Overview

Requests the address resolution service middleware to update properties. When an update of properties specified as requiring a status change notification is accepted, the address resolution service middleware will broadcast the changed values. When the properties selected to be updated are not specified as requiring a status change notification, notifications will be made as specified in the notification method designation field.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties to be updated belong.
Number of EPC values (update)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be written into the properties to be updated.
Notification method designation	0: No notification 1: Broadcast notification 2: Notified individually to all nodes registered in the address resolution service DB 3: Notification by means of class-specific broadcasts to all classes registered in the address resolution service DB

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Number of EPC values (update, acceptance)	The number of properties updated.
List (update, acceptance, EPC)	A list of the properties updated (EPC).

(11) Array property update request service

① Overview

Requests the address resolution service middleware to update array properties. When an update of properties specified as requiring a status change notification is accepted, the address resolution service middleware will broadcast the changed values. When the properties selected to be updated are not specified as requiring a status change notification, notifications will be made as specified in the notification method designation field.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the array properties to be updated belong.
Update (array, EPC)	The EPC values of the array properties to be updated.
Number (update, array element)	The array numbers of the array elements to be updated.
Data (update, array, EPC)	The values to be written into the array elements of the array properties to be updated.
Notification method designation	0: No notification 1: Broadcast notification 2: Notified individually to all nodes registered in the address resolution service DB 3: Notification by means of class-specific broadcasts to all classes registered in the address resolution service DB

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected

(12) Property reference request service

① Overview

Requests the address resolution service middleware to reference properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the properties to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.

④ Response data

Data name	Explanation
Processing result	0: Reference accepted 1: Reference rejected
Response (reference, EPC, number)	The number of properties that were actually referenced.
Response list (reference, EPC)	A list of the EPC values of the properties that were actually referenced.
Response list (reference, EPC, size)	A list of the sizes of the properties that were actually referenced.
Response list (reference, EPC, type)	A list of the types of the properties that were actually referenced.
Response list (reference, EPC, data)	A list of the property values referenced.

(13) Status notification method designation property update request service

① Overview

Requests the address resolution service middleware to update status notification method designation properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the status notification method designation properties to be updated

	belong.
Number (update, EPC)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be written into the properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Number (update, acceptance, EPC)	The number of properties updated.
List (update, acceptance, EPC)	A list of the EPC values updated.

(14) Status notification method designation property reference request service

① Overview

Requests the address resolution service middleware to reference status notification method designation properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the status notification method designation properties to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.

④ Response data

Data name	Explanation
Processing result	0: reference accepted 1: reference rejected
Response (reference, number of EPC values)	The number of properties that were actually referenced.
Response list (reference, EPC)	A list of the EPC values of the properties that were actually referenced
Response list (reference, EPC, data)	A list of the property values that were actually referenced.

(15) Set control method designation property update request service

① Overview

Requests the address resolution service middleware to update Set control method designation properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the Set control method designation properties to be updated belong.
Number (update, EPC)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be written into the properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Number (update, acceptance, EPC)	The number of properties updated.
List (update, acceptance, EPC)	A list of the EPC values updated.

(16) Set control method designation property reference request service

① Overview

Requests the address resolution service middleware to reference Set control method designation properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the Set control method designation properties to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.

④ Response data

Data name	Explanation
Processing result	0: Reference accepted 1: Reference rejected
Response (reference, number of EPC values)	The number of properties that were actually referenced.
Response list (reference, EPC)	A list of the EPC values of the properties that were actually referenced
Response list (reference, EPC, data)	A list of the property values referenced.

(17) “Action link method designation” property update request service

① Overview

Requests the address resolution service middleware to update “Action link method designation” properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the “action link method designation” properties to be updated belong.
Number (update, EPC)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be written into the properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Number (update, acceptance, EPC)	The number of properties updated.
List (update, acceptance, EPC)	A list of the EPC values updated.

(18) “Action link method designation” property reference request service

① Overview

Requests the address resolution service middleware to reference “Action link method designation” properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the “action link method designation” properties to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.

④ Response data

Data name	Explanation
Processing result	0: Reference accepted 1: Reference rejected
Response (reference, number of EPC values)	The number of properties that were actually referenced.
Response list (reference, EPC)	A list of the EPC values of the properties that were actually

	referenced
Response list (reference, EPC, data)	A list of the property values referenced.

(19) “Trigger link method designation” property update request service

① Overview

Requests the address resolution service middleware to update “trigger link method designation” properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the “trigger link method designation” properties to be updated belong.
Number (update, EPC)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be written into the properties to be updated.

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update rejected
Number (update, acceptance, EPC)	The number of properties updated.
List (update, acceptance, EPC)	A list of the EPC values updated.

(20) “Trigger link method designation” property reference request service

① Overview

Requests the address resolution service middleware to reference “trigger link method designation” properties.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the home node object to which the “trigger link method designation” properties to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.

④ Response data

Data name	Explanation
Processing result	0: Reference accepted 1: Reference rejected
Response (reference, number of EPC values)	The number of properties that were actually referenced.
Response list (reference, EPC)	A list of the EPC values of the properties that were actually referenced
Response list (reference, EPC, data)	A list of the property values referenced.

3.7.6 Non-home node object management services

(1) "Automatic non-home node object generation" request service

① Overview

Requests automatic generation of an object for another node. This service can only generate objects for the functions registered in the address resolution service DB. Properties are generated within the middleware using a communication-based method after performing an intra-middleware check that is also communication-based.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the other node object to be generated.

④ Response data

Data name	Explanation
Processing result	0: Generation accepted 1: Generation rejected

(2) "Automatic non-home node object generation" result notification service

① Overview

Reports the result of automatic generation of an object for another node.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Object generation result	0: Generation successfully completed 1: Generation failed
Control ID	The control ID of the other node object that was generated.

④ Response data

None

(3) “Property designation-based non-home node object generation” request service

① Overview

Generates, for another node, an object that has the specified properties. This service can only generate objects for the functions registered in the address resolution service DB.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the other node object to be generated.
Number of properties	The number of properties contained in the object to be generated.
EPC code list	A list of the EPC values of the properties.
List (EPC, size)	A list of the sizes of the properties.
List (EPC, type)	A list of the types of the properties.
Number of array properties	The number of array properties that the object to be generated is going to have.
List (array, EPC, code)	A list of the EPC values of the array properties.
List (array, EPC, size)	A list of the sizes of the array properties.
List (array, EPC, type)	A list of the types of the array properties.
List (number of array elements)	A list of the number of array elements of the array properties.

④ Response data

Data name	Explanation
Processing result	0: Generation successfully completed 1: Generation failed
Control ID	The control ID of the other node object that was generated.

(4) Non-home node object deletion request service

① Overview

Requests the deletion of an object of another node.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the object to be deleted.

④ Response data

Data name	Explanation

Processing result	0: Deletion successfully completed 1: Deletion rejected

(5) Property update message transmission request service

① Overview

Requests the address resolution service middleware to send a message to update the values of the specified properties. The properties to be updated must be located in an object of a node other than the home node. When a request is made to send a message to update the values of properties that requires a response, the properties will not be available for another update until a response message is received or the timeout period expires. The values of the corresponding properties in the object(s) in the node(s) other than the home node will be updated by reading out the values of the actual pieces of equipment or upon receipt of a notification.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the properties selected to be updated belong.
Number (update, EPC)	The number of properties to be updated.
List (update, EPC)	A list of the EPC values of the properties to be updated.
List (update, EPC, data)	A list of the values to be updated.
Broadcast request flag	A flag specifying whether to only update properties for the functions specified by control IDs or to perform a broadcast to all functions of the same type (in terms of the X1 and X2 values of the EOJ) as those of the specified functions. 0: Unicast (specified functions only) 1: Intra-domain broadcast (functions of the same type located within the domain) 2: Broadcast over the home subnet (functions of the same type located within the home subnet).
Response request flag	A flag specifying whether or not to request the function in question to return a response. 0: Response required 1: Response not required

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update request rejected

(6) Array property update message transmission request service

① Overview

Requests the address resolution service middleware to send a message to update the values of

the specified array properties. The properties to be updated must be located in an object of a node other than the home node. When a request is made to send a message to update the values of properties that requires a response, the properties will not be available for another update until a response message is received or the timeout period expires. The values of the corresponding properties in the object(s) of the node(s) other than the home node will be updated by reading out the values of the actual pieces of equipment or upon receipt of a notification.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the array properties selected to be updated belong.
Size (update, array, EPC)	The EPC values of the array properties to be updated.
Number (update, array element)	The element number for the array elements to be updated.
Data (update, array, EPC)	The values to be updated.
Broadcast request flag	A flag specifying whether to only update properties for the functions specified by control IDs or to perform a broadcast to all functions of the same type (in terms of the X1 and X2 values of the EOJ) as those of the specified functions. 0: Unicast (specified functions only) 1: Intra-domain broadcast (functions of the same type located within the domain) 2: Broadcast over the home subnet (functions of the same type located within the home subnet).
Response request flag	A flag specifying whether or not to request the function in question to return a response. 0: Response required 1: Response not required

④ Response data

Data name	Explanation
Processing result	0: Update accepted 1: Update request rejected

(7) Property reference message transmission request service

① Overview

Requests the address resolution service middleware to send a message to reference the values of the specified properties. The properties to be referenced must be located in an object of a node other than the home node. When a request is made to send a message to reference the values of properties, the properties will not be available for another reference until a response message is received or the timeout period expires. The values of the corresponding properties in the object(s) in a node(s) other than the home node will be updated upon receipt of a

reference response.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the properties selected to be referenced belong.
Number (reference, EPC)	The number of properties to be referenced.
List (reference, EPC)	A list of the EPC values of the properties to be referenced.
Broadcast request flag	A flag specifying whether to only reference properties for the functions specified by control IDs or to perform a broadcast to all functions of the same type (in terms of the X1 and X2 values of the EOJ) as those of the specified functions. 0: Unicast (specified functions only) 1: Intra-domain broadcast (functions of the same type located within the domain) 2: Broadcast over the home subnet (functions of the same type located within the home subnet).

④ Response data

Data name	Explanation
Processing result	0: Reference accepted 1: Reference request rejected

(8) Property change notification service

① Overview

Notifies the application software that the properties of an object of another node have been changed by a notification or reference response (reception). When the application software does not reference the changed properties after receiving a property change notification, no property change notification will follow the reception of succeeding notifications and reference responses.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the non-home node object whose properties have changed.
Number (change, EPC)	The number of properties changed.
List (change, EPC)	A list of the EPC values of the changed properties.

④ Response data

Data name	Explanation
Processing result	0: Notification accepted 1: Notification rejected

(9) Timeout notification service

① Overview

Notifies the application software that no update response or reference response was received during the timeout period.

② Direction

Address resolution service middleware → application software

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the properties determined to require a timeout notification belong.
Number (notification, EPC)	
List (notification, EPC)	
Broadcast request flag	0: Unicast 1: Broadcast
Control ID of the recipient of the notification	

④ Response data

Data name	Explanation
Processing result	0: Notification accepted 1: Notification rejected

(10) Property reference request service

① Overview

References the values of the specified properties by reading out the properties of an object of a non-home node.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the properties selected to be referenced belong.
Number (reference, EPC)	
List (reference, EPC)	
List (reference, EPC, size)	
List (reference, EPC, type)	

④ Response data

Data name	Explanation
Processing result	0: Accepted 1: Rejected
Number (reference, EPC)	
List (reference, EPC)	
List (reference, EPC, size)	
List (reference, EPC, type)	
List (reference, EPC, data)	

(11) Array property reference request service

① Overview

References the values of the specified array properties by reading out the properties of an object of a non-home node.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation
Control ID	The control ID of the function to which the properties selected to be referenced belong.
EPC (reference)	
Size (reference, EPC)	
Type (reference, EPC)	
Array element number (reference)	

④ Response data

Data name	Explanation
Processing result	0: Accepted 1: Rejected
EPC (reference)	
Size (reference, EPC)	
Type (reference, EPC)	
Data (reference, EPC)	

(12) “Changed property list” request service

① Overview

Requests a list of the properties that have not been referenced after a value change.

② Direction

Application software → address resolution service middleware

③ Input data

Data name	Explanation

Control ID	The control ID of the other node object for which a property list is requested.

④ Response data

Data name	Explanation
Processing result	0: Accepted 1: Rejected
List (value change, EPC)	

(13) “Changed object list” request service

① Overview

Requests a list of the objects having properties that have not been referenced after a value change.

② Direction

Application software → address resolution service middleware

Input data

None

④ Response data

Data name	Explanation
Processing result	0: Accepted 1: Rejected
List (value change, object, control ID)	

Chapter 4 File Transfer Service Middleware

This chapter defines the requirements for the file transfer service middleware that transfers files stored in equipment objects located in nodes as well as the requirements for the file reception service class and file transmission service class. The application software can transfer files between ECHONET nodes by using the file transfer service middleware. The middleware is intended for transfers of files of up to several kilobytes in size and is used to transfer still pictures taken with cameras, recipe data for microwave ovens, and other similar types of data.

4.1 Functions of the File Transfer Service Middleware

The required functions for the file transfer service middleware are listed below together with brief explanations thereof.

① PUSH-type file transmission function

A function that allows a transmitting node having a file transmission service object to autonomously transmit a file to a receiving node having a file reception service object in response to an instruction from an internal application software program.

② PULL-type file transmission function

A function that allows a transmitting node having a file transmission service object to transmit a file to a receiving node having a file reception service object in response to a file transmission request from the receiving node.

③ PUSH-type file reception function

A function that allows a receiving node having a file reception service object to receive files that are autonomously transmitted by a transmitting node having a file transmission service object by means of the PUSH-type file transmission function (See ① above).

④ PULL-type file reception function

A function that allows a receiving node having a file reception service object to send a file transmission request to a transmitting node having a file transmission service object in response to an instruction from an internal application software program and receive the file transmitted by the transmitting node by means of the PULL-type file transmission function (See ② above).

File transfers must be achieved by either the PUSH-type or PULL-type functions. The file transfer service middleware must be implemented in such a way that it can control the PUSH-type functions, PULL-type functions or both according to the characteristics of the

application software.

4.2 File Reception Service Class

This section specifies the requirements for the file reception service class, so that the operation status of the file reception service, equipment object data used for the file reception service and split data size for split data transmission can be published within the domain. An instance of this service class can only perform file reception processing for a single instance of the file transmission service class (See 4.3) of the transmitting node. The object codes for the file reception service class are as follows:

Class group code: 0x0D
Class code: 0x02
Instance code: 0x01 to 0x7F

Table 4.1 lists the properties of the file reception service class.

Table 4.1 List of Properties of the File Reception Service Class

Property name	EPC	Contents of property Range (decimal notation)	Data type	Data size	Unit	Access rule	Required	Announcement on status change	Remarks
Object data for PUSH-type reception	0xC0	Indicates the object data for PUSH-type reception that is receivable. Maximum size for files (in bytes): unsigned short*1 + Number of pieces of file storage data (= N): unsigned char*1 + [File storage equipment data: unsigned char*3 + number of files: unsigned short] *N	–	243 bytes	–	Get	※		
Data split data	0xD0	Maximum number of bytes (EDT) the receiving side can receive 1 to 247	Unsigned short	2 bytes	–	Get			

※ When the PUSH-type file reception function is to be used, the “object data for PUSH-type reception” property must be implemented.

4.2.1 Operation status

Specifies the operation status (ON/OFF) of the file reception service function and acquires the current operation status. The value is 0x30 when the operation status is ON (i.e. the service is active) and 0x31 when the operation status is OFF (i.e. the service is not active). When the operation status is OFF, access to other properties is not guaranteed.

4.2.2 Object data for PUSH-type reception

The property for PUSH-type reception that gives the maximum size (in bytes) for files and file source equipment object and property data. When the PUSH-type file reception function is to be used, this property must be implemented. There must be no overlapping of file source equipment data values among instances.

<Detailed explanation of the components>

- Maximum size for files: The maximum number of bytes a file can contain (after assembly) in order for it to be received by means of the PUSH-type file reception function.
- Number of pieces of file storage data: The number of pieces of file storage data (max. = 40).
- Piece of file storage data: A "file storage equipment data" pair (i.e. data on each piece of equipment that stores PUSH-receivable files) and the "number of files" (i.e. the number of receivable files for each piece of equipment that stores PUSH-receivable files).

File storage equipment data: The class group code, class code and property code of a piece of file storage equipment

Number of files: The number of files that can be received and retained (each piece of file storage equipment).

0 to 0xFFFE = number of files, 0xFFFF = indicates that there is no limit.

<Composition>

The composition of this property is as shown in Fig. 4.1.

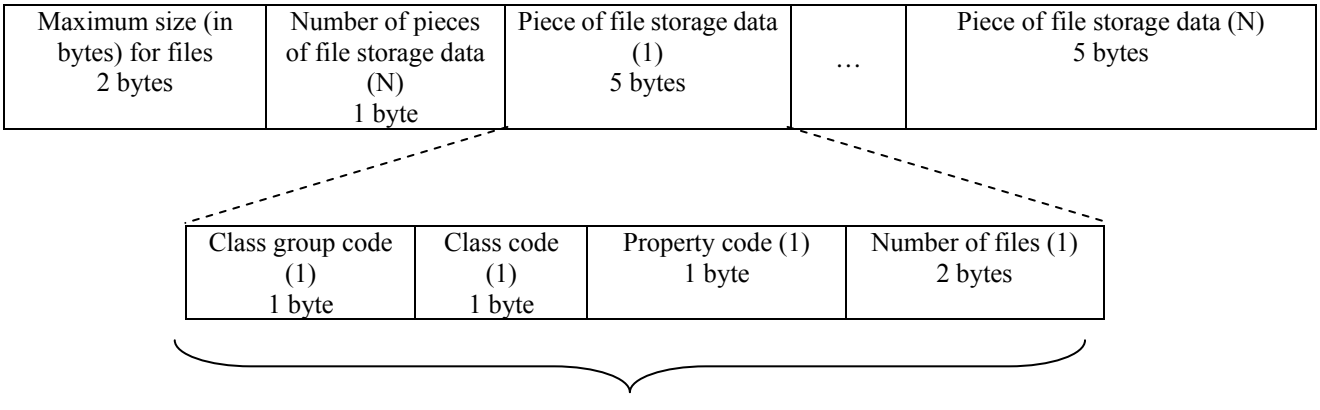


Fig. 4.1 Composition of the "Object Data for PUSH-type Reception" Property

4.2.3 Data split data

The property that indicates the maximum receivable split data size. The transmitting node reads the property value, compares it with the home node's maximum split data size and adopts the smaller one to split the original file into split data pieces for transmission.

<Detailed explanation of the component>

・Indicates the maximum number of bytes (EDT) the receiving side can receive.

<Timing of reading, writing, notification, etc.>

・The reading is performed before the transmitting side makes the transmission data notification.

4.3 File Transmission Service Class

This section specifies the requirements for the file transmission service class, so that the operation status of the file transmission service, equipment object data used for the file transmission service and transmission file data can be published within the domain. An instance of this service class can only perform file transmission processing for a single instance of the file reception service class (See 4.2) of the receiving node. The object codes for the file transmission service class are as follows:

Class group code: 0x0D
Class code: 0x03
Instance code: 0x01 to 0x7F

Table 4.2 lists the properties of the file transmission service class.

Table 4.2 List of Properties of the File Transmission Service Class (1/2)

Property name	EPC	Contents of property Range (decimal notation)	Data type	Data size	Unit	Access rule	Required	Announcement on status change	Remarks
Object data for PULL-type transmission	0xC1	Indicates the object data for PULL-type transmission that is transmittable. Maximum size for files (in bytes): unsigned short*1 + Number of pieces of file provider data: unsigned char*1 + [File provider equipment data: unsigned char*4 + number of files: unsigned short] *N	–	243 bytes	–	Get	※1		

※ 1 When the PULL-type file transmission function is to be used, the “object data for PULL-type transmission” property must be implemented.

Table 4.2 List of Properties of the File Transmission Service Class (2/2)

Property name	EPC	Contents of property Range (decimal notation)	Data type	Data size	Unit	Access rule	Required	Announcement on status change	Remark
Transmission and reception setting	0xD0	A transmission trigger or the transmission or reception result. Normal completion = 0x00 Forced termination = 0x01 Transmission start = 0x20	Unsigned char	1 byte		Set	○		
Transmission status	0xD1	Indicates the current transmission status. Ready = 0x00 Transmission in progress = 0x20	Unsigned char	1 byte	–	Get	○		
File provider data	0xD2	Object data for transmission. File provider equipment data: unsigned char*4 + File number: unsigned short	–	6 bytes	–	Anno Set	○ ※2		
Transmission file data	0xD3	Data about the file to be transmitted. Name: unsigned char*8 + Attribute: unsigned char*3 + Date: unsigned char*7 + File size (in bytes): unsigned short*1 + Check code: unsigned char*1	–	21 bytes	–	Anno	○		
Data to be transmitted	0xD4	Split transmission of the file to be transmitted. [Array element No.: unsigned short*1 + total number of array elements: unsigned short*1 + split data size: unsigned short*1 + split data pieces: unsigned char* (split data size)] * the number of split data pieces	–	(6 + split data size) x the number of split data pieces	–	Anno M GetM	○		

※2 When the PULL-type file transmission function is to be used, the Set access rule must be implemented in the "object data" property.

4.3.1 Operation status

Specifies the operation status (ON/OFF) of the file transmission service function and acquires the current operation status. The value is 0x30 when the operation status is ON (i.e. the service is active) and 0x31 when the operation status is OFF (i.e. the service is not active). When the operation status is OFF, access to other properties is not guaranteed.

4.3.2 Object data for PULL-type transmission

A property for PULL-type file transmission that gives the maximum size for files (in bytes) and file provider equipment object and property data. There must be no overlapping of file provider equipment data values among instances.

When the PULL-type file transmission function is to be used, this property must be implemented.

<Detailed explanation of the components>

- Maximum size for files: The maximum number of bytes a file can contain (before splitting) in order for it to be transmitted by means of the PULL-type file transmission function.
- Number of pieces of file provider data: The number of pieces of file provider data (max. = 40).
- Piece of file provider data: A "file provider equipment data" pair (i.e. data on each piece of equipment that provides files that can be transmitted using the PULL-type file transmission function) and "the number of files" (i.e. the number of files each piece of file provider equipment has).

File provider equipment data: A combination of an "equipment object represented by the class group code, class code and instance code of a piece of file provider equipment that provides files that can be transmitted using the PULL-type file transmission function" and the "property code."

Number of files: The number of files that each piece of file provider equipment has.

<Timing of reading, writing, notification, etc.>

- The "number of files" value of this property cannot be changed when the "transmission status" property value is "transmission in progress."

<Composition>

・The composition of this property is as shown in Fig. 4.2.

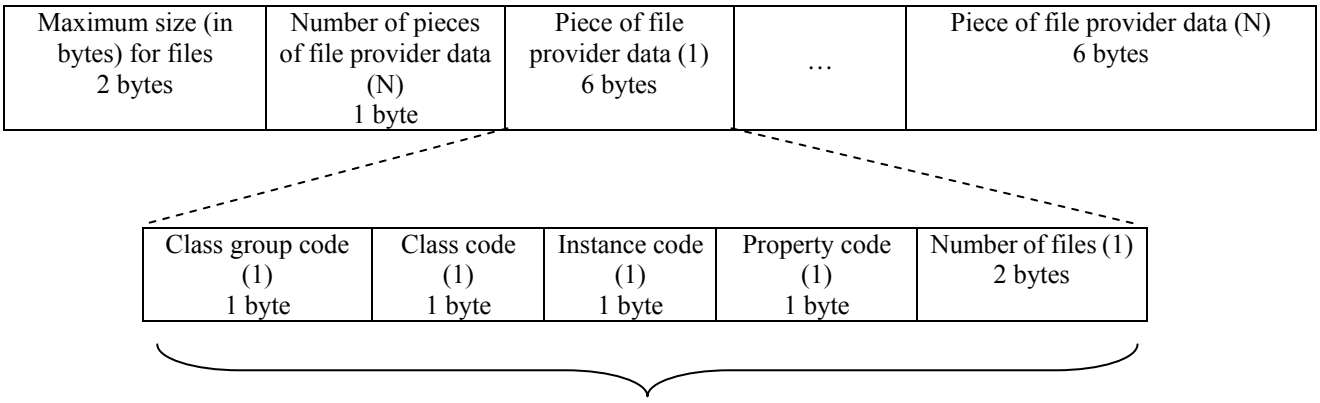


Fig. 4.2 Composition of the "Object Data for PULL-type Transmission" Property

4.3.3 Transmission and reception setting

The property used by the receiving node to give a transmission trigger or the transmission or reception result. When a value is written into this property, it will be immediately reflected in the "transmission status" property.

<Property value>

・Normal completion = 0x00 (The transmission has been completed successfully.)
・Forced termination = 0x01 (The transmission has been terminated by the receiving node.)
・Transmission start = 0x20 (Commencement of transmission processing.)

<Timing of reading, writing, notification, etc.>

・When a value is written into this property, it will be immediately reflected in the "transmission status" property. If the "transmission start" value is written into this property, the "transmission status" property will be set to "transmission in progress." If either of the two other values is written into this property, the "transmission status" property will be set to "ready." When the "transmission status" property value is "transmission in progress," only the receiving node that has written the "transmission start" value into this property can write a value into this property.
・Transmission processing can be interrupted during the transmission of data by writing the "normal completion" or "forced termination" value into this property.

4.3.4 Transmission status

This property indicates whether or not the instance is performing transmission processing. When the property value is “transmission in progress,” no change can be made to the “transmission file data” or “transmission data” property.

<Property value>

・Ready = 0x00 (On standby.)

・Transmission in progress = 0x20 (Transmission processing is being performed.)

<Timing of reading, writing, notification, etc.>

・When a value is written into the “transmission and reception setting” property, it will be reflected in the “transmission status” property. Table 4.3 shows how this property changes in value as a result of each of the 3 values being written into the “transmission and reception setting” property.

Table 4.3 “Transmission Status” Property Value Changes Following “Transmission and Reception Setting” Property Value Changes, etc.

“Transmission status” value	Events Causing a Shift to “Ready” or “Transmission in Progress”
Ready (0x00)	1. Immediately after a node startup. 2. The writing into the “transmission and reception setting” property of 0x00 (normal completion) or 0x01 (forced termination) (immediately after Set). 3. Upon expiration of the timeout period.
Transmission in progress (0x20)	1. The writing into the “transmission and reception setting” property of 0x20 (transmission start)(immediately after Set).

4.3.5 File provider data

This property is used to specify the piece of file provider equipment data to be used for the transmission processing. In a PUSH-type file transfer, the transmitting node reports the value of this property to the receiving node to indicate the transmitting node’s intention to send the files stored in the piece of file provider equipment identified by this property. In a PULL-type file transfer, the receiving node writes an appropriate value into the “file provider data” property of the transmitting node to request the transmitting node to prepare (copy) the files stored in the piece of file provider equipment. The transmitting node then reports the value of the “file provider data” property to the receiving node regardless of whether or not the attempt to prepare (copy) the file is successful.

<Property value>

・Piece of file provider equipment data: The piece of file provider equipment data to be used

for the transmission that is included in the object data for the appropriate transmission type.

・File number: The number of the file of the piece of file provider equipment. The numbering method shall be implementation-dependent (transmitting node). However, when the property indicated in the piece of file provider equipment data is an array property, the file numbering must be the same as the array element numbering.

<Timing of reading, writing, notification, etc.>

・In the case of PUSH-type transmission, the value of this property is reported to the file reception service object of the receiving node.

・In the case of PULL-type transmission, the receiving node writes the appropriate values into this property. Upon reception of this write request message, the file provider data will be reported back to the file reception service object of the receiving node. If an incorrect value is written, the file provider data will be reported with the file number set to 0xFFFF.

<Composition>

The composition of this property is as shown in Fig. 4.3.

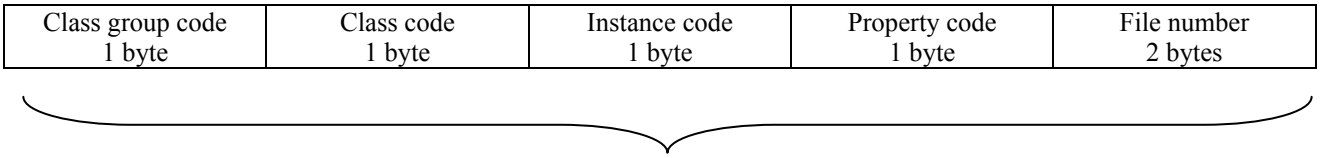


Fig. 4.3 Composition of the "File Provider Data" Property

4.3.6 Transmission file data

This property indicates the name, attribute, date, size (in bytes), etc. of the file to be transmitted. This property is used to obtain information for selecting the file to transmit in a PULL-type transfer.

<Detailed explanation of the components>

・Indicates the name, date, size (in bytes) and check code of the file to be transmitted.

・File name: A file name consisting of eight ASCII characters. The file name is stored in such a way that no space is left before the first bit and the remaining space after the last bit is padded with "0x00"s.

・Attribute: A file attribute name consisting of three ASCII characters. The file attribute name is stored in such a way that no space is left before the first bit and the remaining space after the last bit is padded with "0x00"s.

・Date: Indicates the file date in "Y1Y2/MN/DD HH:MM:SS" format.

Y1: 0x13 to 0x63 (19 to 99)
Y2: 0x00 to 0x63 (00 to 99)
MN: 0x01 to 0x0C (1 to 12)
DD: 0x01 to 0x1F (1 to 31)
HH: 0x00 to 0x17 (0 to 23)
MM: 0x00 to 0x3B (0 to 59)
SS: 0x00 to 0x3B (0 to 59)

In the case of a file with no date, 0x00 will be written into all the sub-fields of the date field. When Y1 is 0x00, the date data will be treated as invalid data.

・File size (in bytes): The total number of bytes contained in the file to be transmitted.
・Check code: 2's complement of the total number of bytes contained in the file to be transmitted.

<Timing of reading, writing, notification, etc.>

・When "0x20" is written into the "transmission and reception setting" property and the "transmission status" property is changed to "transmission in progress," the transmitting node will report the values of this property to the receiving node.
・If the piece of file provider equipment data of the "file provider data" property is not found among the pieces of file provider equipment data of the "object data for PULL-type transmission" property (i.e. the piece of file provider data is inappropriate), the contents of the "transmission file data" property will be reported with "0" written into the file size field.

<Composition>
The composition of this property is as shown in Fig. 4.4.

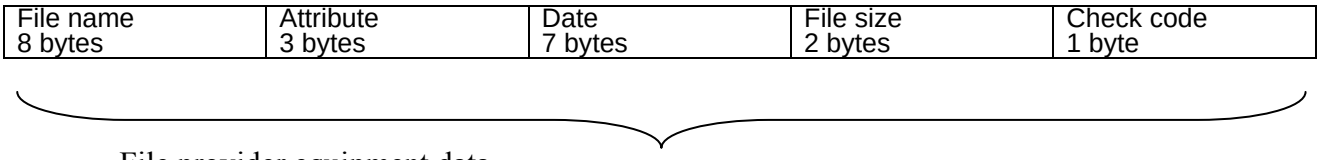


Fig. 4.4 Composition of the "Transmission File Data" Property

4.3.7 Data to be transmitted

This property indicates, in the form of an array, the split data pieces of the size specified as the split data size into which the file to be transmitted is split. The array elements must be contiguous.

<Detailed explanation of the components>

・Array element number: The number of each split data piece. The split data pieces are numbered in the order of their original locations in the "file to be transmitted," beginning with the head split data piece.

・Total number of arrays: The total number of array elements to be sent in split transmission.

・Split data size: The size (in bytes) of the split data pieces. All split data pieces have the same size except for the last array element of an array with two or more array elements and the EDT size is equal to "split data size + 6."

・Split data pieces: The split data pieces into which the file to be transmitted is split. The last array element has the same EDT size as the other array elements. After the split data size, zero padding is applied.

<Timing of reading, writing, notification, etc.>

・After reception of a "transmission file data" notification response from the receiving node, the transmitting node reports the data to be transmitted.

・If the "transmission status" property is not "transmission in progress," an NG response will be returned.

<Composition>

The composition of this property is as shown in Fig. 4.5

Array element number	Total number of array elements	Split data size	Split data piece
2 bytes	2 bytes	2 bytes	"split data size" bytes
		•••	Total number of array elements
Array element number	Total number of array elements	Split data size	Split data piece
2 bytes	2 bytes	2 bytes	"split data size" bytes

Fig. 4.5 Composition of the "Data to Be Transmitted" Property

4.4 File Transfer Sequences

This section specifies the communication sequences used to transfer files using the file transmission and reception service objects.

Because there are basically two types of file transfers, namely PUSH-type file transfers whereby transmitting nodes autonomously send files to receiving nodes and PULL-type file transfers whereby receiving nodes take out files by sending transmission request triggers to transmitting nodes, two different file transfer processing sequences are specified.

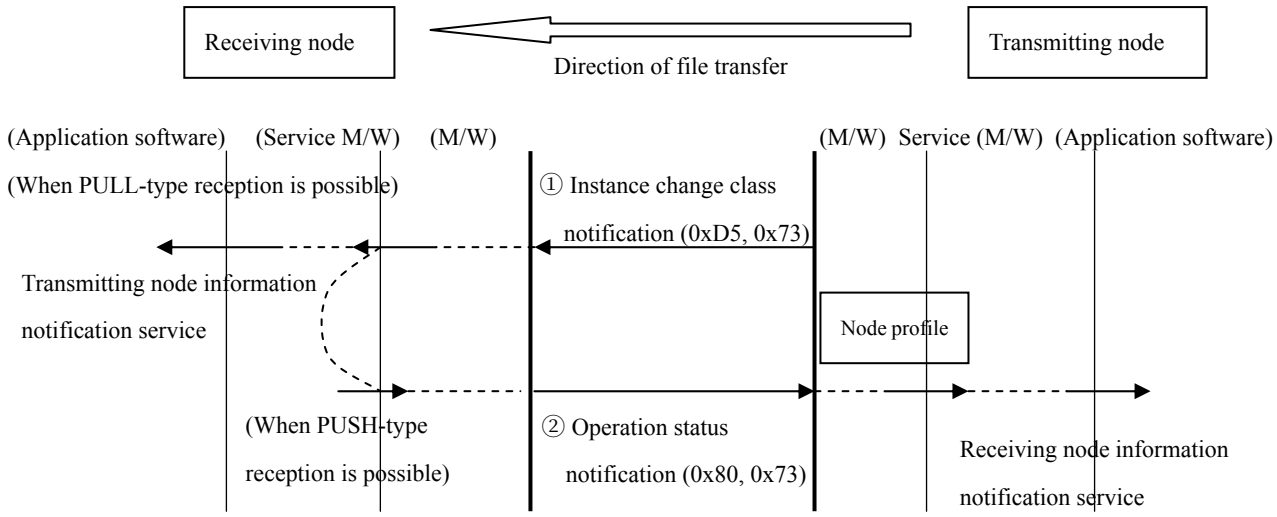
The interface for communication between the services is specified in “4.6 File Transfer Service Interface.”

4.4.1 Equipment startup sequence

In a PUSH-type file transfer, the transmitting node takes advantage of the instance change class notification process of the node startup sequence to acquire the EA value of the receiving node. In a PULL-type file transfer, the receiving node takes advantage of the instance change class notification process of the node startup sequence to acquire the EA value of the transmitting node.

(1) Startup sequence 1

Figure 4.6 shows the sequence for cases in which the receiving node starts up before the transmitting node.



[Note 1] The figures in parentheses indicate the EPC and ESV values of the messages.

[Note 2] □ indicates the node profile object.

Fig. 4.6 Sequence for Cases Where the Transmitting Node Starts Up After the Receiving Node

① Instance change class notification (Transmitting node → receiving node)

[Processing in the transmitting node]

The transmitting node makes an instance change class notification to the receiving node in accordance with the startup sequence (See Part 2).

[Processing in the receiving node]

Upon reception of the notification message, the file transfer service middleware of the receiving node reports the EA value of the transmitting node (which has a file transmission service object) to the application software (transmitting node information notification service). The receiving node's application software manages the EA value of the transmitting node.

② Operation status notification (Receiving node → transmitting node)

[Processing in the receiving node]

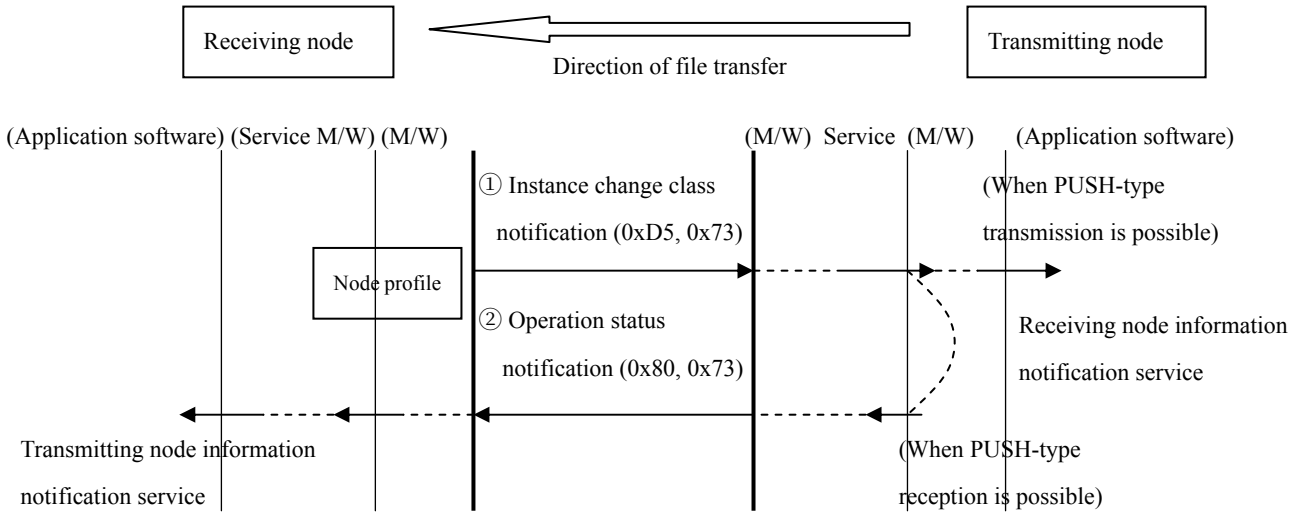
Upon reception of an instance change class notification (①), the receiving node's file transfer service middleware must report to the transmitting node the operation status of all PUSH-type file reception-capable file reception service objects. (Compulsory)

[Processing in the transmitting node]

Upon reception of the notification message, the file transfer service middleware of the transmitting node reports the EA value of the receiving node (which has a file reception service object) to the application software (receiving node information notification service). The transmitting node's application software manages the EA value of the receiving node.

(2) Startup sequence 2

Figure 4.7 shows the sequence for cases in which the transmitting node starts up before the receiving node.



[Note 1] The figures in parentheses indicate the EPC and ESV values of the messages.

[Note 2] □ indicates the node profile object.

Fig. 4.7 Sequence for Cases Where the Receiving Node Starts Up After the Transmitting Node

① Instance change class notification (Receiving node → transmitting node)

[Processing in the receiving node]

The receiving node makes an instance change class notification to the transmitting node in accordance with the startup sequence (See Part 2).

[Processing in the transmitting node]

Upon reception of the notification message, the file transfer service middleware of the transmitting node reports the EA value of the receiving node (which has a file reception service object) to the application software (receiving node information notification service). The transmitting node's application software manages the EA value of the receiving node.

② Operation status notification (Transmitting node → receiving node)

[Processing in the transmitting node]

Upon reception of an instance change class notification (①), the transmitting node's file transfer service middleware must report to the receiving node the operation status of all

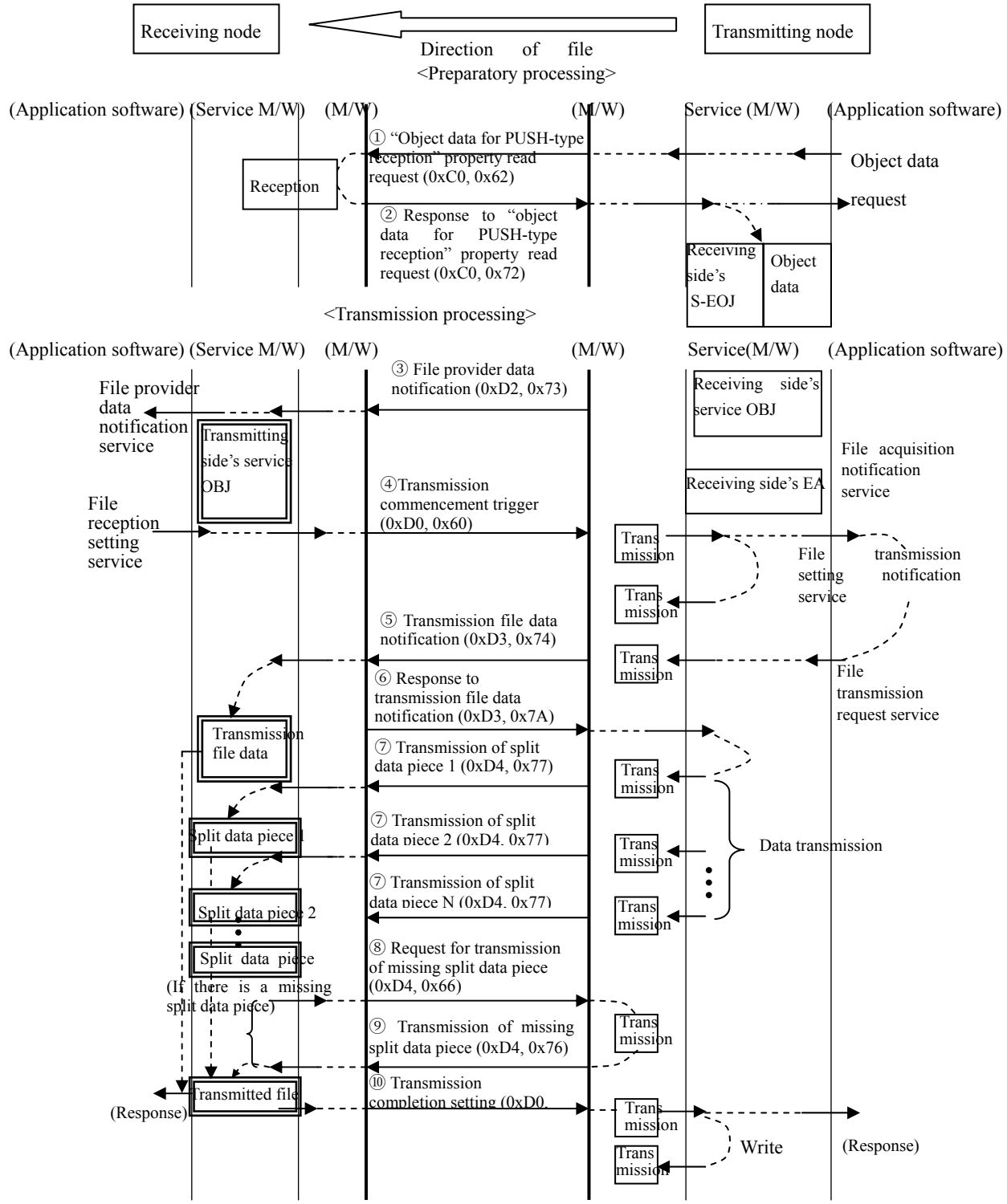
PULL-type file transmission-capable file transmission service objects. (Compulsory)

[Processing in the receiving node]
Upon reception of the notification message, the file transfer service middleware of the receiving node reports the EA value of the transmitting node (which has a file transmission service object) to the application software (transmission node information notification service). The receiving node's application software manages the EA value of the transmitting node.

4.4.2 PUSH-type file transfer sequence

For cases in which the transmitting node autonomously transmits a file, the transmitting node requests the receiving node to report the contents of the "object data for PUSH-type reception" property, reports, as necessary, the contents of the "file provider data" property to the receiving node and then reports the contents of the "transmission file data" property to the receiving node. After this, the transmitting node transmits the file in split transmission.

Figure 4.8 shows a typical communication sequence.



[Note 1] The figures in parentheses indicate the EPC and ESV values of the messages.

[Note 2] "Reception" and "Transmission" indicate the file reception and transmission service objects, respectively. □ indicates data that must be stored in the file transfer service M/W.

Fig. 4.8 PUSH-type File Transfer Sequence

<Preparatory processing>

① “Object data for PUSH-type reception” property read request (Transmitting node → receiving node)

[Processing in the transmitting node]

The transmitting node’s application software sends an “object data for PUSH-type reception” property read request to the receiving node’s reception service object (object data request service). The transmitting node’s file transfer service middleware may assume that two or more reception service objects are implemented in the receiving node and make a general broadcast to all instances.

② Response to “object data for PUSH-type reception” property read request (Receiving node → transmitting node)

[Processing in the receiving node]

Upon reception of an “object data for PUSH-type reception” property read request (①), all reception service objects in the receiving node respond by transmitting the contents of the “object data for PUSH-type reception” property.

[Processing in the transmitting node]

Upon reception of this response message, the transmitting node’s file transfer service middleware stores, as the receiving node’s file reception service object value, the SEOJ value contained in the message in such a way that it is linked with the object data for PUSH-type reception, and returns the response to the object data request service to the application software.

<Transmission processing>

③ File provider data notification (Transmitting node → receiving node)

[Processing in the transmitting node]

The transmitting node’s application software sets the file provider data in the file transfer service middleware (file acquisition notification service).

In response, the file transfer service middleware performs a file provider data notification (transmitting node → receiving node).(Compulsory)

During this process, the file transfer service middleware determines the SEOJ and DEOJ values of the notification message as follows:

SEOJ: A file transmission service object is randomly selected to be used as the point of contact for the transmission.

DEOJ: A file reception service object that can handle the file provider equipment data reported by the file acquisition notification service is selected from the receiving node's file reception service objects (which were acquired in ②).

[Processing in the receiving node]
The receiving node's file transfer service middleware stores the SEOJ value contained in this notification message as the transmitting node's file transmission service object (file provider data notification service).

④ Transmission commencement trigger (Receiving node → transmitting node)
[Processing in the receiving node]
Upon reception of the file provider data notification (③), the receiving node's application software sets, in the file transfer service middleware, the result of the judgment as to whether or not the file is receivable (file reception setting service).

The file transfer service middleware of the receiving node sends a transmission commencement trigger to set the "transmission and reception setting" property to "0x20" (transmission start) to the sender of the file provider data notification (file transmission service object), so that it can receive the file. If the file specified by the file provider data is not acquirable or if the receiving node chooses not to receive the file, a write request will be made to set the "transmission and reception setting" property to "0x01" (forced termination). (Compulsory)

[Processing in the transmitting node]
The transmitting node's file transfer service middleware reports the result of the writing of the value into the receiving node's "transmission and reception setting" property to the application software (file transmission setting notification service).

The transmitting node's file transfer service middleware writes the appropriate value into the "transmission status" property according to the value written into the "transmission and reception setting" property. (Compulsory)

⑤ Transmission file data notification (Transmitting node → receiving node)
[Processing in the transmitting node]
Upon reception of the file transmission setting service's notification, the transmitting node's application software registers, in the file transfer service middleware, the file to be transmitted and the transmission file data (file transmission request service).

In response to the writing of the above-mentioned data, the file transfer service middleware

reports the transmission file data to the receiving node's file reception service object. (Compulsory)

⑥ Response to transmission file data notification (Receiving node → transmitting node)

[Processing in the receiving node]

The receiving node sends a response to the notification back to the transmitting node.

⑦ Transmission of split data pieces (Transmitting node → receiving node)

[Processing in the transmitting node]

Upon reception of the response to the transmission file data notification (⑥), the transmitting node's file transfer service middleware splits the file to be transmitted into split data pieces of the specified split data size, specifies the receiving node's EA value and file reception service object, and sends the split data pieces in sequence in ascending order, beginning with the one with the smallest array element number. (Compulsory)

⑧ Request for transmission of missing split data piece (Receiving node → transmitting node)

[Processing in the receiving node]

If there is a split data piece that is not properly received, the receiving node's file transfer service middleware makes a request to the transmitting node for transmission of the missing split data piece, with the array element number of the missing split data piece specified.

⑨ Transmission of missing split data piece (Transmitting node → receiving node)

[Processing in the transmitting node]

Upon reception of the request for transmission of the missing split data piece (⑧) from the receiving node, the transmitting node's middleware sends the split data piece having the specified array element number to the receiving node.

⑩ Transmission completion setting (Receiving node → transmitting node)

[Processing in the receiving node]

If the file is successfully received, the receiving node's file transfer service middleware writes "0x00" (normal completion) into the "transmission and reception setting" property of the file transmission service object of the transmitting node. If it is necessary to terminate the transmission processing for some reason, the receiving node's file transfer service middleware writes "0x01" (forced termination) into the "transmission and reception setting" property of the file transmission service object of the transmitting node. (Compulsory)

The receiving node's file transfer service middleware combines the split data pieces

obtained in "⑦ Transmission of split data pieces" back into a single file and registers it in the application software together with the file data obtained in "⑤ Transmission file data notification." If the file is not properly received, the receiving node's file transfer service middleware reports it to the application software (response, file reception setting service).

[Processing in the transmitting node]

In response to the transmission completion setting by the receiving node, the transmitting node sets the "transmission status" property to "0x00" (ready). To provide for cases in which no transmission completion setting is made, the transmitting node shall have a timeout period of 1 minute or longer in the processing to request the split data pieces and wait for a transmission completion setting frame. When this timeout period expires, the transmitting node shall set the "transmission status" property to "0x00" (ready). The transmitting node shall be able to receive a retransmission request from the receiving node that is accepted within a period of 1 minute. (Compulsory)

When the receiving node makes the transmission completion setting or forced file transmission termination setting (⑩), the transmitting node's file transfer service middleware will report it to the application software (response, file transmission service).

Figures 4.9 and 4.10 show examples of the EDATA section of the ECHONET frame for each message type.

① "Object data for PUSH-type reception" property read request

OHD	DEOJ	EPC	ESV
1byte	3bytes	1byte	1byte
0x82	0x0D,0x02	0xC0	0x62

② Response to "object data for PUSH-type reception" property read request

OHD	SEOF	EPC	ESV	Maximum file size (in bytes)	Number of pieces of file storage data
Byte	3bytes	1byte	1byte	2 bytes	1byte(=N)
0x81	0x0D,0x02	0xC0	0x72		

Piece of file storage data (1)					
Class group code	Class code	Property	Number of files	・・・	Piece of file storage data (N)
1byte	1byte	1byte	2 bytes		5 bytes

③ File provider data notification

OHD	SEOJ	DEOJ	ESV	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x03	0x0D,0x02	0xD2	0x73

File provider equipment class group code	File provider equipment class code	File provider equipment instance code	File provider property	File number
1byte	1byte	1byte	1byte	2bytes

④ Transmission commencement trigger

OHD	SEOJ	DEOJ	EPC	ESV	Transmission and reception setting
1byte	3bytes	3bytes	1byte	1byte	1byte
0x83	0x0D,0x02	0x0D,0x03	0xD0	0x60	0x20

⑤ Transmission file data notification

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x03	0x0D,0x02	0xD0	0x74

File name	Attribute	Date	File size (in bytes)	Check code
8 bytes	3 bytes	7 bytes	2 bytes	1byte

Fig. 4.9 Examples of Messages Used in a PUSH-type File Transfer (1)

⑥ Response to transmission file data notification

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3byte	1byte	1byte
0x83	0x0D,0x02	0x0D,0x03	0xD3	0x7A

⑦ Transmission of split data pieces

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x03	0x0D,0x02	0xD4	0x77

Array element number	Total number of array elements	Split data size	Split data piece (split data size) bytes
2 bytes	2 bytes	2 bytes	

⑧ Request for transmission of missing split data piece

OHD	SEOJ	DEOJ	EPC	ESV	Specified array element number
1byte	3bytes	3bytes	1byte	1byte	2 bytes
0x83	0x0D,0x03	0x0D,0x03	0xD4	0x66	

⑨ Transmission of missing split data piece

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x03	0x0D,0x02	0xD4	0x76

Array element number	Total number of array elements	Split data size	Split data piece (split data size) bytes
2 bytes	2 bytes	2 bytes	

⑩ Transmission completion setting

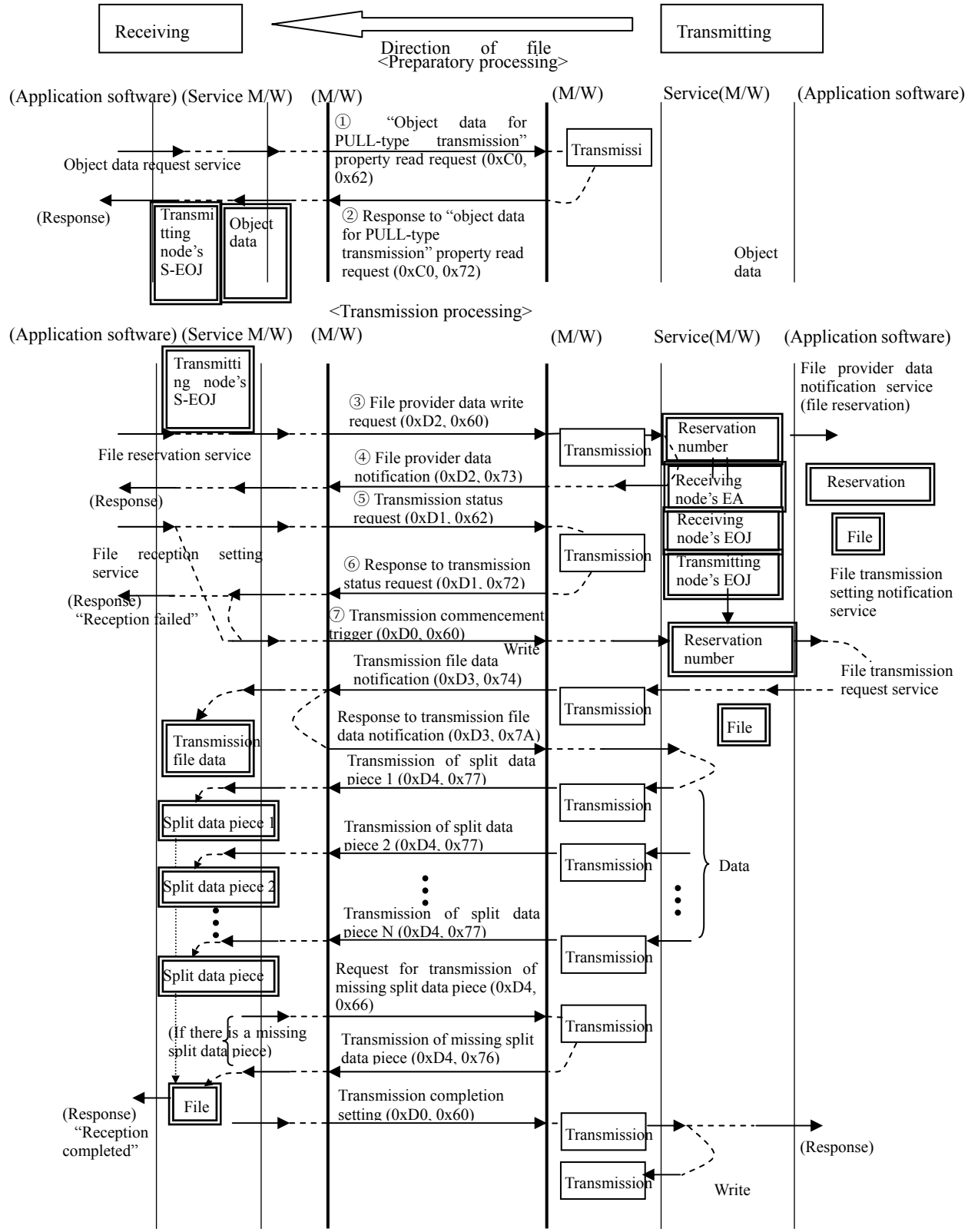
OHD	SEOJ	DEOJ	EPC	ESV	Transmission and reception setting
1byte	3bytes	3bytes	1byte	1byte	
0x83	0x0D,0x02	0x0D,0x03	0xD0	0x60	

Fig. 4.10 Examples of Messages Used in a PUSH-type File Transfer (2)

4.4.3 PULL-type file transfer sequence

For cases in which the receiving node requests the transmitting node to transmit a file, the receiving node first requests the contents of the “object data for PUSH-type transmission” property to confirm the file to be transmitted. The receiving node then obtains the transmission file data as necessary, and sends a transmission commencement trigger. In response to this trigger, the transmitting node reports the transmission file data of the specified object. The processing after this is the same as that for a PUSH-type file transfer.

Figure 4.11 shows a typical communication sequence.



[Note 1] The figures in parentheses indicate the EPC and ESV values of the messages.

[Note 2] "Reception" and "Transmission" indicate the file reception and transmission service objects, respectively. □ indicates data that must be stored in the file transfer service M/W.

Fig. 4.11 PULL-type File Transfer Sequence

<Preparatory processing>

① "Object data for PULL-type transmission" property read request (Receiving node → transmitting node)

[Processing in the receiving node]

The receiving node's application software sends an "object data for PULL-type transmission" property read request to the transmitting node (object data request service). The receiving node's file transfer service middleware may assume that two or more transmission service objects are implemented in the transmitting node and make a general broadcast to all instances.

② Response to "object data for PULL-type transmission" property read request (Transmitting node → receiving node)

[Processing in the transmitting node]

Upon reception of an "object data for PULL-type transmission" property read request (①), all transmission service objects in the transmitting node respond by transmitting the contents of the "object data for PULL-type transmission" property.

[Processing in the receiving node]

Upon reception of this response message, the receiving node's file transfer service middleware stores, as the transmitting node's file transmission service object value, the SEOJ value contained in the message in such a way that it is linked with the object data for PULL-type transmission, and returns the response to the object data request service to the application software.

<Transmission processing>

The file transfer is triggered by the transmission of a file reception request by the receiving node's application software to the file transfer service middleware and is performed in the following sequence:

③ File provider data write request (Receiving node → transmitting node)

[Processing in the receiving node]

The receiving node's application software writes into the file transfer service middleware the EA value of the transmitting node that corresponds to the transmitting node's piece of file provider equipment that contains the file it wants to reserve and the file provider data (file reservation service).

In response, the file transfer service middleware makes a request to write the appropriate value ("file provider data" property) to the transmitting node's file transmission service object. (Compulsory)

During this process, the file transfer service middleware determines the SEOJ and DEOJ values of the write request message as follows:

SEOJ: A file reception service object is randomly selected to be used as the point of contact for the reception.

DEOJ: A file transmission service object that can handle the file provider equipment data reported by the file reservation service is selected from the transmitting node's file transmission service objects (which were acquired in ②).

[Processing in the transmitting node]
The transmitting node's file transfer service middleware stores the SEA, SEOJ and DEOJ values contained in the file provider data write request message as the receiving node's EA value, file reception service object value and home node file transmission service object value, respectively. The transmitting node's file transfer service middleware also generates a file reservation number and reports it to the application software together with the file provider data (file provider data notification service). A unique reservation number is given to each combination of SEA, SEOJ and DEOJ values.

In response to the notification of the file provider data, the transmitting node's application software prepares (copies) the file identified by the equipment object and properties specified in the file provider data and manages it together with the reservation number contained in the file provider data notification message. If two or more file provider data notification messages are received consecutively that have the same reservation number, all messages except the last one will be treated as invalid messages. The copy file is discarded upon expiration of the timeout period. Timeout shall be application implementation-dependent.

④ File provider data notification (Transmitting node → receiving node)
[Processing in the transmitting node]
The transmitting node's file transfer service middleware checks whether or not the piece of file provider equipment data acquired as a result of the file provider data write request (③) is one of the pieces of file provider equipment data of the "object data for PULL-type transmission" property of the file transmission service object. If it is (i.e. the piece of file provider data is appropriate), the transmitting node's file transfer service middleware will report back the file provider data. If not (i.e. the piece of file provider data is inappropriate), or if the piece of file provider equipment data acquired as a result of the file provider data write request is one of the pieces of file provider equipment data of the "object data for PULL-type transmission" property of the file transmission service object but it is not possible to prepare (copy) the file, the file provider data will be reported back with the file

number set to 0xFFFF. (Compulsory)

[Processing in the receiving node]
Upon reception of this notification, the receiving node's file transfer service middleware returns a response to the application software (response to file reservation service).

⑤ Transmission status request (Receiving node → transmitting node)
[Processing in the receiving node]
If it is confirmed that the transmitting node has successfully prepared the file from the response to the file reservation service, the receiving node's application software will write the appropriate value in the file transfer service middleware to indicate its intention to receive the file (file reception setting service).

The receiving node's file transfer service middleware makes a request for the "transmission status" property to the transmission service object of the transmitting node that was identified as a result of the "Object data for PULL-type transmission" property read request (①).

⑥ Response to transmission status request (Transmitting node → receiving node)
[Processing in the transmitting node]
The transmitting node sends a response to the transmission status request (⑤).

⑦ Transmission commencement trigger (Receiving node → transmitting node)
[Processing in the receiving node]
Upon completion of the file reception setting service or upon reception of the transmission status response (ready), the file transfer service middleware sends a transmission commencement trigger to set the "transmission and reception setting" property to "0x20" (transmission start) to the sender node (file transmission service object). (Compulsory)

[Processing in the transmitting node]
Upon reception of the transmission commencement trigger from the receiving node, the transmitting node's file transfer service middleware changes the "transmission status" property to "0x20" (transmission in progress). (Compulsory)

The transmitting node's file transfer service middleware also performs a search to find the file reservation number using the SEA, SEOJ and DEOJ values contained in the transmission commencement trigger message and reports the reservation number to the application software (file transmission setting notification service).

[Note] The processing for the transmission file data notification and succeeding actions is

the same as that for a PUSH-type file transfer.

If the transmitting node receives another transmission commencement trigger before receiving a transmission completion setting message, it must perform the processing for the transmission file data notification and the succeeding actions.

If a write is performed to set the “transmission and reception setting” property to “0x00” (normal completion) or “0x01” (forced termination), the reservation number will be discarded.

Figures 4.12 and 4.13 show examples of the EDATA section of the ECHONET frame for each message type.

① "Object data for PULL-type transmission" property read request

OHD	DEOJ	EPC	ESV
1byte	3bytes	1byte	1byte
0x82	0x0D,0x03	0xC1	0x62

② Response to "object data for PULL-type transmission" property read request

OHD	SEOJ	EPC	ESV	Maximum file size (in bytes)	Number of pieces of file provider data
1byte	3bytes	1byte	1byte	2 bytes	1byte (=N)
0x81	0x0D,0x03	0xC1	0x72		

Piece of file provider data (1)						
Class group code	Class code	Instant	Property	Number of files	. . .	Piece of file provider data (N)
1byte	1byte	1byte	1byte	2 bytes		6 bytes

③ File provider data write request

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x02	0x0D,0x03	0xD2	0x60

File provider equipment class group code	File provider equipment class code	File provider equipment instance code	File provider property	File number
1byte	1byte	1byte	1byte	2 bytes

④ File provider data notification

OHD	SEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x03	0x0D,0x03	0xD2	0x74

File provider equipment class group code	File provider equipment class code	File provider equipment instance code	File provider property	File number
1byte	1byte	1byte	1byte	2 bytes

Fig. 4.12 Examples of Messages Used in a PULL-type File Transfer (1)

⑤ Transmission status request

OHD	DEOJ	DEOJ	EPC	ESV
1byte	3bytes	3bytes	1byte	1byte
0x83	0x0D,0x02	0x0D,0x03	0xD1	0x62

⑥ Response to transmission status request

OHD	DEOJ	DEOJ	EPC	ESV	Transmission status
1byte	3bytes	3bytes	1byte	1byte	0x00
0x83	0x0D,0x03	0x0D,0x02	0xD1	0x72	

⑦ Transmission commencement trigger

OHD	SEOJ	DEOJ	EPC	ESV	Transmission and reception setting
1byte	3bytes	3bytes	1byte	1byte	1byte
0x83	0x0D,0x02	0x0D,0x03	0xD0	0x60	0x20

Fig. 4.13 Examples of Messages Used in a PULL-type File Transfer (2)

4.5 File Transfer Service Middleware Requirements

The file transfer service middleware requirements that must be satisfied are as follows:

① A file reception service object must be implemented in all receiving nodes.
② A file transmission service object must be implemented in all transmitting nodes.
③ To achieve a PUSH-type file transfer, the "object data for PUSH-type reception" property must be implemented in the file reception service object of the receiving node.
④ To achieve a PULL-type file transfer, the "object data for PULL-type transmission" property must be implemented in the file transmission service object of the transmitting node.
⑤ The requirements specified in "4.4 File Transfer Sequences" must be satisfied.

4.6 File Transfer Service Interface

This section specifies the services to be provided by the file transfer service API and shows examples of data exchange between the application software and file transfer service middleware.

4.6.1 Services relating to PUSH-type file transmission (transmitting node)

(1) Receiving node address notification service

① Overview

Notifies the application software of the ECHONET address of the receiving node to which a file is to be transmitted in a PUSH-type file transfer.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Receiving node's EA	The ECHONET address of the receiving node.

④ Response data

None

(2) "Object data for PUSH-type reception" request service

① Overview

Acquires the contents of the "Object data for PUSH-type reception" property from the receiving node.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Receiving node's EA	The ECHONET address of the receiving node.

④ Response data

Data name	Explanation
Maximum size for files	The maximum number of bytes a file can contain in order for it to be received by means of the PUSH-type file reception function.
Number of pieces of file storage data (= N)	The number of pieces of file storage data.
File storage equipment data	The class group code, class code and property code of the equipment object that stores the file(s).
Number of files	The number of files that can be received and retained (each piece of file storage equipment).
…	(N pieces of file provider data)

(3) File acquisition notification service

① Overview

Makes a request to notify the contents of the "file provider data" property to the receiving node.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Receiving node's EA	The ECHONET address of the receiving node.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.

④ Response data

None

(4) File transmission setting notification service

① Overview

Notifies the application software that there has been a "transmission and reception setting" write request from the receiving node.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Transmission and reception setting	0x20: Transmission start 0x01: Forced termination

④ Response data

None

(5) File transmission request service

① Overview

Writes data into the home node's file transmission service object to transmit a file.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
File name	The name of the file to be transmitted, which consists of eight ASCII characters (8 bytes).
Attribute	The attribute of the file to be transmitted, which consists of three ASCII characters (3 bytes).
Date	The date (dominical year, month, day, hour, minute and second) of the file to be transmitted.
File size	The size (in bytes) of the file to be transmitted.
Pointer	A pointer to the file to be transmitted.

④ Response data

Data name	Explanation
Transmission result	0: Transmission completed successfully 1: Transmission failed

4.6.2 Services relating to PUSH-type file reception (receiving node)

(1) File provider data notification service

① Overview

When a "file provider data" property notification is received from the transmitting node in a PUSH-type file transfer, the received data will be reported to the application software.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Transmitting node's EA	The ECHONET address of the transmitting node.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.

④ Response data

None

(2) File reception setting service

① Overview

If the file provider data received in the file provider data notification message indicates that the file in question is receivable, a request will be made to write the value to start transmission into the "transmission and reception setting" property of the transmitting node's file transmission service object, to receive the file.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Transmission and reception setting	The value to be written into the "transmission and reception setting" property. 0x20 = Reception permitted, 0x01 = Reception prohibited
Sender's EA	The ECHONET address of the sender node.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.

④ Response data

Data name	Explanation
Reception result	0: Reception completed successfully 1: Reception failed
File name	The name of the file to be received, which consists of eight ASCII characters (8 bytes).
Attribute	The attribute of the file to be received, which consists of three ASCII characters (3 bytes).
Date	The date (dominical year, month, day, hour, minute and second) of the file to be received.
File size	The size (in bytes) of the file to be received.
Pointer	A pointer to the file to be received.

4.6.3 Services relating to PULL-type file transmission (transmitting node)

(1) File provider data notification service (file reservation)

① Overview

When the receiving node writes data into the "file provider data" property of the transmitting node, the value(s) written will be reported to the application software.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Reservation number	The administration number of the registered (reserved) piece of file provider data.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.

④ Response data

None

(2) File transmission setting notification service

① Overview

Notifies the application software that a transmission commencement trigger has been received from the receiving node.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Reservation number	The administration number of the registered (reserved) piece of file provider data.
Transmission and reception setting	0x20: Transmission start 0x01: Forced termination

④ Response data

None

(3) File transmission request service

① Overview

Writes data into the home node's file transmission object to transmit a file.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Reservation number	The administration number of the registered (reserved) piece of file provider data.
File name	The name of the file to be transmitted, which consists of eight ASCII characters (8 bytes).
Attribute	The attribute of the file to be transmitted, which consists of three ASCII characters (3 bytes).
Date	The date (dominical year, month, day, hour, minute and second) of the file to be transmitted.
File size	The size (in bytes) of the file to be transmitted.
Pointer	A pointer to the file to be transmitted.

④ Response data

Data name	Explanation
Transmission result	0: Transmission completed successfully 1: Transmission failed

4.6.4 Services relating to PULL-type file reception (receiving node)

(1) Transmitting node data notification service

① Overview

Notifies the application software of the ECHONET address of the transmitting node that sends a file in a PUSH-type file transfer.

② Direction

File transfer service middleware → application software

③ Input data

Data name	Explanation
Transmitting node's EA	The ECHONET address of the transmitting node.

④ Response data

None

(2) “Object data for PULL-type transmission” request service

① Overview

Requests the file transmission object of the transmitting node to send the contents of the “object data for PULL-type transmission” property.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Transmitting node’s EA	The ECHONET address of the transmitting node.

④ Response data

Data name	Explanation
Maximum size for files	The maximum number of bytes a file can contain in order for it to be transmitted.
Number of pieces of file provider data (= N)	The number of pieces of file provider data, each of which consists of a piece of file provider equipment data and the number of files the piece of file provider equipment has.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that stores the file(s).
Number of files	The number of files the piece of file provider equipment has).
…	(N pieces of file provider data)

(3) File reservation service

① Overview

Reserves the file having the specified file provider equipment object, property and file number for transmission (file transmission service object of the transmitting node). The file transfer service middleware writes the specified data into the transmitting node by means of a file provider data write request and receives a file provider data notification from the transmitting node.

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Sender’s EA	The ECHONET address of the sender node.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.

④ Response data

Data name	Explanation
Reservation result	0: Reservation completed successfully 1: Reservation failed

(4) File reception setting service

① Overview

Requests the transmission of the file that was reserved by the file reservation service (transmitting node's file transmission service object). The file transfer service middleware controls the sequence from the confirmation of the transmitting node's transmission status to the transmission of the transmission commencement trigger to the reception of the file to the reception completion setting, and, upon completion of the file reception, returns "reception completed."

② Direction

Application software → file transfer service middleware

③ Input data

Data name	Explanation
Transmission and reception setting	The value to be written into the "transmission and reception setting" property. 0x20 = Reception permitted, 0x01 = Reception prohibited
Sender's EA	The ECHONET address of the sender node.
File provider equipment data	The class group code, class code, instance code and property code of the equipment object that provides the file.
File number	The number of the file of the piece of file provider equipment.
Transmission status inquiry flag	0: Inquiry disabled 1: Inquiry enabled

④ Response data

Data name	Explanation
Reception result	0: Reception completed successfully 1: Reception failed
File name	The name of the file to be received, which consists of eight ASCII characters (8 bytes).
Attribute	The attribute of the file to be received, which consists of three ASCII characters (3 bytes).
Date	The date (dominical year, month, day, hour, minute and second) of the file to be received.
File size	The size (in bytes) of the file to be received.
Pointer	A pointer to the file to be received.

Chapter 5 Link Setting Service Middleware

This chapter specifies the requirements for the action link setting and trigger link setting classes that are used to make settings for action and trigger links between nodes. The action link setting function actively controls other pieces of equipment upon intra-node property value changes and the trigger link setting function changes the values of properties located in the home node as specified in received messages. This service middleware achieves inter-node communication that does not require controllers.

5.1 Functions of the Link Setting Service Middleware

The required functions for the link setting service middleware are listed below together with brief explanations thereof.

① Action link service

Transmits a message when a node property value change has occurred in a node having an action link setting service object and the changed value meets the conditions for message transmission.

② Trigger link service

When a node having a trigger link setting service object receives a message and the message meets the trigger conditions, the trigger link service alters the specified property value.

It is not required to implement both the action and trigger link services. The action link service, trigger link service or both shall be implemented according to the characteristics of the application software.

5.2 Action Link Setting Service Class

This section specifies the requirements for the action link setting service class, so that action conditions for the action link service, action message enable/disable flags and action message makeup information can be published within the domain. The maximum allowable number of action links is the number of instances belonging to the action link setting object implemented in the piece of equipment. The object codes for the action link setting service class are as follows:

Class group code: 0x0D
Class code: 0x04
Instance code: 0x01 to 0x7F

Table 5.1 lists the properties of the action link setting service class.

Table 5.1 List of Properties of the Action Link Setting Service Class

Property name	EPC	Contents of property Value range	Data type	Size (in bytes)	Access rule	Required	Announcement on status change	Remarks
Action conditions	0xA0	Indicates the conditions for the linked action. (See the explanation below in the section entitled "Action Conditions.")	Unsigned char	Max. 247	Set/Get	○		
Action message enable/disable flag	0xA1	Indicates whether the current action setting is enabled or not. (See the explanation below in the section entitled "Action Message Enable/Disable Flag.")	Unsigned char	1	Set/Get	○		
Action message makeup information	0xA2	Information on the message that is to be transmitted immediately after the action condition is satisfied. (See the explanation below in the section entitled "Action Message Makeup information.")	Unsigned char	Max. 247	Set/Get	○		

5.2.1 Action conditions

The "action conditions" property indicates the conditions that must be satisfied in order for the message specified by the "action message makeup information" property (action message) to be transmitted. As many action conditions as necessary can be specified (i.e. action condition 1, action condition 2, and so on), as long as the 247 byte limit is not exceeded. When two or more action conditions are specified, all of the action conditions must be satisfied in order for the message specified by the "action message makeup information" property to be transmitted. When the action conditions for different instances are satisfied simultaneously and the action messages to be transmitted are identical, it is not required to transmit the same message twice (or more). The composition of the "action conditions" property is as follows:

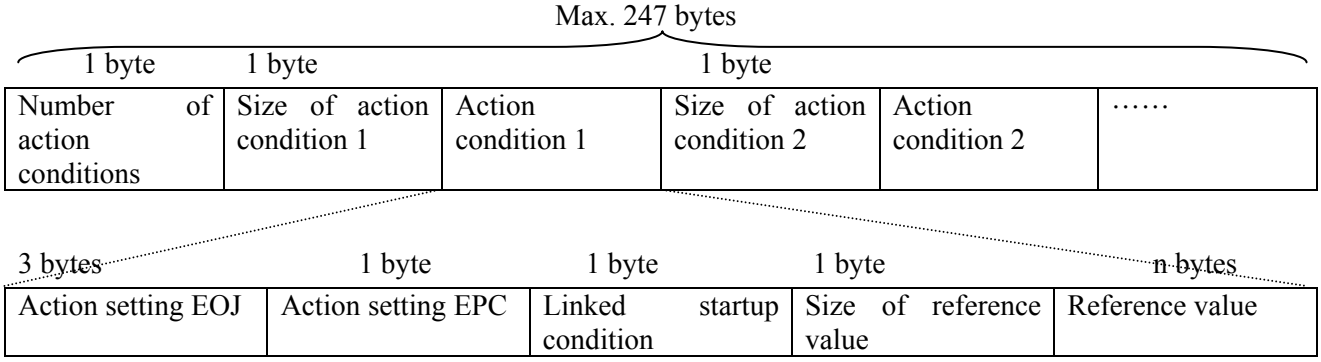


Fig. 5.1 Composition of the "Action Conditions" Property

① Number of action conditions

Indicates the number of "action condition" fields that follow the "number of action conditions" field. When this field contains "0," it is deemed that no action condition has been specified, and no action message will be transmitted. When "0" is written into this field, the action message enable/disable flag of the same instance will be set to "disable" and the data of the "action message makeup information" property will be cleared using "0x00."

② Size of action condition

Each "action condition" field is preceded by a "size of action condition" field that indicates the size of data contained in the "action condition" field.

③ Action condition

Each "action condition" field consists of the "action setting EOJ," "action setting EPC," "linked startup condition" "size of reference value" and "reference value" sub-fields.

A) Action setting EOJ

Indicates the EOJ value of the object being monitored.

B) Action setting EPC

Indicates the EPC value of the property being monitored.

C) Linked startup condition

The "linked startup condition" sub-field specifies the linked startup condition in terms of the relationship between the value of the property being monitored (i.e. the property specified by the action setting EPC value and the action setting EOJ value) and the reference value. If the linked startup condition is satisfied (and all other linked startup conditions, if any, are satisfied), the message will be transmitted in accordance with the secure setting and action message makeup information. Each "linked startup condition" field shall contain one of the following values:

0x00: No linked startup
0x01: The property value is equal to (=) the reference value.
0x02: The property value is larger than (>) the reference value.
0x03: The property value is smaller than (<) the reference value.
0x04: The property value is equal to or larger than (≥) the reference value.
0x05: The property value is equal to or smaller than (≤) the reference value.
0x06: The property value is not equal to (≠) the reference value.
0x07: Upon a property value change

D) Size of reference value

Indicates the size of the reference value.

E) Reference value

The value that is compared with the value of the property being monitored to determine whether or not the linked startup condition is satisfied. In the case of an array property, the first 2 bytes shall be used for the array element number and the third and succeeding bytes shall be used for the value for comparison.

5.2.2 Action message enable/disable flag

The action message enable/disable flag indicates whether or not to transmit the message specified by the current action message makeup information. When the value of this flag is "disable," no action message will be transmitted even if all action conditions are satisfied. The composition of the action message enable/disable flag is as follows:

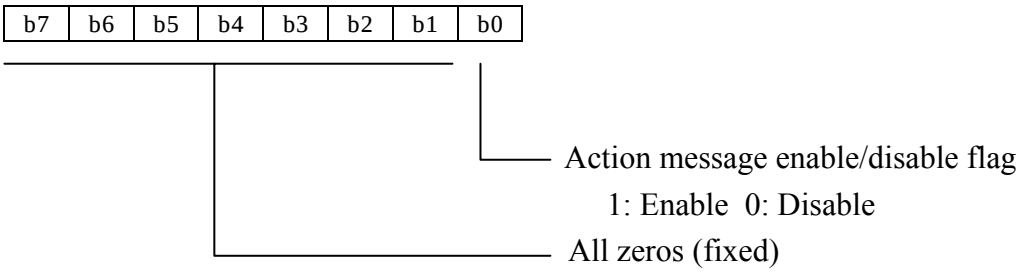


Fig. 5.2 Action Message Enable/Disable Flag

When the b0 bit contains "0," no action message will be transmitted even if all action conditions are satisfied.

5.2.3 Action message makeup information

The "action message makeup information" property contains information on the message that will be transmitted immediately after all action conditions are satisfied. When the service middleware receives a request to write action message makeup information into the "action message makeup information" property, the service middleware will write the data into the "action message makeup information" property if the received action message makeup information satisfies all of the following conditions:

・When the "encryption" or "authentication" bit of the "secure setting" field contains the value to enable encryption or authentication, respectively, the node must have the secure message transmission function.

・The composition of the “secure setting” field must be as specified in Fig. 5.4.

・The composition of the “message format” field must be as specified in Fig. 5.5.

・When the “message format designation” sub-field of the “message format” field contains the value to use the “basic format,” the “number of target properties” field must contain “1.”

・When the “message format designation” sub-field of the “message format” field contains the value to use the “composite format,” the “number of target properties” field must contain “1” or a larger value.

・When the “message format designation” sub-field of the “message format” field contains the value to use the “composite format,” the node must have the composite format message transmission function.

・The composition of the “EA data” field must be as specified in Part 2, Section 4.2.2 of this ECHONET Specification.

・The composition of the “OHD data” field must be as specified in Part 2, Section 4.2.5 of this ECHONET Specification.

・The compositions of the “SEOJ data” and “DEOJ data” fields must be as specified in Part 2, Section 4.2.6 of this ECHONET Specification.

・When an SEOJ value is specified in the “OHD data” field, the ECHONET object specified by the SEOJ value must be present in the node.

・The “ESV data” field must contain one of the service code values, which are explained later.

・The sum of the values (in bytes) of the property data counters must be equal to the sum of the sizes (in bytes) of all “target property” and “property value” data pieces.

・When the value contained in the “ESV data” field indicates a notification, the ECHONET property or properties specified by the “target property” data must be present in the node.

・When the value contained in the “ESV data” field indicates a write request, the relevant property value data must exist.

If any of the above conditions is not satisfied, the received message will be discarded and the processing will be terminated.

The composition of the "action message makeup information" property is as follows:

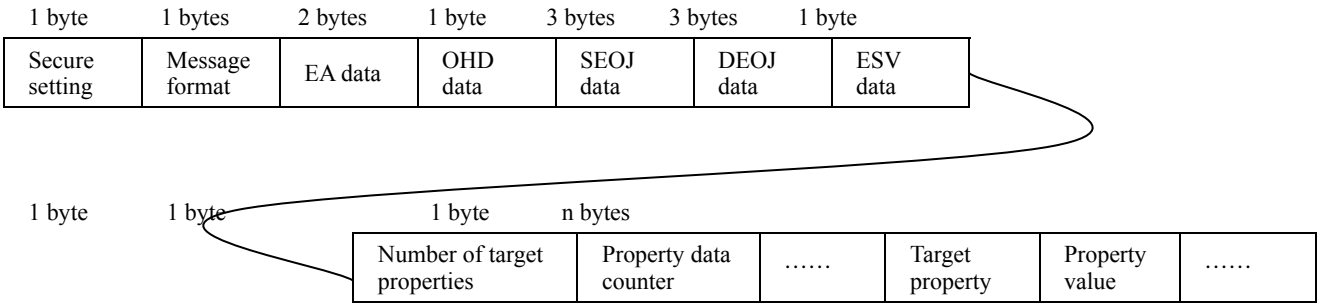


Fig. 5.3 Composition of the "Action Message Makeup Information" Property

① Secure setting

The "secure setting" field indicates whether the action message is to be transmitted as a plain message or secure message. When the action message is to be transmitted as a secure message, this field also indicates how the message is to be made secure. The composition of the "secure setting" field is as follows:

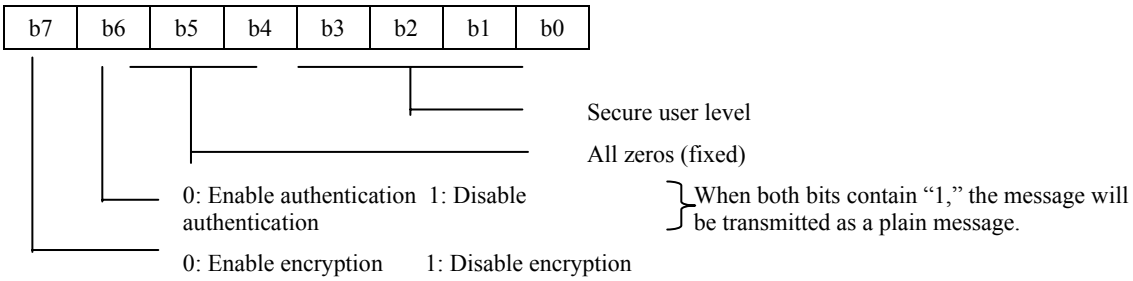


Fig. 5.4 Composition of the "Secure Setting" Field

When both the b7 and b6 bits contain "1," the action message will be transmitted as a plain message and the b3 to b0 bits will be ignored. Otherwise, the action message will be converted into a secure message before transmission. In the latter case, the b0 to b3 bits indicate the secure user level and the b6 and b7 bits indicate whether or not to enable the authentication and encryption functions, respectively. If a node that does not have the secure message transmission function receives an action message makeup information write request whose "secure setting" field indicates that the authentication or encryption function is enabled, that action message makeup information shall not be written.

② Message format

The “message format” field indicates the format of the message to be transmitted. The composition of the “message format” field is as follows:

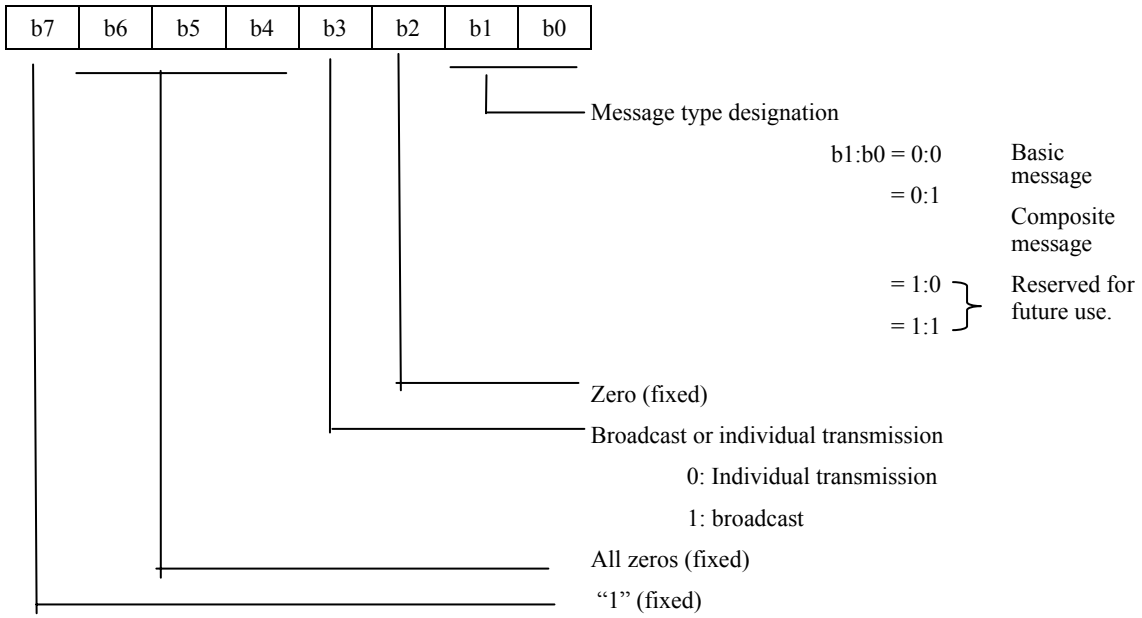


Fig. 5.5 Composition of the “Message Format” Field

The format of an action message is specified by the combination of the value contained in the b1 bit and the value contained in the b0 bit. If the b1 and b0 values are both 0, the action message will be transmitted in the basic format, and if the b1 and b0 values are 0 and 1, respectively, the action message will be transmitted in the composite format. If the b3 bit contains “1,” the action message will be transmitted individually to the specified address, and if the b3 bit contains “0,” the action message will be transmitted to the broadcast address. If a node that does not have the composite message transmission function receives an action message makeup information write request whose “message format” field indicates that the composite format is selected, that action message makeup information shall not be written.

③ EA data

The “EA data” field indicates the ECHONET address of the destination of the action message. When the b3 bit of the “message format” field contains “1” (broadcast), the “EA data” field indicates the broadcast address for a specific group of ECHONET addresses or the broadcast address for a general broadcast to all ECHONET addresses. The composition of the “EA data” field for broadcast is as specified in Part 2, Section 4.2.2 (Fig. 4.4) of this ECHONET Specification.

④ OHD data

The “OHD data” field indicates the OHD value of the action message.

⑤ SEOJ data

The “SEOJ data” field indicates the SEOJ value of the action message. If no SEOJ value is specified in the “OHD data” field, the “SEOJ data” field shall have no meaning and “0xFFFFFF” shall be stored in the “SEOJ data” field.

⑥ DEOJ data

The “DEOJ data” field indicates the DEOJ value of the action message. If no DEOJ value is specified in the “OHD data” field, the “DEOJ data” field shall have no meaning and “0xFFFFFF” shall be stored in the “DEOJ data” field.

⑦ ESV data

The “ESV data” field indicates the ESV value of the action message. When “composite format” is specified in the “message format” field, the “ESV data” field indicates the CpESV value. The “ESV data” field must contain one of the ECHONET service code values listed below. If an action message makeup information write request is received that has an “ESV data” value other than those specified below, that action message makeup information shall not be written.

Table 5.2 ESV Values for the Basic Format

Service code	Service	Symbol
0x60	Property value write request not requiring a response	SetI
0x61	Property value write request requiring a response	SetC
0x64	Element designation-type property value write request not requiring a response	SetMI
0x65	Element designation-type property value write request requiring a response	SetMC
0x73	Property value notification not requiring a response	INF
0x74	Property value notification requiring a response	INFC
0x77	Element designation-type property value notification not requiring a response	INFM
0x78	Element designation-type property value notification requiring a response	INFMC

Table 5.3 CpESV Values for the Composite Format

Service code	Service	Symbol
0x60	Property value write request not requiring a response	CpSetI
0x61	Property value write request requiring a response	CpSetC
0x73	Property value notification not requiring a response	CpINF
0x74	Property value notification requiring a response	CpINFC

⑧ Number of target properties

When the action message is a composite message, the “number of target properties” field shall indicate the number of target properties. When the action message is a basic message, the “number of target properties” field shall contain “1.” The “number of target properties” field is followed by the “property data counter,” “target property” and “property value” fields, which are explained below.

⑨ Property data counter

The “action message makeup information” property may have one or more “property data counter” fields. The number of “property data counter” fields shall be equal to the number indicated in the “number of target properties” field. The “property data counter” fields shall be contiguous and each “property data counter” field shall indicate the sum of the sizes (in bytes) of the corresponding “target property” and “property value” fields.

⑩ Target property

The “action message makeup information” property may have one or more “target property” fields. The number of “target property” fields shall be equal to the number indicated in the “number of target properties” field. Each of the “target property” fields, which indicates one of the EPC values of the action message, shall be paired with the corresponding “property value” field, and the pairs of fields shall be contiguous.

⑪ Property value

The “action message makeup information” property may have one or more “property value” fields. The number of “property value” fields shall be equal to the number indicated in the “number of target properties” field. Each of the “property value” fields, which indicates one of the EDT values of the action message, shall be paired with the corresponding “target property” field, and the pairs of fields shall be contiguous.

When the value contained in the “ESV data” field indicates a notification, no “property value” data shall be stored and the values held by the relevant properties, at the time of action message transmission, shall be used.

5.3 Trigger Link Setting Service Class

This section specifies the requirements for the trigger link setting service class, so that trigger processing information for the trigger link service, trigger message enable/disable flags and trigger message makeup information can be published within the domain. The maximum allowable number of trigger links is the number of instances belonging to the trigger link setting object implemented in the piece of equipment. The object codes for the trigger link setting service class are as follows:

Class group code: 0x0D
Class code: 0x05
Instance code: 0x01 to 0x7F

Table 5.4 lists the properties of the trigger link setting service class.

Table 5.4 List of Properties of the Trigger Link Setting Service Class

Property name	EPC	Contents of property Value range	Data type	Size (in bytes)	Access rule	Required	Announcement on status change	Remark
Trigger processing information	0xA0	Provides information on the processing to be performed upon reception of a trigger message. (See the explanation below in the section entitled “Trigger Processing Information.”)	Unsigned char	Max. 247	Set/Get	○		
Trigger message enable/disable flag	0xA1	Indicates whether the current trigger setting is enabled or not. (See the explanation below in the section entitled “Trigger Message Enable/Disable Flag.”)	Unsigned char	1	Set/Get	○		
Trigger message makeup information	0xA2	Contains information on the trigger message. (See the explanation below in the section entitled “Trigger Message Makeup information.”)	Unsigned char	Max. 247	Set/Get	○		

5.3.1 Trigger processing information

The “trigger processing information” property provides information on the piece or pieces of trigger processing to be performed immediately after the trigger conditions are satisfied. A piece of trigger processing is the overwriting, with the value specified in the “value to write” sub-field of a “piece of trigger processing” field, of the property located in the home node that is specified by the “trigger setting EOJ” and “trigger setting EPC” sub-fields of the same “piece of trigger processing” field. As many pieces of trigger processing as

necessary can be specified (i.e. trigger processing piece 1, trigger processing piece 2, and so on). When two or more pieces of trigger processing are specified, all the pieces of trigger processing will be performed immediately after the trigger conditions are satisfied. The composition of the "trigger processing information" property is as follows:

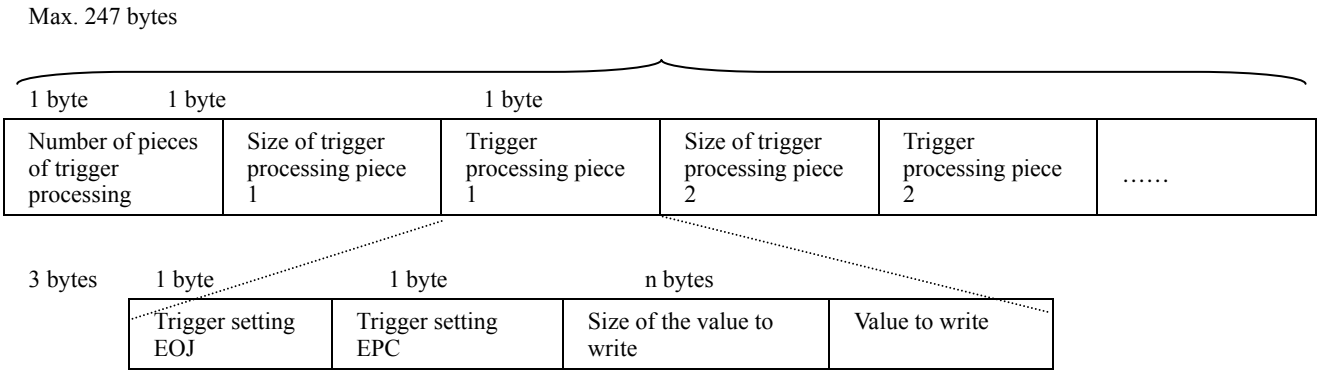


Fig. 5.6 Composition of the "Trigger Processing Information" Property

① Number of pieces of trigger processing

The "number of pieces of trigger processing" field indicates the number of pieces of trigger processing specified in the "trigger processing information" property. When this field contains "0," it is deemed that no piece of trigger processing has been specified, and no trigger processing will be performed even if all fields of the "trigger message makeup information" property contain valid values. When "0" is written into the "number of pieces of trigger processing" field, the trigger message enable/disable flag of the same instance will be set to "disable" and the data of the "trigger message makeup information" property will be cleared using "0x00."

② Size of trigger processing piece

Indicates the size of data contained in the succeeding "trigger processing piece" field.

③ Trigger processing piece

Each "trigger processing piece" field consists of the "trigger setting EOJ," "trigger setting EPC," "size of the value to write" and "value to write" sub-fields

A) Trigger setting EOJ

Indicates the EOJ value of the object on which the pieces of trigger processing are to be performed.

B) Trigger setting EPC

Indicates the EPC value of the property on which the pieces of trigger processing are to be performed.

C) Size of the value to write

Indicates the size (in bytes) of the value to write.

D) Value to write

Indicates the value to be written into the property specified by the trigger setting EOJ and trigger setting EPC values immediately after the trigger conditions are satisfied. In the case of an array property, the first 2 bytes shall be used for the array element number and the third and succeeding bytes shall be used for the value to write.

5.3.2 Trigger message enable/disable flag

The trigger message enable/disable flag indicates whether or not to perform the pieces of trigger processing specified in the "trigger processing information" property after the trigger conditions are satisfied. When the value of this flag is "disable," no trigger processing will be performed even if the trigger conditions are satisfied. The composition of the trigger message enable/disable flag is as follows:

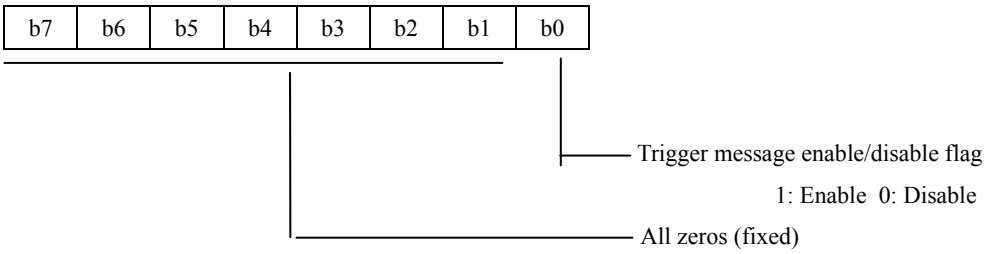


Fig. 5.7 Composition of the Trigger Message Enable/Disable Flag

When the b0 bit contains "0," no trigger processing will be performed even if the trigger conditions are satisfied.

5.3.3 Trigger message makeup information

The "trigger message makeup information" property contains the information to be compared with the information contained in the received message to determine whether or not the trigger conditions are satisfied. The composition of the "trigger message composition information" property is as follows:

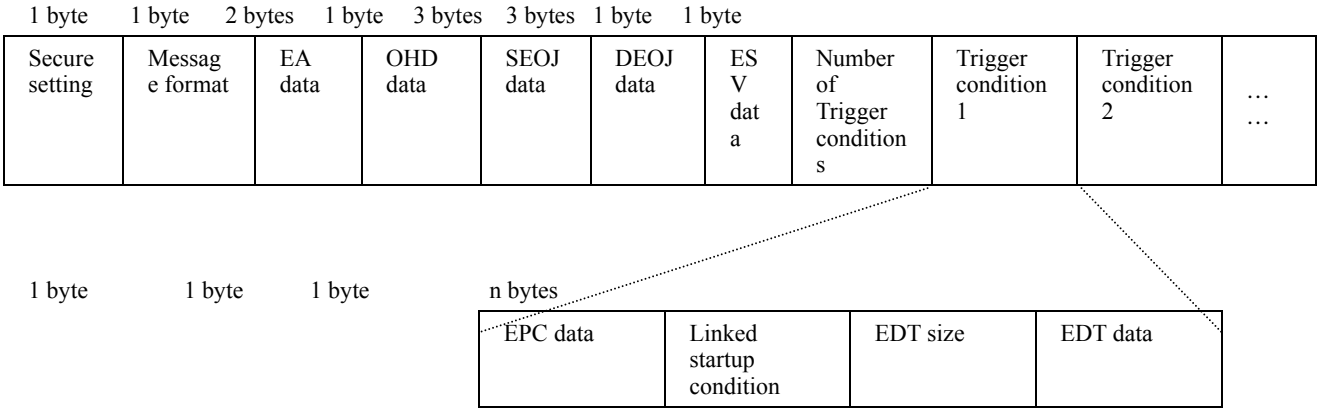


Fig. 5.8 Composition of the "Trigger Message Makeup Information" Property

① Secure setting

The "secure setting" field indicates whether the message to be received is a plain message or secure message. The composition of the "secure setting" field is as follows:

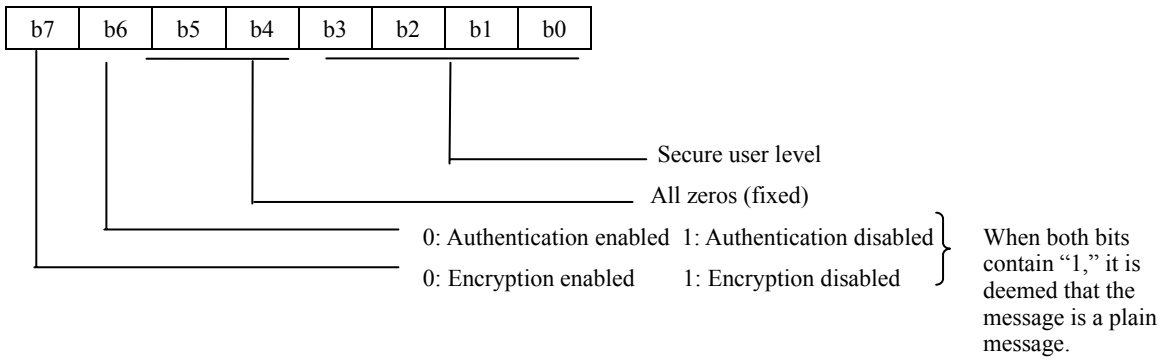


Fig. 5.9 Composition of the "Secure Setting" Field

When both the b7 and b6 bits contain "1," it shall be deemed that the message to be received for use for the trigger judgment is a plain message (i.e. a message to which neither the encryption function nor the authentication function has been applied), and the received message must be a plain message in order for the trigger conditions to be regarded as having been satisfied.

Otherwise, it shall be deemed that the message is a secure message, and the action message must be converted into a secure message before transmission (in which case the b0 to b3 bits indicate the secure user level and the b6 and b7 bits indicate whether or not the authentication and encryption functions, respectively, are enabled.)

② Message format

The “message format” field indicates the format of the received message. The EHD value of the received message and the “message format designation” bits of the “message format” field must match in order for the trigger conditions to be regarded as having been satisfied. However, when the b1 and b0 bits contain “1” and ”0,” respectively, the trigger conditions shall be regarded as having been satisfied irrespective of whether the received message is a basic or composite message. The received message must be an individually transmitted message when both the b3 and b2 bits contain “0,” and a broadcast message when both the b3 and b2 bits contain “1.” When the b2 bit contains “1,” the trigger conditions shall be regarded as having been satisfied irrespective of whether the received message is an individually transmitted message or a broadcast message. The composition of the “message format” field is as follows:

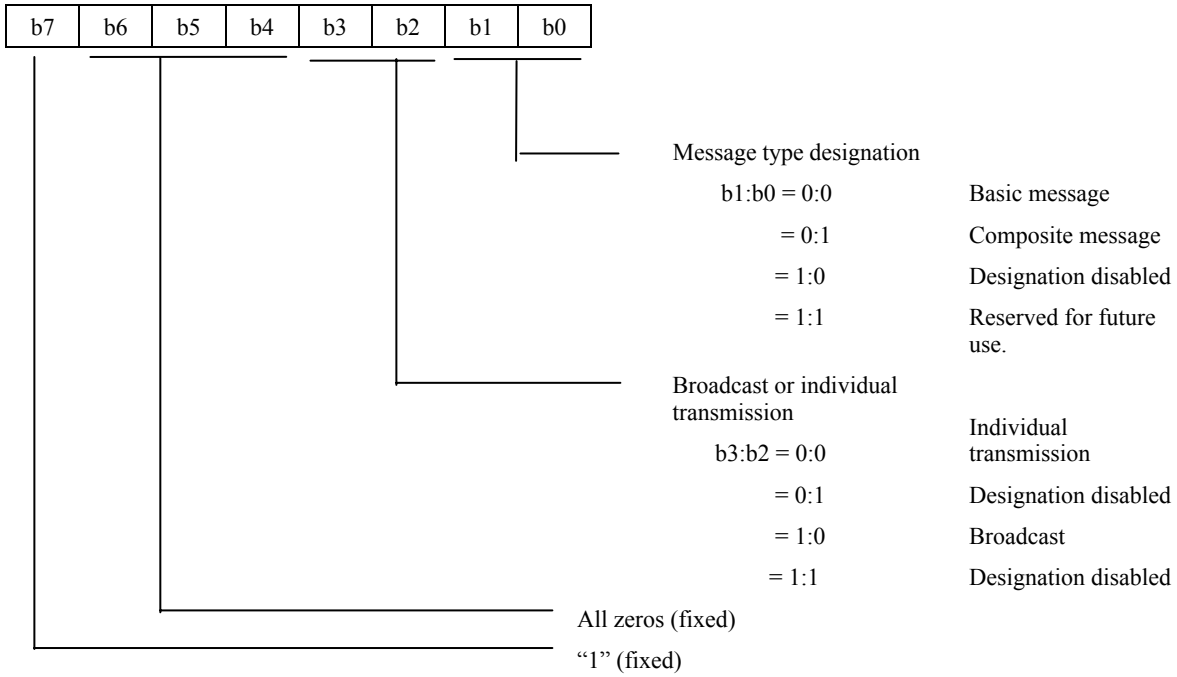


Fig. 5.10 Composition of the “Message Format” Field

③ EA data

The “EA data” field indicates the SEA value of the received message. The SEA value of the received message must be identical to the value contained in the “EA data” field in order for the trigger conditions to be regarded as having been satisfied.

④ OHD data

The “OHD data” field indicates the OHD value of the received message. The OHD value of the received message must be identical to the value contained in the “OHD data” field in order for the trigger conditions to be regarded as having been satisfied.

⑤ SEOJ data

The “SEOJ data” field indicates the SEOJ value of the received message. The SEOJ value of the received message must be identical to the value contained in the “SEOJ data” field in order for the trigger conditions to be regarded as having been satisfied. If no SEOJ value is specified in the “OHD data” field, the “SEOJ data” field shall have no meaning and “0xFFFFFF” shall be stored in the “SEOJ data” field.

⑥ DEOJ data

The “DEOJ data” field indicates the DEOJ value of the received message. The DEOJ value of the received message must be identical to the value contained in the “DEOJ data” field in order for the trigger conditions to be regarded as having been satisfied. If no DEOJ value is specified in the “OHD data” field, the “DEOJ data” field shall have no meaning and “0xFFFFFF” shall be stored in the “DEOJ data” field.

⑦ ESV data

The “ESV data” field indicates the ESV value of the received message when the received message is a basic message. This field indicates the CpESV value of the received message when the received message is a composite message. The ESV or CpESV value of the received message must be identical to the value contained in the “ESV data” field in order for the trigger conditions to be regarded as having been satisfied.

⑧ Number of trigger conditions

Indicates the number of trigger conditions.

[Trigger conditions]

The “trigger message makeup information” property may have one or more “trigger condition” fields. The number of “trigger condition” fields is equal to the value specified in the “number of trigger conditions” field and each “trigger condition” field consists of the “EPC data,” “linked startup condition,” “EDT size” and “EDT data” sub-fields. If two or more trigger conditions are specified, all the trigger conditions must be satisfied in order for the trigger processing to be performed and the received message must be a composite message in order for the trigger conditions to be regarded as having been satisfied.

⑨ EPC data

The “EPC data” sub-field indicates the EPC value of the received message. The EPC value

contained in the received message must be identical to the EPC value contained in the "EPC data" sub-field in order for the trigger conditions to be regarded as having been satisfied.

⑩ Linked startup condition

The "linked startup condition" sub-field specifies the linked startup condition in terms of the relationship between the EDT value of the EPC value of the received message (whose EPC value matches) and the value contained in the "EDT data" sub-field. When the value contained in the "linked startup condition" sub-field is "no linked startup," the trigger conditions will always be regarded as not having been satisfied. When the value contained in the "linked startup condition" sub-field is "to be regarded as having been satisfied irrespective of the EDT value contained in the message," the trigger conditions will always be regarded as having been satisfied. If the received message does not contain an EDT value (Get, etc.), only the EPC value match check will be performed and no linked startup conditions will be used. Each "linked startup condition" field shall contain one of the following values:

0x00: no linked startup

0x01: The EDT value contained in the message is equal to (=) the value contained in the "EDT data" sub-field.

0x02: The EDT value contained in the message is larger than (>) the value contained in the "EDT data" sub-field.

0x03: The EDT value contained in the message is smaller than (<) the value contained in the "EDT data" sub-field.

0x04: The EDT value contained in the message is equal to or larger than (≥) the value contained in the "EDT data" sub-field.

0x05: The EDT value contained in the message is equal to or smaller than (≤) the value contained in the "EDT data" sub-field.

0x06: The EDT value contained in the message is not equal to (≠) the value contained in the "EDT data" sub-field.

0x07: To be regarded as having been satisfied irrespective of the EDT value contained in the message.

⑪ EDT size

The "EDT size" sub-field indicates the size of the property specified by the "EPC data" sub-field. When the value contained in the "linked startup condition" sub-field is "no linked startup" or "to be regarded as having been satisfied irrespective of the EDT value contained in the message," or when the received message does not contain an EDT value (Get, etc.), the "EDT size" sub-field shall contain "0."

⑫ EDT data

The “EDT data” sub-field contains the data used to determine whether or not the linked startup condition is satisfied. When the value contained in the “linked startup condition” sub-field is “no linked startup” or “to be regarded as having been satisfied irrespective of the EDT value contained in the message,” the “trigger message makeup information” property has no “EDT data” sub-field. In the case of an array property, the first 2 bytes shall be used for the array element number and the third and succeeding bytes shall be used for the property value.

5.4 Linked Startup Service Middleware Requirements

The linked startup service middleware requirements that must be satisfied are as follows:

As a general rule, the property values of action and trigger link setting service objects must be stored in a nonvolatile memory device.

It is recommended that action link service objects be implemented when it is intended to use the link setting service. For a node having a trigger link service object to achieve a linked action, a function is required in the other node to transmit a trigger message by means of an action link service object, etc. A node having an action link service object, on the other hand, can actively control other pieces of equipment and thus can control the other node even when it does not have a link function, which means that implementing action link service objects facilitates system development.

An action or trigger link service object has, as part of it, a communication definition object for link setting (action setting / trigger setting) that meets the requirements specified in Part 2, Chapter 9. Therefore, it is strongly recommended that action or trigger link service objects be implemented rather than communication definition objects for link setting (communication definition class group) if it is intended to use the link setting function.

5.5 Link Setting Service Interface

This section specifies the services to be provided by the link setting service API and shows examples of data exchange between the application software and link setting service middleware.

5.5.1 Trigger link service

(1) Trigger processing notification service

① Overview

Reports the contents of the piece or pieces of trigger processing to the application software after the trigger conditions are satisfied.

② Direction

Trigger link service middleware → application software

③ Input data

Data name	Explanation
ECHONET object	Specifies the ECHONET object of the target property for the trigger processing.
Property	Specifies the target property for the trigger processing.
Property value	Specifies the property value to write.

④ Response data

None

Chapter 6 Group Broadcast Number Management Service Middleware

This chapter specifies the requirements for the group broadcast number management service middleware and group broadcast number management service class, which are implemented in controllers and other nodes capable of setting group broadcast numbers (as defined in Part 2, "9.11.1 Detailed specifications for the node profile class") in other nodes. In a system which has two or more controllers or other nodes capable of setting broadcast group numbers in other nodes, the implementation of the group broadcast number management service class allows latest information on the group broadcast numbers that have been set to be published in the network, thereby allowing such controllers etc. to reference group broadcast numbers that have been set in other nodes by other controllers etc. and coordinate broadcast numbers with other controllers.

Applications capable of setting group broadcast numbers in other nodes use the group broadcast number management service middleware to set group broadcast numbers in other nodes. The group broadcast number management service middleware updates the information contained in the home node's group broadcast number management service class on the group broadcast numbers that have been set and sends write requests to the group broadcast number properties of the node profile objects of the target nodes for the setting of group broadcast numbers. For the specifications for the ECHONET addresses of senders and recipients of group broadcast messages, refer to Part 2, "4.2.2 Source/Destination ECHONET Address (SEA/DEA)."

6.1 Functions of the Group Broadcast Number Management Service Middleware

The functions the group broadcast number management service middleware must provide are as follows:

(A) Group broadcast number setting function

When an application of the home node sets a group broadcast number in a non-home node equipped with the group broadcast function using the group broadcast number management service middleware, the group broadcast number setting function sets the group broadcast number in the group broadcast number property of the node profile object of the non-home node using the Set access rule.

(B) Node management function for group broadcast number setting

This function stores, whenever an application of the home node sets a group broadcast number in a non-home node using the group broadcast number management service middleware, the EA value of the non-home node. However, when there is another node equipped with a function to set group broadcast numbers, the EA values stored with the "node management function for group broadcast number setting" do not necessarily match the EA values of the nodes which reside in the system and in which group broadcast numbers have been set.

6.2 Group Broadcast Number Management Service Class

This section specifies the requirements for the group broadcast number management service class, which is a service class to allow the operation status of the group broadcast number management service and the group broadcast numbers that have been set by the service to be published within the domain. The object codes of the group broadcast number management service class are as follows:

Class group code: 0x0D

Class code: 0x06

Instance code: 0x01 to 0x7F

Table 6-1 Properties of the Group Broadcast Number Management Service Class

Property name	EPC	Content 値域(10 進表記)	Data type	Data size	Access rule	Compulsory	Status change announcement	Remark
Information on the group broadcast numbers that have been set	0x C0	Provides information on the group broadcast numbers that have been set First byte: Number of group broadcast numbers that have been set Second and succeeding bytes: If the number of group broadcast numbers that have been set is 8 or less, the group broadcast numbers will be listed. For the specifications for the case where the number of group broadcast numbers that have been set is 9 or more, refer to Section 6.2.2.	unsigned char × (Max)33	(Max)33	Get	○		

6.2.1 Operation status

Used to set the operation status (ON/OFF) of the group broadcast number management service functions and acquire information on the operation status. The value “0x30” indicates that the operation status is ON (service enabled) and the value “0x31” indicates that the operation status is OFF (service disabled). When the operation status is OFF, access to other properties is not guaranteed. In the case where the functions of this class become operational as soon as the node starts operating, this property may be implemented with the property value fixed at 0x30 (operation status = ON).

6.2.2 Information on the group broadcast numbers that have been set

This property indicates the group broadcast numbers that have been set in other nodes by an application using the group broadcast number management service middleware.
If the number of group broadcast numbers that have been set is 8 or less, the first byte

indicates the number of group broadcast numbers that have been set and the second and succeeding bytes list the group broadcast numbers that have been set. If the number of group broadcast numbers that have been set is 9 or more, the first byte indicates the number of group broadcast numbers that have been set and the second and succeeding bytes provide information in the description format explained below. The bit value "1" indicates that a group broadcast number has been set in 1 or more nodes, and the bit value "0" indicates that no group broadcast number has been set.

	Bit0	Bit1	Bit2	Bit3	Bit4	Bit5	Bit6	Bit7
2byte	00	01	02	03	04	05	06	07
3byte	08	09	0A	0B	0C	0D	0E	0F
4byte	10	11	12	13	14	15	16	17
5byte	18	19	1A	1B	1C	1D	1E	1F
6byte	20	21	22	23	24	25	26	27
7byte	28	29	2A	2B	2C	2D	2E	2F
8byte	30	31	32	33	34	35	36	37
9byte	38	39	3A	3B	3C	3D	3E	3F
10byte	40	41	42	43	44	45	46	47
11byte	48	49	4A	4B	4C	4D	4E	4F
12byte	50	51	52	53	54	55	56	57
13byte	58	59	5A	5B	5C	5D	5E	5F
14byte	60	61	62	63	64	65	66	67
15byte	68	69	6A	6B	6C	6D	6E	6F
16byte	70	71	72	73	74	75	76	77
17byte	78	79	7A	7B	7C	7D	7E	7F
18byte	80	81	82	83	84	85	86	87
19byte	88	89	8A	8B	8C	8D	8E	8F
20byte	90	91	92	93	94	95	96	97
21byte	98	99	9A	9B	9C	9D	9E	9F
22byte	A0	A1	A2	A3	A4	A5	A6	A7
23byte	A8	A9	AA	AB	AC	AD	AE	AF
24byte	B0	B1	B2	B3	B4	B5	B6	B7
25byte	B8	B9	BA	BB	BC	BD	BE	BF
26byte	C0	C1	C2	C3	C4	C5	C6	C7
27byte	C8	C9	CA	CB	CC	CD	CE	CF
28byte	D0	D1	D2	D3	D4	D5	D6	D7
29byte	D8	D9	DA	DB	DC	DD	DE	DF
30byte	E0	E1	E2	E3	E4	E5	E6	E7
31byte	E8	E9	EA	EB	EC	ED	EE	EF
32byte	F0	F1	F2	F3	F4	F5	F6	F7
33 byte	F8	F9	FA	FB	FC	FD	FE	FF

(Note) Bit value = 0: No group broadcast number has been specified

Bit value = 1: A group broadcast number has been specified

6.3 Compulsory Requirements

With regard to the group broadcast number management service middleware, the following requirements must be satisfied:

(A) In the case where this class is implemented, the "information on the group broadcast numbers that have been set" property must be implemented without exception.

(B) The area that can be used for group broadcast numbers shall be as shown in Table 6-2. The setting of group broadcast numbers in other nodes through the use of this class is permitted in the range between 0x00 and 0x6F inclusive.

Table 6-2 Group Broadcast Number Setting Ranges

Name of area	Range
Network setting area	0x00～0x6F
Manual setting area	0x70～0x7F
For future reserved	0x80～0xFF

(C) Whenever a group broadcast number is set in a non-home node, the EA value of the non-home node shall be stored.

6.4 Group Broadcast Number Management Service Interface

This section describes the services the group broadcast number management service API should provide, and shows examples of data exchanges between the application software and group broadcast number management service middleware.

6.4.1 Group broadcast number management service

(1) Group broadcast number addition service

1) Overview

The application provides the group broadcast number management service middleware with the data showing the number of group broadcast numbers to add and the group broadcast numbers to add, together with the EA value of the destination node. The group broadcast number management service middleware provides these data to the ECHONET communication processing section for the transmission of a request for a write to the group broadcast number property of the node profile class with the group broadcast numbers to add and the group broadcast numbers that have already been set in the target node combined. The setting of group broadcast numbers shall use node-to-node messages and shall not use broadcast messages.

The group broadcast number management service class shall store, whenever it receives this service, the EA value of the node being managed in relation to setting for each group broadcast number.

2) Direction

Application → Group broadcast number management service middleware

3) Input data

Data name	Explanation
EA value of the destination node	ECHONET address of the destination node
Number of group broadcast numbers to set	Number of group broadcast numbers to add
Group broadcast numbers	The group broadcast numbers to add in the destination node

4) Response data

None

5) Others

When an application adds a group broadcast number in another node, the group broadcast number management service middleware shall operate as follows:

If the group broadcast number management service middleware receives a request from the application to add the group broadcast number "3" in a node in which the group broadcast numbers "1" and "2" have been set, the data shall be passed to the ECHONET communication processing section in such a way that a write request will be sent with "1," "2" and "3" specified as the property values of the group broadcast number property of the node profile class, because the method to write data to the group broadcast number property is overwriting.

(2) Group broadcast number deletion service

1) Overview

The application provides the group broadcast number management service middleware with the data showing the number of group broadcast numbers to delete and the group broadcast numbers to delete, together with the EA value of the destination node. The group broadcast number management service middleware passes the data to the ECHONET communication processing section in such a way that a request for a write to the group broadcast number property of the node profile class will be sent with the group broadcast numbers to delete removed from the set of the group broadcast numbers that have been set in the setting alteration target node. The setting of group broadcast numbers shall use node-to-node messages and shall not use broadcast messages.

The group broadcast number management service class shall store, whenever it receives this service, the EA value of the node being managed in relation to setting for each group broadcast number.

2) Direction

Application → Group broadcast number management service middleware

3) Input data

Data name	Explanation
EA value of the destination node	ECHONET address of the destination node
Number of group broadcast numbers to delete	Number of group broadcast numbers to delete
Group broadcast numbers	The group broadcast numbers to delete

4) Response data

None

5) Others

When an application deletes a group broadcast number in another node, the group broadcast number management service middleware shall operate as follows:

If the group broadcast number management service middleware receives a request from the application to delete the group broadcast number "3" in a node in which the group broadcast numbers "1" and "2" have been set, the data shall be passed to the ECHONET communication processing section in such a way that a write request will be sent with "1" and "2" specified as the property values of the group broadcast number property of the node profile class, because the method to write data to the group broadcast number property is overwriting.

(3) "By group broadcast number" node information acquisition service

1) Overview

By passing a group broadcast number to the group broadcast number management service middleware, the application acquires from the group broadcast number management service middleware the EA values of the nodes in which the group broadcast number has been set.

2) Direction

Application → Group broadcast number management service middleware

3) Input data

Data name	Explanation
Number of group broadcast numbers to set	A group broadcast number is entered to acquire the EA values of the nodes in which the group broadcast number has been set.

4) Response data

Data name	Explanation
Number of nodes in which the group broadcast	Indicates the number of nodes in which the entered group broadcast number has been set.

number has been set	
EA values of the nodes in which the group broadcast number has been set	Lists the ECHONET addresses of the nodes in which the entered group broadcast number has been set.

(4) “By node” group broadcast number acquisition service

1) Overview

By passing to the group broadcast number management service middleware the EA value of a node in which group broadcast numbers have been set, the application acquires from the group broadcast number management service middleware the group broadcast numbers that have been set in the node with the EA value.

2) Direction

Application → Group broadcast number management service middleware

3) Input data

Data name	Explanation
EA	ECHONET address

4) Response data

Data name	Explanation
Number of group broadcast numbers that have been set in the node	Indicates the number of group broadcast numbers that have been set in the node with the entered EA value.
Number of group broadcast numbers to set	Lists the group broadcast numbers that have been set in the node with the entered EA value.

Chapter 7 EMS Service Middleware for Housing (Suggested Practical Applications)

This chapter provides examples of EMS service middleware for housing, as a basis for the future development of service objects.

7.1 System Model

As a system model for housing-dedicated EMS (Energy Management Service), a home-dedicated peak-cut EMS is designed as a control system to prevent total current consumption in the target range (usually within a single home) from exceeding a set value. The following methods might be adopted to implement such a system:

(A) Housing-dedicated feedback-type peak-cut EMS

When total current consumption exceeds the set value, the capacity of devices in operation is reduced according to rules specified by the controller.

(B) Housing-dedicated feed forward type peak-cut EMS

When the device capacity is to be changed, this intention is applied to the controller, which then compares the current total consumption with the set value and assigns a consumable current value to the device.

(C) Housing-dedicated hybrid-type peak-cut EMS

When the device capacity is to be changed, this intention is applied to the controller, which then compares current total consumption with the set value and assigns the consumable current to the device. As a result, if the total current consumption exceeds the set value, the controller reduces the capacity of the device in operation according to a specific rule. (Hybrid of methods A and B)

Figure 6.1 shows the network configuration on which this system is based. In the figure, the shaded devices are target devices for housing-dedicated EMS control.

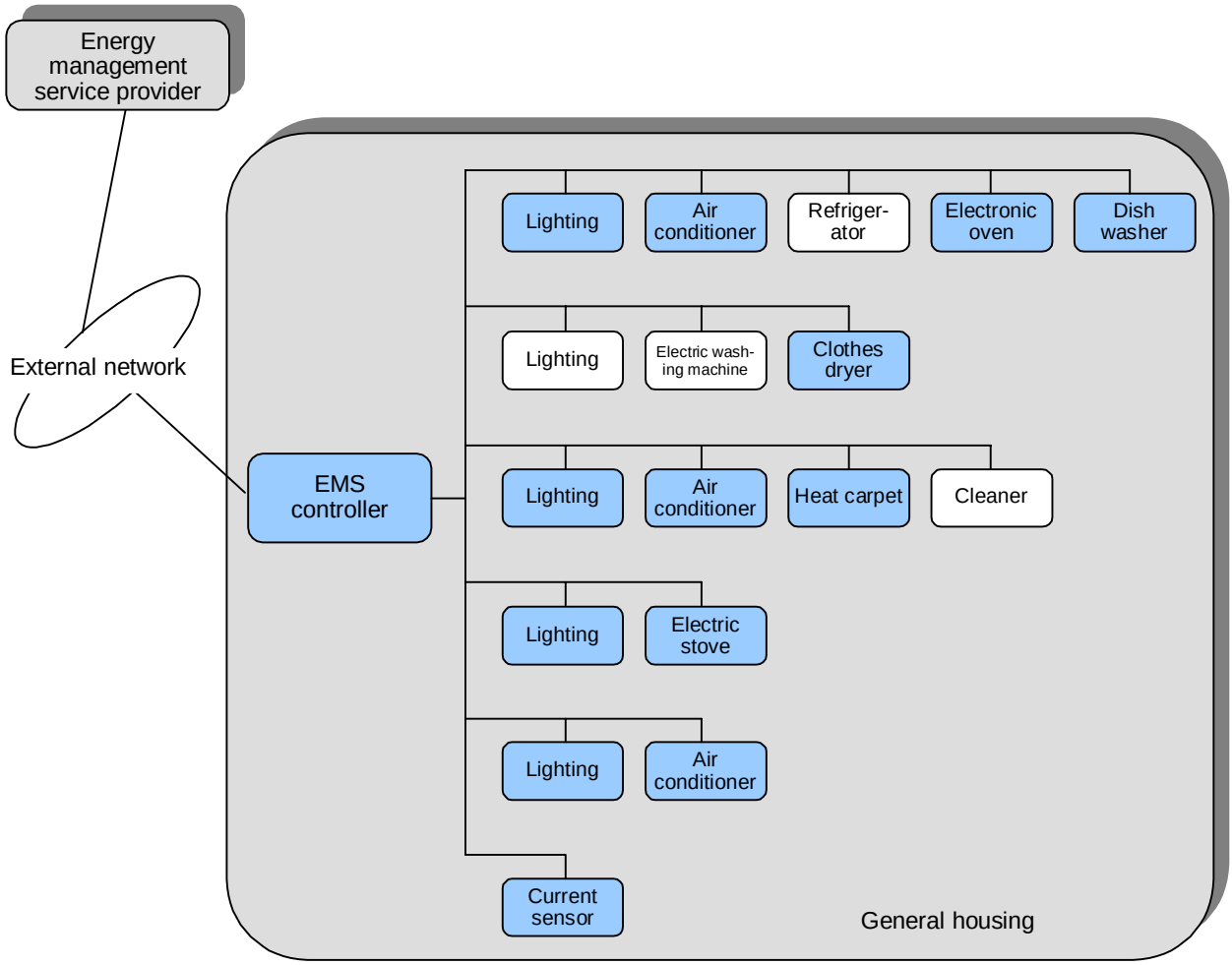


Fig. 7.1 Housing-Dedicated EMS System and ECHONET Network

7.2 Housing-Dedicated EMS Functions

Regarding the three types of housing-dedicated EMS described in the previous section, this section outlines the control to be exerted by the EMS controller and describes its assumed functions.

7.2.1 Housing-dedicated feedback-type peak-cut EMS

(1) Outline of controller control

Figure 3.2 shows the outline of control to be exerted by the controller supervising housing-dedicated feedback-type peak-cut EMS control.

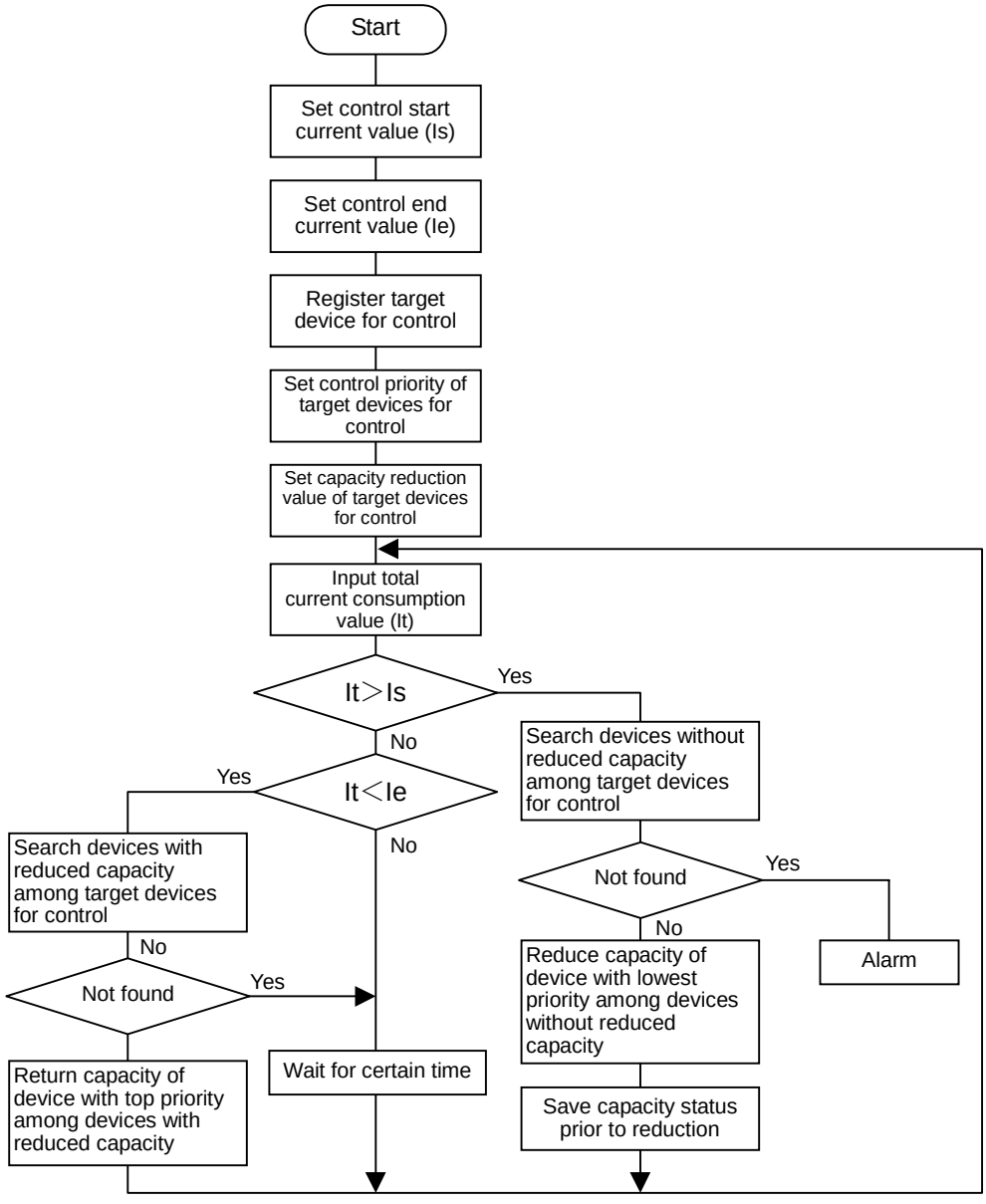


Fig. 7.2 Outline of Housing-Dedicated Feedback-Type Peak-Cut EMS Control

(2) Controller functions

The controller functions required to implement housing-dedicated feedback-type peak-cut EMS control are described below.

(a) Control-target device managing function

A function to register the target ECHONET devices for EMS control and set and hold the parameters for individual devices. The parameters to be set are shown below.

- Control priority
- Capacity reduction value
- Recovery capacity value (capacity value immediately before capacity reduction)

The target capacity for control differs with each device type. Table 3.1 shows device types and examples of control-target capacity for these devices.

Table 7.1 Device Types and Control-Target Capacity

Device type	Control-target capacity
Lighting	Lighting, power ON/OFF
Air conditioner	Set temperature, power ON/OFF
Dish washer	Water temperature, suspension
Clothes dryer	Heater power, suspension
Hot carpet	Set temperature, power ON/OFF
Electric stove	Heater power, power ON/OFF

(b) Current value monitoring function

A function to monitor the current value, which is a control start condition, and to set and hold the control start current value and control end current value as a reference for device control.

(c) Device control function

A function to exert peak-cut control according to the data of the control-target device managing function and the current value monitoring function.

7.2.2 Housing-dedicated feed forward type peak-cut EMS

(1) Outline of controller control

Figure 3.3 shows the outline of control to be exerted by the controller supervising housing-dedicated feed forward type peak-cut EMS control.

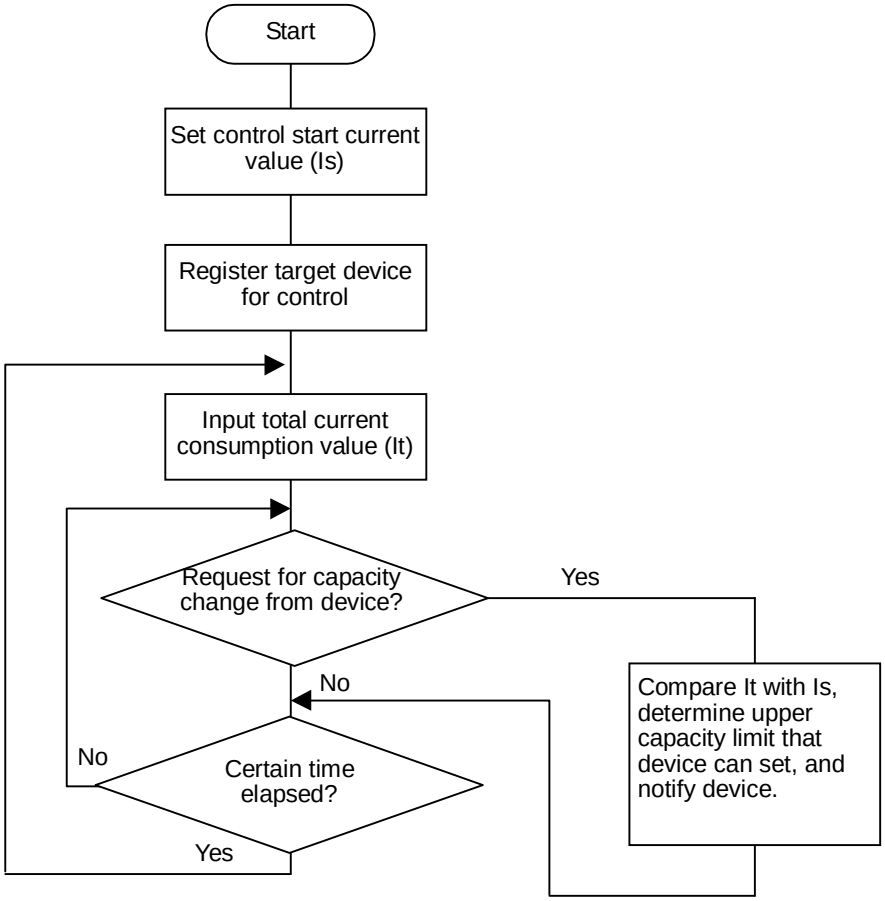


Fig. 7.3 Outline of Housing-Dedicated Feed Forward Type Peak-Cut EMS Control

(2) Controller functions

The functions of the controller required to implement housing-dedicated feed forward type peak-cut EMS control are described below.

(a) Control-target device managing function

A function to register the target ECHONET devices for EMS control

(b) Current value monitoring function

A function to monitor the current value, which is a control start condition, and to set and hold the control start current value and control end current value as a reference for device control.

(c) Device control function

A function to exert peak-cut control according to the data of the control-target device managing function and the current value monitoring function.

7.2.3 Housing-Dedicated hybrid-type peak-cut EMS

(1) Outline of controller control

Figure 7.4 shows the outline of control to be exerted by the controller supervising housing-dedicated hybrid-type peak-cut EMS control.

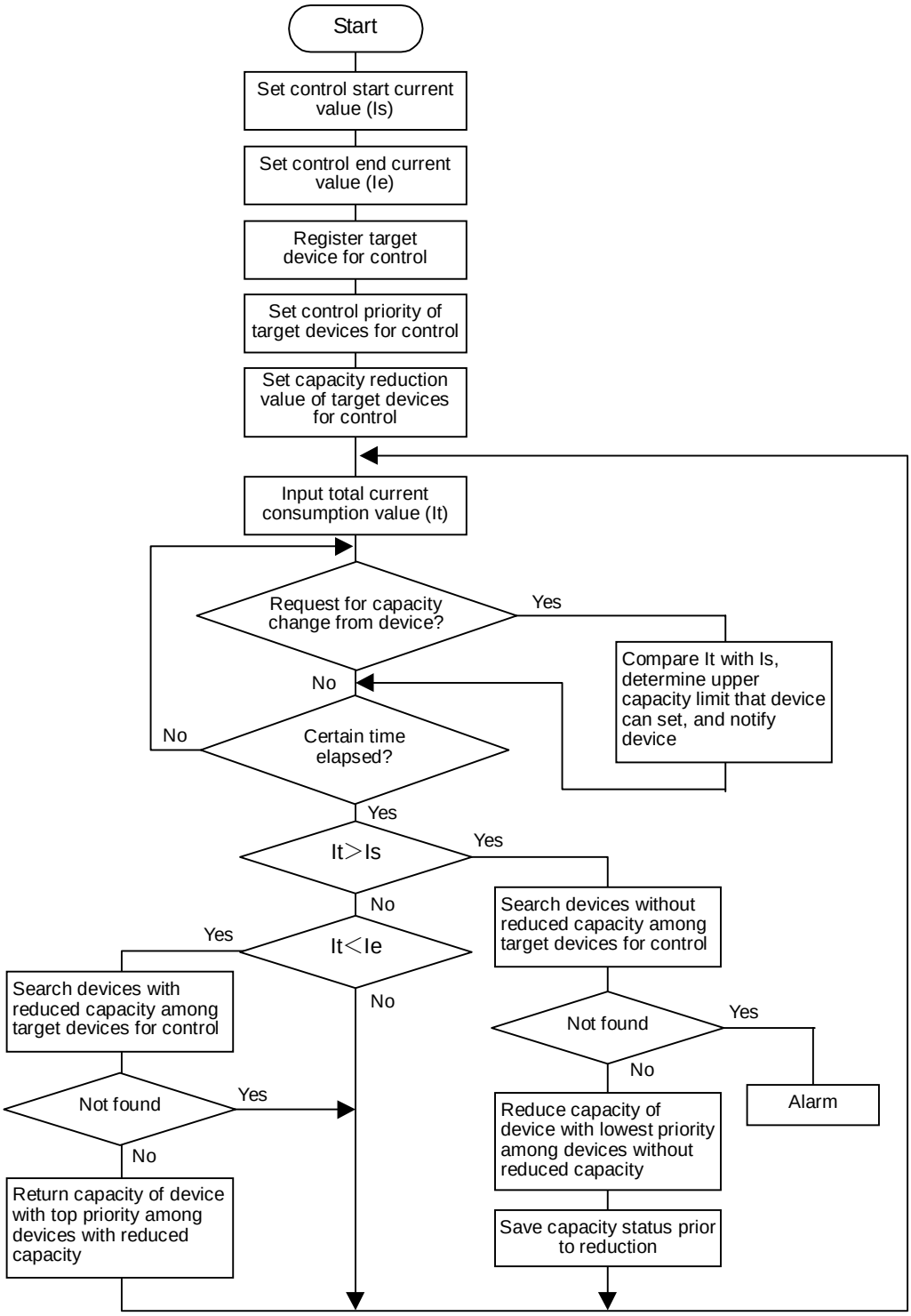


Fig. 7.4 Outline of Housing-Dedicated Feed Forward Type Peak-Cut EMS Control

(2) Controller functions

The controller functions required to implement the housing-dedicated hybrid-type peak-cut EMS control are described below.

(a) Control-target device managing function

A function to register the target ECHONET devices for EMS control and to set and hold the parameters for individual devices. The parameters to be set are shown below.

- Control priority
- Capacity reduction value
- Recovery capacity value (capacity value immediately before capacity reduction)

The target capacity for control differs with each device type. For examples of device types and control-target capacity, see Table 3.1.

(b) Electric current monitoring function

A function to monitor the amount of electric current, which is used as the trigger for activating the control function, and to set and retain the electric current values to start and stop the control function.

(c) Equipment control function

A function to select, based on the control-target equipment management function data and the electric current monitoring function data, the piece(s) of equipment for which the peak-cut control function is to be started or stopped and provide, based on the same data, instructions to start or stop the peak-cut control function for the selected piece(s) of equipment

7.3 Housing-Dedicated EMS Service Middleware Functions

7.3.1 Basic concept

The functions for housing-dedicated EMS shown as an example of housing-dedicated EMS in the previous section are divided into one part for general-purpose use and one special part for control. The former part is modeled and defined as single housing-dedicated EMS service middleware.

◇ General-purpose part

Control-target device managing function
Current value monitoring function

◇ Special part for control

Device control function

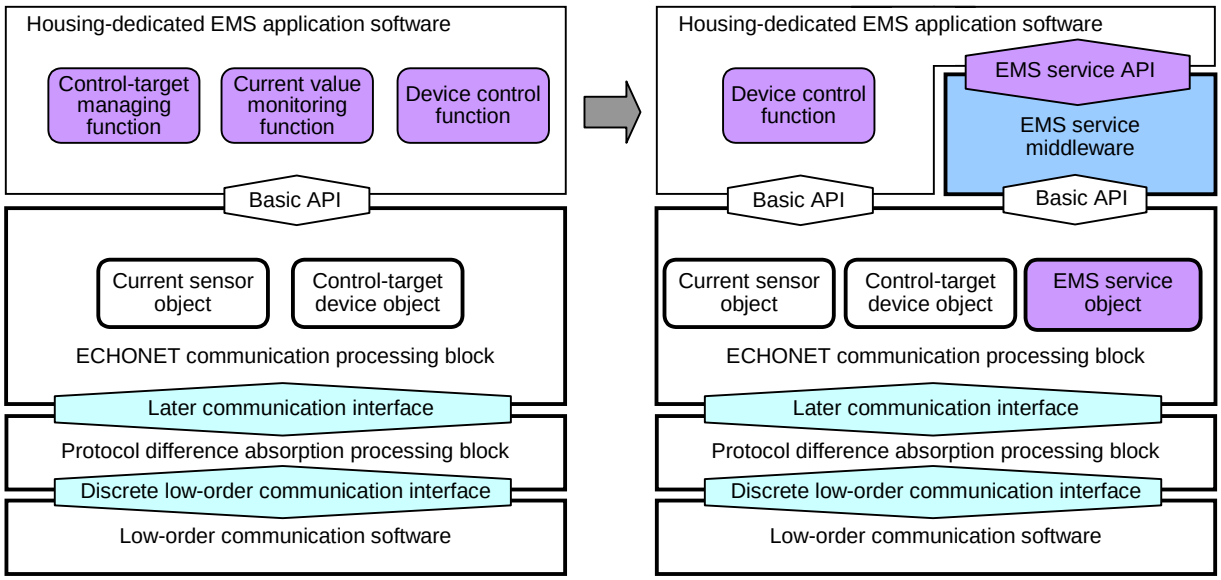


Fig. 7.5 Housing-Dedicated EMS Service Middleware and Service Object

7.3.2 Detailed functions of housing-dedicated EMS service middleware

Table 7.2 shows the functions of the housing-dedicated EMS service middleware with a control-target device managing function and a current value monitoring function for housing-dedicated EMS.

Table 7.2 Housing-Dedicated EMS Service Middleware Functions

Function name	Contents of function
Addition of control-target device	Adds device specified by ECHONET address (EA) and ECHONET object code (EOJ) to EMS control-target device list as EMS control-target device, and creates unique EMS device ID (Apparatus ID) associated with set of EA and EOJ at 1:1.
Control parameter setting	Sets control priority, capacity reduction value, recovery capacity value, and upper limit capacity value for device specified by EMS device ID (ApparatusID).
Deletion of control-target device	Deletes device specified by EMS device ID (ApparatusID) from EMS control-target device list.
Current sensor setting	Holds both EA and EOJ of current sensor to measure current value as control reference.
Measurement time interval setting	Holds current measuring time interval of current sensor.
EMS condition setting	Sets and holds control start current value and control end current value as control reference.
Current value measurement	Gets and holds measured value of current sensor at each measurement time interval.
Acquisition of current value	Gets the up-to-date current value measured by the specified current sensor.
Search for EMS control-status device group	Searches EMS device IDs (ApparatusIDs) of all devices under EMS control in EMS control-target device list.
Search for EMS non-control-status device group	Searches EMS device IDs (ApparatusIDs) of all devices not under EMS control in EMS control-target device list.
Search for EMS control-status device	Searches EMS device ID (ApparatusID) of device with top priority among all devices under EMS control in EMS control-target device list.
Search for EMS non-control-status device	Searches EMS device ID (ApparatusID) of device with lowest priority among all devices not under EMS control in EMS control-target device list.
Event generation	Compares measured value of specified current sensor with control start current value and control end current value and generates an event. Measured value > Control start current value Generation of EV_ST Measured value < Control end current value Generation of EV_SP
Capacity reduction	Sets capacity of device specified by EMS device ID (ApparatusID) to capacity reduction value. The capacity prior to this change is stored as a capacity recovery value.
Capacity recovery	Sets capacity of device specified by EMS device ID (ApparatusID) to capacity recovery value.

7.4 Housing-Dedicated EMS Service Object

7.4.1 Basic concept

Housing-dedicated EMS service classes are defined to open the functions defined in the EMS service middleware (which are shown as an example in the previous section) to another device or the controller.

7.4.2 Detailed definitions of housing-dedicated EMS service classes

Table 7.3 provides definitions of the housing-dedicated EMS service class properties.

Table 7.3 Housing-Dedicated EMS Service Middleware (1/2)

Property name	EPC	Contents of property Value range (decimal notation)	Data type	Data size	Unit	Access rule	Man-datory	Announce-ment at state change	Remark
Number of control-target devices	0xC0	Total number of target devices for EMS control 0x00～0xFF（0～255）	Unsigned char	1 byte	–	Get	○		
Control-target device list	0xC1	Arrangement of sets of (EA, EOJ). The element number becomes the EMS device ID (ApparatusID).	Array	10 bytes × Max. 255	–	Get GetM	○		
Control priority list	0xC2	Control priority of EMS-target devices. The element number becomes the EMS device ID (ApparatusID). 0x00～0xFF（0～255）	Array	1 byte × Max. 255	–	Get GetM	○		
Capacity reduction value list	0xC3	Capacity reduction value of EMS-target device. The element number becomes the EMS device ID (ApparatusID).	Array	2 bytes × Max. 255	–	Get GetM	○		
Recovery capacity value	0xC4	Recovery capacity value of EMS-target device. The element number becomes the EMS device ID (ApparatusID).	Array	2 bytes × Max. 255	–	Get GetM	○		
Upper limit capacity value list	0xC5	Upper limit capacity value of EMS-target device. The element number becomes the EMS device ID (ApparatusID).	Array	2 bytes × Max. 255	–	Get GetM	○		
EMS control status list	0xC6	List of flags to indicate whether the EMS control-target device is under control. The element number becomes the EMS device ID (ApparatusID). 0x30: Under control 0x31: Not under control	Array of char	1 byte × Max. 255	–	Get GetM Set SetM	○		
Current sensor	0xC7	EA and EOJ of current sensor as control reference	Unsigned char	6 bytes	–	Get/Set	○		
Measuring time interval	0xC8	Time interval in getting measured value of current sensor 0x0000～0xFFFF	Short	2 bytes	Sec.	Get	○		

Table 7.3 Housing-Dedicated EMS Service Middleware (2/2)

Property name	EPC	Contents of property Value range (decimal notation)	Data type	Data size	Unit	Access rule	Man-datory	Announce-ment at state change	Remark
Control start/end current value	0xC9	Current value at start/end of EMS control 2 high-order bytes: Control start current value 2 low-order bytes: Control end current value	Short × 2	4 bytes	A	Get	○		
Current value	0xCA	Up-to-date current value input from current sensor 0x00–0xFD (0–253)	Unsigned char	1 byte	A	Get/Set	○		
Event	0xCB	Holds an event Not generated: 0x30 EV_ST: 0x31 EV_SP: 0x32	Unsigned char	1 byte	–	Get			

7.5 Housing-Dedicated EMS Service API

This specification is under review in accordance with the housing-dedicated EMS service middleware specifications of Sections 1 to 3 shown at the Sample Proposal level. Only function items are presented as a Sample Proposal for the API specifications.

7.5.1 Basic concept

This service API is designed around a basic concept specified by the service object API. APIs are defined as those that are executed by calling the previously specified service middleware function.

7.5.2 List of function items

Table 3.5 shows a list of function items of the service API for accessing the housing-dedicated EMS service middleware.

Table 7.4 List of Housing-Dedicated EMS Service API Function Items

API name	Contents of function
Equipment search	Searches through the list of control-target equipment for pieces of equipment that match the specified condition(s) and returns the EMS equipment ID (ApparatusID) values of the pieces of equipment found. Search conditions: "Equipment under EMS control," "equipment not under EMS control," "equipment with highest priority for control" and/or "equipment with lowest priority for control"
Electric current sensor setting	Takes the EA and EOJ values from the electric current sensor installed to measure the amount of electric current (which is used as the trigger for activating the control function) and writes the values into the "electric current sensor" property of the EMS service object.
Measurement interval setting	Takes the electric current measurement interval value for the electric current sensor and stores the values in the "electric current sensor" property of the EMS service object.
EMS condition setting	Takes the electric current values to start and stop the control function and writes them into the "electric current value to start/stop control" property of the EMS service object.
Electric current measurement start/stop	Provides instructions to the service middleware to start/stop the measurement of the amount of electric current.
Electric current data acquisition	Reads out the value of the "amount of electric current" property of the EMS service object.
Event data acquisition	Returns the value of the "event" property of the EMS service object.
Capacity reduction	Inputs EMS device ID (ApparatusID), sets capacity of target device in capacity reduction value, and holds capacity prior to change as capacity recovery value.
Capacity recovery	Inputs EMS device ID (ApparatusID) and sets capacity of device specified by EMS device ID (ApparatusID) in capacity recovery value.

Chapter 8 Small Building/Store-Dedicated EMS Service Middleware (Sample Proposal)

This chapter describes examples of the small building/store-dedicated EMS service middleware and service objects as a basis for future discussion.

8.1 System Model

The functional requirements for the small building/store-dedicated power EMS system are analyzed. Based on these results, the small building/store-dedicated power EMS service middleware functions are defined and indicated as small building/store-dedicated EMS service objects. Figure 8.1 shows an example of a store energy management system configuration.

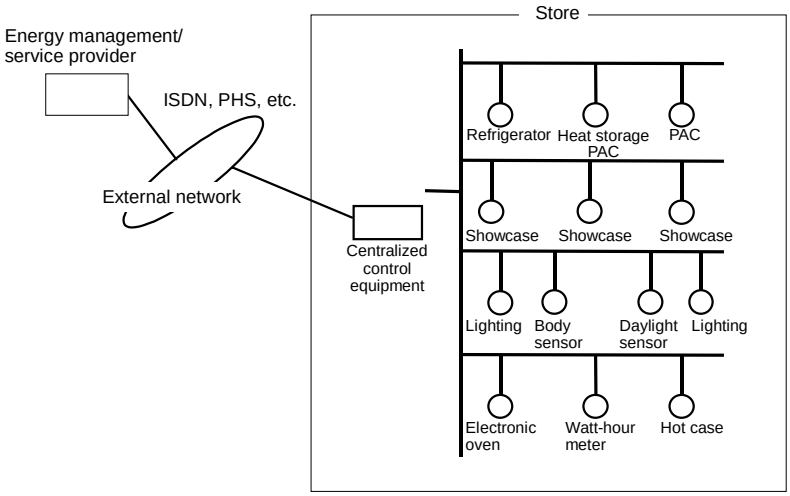


Fig. 8.1 Example of a Store EMS Configuration

Figure 8.2 shows an example of a small-building energy management system configuration.

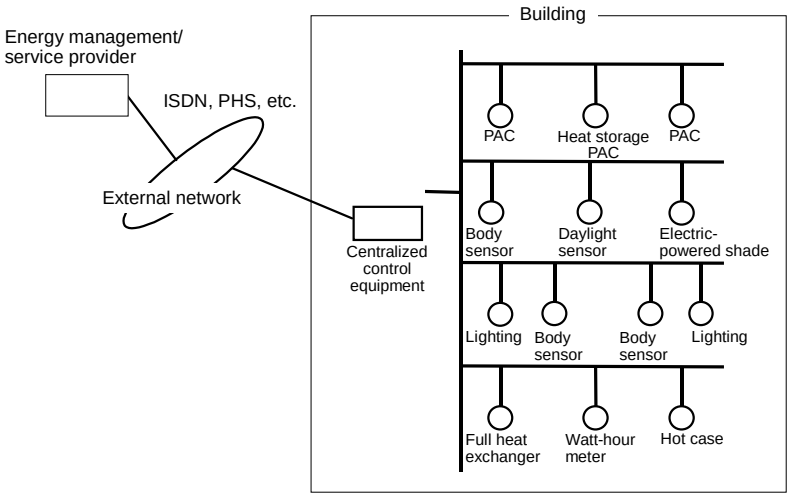


Fig. 8.2 Example of a Small-Building EMS Configuration

8.2 Small Building/Store-Dedicated EMS Functions

This section describes the functions assumed for small building/store-dedicated EMS and shows examples of each control function.

(1) Energy-saving control

- Linked action control

  Outline: Works with existent/non-existent and lighting detecting outputs, changes the operation status of each device, and executes optimum operation.

Event	Control-target device and contents of control
Human detection sensor Existent/non-existent detection	Change of operation status of lighting, etc. ON/OFF, brightness adjustment
Daylight sensor detection brightness	Lighting brightness adjustment and shade adjustment by detecting brightness

(2) Demand linked action function

- Inter-device linked action control

  Outline: Prevents simultaneous operation between specific devices.

Event	Control-target device and contents of control
Electronic oven ON/OFF	Change of hot case operation status and air conditioner set temperature control/control release

(3) Peak shift function

- Heat storage application device peak shift control

  Outline: Heat storage device control to shift air conditioner load, such as daytime cooling to nighttime power.

Event	Control-target device and contents of control
Cold/heat storage start time	Heat storage PAC: Cold/heat storage enabled
Cold/heat storage end time	Heat storage PCA: Cold/heat storage disabled

(4) Demand time zone power reduction control

- Short time zone peak shift control

  Outline: Avoids demand time zone and shifts energy use before and after an event.

Event	Control-target device and contents of control
Demand time zone start	Air conditioner setting temperature control (one degree up or down)
Demand time zone end	Air conditioner setting temperature control release (user-set temperature)

- Power consumption control and power consumption monitoring priority control

  Outline: Monitors power consumption in facilities and controls the device operation depending on increase/decrease from specified value according to previously determined priority.

Event	Control-target device and contents of control
Based on monitoring demand time zone start or power consumption, it is predicted that the specified value will be exceeded.	Changes operation contents of each device based on device priority and power monitoring result. Control example: Priority Low → high Capacity saving to air conditioner → Hot case thermo ON disabled → Reduction of lighting brightness

8.3 Small Building/Store-Dedicated EMS Service Object

8.3.1 Small building/store-dedicated EMS service class

The small building/store-dedicated EMS service classes resulting from modeling the small building/store EMS functions described in Section 3.2 are defined. Table 4.1 shows their relationship with small building/store-dedicated EMS service middleware functions.

Table 8.1 List of Small Building/Store-Dedicated EMS Service Classes

Functional classification	Class name	Outline of function working with the service middleware
(1) Demand control	(A) Demand time zone power consumption monitoring priority control class	Monitors power consumption only in demand time zone and controls it based on priority upon occurrence of each event of specified power consumption conditions.
	(B) Demand linked action control class	Demand control linked action control between devices.
	(C) Demand time zone linked action control class	Executes linked action control of demand control only in demand time zone.
	(D) Demand schedule control class	Performs specification operation at specified time.
(2) Peak shift control	Peak shift control class	Starts or ends heat storage in connection with peak shift by heat storage equipment.
(3) Energy-saving control	Energy-saving control class	Linked action control between devices

Figure 7.3 shows a small building/store-dedicated service object model diagram. The service middleware function corresponding to each object is to monitor the occurrence of an event corresponding to the purpose and to operate the device on the specified algorithm as basic operations. Properties and services such as those shown in the small building/store EMS service object are provided as super-classes of these classes. The algorithm (method) for determining execution of the monitoring target, operation target, and operation contents differs with the service middleware corresponding to each object. Demand control objects have the four sub-classes shown in Table 3.1.

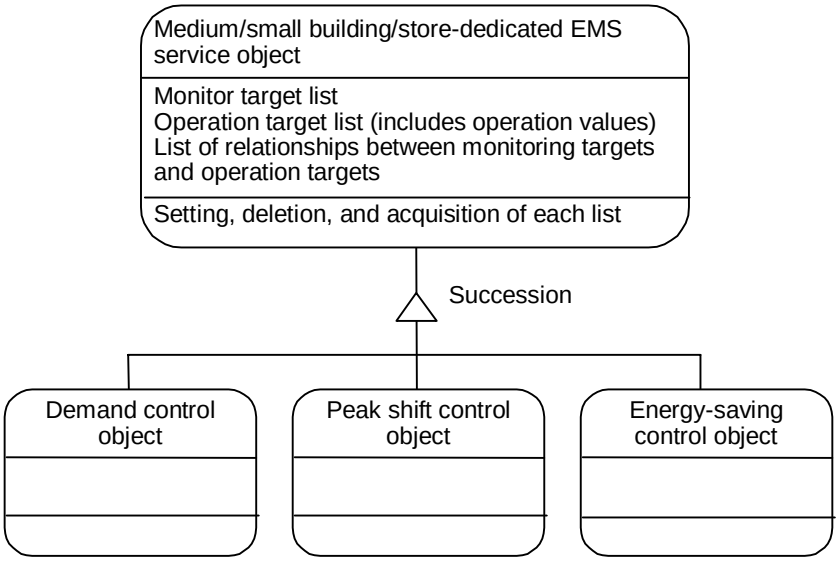


Fig. 8.3 Small Building/Store-Dedicated EMS Service Object Model Diagram

8.3.2 Details of small building/store-dedicated service classes

The details of each previously described service object are provided using data definitions that may be properties to be handled.

The target device objects for operation, monitoring, etc. (groups of properties and values of individual instances) are described below.

{Echonet Device Object} = {Echonet Address} + {Class} + {Instance} + {Property} + {Property Value}

This notation is based on the data dictionary notation method in the general function modeling technique.

{ } denotes data name.

{A} = {B} + {C} means that abstracted data A consists of data B and data C.

{B} = {a | b | c} means that data B may take the value of a, b, or c.

{A}n means that set n (n is at least one or more) of the information defined by data A is defined.

(1) Demand control

(A) Demand time zone power consumption monitoring priority control class

If the power consumption (Wh) exceeds the specified power value (Wh) during monitoring, the devices are operated in order of the preset order of priority.

- Monitoring target and conditions
  {Monitoring-target object} = {Echonet Device Object} + {Event}
  {Event} = {Over | Under| }
- Operation priority and contents
  {Contents of operation priority} = {{Priority} + {Echonet Device Object} } n

(B) Demand linked action control class

The contents of the operation of one device are changed based on the event of another device. Events and operation targets are associated with each other. Used for sensor linked action and device linked action control, this class has the following functions:

(a) Individual 1:1 linked action control

One device is operated for one input event.

- Linked action event conditions
  {Linked action condition} = {Echonet Device Object} + {Event}
  {Event} = {Over | Under}
- Operation target
  {Operation target} = {Echonet Device Object}

(b) Group linked action control 1:n n > 1

Multiple devices are operated for one input event.

- Linked action event condition
  {Linked action condition} = {Echonet Device Object} + {Event}
- Operation target
  {Operation target} = { {Echonet Device Object} } n

(c) Multiple event/group linked action control {n:1} n n > 1

Multiple devices are operated using OR/AND of multiple event conditions as an event.

- Linked action event condition
  {Event condition} = { {Echonet Device Object} + {Event} } n
- Relationship between event conditions
  {Event relation} = {Or | And}
- Operation target
  {Operation target} = { {Echonet Device Object} }n

(C) Demand time zone linked action control class

Demand control linked action control is executed only in the demand time zone. This has the following functions:

a) Demand time zone setting
  {EMS schedule} = {{Start time} + {End time}} n
  {Start time | End time} = {Day of the week} + {Hour} + {Minute}
  {Day of the week} = {Every day | Holiday | Weekday | Day | to | Sat.}

Other functions are the same as for (B) Demand linked action control.

(D) Demand schedule control class

The specified device is operated at the specified time.

- Specified time
  {Schedule} = {{Schedule No.} + {Schedule type} + {*Schedule value} n
  {Schedule type} = {Weekly | Daily}
  * {Weekly schedule value} = {Day of the week} + {Hour} + {Minute}
  * {Daily schedule value} = {Hour} + {Minute}
- Specification operation
  {Operation target} = {Schedule No.} + {Echonet Device Object} } n

(2) Peak shift control class

Peak shift control is executed by heat storage equipment. This has the following functions:

Heat storage enable/disable control

Cold/heat storage operation enable/disable is operated for the target heat storage equipment in the heat storage contract time.

Heat storage start time → Cold/heat storage operation enable

Heat storage end time → Cold/heat storage operation disable

- Heat storage time zone setting
  {Heat storage time zone}
  = {{Start time} + {End time} n
  {Time} = {Day of the week} + {Hour} + {Minute}
  {Day of the week} = {Every day | Sun. | to |Sat.}
- Operation target
  {Operation target} = { {Echonet Device Object} } n
  (Cold/heat storage enable/disable property of heat storage equipment)

(3) Energy-saving control class
The function is the same as for (B) Demand linked action control. Conditions and operation priority differ.

8.4 Sequence

Figure 8.4 shows the relationship between the small building/store-dedicated EMS service object and an application using an event trace diagram as an object model sequence.

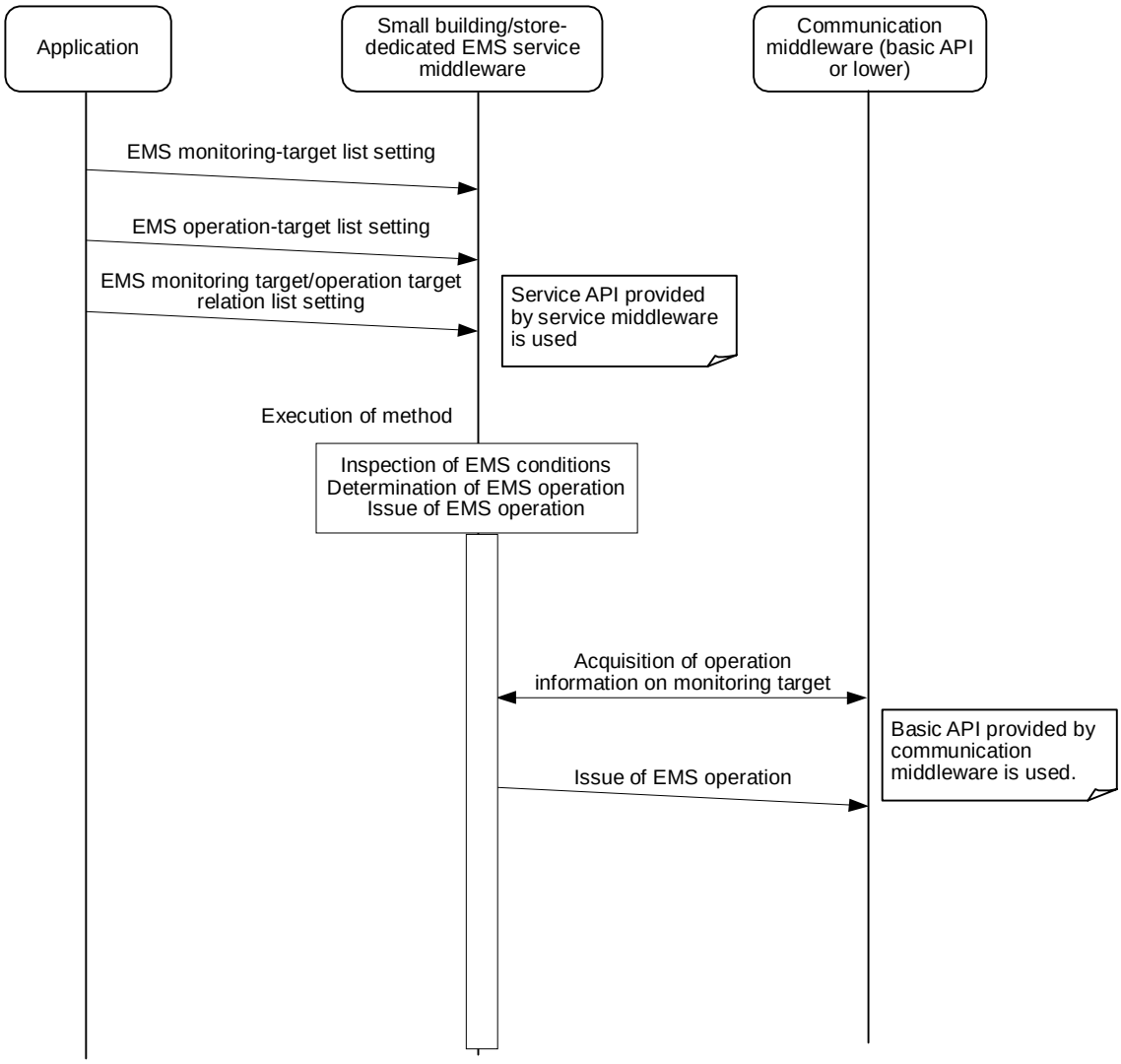


Fig. 8.4 Basic Operation Sequence of Small Building/ Store-Dedicated EMS Service Middleware

8.5 EMS Service API for Stores and Small Buildings

The specifications described in the following subsections are under development based on the specifications for the EMS service middleware for stores and small buildings, which are described in Sections 8.1 through 8.4 down to the level of suggested practical applications. Therefore, this section only describes the API specifications in terms of functions on the level of suggested practical applications.

8.5.1 Basic concept

The service API is designed based on the basic concept described in the section entitled "EMS Service Object for Stores and Small Buildings," as an API simplified to the maximum extent possible, which provides an object access interface of the super class level that abstracts all functions of the EMS service middleware for stores and small buildings as specified in Section 8.3. That is, the EMS service for stores and small buildings monitors the operation status of pieces of equipment installed on the premises using operation status data (including measurements from sensors, etc.), and when any of the predefined conditions occurs in the piece or pieces of equipment being monitored, the manipulation action that was specified in advance by the service object-specific algorithm will be performed. For this reason, this API provides the condition setting service for such monitoring and manipulation of pieces of equipment.

8.5.2 List of function items

Table 8.2 shows a list of service API function items to access the small building/store-dedicated EMA service object.

Table 8.2 List of Function Items of the Small Building/ Store-Dedicated EMS Service API

API name	Definition of function
EMS monitoring-target setting API	An API to set a monitoring target as a condition for executing EMS control. This API sets the monitoring target condition to execute EMS control for operation information value (event) changes (occurrence of event, excess or lowering of value from the specified value, etc.) on the operation information of a specific device in the EMS time zone for each EMS type (demand control, peak shift, energy-saving) in the service middleware.
EMS operation-target setting API	An API to set the contents of operation to the EMS operation target as a condition for executing EMS control. The contents are a request for updating the operation information of a specific device.
EMS control contents setting API	This API sets a combination of the above monitoring target and an operation target as a condition for executing EMS control. The service middleware issues operation to the operation target on occurrence of a monitoring condition according to this combination condition.