Part X　ECHONET System Design Guidelines

Revision History

・Version1.0	March 18th	2000 released	Open to consortium members
	July	2000	Open to the public
・Version1.01	May 23rd	2001	Open to the public
(incl. additions and corrections to Version 1.0; Parts 1, 8, 9, and 10 are unchanged from Version 1.0)			
・Version2.00	August 7th	2001	Open to consortium members
(only changes from Version 1.11 in Parts 2 and 4)			
・Version2.01	December 18th	2001	Open to consortium members
・Version2.10 Preview	December 28th	2001	Open to consortium members
・Version2.10 Draft	February 15th	2002	Open to consortium members
・Version2.10	March 7th	2002	Open to consortium members
・Version2.11	April 26th	2002	Open to consortium members

Contents

Part X ECHONET System Design Guidelines i

Chapter1 Overview 1-1

1.1 Basic Concept 1-1

Chapter2 Network Configuration 2-1

2.1 Network Configuration Requirements 2-1

Chapter3 Distributed Management System 3-1

3.1 Overview 3-1
3.2 Design Guidelines 3-2
3.3 System Design 3-3
3.3.1 System Architecture 3-3
3.3.2 System Situation-based Design 3-4
3.4 Approach to System Entry, Quittance, Registration, and Deletion 3-6
3.5 Methods of Confirming ECHONET Node Existence 3-7
3.6 System Configuration Data 3-8
3.7 System Startup 3-9
3.7.1 Definition of System Startup Situation 3-9
3.7.2 System Startup Processing 3-9
3.7.3 ECHONET Instance Management 3-10
3.8 Normal System Operation 3-11
3.8.1 Definition of Normal System Operation Situation 3-11
3.8.2 Processing During Normal System Operation 3-11
3.9 System Errors 3-12
3.9.1 Definition of System Error Situation 3-12
3.9.2 System Error Processing 3-12
3.10 System Maintenance 3-13
3.10.1 Definition of System Maintenance Situation 3-13
3.10.2 System Maintenance Processing 3-13
3.11 Traffic Specifications 3-13

Chapter1 Overview

1.1 Basic Concept

This Part will describe guidelines for network configuration and system design when designing an ECHONET-based system.

There are many types of housing, and device installation location, installation formats, and types of use will differ accordingly. In ordinary homes, users will install various devices from various vendors over time. Therefore, the system must be capable of incorporating devices from various vendors in various installations over a long period of time. ECHONET makes it possible to build a network utilizing the characteristics of ECHONET network media and to use this to create systems with a variety of architectures.

To provide users with a stable, reliable, easy-to-use multivendor system, the network must be suited to device use and must adopt a consistent design philosophy based on a unified system model. This Volume provides guidelines for the design of system architectures and network configurations based on the ECHONET specification.

The network configuration guidelines will provide a summary of network configuration constraints and requirements as seen from the perspective of system configuration.

Regarding the minimum functions to be implemented between devices and controllers, the system architecture design guidelines will present an example of a recommended system design example in the form of a distributed management system design that achieves mutually independent expandability with minimal links between controllers and devices.

Integrated control systems are another possibility. These specifications will be revised in accordance with changing user needs and verification and will also identify necessary system architectures for inclusion in this section.

Chapter2 Network Configuration

2.1 Network Configuration Requirements

(1) Network characteristics and usage in ECHONET

Power line: A wired network that can serve as the core of a facility-type system because of its ability to penetrate walls.

Special low-power wireless: A wireless network that can penetrate walls but is easily impacted by the environment. Depending on how it is used, it has the potential to serve as the core of an appliance-type system.

Infrared: Cannot penetrate walls; in appliance -type systems, is used for communications between devices within the same room.

Twisted-pair cable: Requires special wiring, but provides highly reliable communications.

(2) Connections between networks using different protocols

Networks using different lower-layer protocols are connected via an ECHONET Router. Each side of the ECHONETRouter becomes a different subnet, which is identified by a Net ID. When there is only one subnet, the Net ID assumes the default value, and the Node ID for each of the ECHONET Node is unique. When there is more than one subnet, the uniqueness of each ECHONET Node ID within each subnet is guaranteed, so ECHONETRouters cooperate to assign unique Net IDs to ensure that each ECHONET Address will remain unique throughout the domain. ECHONETRouters have the function of assigning Net IDs to ECHONET Nodes.

(3) ECHONET Router hop count

When communicating beyond the ECHONET Router, the number of times the ECHONET Router is passed is called the hop count in the ECHONET routing specifications. The maximum hop count is seven for the protocol and about three during actual use. In a home network, it is possible to connect a core network, networks for each room, and ECHONET Controllers and ECHONET Devices with two subnets. Given this network configuration, when communicating between an ECHONET Node (an ECHONET-compliant remote control) on a subnet in one room and an ECHONET Node (some kind of ECHONET Device) in another room, it would be possible to configure the system with three subnets, i.e., two hops. Therefore, a hop count of about three is recommended.

The reason for creating a limit on the number of hops is to prevent unnecessary increases in traffic when routing processing sends an incorrect frame around the network.

(4) Warning regarding duplicate routes

When configuring the network, it is important to ensure that transmission frames from the same originating device are not transmitted to the same device via a number of channels, producing unnecessary traffic and unexpected errors.

(5) Transmission delays

Transmission capabilities vary with the characteristics of the network's lower-layer protocol. For applications that place no constraints on network configuration, response wait time must be taken into account. It is possible to configure a highly reliable system by selecting a message format that confirms a response at the data link level or at the application level.

(6) Networks and domains

A domain consists of an ECHONET Router and at least one subnet which has been assigned a unique Net ID.

(7) Networks and systems

A system consists of ECHONET Nodes connected to a network comprising an ECHONET Router and at least one subnet. When ECHONET Node or subnets of other systems are to be incorporated, even partially, into a system, both systems must belong to the same domain. In other words, each Net ID must be unique throughout the networks that make up both systems. The device shown in the center of Fig. 2.1 belongs to both System A and System B. The two routers connected to the dotted-line network must have the same Net ID on both sides.

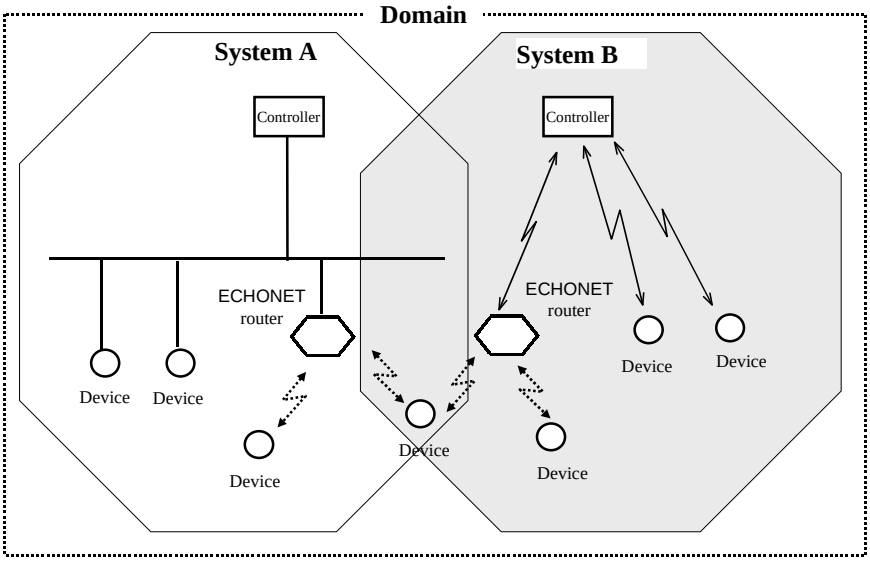

Fig. 2.1 Networks and systems

(8) ECHONET Node movement and Node IDs, Net IDs, and ECHONET Addresses

In a home network, it is assumed that ECHONET Nodes will be moved from room to room. This creates the potential for changing subnets and Node IDs, which are assigned in linkage with the MAC Address. In other words, ECHONET Addresses will change.

Chapter3 Distributed Management System

3.1 Overview

This section presents design guidelines for the configuration of a multivendor, distributed management system based on the ECHONET specification.

Most current devices incorporate microcomputers and perform intelligent, autonomous operation. A system that links such devices obtains from each only the data it requires or communicates to the device the desired operation and encourages the device to perform the operation itself. In this way, links are minimized to prevent interference with mutual development. Because the intended scope of applications for ECHONET is based on a system that will develop over time, this type of architecture is considered an appropriate response.

In the distributed management system defined in this section, autonomous functions are distributed for monitoring and operation by controllers. In other words, the system architecture assumes a format characterized by distributed control, in which monitoring and operation are performed from the necessary places.

These specifications are to be applied to the configuration of an ECHONET-compliant system based on the distributed management system philosophy described above. In this section, the term ECHONET will be omitted whenever possible; thus, unless otherwise noted, “Node” signifies an ECHONET Node as defined in Volume I; “Device,” an ECHONET Device such as home appliances, building equipment, sensors, and actuators; “Controller,” an ECHONET Device responsible for central control of the aforementioned devices or a remote control; “Router,” an ECHONET Router, “Adapter,” an ECHONET Adapter, and “Flex Device,” an ECHONET Flex Device. Also unless otherwise noted, a Device is deemed to contain a Node that consists of an Adapter and a Flex Device.

3.2 Design Guidelines

Design guidelines are as indicated below.

(1) Reliability and safety

The system must be designed to be fail-safe; it must consider safety and reliability and ensure that system and device operation will not harm users. Also, to the greatest extent possible, it should be based on the concept of distributing danger so that errors and malfunctions in individual ECHONET Controllers, ECHONET Devices, and networks will not affect other entities.

(2) Simple setup

In consideration of users and installers, the system should allow simple setup and easy installation of devices and systems by anyone.

(3) Consistency and accuracy of control and data

Data to be shared by more than one device or controller must be consistent and accurate. The system must prevent inconsistencies in system data and unnecessary, dangerous, or uncomfortable operations by devices.

(4) Device movement

Because movement of devices is expected in ordinary homes, application systems and networks must allow this and make re-setup procedures unnecessary. The system must also maintain consistency of control and monitoring data, as described above.

(5) Expandability

Home appliances, equipment, and systems have a long lifetime, and over time new devices and technologies will be incorporated. Therefore, expandability should be taken into account whenever possible.

3.3 System Design

3.3.1 System Architecture

Fig. 3.1 shows one possible system configuration. In the Fig., the octagons describe the scope of each system. System A is manufactured by Vendor A and contains four devices. System B is made by Vendor B and also contains four devices.

As shown in the Fig., each system has a controller that is responsible for central management of the operating data for each device.

In the descriptions that follow in this section, the terms “controller” and “device” signify nodes having these functions and do not limit implementation by actual products; a given node may incorporate the functions of both a controller and a device.

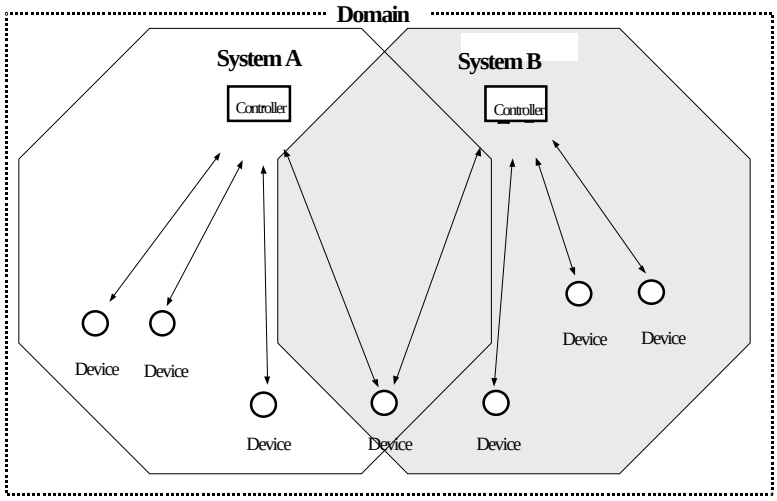

Fig. 3.1 System configuration for distributed management system

Unless otherwise specified, the system architecture, which consists of the relationships and division of functions between the controllers and various devices, has the following division of roles and functions:

・Each device and controller are designed to achieve a high degree of independence. This prevents the obstruction of development by devices and controllers (system) and also prevents controller malfunctions from impacting device operation.

・Operating data for each device is collected with the device acting as a server and the controller as a client. To the extent that it does not interfere with device operation or data security, devices actively disclose operating data and attempt to utilize it effectively in the system. Controllers guarantee the external security of the data for each device.

・Aside from cases in which the controller restricts its own operation, each device in principle

operates independently and with no awareness of the controller. However, when the controller generates an operation restricting device operation, devices manage it and, in the case of a controller malfunction, independently remove the restriction to assure safe operation.

・When a single device is operated by a number of controllers, the system must be designed to maintain consistency and to prevent hunting, etc.

The issue of whether to provide this as part of ECHONET is currently being studied.

Example 1: Use of a controller with proxy functions

A specific controller centrally manages the devices under its control and uses a proxy function to display these devices to other controllers as virtual device objects, which can be accessed by the other controllers. Thus, controller operations are issued to devices after consistency is assured by a controller application having proxy functions.

Example 2: Use of a communications lock mechanism

By locking the communications channel of the device requiring consistency from a specific controller, it is possible to prevent the device from receiving commands from other controllers while the lock is active. Study is needed on whether individual functions can be locked, etc.

・Group device actions (including linked operation) are managed by controllers without regard to devices.

・Inside the system, the range within which data can be communicated is treated as a domain, and therefore domain identification is not necessary at the application level.

・The system design assumes a multi-access network. To enable applications to perceive the network as being multi-access, it should incorporate the processing required for each lower-layer medium into the appropriate protocol difference absorption layer.

3.3.2 System Situation-based Design

(1) System Situations

Fig. 3.2 shows the definition of a situational model for the entire system which will serve as the basis for the design of controller and device system operation. Here, these situations are shown as states and defined using state diagrams. Definitions of each situation in the Fig. are shown below.

・System startup

Situation in which controllers and devices share unique identifier data and mutual connection data required for system operation.

・Normal system operation

Situation in which system is operating normally and there are no obstacles to the execution of system applications (or if obstacles exist, they are not significant enough to affect system operation).

・System error

Situation in which there are significant obstacles to system operation in the form of communications channel errors, individual device errors (including Adapters and Flex Devices), or controller errors.

・System maintenance

Situation in which actions to recover system errors are taken or in which system operating data is collected. This need not involve terminating the operation of individual devices. System operation itself may also be temporarily shut down as necessary.

As shown in Fig. 3.2, when each situation is viewed as a state, it becomes possible in principle to design each status to enable parallel operation (shown by the dotted line).

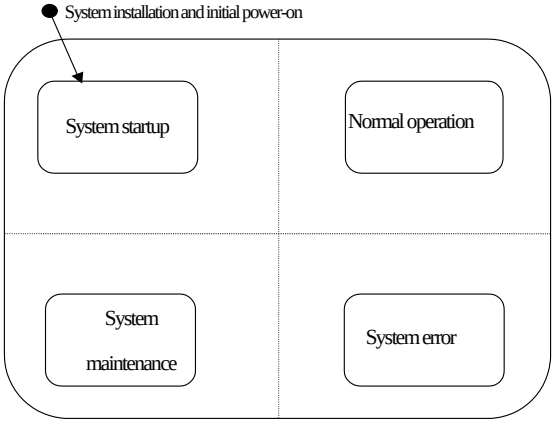

Fig. 3.2 System situation model

(2) Situation models for devices

Design should enable parallel operation of individual controllers and devices, depending on the properties of their applications, based on the situational model described above. The following cases should be considered:

Example 1: Incorporation of a new device—whether new devices can be incorporated into the system during system operation, or whether system operation must first be shut down.

Example 2: Collection of maintenance data—whether collection of device maintenance data or maintenance operations will be allowed during system operation, or whether system operation must first be shut down.

3.4 Approach to System Entry, Quittance, Registration, and Deletion

The existence of each ECHONET node is defined as follows within an ECHONET domain. In the section below, entry and quittance concern the definition of an ECHONET node's reason for existence at the ECHONET network level, while registration and deletion concern the definition of an ECHONET node's reason for existence at the application system level.

(1) Entry

State in which an ECHONET node is connected to an ECHONET network and communications is possible, i.e., the node profile objects can be accessed. "Entry" refers to this state.

New entry refers to the acquisition of an EA, which involves four processes:

[1]MAC determination→[2]Node ID →[3]Net ID determination (Net ID read request)→[4]EA determination

Re-entry refers to entering with node profile data from a previous entry (i.e., entering with data stored from before quittance) (see (2) Quittance below).

(2) Quittance

State in which an ECHONET node is removed from an ECHONET network (includes power-down) and in which communication is impossible, i.e., the node profile objects cannot be accessed.

(3) Registration

Act of storing given ECHONET node application data and EA in the system. Created as an instance list or as linked data in some ECHONET node. Unrelated to physical entry status.

System registration is performed using one of the following processes:

[1]EA data recognition →[2]Given node function recognition (functions and unique identifier data are obtained from node profile class data)→[3]Registration in instance lists of other devices as communications partner

→[3]Or: Registration as linked data (either trigger or action)

Or:

[1]Registration in linked data or instance list by device setup method or setup device via network

(4) Deletion

Act of deleting ECHONET node data and EA from the system. Deleted from instance list or linked data of some ECHONET node. Unrelated to physical entry status.

The following node management methods will not be specified:

Automatic system deletion of already-registered nodes withdrawn from ECOHNET for moving, etc.

→It is possible to use the unique identifier data described in the node profile to know node movement. By confirming the unique identifier data, it is possible to change past registration data for re-entry, etc.

Intentional quittance vs. quittance caused by network malfunction. It is impossible to distinguish between the two. These are to be managed by the appropriate application.

Handling of instances during registration and deletion will be discussed in Section 3.7.3.

3.5 Methods of Confirming ECHONET Node Existence

It is sometimes desirable for various ECHONET nodes to confirm or monitor the existence of a given node. Possible reasons include the following:

・When the nodes are in a mutual control relationship, necessitating a change in control when one of the nodes does not exist.

・When there is an immediate need to detect an error, change control, and display it to the user.

Notification of node entry within the domain is performed by announce (general broadcast) using the basic sequence described in Volume II, Section 5.3. (After entry, each application collects necessary data from the given node and registers it with the system.)

After entry, the function of managing whether quittance has occurred or not is defined as the "communications partner management function."

Communications partner management function: Uses possibility of communications to determine whether the communications partner node is in a state of entry or quittance with respect to the ECHONET network .

【Management Methods】

(1) A managing ECHONET node performs arbitrary communications to a managed ECHONET node with arbitrary timing and decides entry/quittance status based on the response received.

(2) A managing ECHONET node performs arbitrary communications to a managed ECHONET node with periodic timing and decides entry/quittance status based on the response received.

(3) A managing ECHONET node sets a communications definition for arbitrary property access with the objective of communications partner management with a managed ECHONET node and decides entry/quittance status based on the results of this communication.

(4) A managing ECHONET node decides entry/quittance status based on the results of ordinary communication with a managed ECHONET node (i.e., not solely with the objective of communications partner management). It need not be the type of communication specified in the communications definition, such as notification or periodic communications.

ECHONET does not require the special type of communication called "heartbeats." Each

application system achieves the aforementioned objectives using one of the above methods.

3.6 System Configuration Data

Data that must be clarified for each individual node or between nodes, where each node is a device, controller, or router and a system component, is defined as system configuration data. Table 3.1 shows the nodes that store this data, the storage method (volatile/non-volatile memory: whether data is erased when power is turned off), and conditions for erasing the data.

Items indicated within brackets [ ] in Table3.1 are used as necessary and are not mandatory. Also, to reduce the cost of devices, non-volatile memory will allow static setting of data in ROM, etc., at the factory.

Table 3.1 System configuration data and storage methods in the node profile class

	System configuration data name	Data definition	Data source	Node in which data is to be stored	Storage method: volatile/non-volatile	Conditions for data modification/erasure
1	ECHONET address	ECHONET address of each device	Individual nodes	Individual nodes	Volatile possible if address is re-decided at startup Non-volatile recommended	(modifiable) Subnet movement, etc.
2	Itself node instance list data	Data regarding itself node instance list pages 1 and 2 in node profile	Individual nodes	Individual nodes	Volatile possible if system is reconfigured at startup	(modifiable) Change in device comprising node, etc.
3	Property map	Property support content for each instance (for identifying implemented functions)	Individual nodes	Individual nodes	Volatile possible if system is reconfigured at startup	When modifications can be made using switches or external settings, modifications are possible when this method is used.
4	Other node EA list	EA data of communications partner node	Controllers Routers [Devices] (Settings from a settings device, etc.)	Controllers Routers [Devices]	Non-volatile recommended but not required because data is optional	(modifiable) Initialization
5	Unique identifier data	Unique data that applications use to uniquely identify devices. (ECHONET addresses may change when their connection to the network moves, but this data remains constant.)	Individual nodes	[Individual nodes]	Non-volatile	(modifiable) Rewrite settings For assuring uniqueness within the system when, after recognizing the unique data for a device, the system associates it with its own sequence number for unique management within the system and rewrites it.

ECHONET defines and specifies system configuration data as node profile objects and as individual device object properties, and these data are implemented by each node. See Volume II for more information on node profile objects.

3.7 System Startup

3.7.1 Definition of System Startup Situation

System startup refers to the action of completing registration and deletion operations by exchanging and sharing system configuration data necessary for system operation between the nodes (i.e., the controllers and devices, including both Adapters and Flex Devices) comprising the system. The situation in which this action is performed is referred to as system startup. Once system startup is completed, the relationship for the given application system has been established between the nodes.

In ordinary homes there will be frequent movement of devices and connection of new devices during system operation; therefore, functions making it possible to add devices while the system is operating are required.

3.7.2 System Startup Processing

The following explanation of system startup processing will follow the main steps in the procedure.

(1) Power on

(2) Establish network

Determine lower-layer physical layer address: Determined as per a method specified for each lower-layer transmission protocol, to be provided separately.

Example: Manual, automatic (address server YES/NO)

(3) Establish system component elements

Determine ECHONET address: Determined as per a method specified for each lower-layer transmission protocol, to be provided separately.

Establish routing data: Established as per ECHONET routing data establishment method specifications, to be provided separately.

(4) Establish system configuration data

Establish application system registration data between nodes. Registration data is exchanged as the previously described configuration data between nodes or obtained and created from user settings and is shared between controllers and devices as necessary. Communications middleware uses this data to create an object instance of the communication partner within the communications middleware, confirm the functions of each device (accessible services and properties), and determine the method for acquiring operating data (monitor? receipt of notification? periodic communications? notification address?), thereby establishing the system architecture.

(5) Changes to system component elements

Processing undertaken in response to the registration of a new device in the system or to the deletion of an already registered device. Because the quittance of already registered devices and the re-entry of withdrawn devices is expected to be performed frequently in homes, the issue of whether to include this processing at startup will be decided based on the unique characteristics of each system.

3.7.3 ECHONET Instance Management

(1) Self node instance management

ECHONET nodes manage the instances they disclose under their own responsibility by assigning instance numbers. Other nodes use these instance numbers to identify the various instances of other nodes. Therefore, when a node discloses the same device, the instance number must be stored in a non-volatile manner to prevent changes at system startup (note that it need not be stored in memory; it need only be assured, for example, by user operation and the instance creation algorithm).

(2) Other node instance management

ECHONET nodes use the instance numbers disclosed by other nodes to identify actual states within each node. Normally, therefore, this data is managed in combination with the ECHONET address of the given node and other properties shown by the node profile objects to enable continuous identification. For example, the typical method would be to determine the actual state using a combination of the ECHONET address and the instance number shown by the given node. When a node is consciously designed to handle data from other nodes, it should clearly specify those nodes as communications partners.

(3) Mutual management of instances within domains

Here, the approach to handling of communications within the domain using instance lists through ECHONET will be presented. By disclosing a database of instance lists for itself and other nodes, a node can manage instances within the domain via the network by reading and writing under its own responsibility.

By making it possible to set and obtain via the network the nodes with which each node has relationships, this has the following objectives:

・It enables an overview of node relationship data and clarifies the decomposition point of responsibility when troubleshooting.

・It facilitates system configuration maintenance within the domain.

・It enables automation of the registration and deletion processes.

・In applications with a clear algorithm for linked operation, it enables linkages.

Thus, when a node is consciously designed to handle data from other nodes, it is recommended that it display an EA list of the other nodes as communications partners. The following example will discuss the need for disclosure of this list.

Example: Sensor announces measurement results via broadcast.

Here, the sensor itself is conducting communication without an awareness of specific nodes and has no communications partners. Thus, it need not be managed as another node. The controller using the value broadcast by this sensor will include this sensor as a communications partner on its other node EA list.

When an instance is discarded from a domain, the nodes having relationships with this node can be identified by searching this other node EA list. It is also possible to manipulate the other node EA list of these nodes to delete it from the list. It should be noted that the instance has become unnecessary in only one system, and may still be needed in another system. It is also uncertain whether the instance was actually discarded or simply withdrawn temporarily. Therefore, the actions of other node EA list registration and deletion are to be performed at the application level. Notification that an instance was deleted elsewhere is reference data, and the decision of whether to delete it in the current system should be made by the application itself. Thus it is necessary to design the other node EA list registration and deletion processes such that, when performing them remotely from the network, there is adequate responsibility on both the acting and receiving sides.

Explanation

The basic approach towards linked operation of applications is linked operation between ECHONET object instances. Ordinarily, it would be necessary for linkages between devices to handle instance relationship data. Version 1.0 β, however, specifies only those properties handling only the EA of the related instances. When considering linked algorithms for applications, it is possible for nearly all functions to create a linked relationship between instances by knowing the EA. Therefore, only the minimum required functions were defined. The need for handling instance relationship data will be studied in the future following verification of reliability and ease of system configuration.

3.8 Normal System Operation

3.8.1 Definition of Normal System Operation Situation

Normal operation refers to the situation in which nodes connected to the system have obtained the necessary system configuration data, and operation is performed in accordance with the role to be played by system applications.

3.8.2 Processing During Normal System Operation

(1) Operating data setting and getting

Each node receives operating data operations (property settings) within a range that does not cause mechanical damage or other inconsistencies in its own operation. With respect to the acquisition of operating data, it discloses as much data as possible. The security of acquired operating data outside the ECHONET domain is concentrated in the functions of the gateway, which is external connections. When an inconsistency is encountered, however, the device itself takes responsibility and acts to prevent it (e.g., when the individual units of a multi-unit air conditioning system cannot be freely operated).

(2) Notification of operating data

When operating data changes, the controller is notified immediately, or else an active notification function from a device, such as periodic notification of a value, is sent to the addresses obtained during startup processing.

(3) Operating data update frequency

Each device or controller displays to the greatest extent possible the timing restrictions and update frequencies for operating data (i.e., device property data) values. This prevents unnecessary increases in traffic and enables the design of reasonable constraints on system time, such as error detection.

(4) Device movement

Devices registered with the system remain components until they are deleted; normally they are not erased by the system when temporarily withdrawn from the network because of a power outage, etc. Particularly in home applications, where movement of devices between rooms is expected, new registration data should be established as quickly and as automatically as possible following device movement. Also, each device should have a specific identifier for identification by applications. This identifier should be set when the device is shipped or based on controller settings, etc., after incorporation in the system, and should be stored in non-volatile memory. Also, it should be possible to set this identifier using communications even when it was set at the factory.

3.9 System Errors

3.9.1 Definition of System Error Situation

A system error is defined as a situation in which a network failure or controller (including Adapters and Flex Devices) or device malfunction prevent the system from achieving its stated objectives.

3.9.2 System Error Processing

Three causes of system errors are defined below. When the responsible cause is removed or resolved and the system recovers, the re-establishment of system configuration data is to be performed as automatically as possible. For problems with serious safety implications, the recovery procedure should be handled through human intervention.

(1) Communications errors and network errors

Communications errors or network errors refer to situations in which communications between nodes becomes impossible due to a network disconnection (including router failures) or a high level of noise, and in which normal communications cannot be performed, even after processing to recover routers or to recover the performance disclosed for each lower-layer protocol (error correction, retransmission, etc.). When the problem lies in the nodes, it is referred to as a communications error; when the problem is in the network transmission media, it is referred to as a network error. Here, each application performs the minimum required recovery processing in accordance with the importance of

the given communication in an attempt to continue system operation.

It is important to design the system to prevent frequent abnormal shutdowns of the system caused by increased traffic from unnecessary re-transmissions or by overly sensitive reactions to communications errors.

(2) Device errors

The system should have the function of notifying other parts of the system of device errors detected by individual devices. It should provide the controller with the necessary data, as determined by device characteristics, in accordance with the degree of urgency and level of the error. When an error occurs, it is important to prevent sharp increases in traffic other than the error announcement specified in the properties of each device object.

When a device consists of an Adapter and a Flex Adapter, it is to be configured in such a way that it is possible to determine with certainty when the Adapter malfunctions and to carry out appropriate recovery measures (e.g., when an Adapter resets independently, it should emit a Stop Notification Service continuously until the Flex Device returns the results of processing).

(3) Controller errors

The system should be designed such that controller errors do not affect the independent operation of individual devices.

3.10 System Maintenance

3.10.1 Definition of System Maintenance Situation

System maintenance is defined as the situation in which processing for the maintenance of devices or systems is performed, such as the collection of system device operating data or error history data.

3.10.2 System Maintenance Processing

(1) Data collection

Each device actively supports and discloses general maintenance data (when disclosure is mandated in the specifications) and attempts to utilize this data effectively within the system.

(2) Device and system startup and shutdown for maintenance

The system should be configured in such a way that, to the greatest extent possible, it will not be necessary to shut down a device or system to collect the aforementioned maintenance data.

3.11 Traffic Specifications

Stable system operation requires traffic specifications. Each lower-layer protocol and upper-layer application must have traffic control functions to prevent convergence.

In ECHONET, system applications are designed at the instance level. However, when considering communications network traffic and the communications processing capabilities to be implemented by each node, applications must be designed at the node level rather than at the instance level. Thus, system design is necessary for applications.

For example, when requesting data from a specific node, transmitting numerous messages without confirming a response from the intended recipient increases unnecessarily the network load by generating unnecessary messages (when recipient is absent) and inviting unnecessary re-transmissions.

Also, devices typically have limited hardware resources and lack powerful communications processing abilities compared with controllers. The transmission of requests that overload the processing capabilities of such devices renders processing impossible at the affected device, producing the need for re-transmission and increasing traffic unnecessarily.

Thus, this kind of system design should be avoided.