第１０部　ECHONET システム設計指針

改定履歴

・Version1.0　２０００年3月１８日　制定，コンソーシアム会員内公開。
　　２０００年7月　一般公開。
・Version1.01　２００１年5月２３日　コンソーシアム会員内公開。
・Version2.00 draft　２００１年8月7日　コンソーシアム会員内公開。
・Version2.01　２００１年12月19日　コンソーシアム会員内公開。
・Version2.10Preview　２００１年12月28日　コンソーシアム会員内公開。
・Version2.10Draft　２００2年2月15日　コンソーシアム会員内公開。
・Version2.10　２００2年3月7日　コンソーシアム会員内公開。
・Version2.11　２００２年4月２６日　コンソーシアム会員内公開。
・Version3.00Draft　２００２年6月１２日　コンソーシアム会員内公開。
・Version3.00　２００２年8月２９日　コンソーシアム会員内公開。
・Version3.10Draft　２００２年１１月8日　コンソーシアム会員内公開。
・Version3.10　２００２年１２月１８日　コンソーシアム会員内公開。
・Version3.11　２００３年3月7日　コンソーシアム会員内公開。
・Version3.12　２００３年5月２２日　コンソーシアム会員内公開。
・Version3.20Draft　２００３年10月17日　コンソーシアム会員内公開。

	変更部位（目次項目）	追加・変更概要
1	第４章	ECHONET プロパティ値の扱いに関する指針追加
2	第５章	ECHONET セキュア通信運用指針追加
3	3.12	通信定義オブジェクトによる連動設定追加

・Version3.20　２００４年　1月　８日　コンソーシアム会員内公開。
・Version3.21　２００４年　5月２６日　コンソーシアム会員内公開。
・Version3.30　２００４年１２月２日　コンソーシアム会員内公開。
・Version3.40 Draft　２００４年１２月２８日　コンソーシアム会員内公開。
・Version3.40　２００５年　2月　３日　コンソーシアム会員内公開。
・Version3.41　２００５年　5月１１日　コンソーシアム会員内公開。
・Version3.2　２００５年１０月１３日　一般公開。
・Version3.42　２００５年１０月２７日　コンソーシアム会員内公開。
・Version3.50 Draft　２００６年　8月　３日　コンソーシアム会員内公開。
・Version3.50　２００６年　9月２０日　コンソーシアム会員内公開。
・Version3.51 Draft　２００７年　2月　２日　コンソーシアム会員内公開。
・Version3.60　２００７年　3月　５日　コンソーシアム会員内公開。
　　２００７年１２月１１日　一般公開。
・Version4.00 Draft　２０１１年　4月　７日　コンソーシアム会員内公開。
・Version4.00　２０１１年　6月３０日　コンソーシアム会員内公開。

- エコーネットコンソーシアムが発行している規格類は、工業所有権(特許,実用新案など)に関する抵触の有無に関係なく制定されています。
  エコーネットコンソーシアムは、この規格類の内容に関する工業所有権に対して、一切の責任を負いません。
- 本規格発行者は有償・無償を問わず、いかなる第三者に対しても JAVA、IrDA、Bluetooth®、HBS のライセンスを許諾する権限や免責を与える権限を有していません。JAVA、IrDA、Bluetooth®、HBS を使用する場合、当該使用者は自己の責任と判断に基づき、上記規格について使用許可を得るなどの措置が必要です。
- この書面の使用による、いかなる損害も責任を負うものではありません。

目次

第1章 概要

１．１ 基本的考え方

本部では、ECHONET を利用してシステムを設計する際の、ネットワーク構築と、システム設計の指針について記述する。

住宅の形状は様々であり、機器の設置場所、設置形態、利用形態はさまざまである。さらに、一般住宅では、各ユーザはさまざまなベンダーの機器を時間的に分散して設置する。したがって、システムはさまざまなベンダーの機器を、さまざまな設置形態で、長い年月にわたって順次、収容可能でなければならない。ECHONET では、ECHONET の提供するネットワーク伝送メディアの特性を活かしてネットワークを構築し、さらに、これを利用して様々なアーキテクチャのシステムを構築可能である。

信頼性が高く、使い易い、安定的に運転するシステムを、マルチベンダーでユーザに提供するためには、機器の利用形態にあったネットワークの利用と、統一したシステムモデルに基づく一貫性のある設計思想が不可欠であるとの認識から、本部では、ECHONET を利用したネットワーク構築、およびシステムアーキテクチャ設計の指針を示す。

ネットワークの構築指針では、システム構築の視点から見たネットワーク構築の制約、前提条件の整理をする。

システムアーキテクチャの設計指針では、各機器、コントローラ間での、おのおのが実装すべき最低限の機能、役割の設計要件について、コントローラと機器との結合をできるだけ疎とし、相互の独立した発展性を確保する分散型管理システムの設計事例を示し、推奨システム設計事例として開示する。

他には、集中制御型のシステムなどが考えられるが、今後、ユーザのニーズや、実証などを踏まえ本仕様を改訂していくと共に、必要なシステムアーキテクチャを抽出し、本部に収容していく。

第２章 ネットワーク構築

２．１ ネットワーク構築の前提条件

（1） ネットワークの特質と ECHONET での利用の仕方

電灯線：有線のネットワークであり、壁貫通可能なことから、設備系のシステム構築において基幹系となりうるネットワーク。

特定小電力無線：無線のネットワークであり、壁貫通可能だが、環境に左右されやすい。用法によっては設備系のシステム構築において基幹系となりうるネットワーク。

赤外線：壁貫通が不可能なネットワークであり、設備系のシステム構築において、同一室内での機器と機器の間の通信に使用する。

ツイストペア線：配線コストが必要だが、高信頼な通信が可能。

（2） 異種プロトコルのネットワークの接続

異なる下位プロトコルのネットワークを接続する際は、ECHONET ルータを介して接続する。ECHONET ルータの両側でそれぞれ異なるサブネットとなり、これらは Net ID により識別する。サブネットが１つの場合は、Net ID はデフォルト値でもよく、Node ID は各 ECHONET ノードにユニークとなる。２つ以上の場合は、各サブネット内で各 ECHONET ノードのＩＤのユニーク性が保証されているので、Net ID を ECHONET ルータ間の協調によりユニークとすることで、ECHONET アドレスはドメイン全体でユニークとなる。ECHONET ルータは、この Net ID を ECHONET ノードに付与する機能を有する。

（3） ECHONET ルータのホップ数

ECHONET ルータを超えて通信をする場合、ECHONET ルータを超える回数を ECHONET ルーティング仕様におけるホップ数という。ホップ回数の制限は、プロトコル上は7回であるが、運用上は3回程度とする。例えば、なんらかの基幹ネットワークと各部屋内でのネットワークという複数のサブネットを持つ住宅内を想定すると、２つのサブネットで ECHONET コントローラと ECHONET 機器とを接続可能である。こうしたネットワーク構成で一方の部屋のサブネット上の ECHONET ノード（ECHONET 仕様のリモコン）と他方の部屋の ECHONET ノード（なんらかの ECHONET 機器）との間で通信を行う場合、サブネットは3つ、すなわちホップ数は、２回で構成可能である。これらのことから、３回程度を推奨としている。制限を設ける理由は、ルーティング処理により誤ったフレームがネットワーク内をまわり続け、無駄なトラヒックの増大がないようにするためである。

（4） 2 重経路への注意

ネットワークを構築する際は、同一発信元機器の送信フレームが、複数の経路で同一機器に伝達されることにより、無駄なトラヒックや、予期せぬ動作不良を起こすことがない

ように注意すること。

（5） 伝送遅延

ネットワークの下位プロトコルの特質により、伝送能力が異なる。ネットワーク構成を限定しないアプリケーションは、応答の待ち時間を考慮することが必要である。データリンクレベル、あるいはアプリケーションレベルでの応答を確認するメッセージ形式を選択するなどの手段により、信頼性の高いシステムを構築する事が可能である。

（6） ネットワークとドメイン

ドメインは､ユニークなNet IDを与えられている１つ以上のサブネットと､ECHONETルータで構築される。

（7） ネットワークとシステム

システムは、１つ以上のサブネットと、ECHONET ルータで構築されたネットワークに接続されたECHONETノードによって構成される｡他のシステムと同一のECHONETノード、あるいはサブネットを一部でもシステムに収容する場合は、両システムは、同一ドメインに属さなければならない。すなわち、Net IDは、両システムを構成するネットワーク全体にユニークでなければならない。図２．１における中央の機器は、システムＡ、システムＢの両システムに属している。点線のネットワークに接続されている２つのルータは、それぞれの片側で、同じ Net ID が設定されていなければならない。

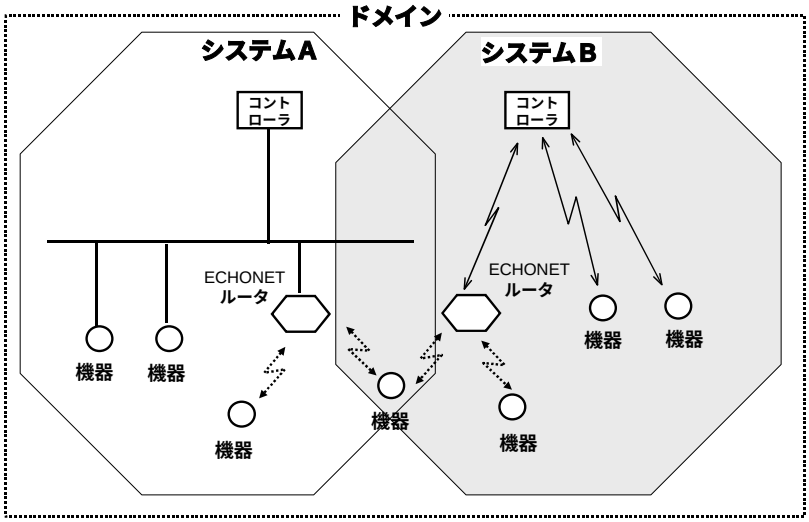


図2．１　ネットワークとシステム

（8） ECHONETノードの移動と、Node ID、Net ID、ＥＣＨＯＮＥＴアドレス

住宅における ECHONET ノードは、部屋から部屋へ移動することが想定される。すなわち、サブネット、およびMACアドレスに連動して決定されるNode IDは変る可能性がある。このことは、ECHONETアドレスが変わることを意味する。

第３章 分散型管理システム

３．１ 概要

本章は、ECHONET を適用し、分散型管理システムをマルチベンダーで構築する際の、システムの設計指針について示す。

現在の多くの機器は、個々にマイクロコンピュータを搭載し、インテリジェントな自律動作を行う。これらを有機的に連携させるシステムは、自己が必要とする情報だけを必要な機器から取得し、あるいは、自己の要求する動作を相手機器に伝え相手機器の責任範囲で動作を促す、互いにできるだけ疎な結合が相互の発展を阻害しない。ECHONET の対象とするフィールドは、時間的にも長い時間をかけて発展していくシステムという特質を持つため、このようなアーキテクチャが適した解のひとつと考える。

本章で定義する分散型管理システムでは、自律した機能が分散して配置され、これらをコントローラが監視、操作する、すなわち、制御は分散、監視、操作は必要な複数の場所からという形態が想定されるシステムのアーキテクチャである。

本規定は、ECHONET を使用し、上記分散型管理システムの思想にもとづくシステムの構築に適用する。本章では、特にことわらない限り以下のように第 1 部で定義する「ECHONET○○」を簡略化して用語を使用する。「ノード」とは第 1 部にて定義する「ECHONET ノード」、「機器」は同様に「ECHONET 機器」のうち、家電機器やビル設備機器、センサ、アクチュエータなどを指し、「コントローラ」とは、「ECHONET 機器」のうち、前述の家電製品等を監視、操作する集中制御装置や、リモコンなどを指す。また、「ルータ」とは「ECHONET ルータ」を意味し、「アダプタ」は「ECHONET アダプタ」を、「フレックス機器」は「ECHONET フレックス機器」を意味する。なお、特にことわらない限り「機器」は、「アダプタ」と「フレックス機器」により構成されるノードも含むものとする。

３．２ 設計指針

下記の事項を設計指針とする。

（1） 信頼性、安全性

システム、機器の運転により、ユーザに危害を加えないよう、安全性、信頼性に留意することを必須とし、フェイルセーフなシステムを設計すること。また、可能な限りおのおのの ECHONET コントローラ、ECHONET 機器、ネットワークの異常、故障が他に影響をおよぼすことの無いように、危険分散を思想とすること。

（2） 省設定

ユーザ、施工者等に配慮し、誰でも簡単に機器、システムを設置可能とし、省設定なシステムであること。

（3） 制御、情報の一貫性、正確性

複数の機器、コントローラで共有する情報は、一貫性、正確性を確保し、機器が無駄、危険、不快な動作をすることや、システムの情報に矛盾を生じるなどのことがないようにすること。

（4） 機器の移動

一般住宅では、機器の移動が想定されるため、アプリケーションシステム、およびネットワークにおいても、これを可とし、これに伴う再設定動作は不要とし、また上記の制御、監視情報の一貫性なども保持されること。

（5） 拡張性

家電製品、設備機器、システムの寿命は長く、年月とともに新しい機器や、技術の導入が図られる。したがって、可能な限り拡張性を考慮すること。

３．３ システム設計

３．３．１ システムアーキテクチャ

図３．１に、本システムが想定するシステム構成例を示す。図３．１において、各八角形の囲みがシステムの範囲を示している。システムＡは、ベンダーＡによって製造されたシステムで、左側４つの機器を収容している。システムＢは、ベンダーＢによって製造されたシステムで、右側４つの機器を収容している。

図３．１のように、各システムは、コントローラを有し、各機器の運転情報を集中管理している。

なお、本章における以下の説明でコントローラ、機器とは、当該機能を実装したノードを意味し、必ずしも実製品の実装を制限するものではなく、いずれかのノードがコントローラ機能、機器機能を兼ね備えることは可である。

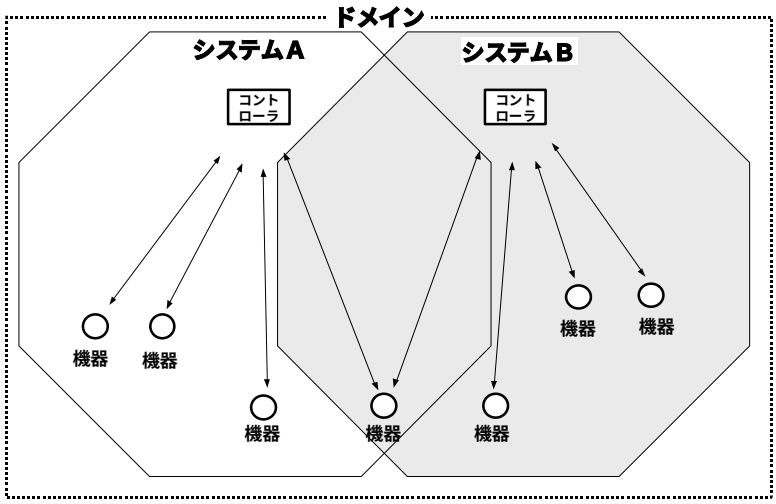


図３．１　分散型管理システムのシステム構成例

コントローラと、各機器の関係、機能分担、すなわちシステムアーキテクチャは、特に本書で規定されている場合を除き、原則下記のような機能、役割分担とする。

・各機器とコントローラは、独立性高く設計する。これにより、機器、コントローラ（システム）の相互の発展を阻害しないようにするとともに、コントローラの故障により、機器の運転への支障が無いようにする。

・各機器の運転情報については、機器をサーバ、コントローラをクライアントとして収集する。機器は、特に自己の運転や、情報のセキュリティに支障が無い場合は、積極的に自己の運転情報を開示し、システムでの有効な活用を図る。各機器の情報の宅外へのセキュリティは、コントローラ側で保証すること。

・各機器は、コントローラが自己の運転を制限している場合以外は、特にコントローラ

を意識することなく、原則的に自律動作するものとする。ただし、コントローラからの運転制限に関する操作が発生している場合には、これを管理し、たとえば、コントローラが故障した場合などは、自律的に制限を解除し、安全な運転を実施することとする。

・複数のコントローラからの同一機器の操作における運転は、ハンチング動作等を招かないように、整合性ある設計をすること。

今後これを ECHONET のしくみで提供することを検討中である。

例１：プロキシ機能をもつコントローラによる実現

特定のコントローラが、管理対象にしている機器を集中的に管理し、これをプロキシ機能で他のコントローラに仮想的な機器オブジェクトを開示し、他のコントローラはこれをアクセスする。これにより、各コントローラの操作は、いったんプロキシ機能を持つコントローラのアプリケーションで整合を取られた後、各機器に発行される。

例２：通信ロック機構

整合性を取りたい機器の通信チャネルを特定のコントローラからロックすることで、その間は、他のコントローラからの指令を受け付けない。特定機能毎にロックするなどの検討が必要である。

・機器のグループ(連動運転等を含む)動作は、機器は意識することなく、コントローラで管理すること。

・システム内部では、通信によって情報交換可能な範囲をドメインの範囲として扱うため、ドメインの識別はアプリケーションでは不要。

・システム設計上は、マルチアクセス方式のネットワークを想定する。アプリケーションからはマルチアクセス方式とみなすことが可能なように、下位の伝送メディア毎に必要な処理を適宜プロトコル差異吸収層にて実装することが好ましい。

３．３．２ システムの場面に基づく設計

（１） システムの場面

コントローラ、機器のシステム運転の設計の基本となるシステム全体の場面モデルを定義し、図３．２に示す。ここでは、それぞれの場面を状態として示し、状態図を用いて定義している。図中の各場面の定義を以下に示す。

・システム立ち上げ

コントローラ、機器間で、システム運転に必要な相互の接続情報、個体識別情報の共有を図る動作を行う場面。

・システム通常運転

システムの定常状態で、システムアプリケーションの実行に障害無い、あるいは、一部の障害が発生していても、支障の無い範囲で動作可能な場面。

・システム異常

通信路の異常や、個々の機器（アダプタやフレックス機器を含む）の異常、またはコントローラの異常によりシステムアプリケーションの実行上障害がある場面。

・システムメンテナンス

システム異常の復旧動作や、システムの運転情報を収集するなど場面。個々の機器の運転を必ずしも停止するわけではない。必要に応じ、システム運転を一時的に停止させる場合もある。

図３．２に示すように、各場面を状態としてとらえると、おのおのの状態は、基本的に並行動作可能（点線で区分している）な設計が可能である。

図３．２　システムの場面モデル

（２） 各機器の場面モデル

上記システムの場面モデルに基づき、個々のコントローラ、機器は、それぞれのアプリケーションの性質により、並行動作可能か否かの設計を行うこと。次のようなケースが考えられる。

例１：新たな機器の収容；新たな機器のシステムへの追加収容を、システムの運転中に可能とするか、あるいはシステムの運転を停止してからとするか。

例２：メンテナンス情報の収集；機器のメンテナンスデータの収集や、メンテナンス作業を、システムの運転中に可能とするか、運転を停止してからとするか。

３．４ システムへの参入、離脱、登録、抹消の考え方

ECHONET ドメイン内において、各 ECHONET ノードの存在について次のように定義する。下記において、ECHONET ネットワークレベルにおける ECHONET ノードの存在意義を定義するものが参入、離脱であり、ECHONET 応用のアプリケーションシステムにおける ECHONET ノードの存在意義を定義するものが登録、抹消である。

（1）参入

ECHONET ノードが ECHONET ネットワークに接続され、通信が可能な状態、すなわちノードプロファイルオブジェクトにアクセス可能な状態。「参入する」とは上記状態になることである。

新規参入とは、EA を取得すること、すなわち、

①MAC 決定→②ノード ID→③NetID 決定（NetID 読み出し要求）→④EA 決定

の４つのプロセスを経て参入となる。

再参入とは、過去に参入していた時のノードプロファイル情報で参入すること(離脱前に保持していた情報で参入すること（離脱→（2）参照)。

（2）離脱

ECHONET ノードが ECHONET ネットワークからはずれ（電源 off も含む)、通信が不可能、すなわちノードプロファイルオブジェクトにアクセスできない状態。

（3）登録

システムに当該 ECHONET ノードのアプリケーション情報が EA とともに記憶保持されること。いずれかの ECHONET ノードに連動情報、またはインスタンスリストとして生成される。参入しているかどうかという物理的な状態とは無関係である。

すなわち、

①ＥＡ認識→②当該ノードの機能認識（ノードプロファイルクラス情報から機能、個体識別情報を取得）→③通信相手として他機器インスタンスリストへの登録

→③または、連動情報（トリガ、アクションのいずれか）として登録

あるいは、

①機器の設定手段、ネットワークを介した設定器などによるインスタンスリスト、連動情報への登録

など、いずれかのプロセスを経てシステムへの登録となる。

（4）抹消

システムから当該 ECHONET ノードの情報が EA とともに消滅すること。いずれかの ECHONET ノードから連動情報、またはインスタンスリストから消滅する。参入してい

るかどうかという物理的な状態とは無関係である。

次のようなノード管理の方法は、規格化しない。

登録済みのノードが移動などのためにエコーネットから離脱した際、これをシステムが自動抹消するか否か

→ノードの移動を知る方法として、ノードプロファイルに記述されている個体識別情報が利用可能である。個体識別情報を確認する事により、再参入時に過去の登録情報の変更利用等が可能である。

故意の離脱と、ネットワーク故障による離脱。これらの違いの検出は不可能である。これらは、適宜アプリケーションにて管理すること。

登録、抹消における、インスタンスの取り扱いは、３．７．３を参照のこと。

３．５ ECHONET ノードの生存状況の確認方法

ある ECHONET ノードの生存状態を、さまざまな他の ECHONET ノードがそれぞれの目的で確認したい、監視したいというニーズがある。それぞれの目的とは、下記のようなものが考えられる。

・相互に制御関係にあり、相手が存在しない状況において制御を変更する必要がある場合

・直ちに異常を検出して、制御を変更する、ユーザに表示するなどの必要がある場合

ノードの参入は、第２部５．３節の基本シーケンスにあるようなアナウンス（一斉同報）にてドメイン内に通知する。(参入後は、各アプリケーションにて当該ノードへの必要な情報収集を行い、システムに登録する)

参入後、上記の離脱か否かを管理する機能を「通信相手管理機能」と定義する。

通信相手管理機能：通信の可否により、通信相手 ECHONET ノードが ECHONET ネットワークに参入状態にあるのか、離脱状態にあるのかを把握すること。

【管理方法】

（1）管理側 ECHONET ノードが、任意のタイミングで被管理側 ECHONET ノードに任意の通信を行い、その応答から参入／離脱を判断する。

（2）管理側 ECHONET ノードが、定期的なタイミングで被管理側 ECHONET ノードに任意の通信を行い、その応答から参入／離脱を判断する。

（3）管理側 ECHONET ノードが、被管理側 ECHONET ノードとの間に特に通信相手管理を目的として任意のプロパティアクセスする通信定義を設定し、当該通信結果により参入／離脱を判断する。

（4）管理側 ECHONET ノードが、被管理側 ECHONET ノードとの間で通常に使用する通信の結果（特に通信相手管理のみを目的とするものではない）により参入／離脱を判断する。これは、通信定義にて設定した通知や、定期的な通信等、通信の種別は問わない。

ECHONETでは、いわゆるハートビートと言われる特定の通信の義務付けは行わない。

上記方法のいずれかを使用して、各アプリケーションシステムは上記目的を実現する。

３．６ システム構成情報

システムの構成要素であるノード（機器、コントローラ、ルータ）の情報として、各ノード、およびノード間で明確にする必要のある情報をシステム構成情報と定義し、また、各システム構成情報は、これを保持すべきノード、保持方法（揮発／不揮発記憶：電源断時などに抹消されるか否か)、クリア条件等について表３．１に示す。

なお、表３．１中［　］内表示は適宜、必要に応じ取り扱うもで、必須ではないものとする。また、機器へのコスト負担を下げるため、不揮発記憶の形態は、ＲＯＭ等で生産時に静的に固定する方法も可能とするものがある。

表３．１　ノードプロファイルクラスにおけるシステム構成情報と保持方法

	システム構成情報名	情報定義	情報源	情報を保持すべきノード	保持方法 揮発／不揮発	情報変更・クリア条件
1	ECHONET アドレス	各機器の ECHONET アドレス	各ノード	各ノード	動作開始時に再決定していれば揮発可 不揮発を推奨	（変更可） サブネット移動時等
2	自ノードインスタンスリスト情報	ノードプロファイルの自ノードインスタンスリストページ1、2に関わる情報	各ノード	各ノード	動作開始時に再構築していれば揮発も可	（変更可） ノードを構成する自機器の変更時
3	プロパティマップ	・各インスタンスのプロパティサポート内容（実装している機能の識別を目的とする）	各ノード	各ノード	動作開始時に再構築していれば揮発可	スイッチや外部設定により変更が可能な場合は、当該方法実行時変更可能
4	他ノード EA リスト	・通信相手とするノードの EA 情報	コントローラ ルータ ［機器］ ※設定器等からの設定	コントローラ ルータ ［機器］	基本的には不揮発を推奨するが、オプション情報のため不揮発必須とはしない	（変更可） 設定初期化時
5	個体識別情報	アプリケーションが機器を個体識別するためのユニークな情報。（ECHONET アドレスは、移動によるネットワークへの接続位置によって変化するおそれがあるが、本情報は変化しない。）	各ノード	［各ノード］	不揮発	（変更可） 書き換え設定時 システムは、機器の個体情報認識後は、これをシステム内の個体管理の独自の通番と関連づけて書き換えることなどにより、システム内でのユニーク性を確保する事ができる

ECHONET では、システム構成情報をノードプロファイルオブジェクトや、各機器オブジェクトのプロパティとして定義、規定されており、各ノードはこれを実装する。ノードプロファイルオブジェクトの詳細は第2部を参照のこと。

３．７ システムの立ち上げ

３．７．１ システム立ち上げ場面の定義

システムの立ち上げとは、システムを構成するノード、すなわちコントローラ、機器（アダプタ，フレックス機器含む）間で、相互にシステム運転に必要なシステム構成情報を交換し、共有を図ることにより、登録、抹消を完了する動作とする。この動作を行う場面をシステムの立ち上げとする。システムの立ち上げが完了すると、ノード間は、当該アプリケーションシステムでの関係を確立した状態となる。

一般に宅内では機器の移動や、システムの運転中に新規の機器を接続するケースが多く、システム運転中でも機器をシステムに収容していく機能が必要である。

３．７．２ システムの立ち上げ処理

システムの立ち上げ処理を、主な手順に沿って示す。

（1） 電源投入

（2） ネットワークの確立

下位物理層のアドレスの決定:別途、下位の伝送プロトコル毎に規定の方法で決定する。

例： 手動、自動（アドレスサーバの有無）

（3） システムの構成要素の確立

ECHONET アドレスの決定：別途、下位伝送プロトコル毎に規定の方法で決定する。

ルーティング情報の確立：別途、規定の ECHONET ルーティング情報の確立方法で、確立する。

（4） システムの構成情報の確立

ノード間の、アプリケーションシステム上の登録情報を確立する。登録情報は、前記システム構成情報をノード間で交換、あるいは、ユーザからの設定により取得、生成し、必要に応じコントローラと機器間で共有する。通信ミドルウェアでは、この情報により、通信ミドルウェア内に通信相手のオブジェクトのインスタンスを生成し、各機器の機能（アクセス可能なサービス、プロパティ）の確認や、運転情報の取得方法（モニタか、通知受信か、定時通信か、通報先は？）などを決定し、システムアーキテクチャを確定する。

（5） システム構成情報の変更

新たな機器のシステムへの登録や、あるいは既登録機器の抹消への対応が発生した場合の処理。既登録機器の離脱や、離脱機器の再参入については、宅内では、通常頻繁に行われると予想されるため、これを立ち上げでの処理に含めるか否かは、当該システムの特質により設計するものとする。

３．７．３ ECHONET におけるインスタンス管理

（1） 自ノードインスタンス管理

ECHONET ノードでは自ノードの公開する各インスタンスに自己の責任においてインスタンス番号を付与して管理する。他のノードは、当該インスタンス番号を利用して、他ノード内の複数のインスタンスを識別する。よって、各ノードでは同じ実体の機器を開示する際に、起動の都度インスタンス番号が変化することの無いように、不揮発保持しなければならない（必ずしもメモリ手段を用いるという意味ではなく、ユーザの運用と、インスタンス生成のアルゴリズムなどによって保証されれば良い）。

（2） 他ノードインスタンス管理

ECHONET ノードでは、他ノードの公開するインスタンス番号を利用して各ノードにおける実体を識別する。よって、通常、当該のノードの ECHONET アドレスや、ノードプロファイルオブジェクトの示す他のプロパティと組み合わせて継続的に識別できるように管理する。たとえば、ECHONET アドレスと、当該ノードの示すインスタンス番号と組み合わせて実体を把握する方法が一般的と考える。各ノードは、意識的に他ノードの情報を取り扱う場合は、そのノードを通信相手として明示すること。

（3） ドメイン内のインスタンスの相互管理

ECHONET を通じて、インスタンスリストを使用したドメイン内での通信関係の取り扱いの考え方を示す。自ノード、あるいは、他ノードのインスタンスリストのデータベースを各ノードが公開することにより、これを各ノードの責任において相互に読み出し、書き込みすることで、ネットワークを介したドメイン内のインスタンスの管理が可能となる。

これは、各ノードがどのノードと関係しているかについて、ネットワークを介して設定、取得が可能となるものであり、つぎのことを目的にしている。

・ノード間の関係情報を一望できるようにし、トラブルシューティング時の責任の分解点を明確にする。

・ドメイン内のシステム構成のメンテナンスを容易にする。

・登録、抹消の自動化を可能にする。

・連携運転のアルゴリズムが明確なアプリケーションにおいては、これにより連動を実現可能である。

よって、各ノードは、意識的に他ノードの情報を取り扱う場合は、そのノードを通信相手として他ノード EA リストとして明示することを推奨する。他ノード EA リストの公開是非について例をあげる。

例；センサは、同報で計測結果をアナウンス。

この場合、センサ自身は、特定ノードを意識しない通信なので、通信相手先を持たない。よって、他ノードとして管理する必要は無い。このセンサの同報値を使用するコントローラ側は、通信相手として当該センサを他ノード EA リストにリストアップする。

あるインスタンスがドメインから廃棄された場合、上記の他ノード EA リストを検索することによって、当該ノードを関係先にしているノードが判明する。また、これらのノードの他ノード EA リストを操作し、リストから抹消操作することが可能である。ここで注意すべきことは、他方のシステムで不要になっただけで、一方のシステムでは必要なこともありえる。また、真に廃棄なのか、単なる一時離脱なのかも不明である。よって、他ノード EA リストの登録, 抹消は、アプリケーションレベルで判断、実施すべき操作である。他で抹消したことを通知することは、参考情報であり、自己の中を抹消してよいかは、自己のアプリケーションにて判断すべきことである。ネットワークから遠隔で他ノード EA リストの登録、抹消操作を行う側、される側共に、責任を十分に考えて実行する設計が必要である。

※ 解説

アプリケーションの連携動作の基本的な考え方は、ECHONET オブジェクトのインスタンス間の連携動作である。本来であれば、機器間の連動はインスタンスとインスタンスの関係情報を取り扱う必要がある。しかし、Ver.1.0 では関係先の EA のみを取り扱うプロパティのみ定義する。アプリケーションの連動アルゴリズムを考える場合、ほとんどの機能が EA を知ることで、インスタンス間の連動の関係を生成、実現することが可能であることから、最低限の機能のみを定義するにとどめた。今後、システム構築の容易さや、信頼性などについて検証後、インスタンス間の関係情報を取り扱うことの要否を検討する。

３．８ システムの通常運転

３．８．１ システムの通常運転場面の定義

システムに接続されたノードが必要なシステム構成情報の取得を完了し、システムのアプリケーションが果たす目的に対応した運転を実行する場面を、通常運転という。

３．８．２ システム通常運転時の処理

（1） 運転情報の操作、取得

各ノードは、自己の運転に機械的損害などの不整合を起こさない範囲で、運転情報の操作を受け付ける（プロパティへの設定）。また、運転情報の取得に対しても、可能な限り公開するものとする。取得した運転情報のドメイン外へのセキュリティは、外部接続をするゲートウェイの機能に集約する。ただし、運転に不整合がある場合は、機器側で責任を持ってこれを防止するように動作すること（たとえば、マルチエアコンで、空調機毎に冷暖自在に運転できない場合など）。

（2） 運転情報の通知

運転情報の変化時にこれを直ちにコントローラに通知する、あるいは、定時間毎に値を通知するなどの機器からの能動的な通知機能は、立ち上げ処理で取得した通知先に通知する。

（3） 運転情報の更新周期

各機器、あるいはコントローラでは、取り扱う運転情報（機器のプロパティ情報）の値の更新周期、タイミング制約を可能な限り明示すること。これにより、必要以上のトラヒックの増加を防止し、あるいは異常検知など、システム内の時間に関する制約条件をリーズナブルな値で設計可能なようにする。

（4） 機器の移動

システムへの登録状態に有る機器は、抹消しない限りは、システム構成要素の一員であり、電源 off などによるネットワークからの一時的な離脱の場合、通常、システムでは当該機器を抹消しない。特に住宅でのアプリケーションでは、機器の部屋間の移動が想定されるため、移動後にすみやか、かつできるだけ自動的に新たな登録情報を確立すること。また、アプリケーションにて機器の個体識別が可能なように、各機器は個体識別子を保持しておくこと。個体識別子は、工場出荷時、あるいは、システムに収容後、コントローラからの設定などにより、設定され、不揮発で記憶しておくことが望ましい。また、工場出荷時に設定する場合でも、通信により設定できることとする。

３．９ システム異常

３．９．１ システム異常場面の定義

ネットワークのダウン、コントローラ、機器（アダプタ、フレックス機器含む）の故障などにより、システムが有する目的を達成できない状況にあることを、システム異常と定義する。

３．９．２ システム異常時の処理

システム異常をきたす原因として、以下の３つを定義する。下記異常原因が除去、解決され、復旧した場合は、システムの構成情報を再度確立、徹底をできるだけ自動で実施する。安全上重要度の高い問題は、適宜人手を介し、復旧動作すること。

（1） 通信異常、ネットワーク異常

ネットワークが切断されている（ルータの故障等も含む）、ノイズ環境が劣悪であるなどの原因でノード間の通信が不可能であり、下位層プロトコル毎に開示している性能を満たすリカバリ処理（誤り訂正、再送等）や、ルータのリカバリ処理を実施後も、正常な通信ができない場合を通信異常、またはネットワーク異常という。各ノードに原因が有る場合

を通信異常、ネットワークの伝送メディアの問題をネットワーク異常と呼ぶ。各アプリケーションは、さらに当該通信の重要度に応じて必要最低限のリカバリ処理を実行し、システムの運転の継続を図る。

必要以上の再送によるトラヒックの増加、あるいは、通信異常発生への過敏な反応により、システムの異常停止が頻繁に発生するようなことが無いようにすること。

（2） 機器異常

各個別の機器が検出する機器の異常は、機器の特質にあわせて必要な情報を、異常の緊急度、レベルに応じてコントローラに提供することで、機器外に通知する機能を有することが望ましい。異常発生時には、各機器オブジェクトのプロパティで規定された異常のアナウンス以外で、著しくトラヒックを増加させないこと。

また、機器が、アダプタとフレックス機器により構成される場合、アダプタに何らかの異常が発生したことを機器が確実に認識でき、リカバリー処理ができるように（例えば、アダプタが単独リセットしてしまったような場合には、アダプタが停止通知サービスをフレックス機器が処理結果を返信してくるまで出しつづけるなど）構成すること。

（3） コントローラ異常

コントローラに何らかの異常が発生した場合には、これにより、各個別の機器は自律した運転に支障をきたさないようにすること。

３．１０ システムメンテナンス

３．１０．１ システムメンテナンス場面の定義

システム内の機器の運転情報や、異常履歴情報を収集するなど、機器、システムのメンテナンスのための処理を実行する場面を、システムメンテナンスという。

３．１０．２ システムメンテナンス時の処理

（1） メンテナンス情報の収集

各機器は、オブジェクト規定で公開が義務づけられた各機種における一般的なメンテナンス情報を積極的にサポート、開示し、システムでの有効利用を図る。

（2） メンテナンスによる機器、システムの運転、停止

上記メンテナンス情報の収集のために、機器、あるいはシステムを停止することは可能な限り不要な構成とする。

３．１１ トラヒック規定

システムを安定的に運転するためには、トラヒックに関する規定が必要である。下位の各プロトコル毎に輻輳しないようなトラヒック制御機能を持つことも必要であり、また、上位アプリケーションにおいてもこれを発生させない規定が必要である。

ECHONET では、システムアプリケーションの設計は各インスタンスを単位として行う。しかし、通信ネットワークとしてトラヒックや、各ノードが実装すべき通信の処理能力を考える際は、インスタンス単位ではなく、各ノード単位となる。こうしたことを考慮してアプリケーションのシステム設計をする必要がある。

例えば、特定のノードに対し情報の要求を実施するケースにおいて、当該相手からの応答を確認せずに複数の電文を送信することは、相手が不在の際には無駄な電文を送付し、さらに無駄な再送を誘発し、ネットワークの負荷を無用に上昇させることになる。

また、各機器は通常、コントローラに比べてH／W資源には制限があり、通信の処理能力は必ずしも高くない。よって、このような機器相手に、機器側の処理能力をオーバーした要求を送信することは、機器側が処理不可となり、結局は再送等を発生し、無用なトラヒックの増加を招く。

よって、上記のようなシステム設計は避ける事が好ましい。

３．１２ 通信定義オブジェクトによる連動設定

連動設定用通信定義オブジェクトを用いて連動設定を行うには、原則としてアクション設定を用いることとする。ただし、トリガ設定を用いて連動設定を行うときは、トリガ設定用の電文を送信する機器が通信定義オブジェクトを持たない場合、もしくはトリガ設定用通信定義オブジェクトしか持たないときに限定する。

第４章 ECHONET プロパティ値の扱いに関する指針

本章では、設定されたプロパティ値が、ECHONET プロパティの定義範囲内であるが、対応する実機器の稼動範囲外である場合のプロパティ値の扱いについて指針を記す。

（1） ECHONET プロパティが対応する実機器の連続値の稼動範囲が、ECHONET プロパティ定義範囲より狭い場合に、ECHONET プロパティに、ECHONET プロパティの上限値および下限値の範囲内で、実機器の上限値および下限値の範囲外の値を設定した時、ECHONET ノード上のアプリケーションは、ECHONET プロパティの上限値と、実機器の上限値との間の値を設定した場合は、実機器の上限値を実機器のプロパティ値及び ECHONET プロパティ値とすることを推奨する。また、ECHONET プロパティの下限値と、実機器の下限値との間の値を設定した場合は、実機器の下限値を実機器のプロパティ値及び ECHONET プロパティ値とすることを推奨する。

例えば、ECHONET プロパティ定義範囲が、0x00～0xFD（０℃～253℃）で、対応する実機器の値の稼動範囲が、0x0A～0x32（10℃～50℃）の場合に、ECHONET プロパティに、実機器の上限値と ECHONET プロパティの上限値との間の値（60℃）を設定した場合には、実機器の稼動範囲の上限値 0x32（50℃）を ECHONET プロパティ値とすることを推奨する。また、実機器の下限値と ECHONET プロパティの下限値との間の値（5℃）を設定した場合には、実機器の稼動範囲の下限値 0x0A（10℃）を ECHONET プロパティ値とすることを推奨する。

参考図を図 ４．1に示す。

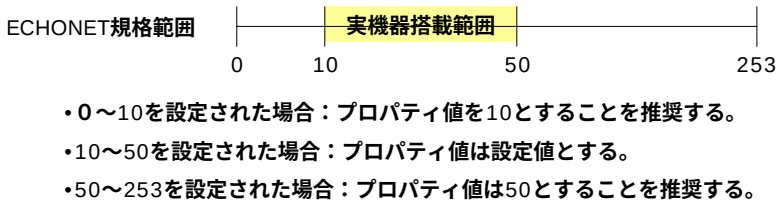


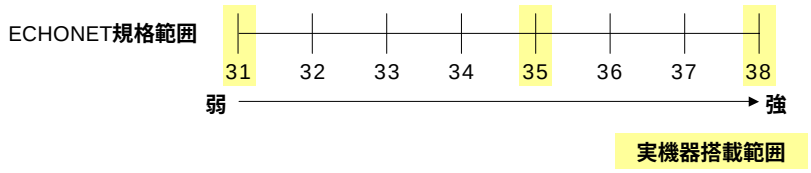
図 ４．１ プロパティ値設定例１

（2） ECHONET プロパティが対応する実機器の値の稼動段階が、ECHONET プロパティ定義範囲より少ない場合に、ECHONET プロパティに、ECHONET プロパティの範囲内で、実機器が保持しない段階値を設定した場合に、ECHONET ノード上のアプリケーションは、設定したプロパティ値に近い値を、実機器のプロパティ値および ECHONET プロパティ値とすることを推奨する。

例えば、ECHONET プロパティ定義範囲が、8 段階 0x31～0x38 で、対応する実機器の値の稼動範囲が、3 段階 0x31, 0x35, 0x38 の場合に、ECHONET プロパティに、8 段階のうち、いずれの値を設定した場合でも、ECHONET ノード上のアプリケーションは、

ECHONET プロパティ定義範囲が、8 段階 0x31～0x38 と、実機器の稼動範囲、3 段階 0x31, 0x35, 0x38 間のマッピングに従い、設定したプロパティ値に近い値を、実機器のプロパティ値および ECHONET プロパティ値とすることを推奨する。

参考図を図 ４．2に示す。



•31,35,38を設定された場合：プロパティ値は設定値とする。

•32,33,34,36,37を設定された場合：設定された値31,35,38の近い値にマッピングを行い、プロパティ値とする。

図 ４．2 プロパティ値設定例２

（3） ECHONET プロパティが対応する実機器の値の実装機能が、ECHONET プロパティ定義範囲より少ない場合に、ECHONET プロパティに、ECHONET プロパティの範囲内で、実機器が保持しない段階値を設定した場合に、ECHONET ノード上のアプリケーションは、設定したプロパティ値を無視することを推奨とし、現在の実機器のプロパティ値を ECHONET プロパティ値とすることを推奨とする。

参考図を図 ４．3に示す。

•41,42,45を設定された場合：プロパティ値は設定値とする。

•43,44を設定された場合：設定した値を無視し、プロパティ値はそのままとする。

図 ４．3 プロパティ値設定例３

第５章 ECHONET セキュア通信運用方法

５．１ 新規機器を ECHONET ネットワークへ参入

ここでは、ECHONET ネットワークへ新規機器が参入する方法を下記に示す。

1. 「新規機器」を ECHONET ネットワークへ参入する。ただし、「新規機器」はセキュア通信には対応していない機器としてネットワークへの接続処理を開始する。
2. 「新規機器」は、ECHONET アドレスを決定する。
3. 「新規機器」は、まず、セキュア通信未対応機器として ECHONET ネットワークに接続し、その他のノードとは平文電文形式でのみ通信が可能となる。
4. 「新規機器」は、セキュア通信が可能な機器へ移行するために、セキュア通信用共有鍵設定ノードは、「新規機器」へユーザ Key を設定する。

５．２ セキュア通信用共有鍵の種類と用途

セキュア通信用共有鍵の種類と用途を表 ５．1に示す。

表 ５．１ セキュア通信用共有鍵の種類と用途

セキュア通信用共有鍵	セキュア通信用共有鍵の用途
シリアル Key	新規に参入する機器へ user セキュア Key を設定する時に使用する。
user セキュア Key	家庭内で、家電機器へ一つ user セキュア Key を設定する。 各々の家電機器は、user セキュア Key 通信を行う。また、Service Provider セキュア Key の設定にも利用する。

５．３ シリアル Key を用いた user セキュア Key の設定方法

1. シリアル Key は、機器に添付されている。シリアル Key は、user セキュア Key を設定するためのセキュア通信用共有鍵として使用する。
2. 新規に機器をセキュア通信可能とするために、セキュア通信用共有鍵設定モードに変換する。
3. ユーザは、シリアル Key をコントローラに入力し、user セキュア Key を機器へセキュア通信で設定する。

4. コントローラは、機器に暗号方式を指定する電文を送信する。
5. user セキュア Key をシリアル Key で暗号化し、認証・暗号方式で機器へ user セキュア Key を書込む。コントローラは、user セキュア Key を通信相手先の機器へ設定する。

５．４ user セキュア Key 設定時のシーケンス

新規に機器をセキュア通信可能にするために、セキュア通信用共有鍵設定モードに変更する。

user セキュア Key は、シリアル Key によって暗号化され、機器へ認証･暗号方式で user セキュア Key を書込む。

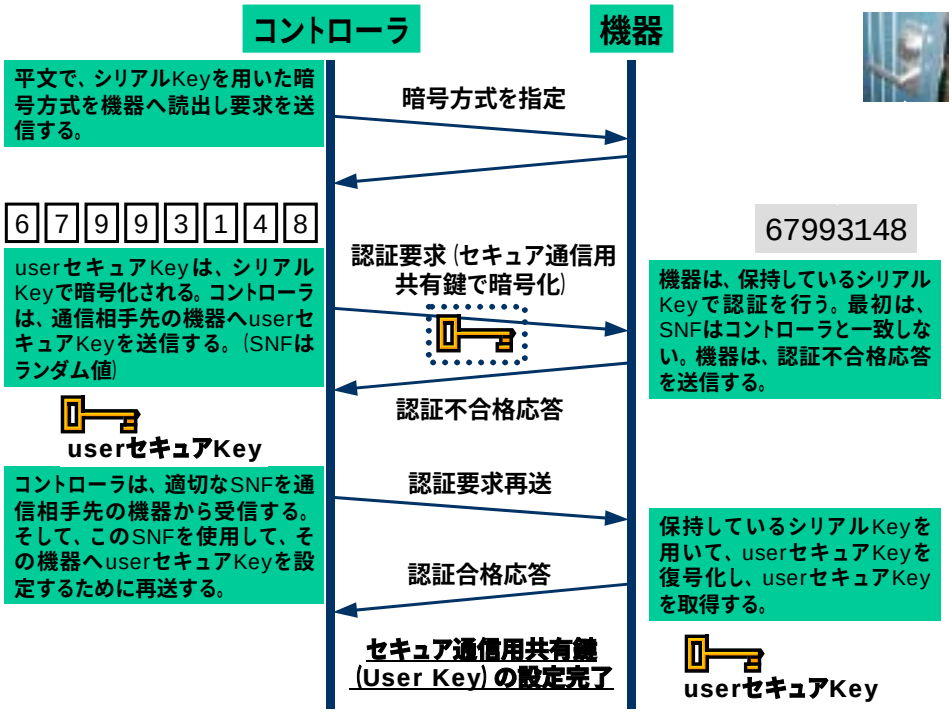


図 ５．１ user セキュア Key 設定時のシーケンス

５．５ user セキュア Key の設定完了

下記に、ホームネットワークに、user セキュア Key を設定するための基本的な方針を記す。

- ✧ 一つの家には、一つの user セキュア Key を設定する。
- ✧ user セキュア Key は、複数の機器に設定可能である。
- ✧ セキュア通信は、共通の user セキュア Key を保持する機器同士で行うことができる。

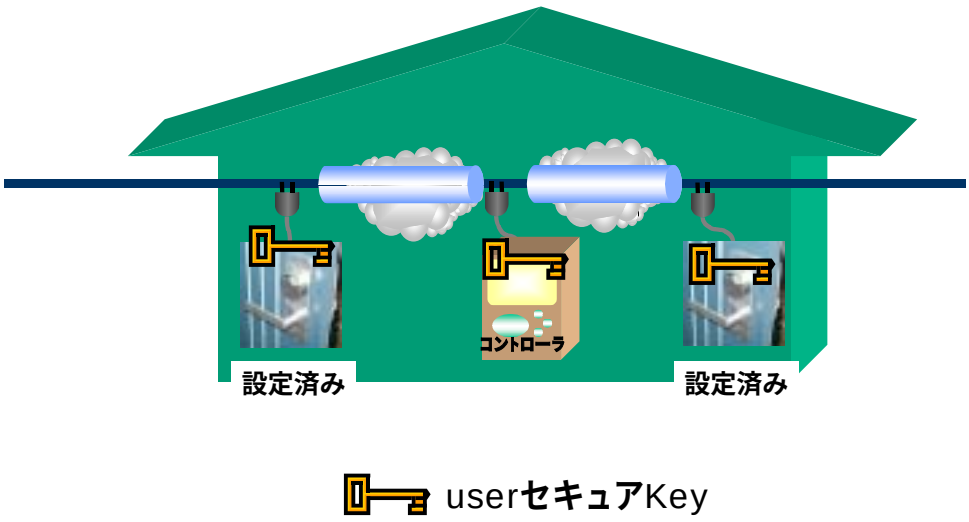


図 ５．２ user セキュア Key 設定完了