OPEN LIB

OPEN LIB EA
Engineering Guide

OPEN LIB EA
エンジニアリングガイド

2014 年 5 月

オーエスエスブロードネット株式会社

第 0.6 版 2014 年 5 月

Printed in Japan

改版履歴

版数	改版年月日	変更内容	備考
0.1	2013/06/30	1～3 章を記述。4.1., 4.1.1, 4.1.2 を記述。 付録 A～C を空欄作成。	初版作成。
0.2	2013/12/27	「4.1.2 IEC62056 オプション」を加筆。OpenADR 対応。	
0.3	2014/01/20	OpenADR から独自のメーターADR 方式への変更に伴い、「4.1.2 IEC62056 オプション」の内容を置換。	開発者への改良仕様提示目的。
0.4	2014/03/31	メーターADR 方式の VPA 側の設定・表示設計に基づき、DR 閾値設定アルゴリズムに関する文章表現を修正。 「4.1.2 IEC62056 オプション」の内容を再修正。	VPA 設計の反映。
0.5	2014/04/20	付録 B に機器を追加。付録 C に通信方式を追加。	
0.6	2014/05/07	3.2.1 の加筆修正（EA 設定ファイル関連情報の更新） 4.1.3 TR-069オプションの更新(既存コードを再整理中のため、現段階の関連情報を一旦削除) 他エンジニアリングガイドとの整合性を取るための一連の文言修正。　他	先行公開版。

参考文献

TR-069/098/106/157/181, Broadband Forum
IEC62056-47, 53, 61, 62, IEC
IEC61968, IEC
PLC G3 Mac Layer Specification, ERDF
PLC G3 Profile Specification, ERDF
OpenADR 2.0b Profile Specification v1.0.0, OpenADR Allience
OpenADR Security Profile v0.02, OpenADR Allience

本書の目的

本書は、OPEN LIB EA の組込み・設定・運用・保守・拡張などの業務に必要な情報をまとめたものです。

本書は､OPEN LIB に主体的に係わるパートナー各社とエンドユーザの技術者および､OPEN LIB に興味を持って下さった全ての方々に対する情報開示を目的に作成されています。本書のインターネット上での再配布および部分的な流用は、個人・法人を問わず自由に行えますが、弊社に著作権の帰属する情報の二次利用に際しては、弊社の著作権を明示して下さい。

本書の対象読者

本書は、OPEN LIB EA の導入設計・運用設定に従事する SE、増設・保守を行う CE および、システム拡張やカスタマイズなどを行うプログラマを対象にしています。

以下の技術に関する知識があると、本書の理解が一層容易になります。

スマートグリッド、DLMS/COSEM、ANSI、PLC、マルチホップ無線、TR-069、CWMP、UDP、TCP/IP、Linux、DBMS、Java、J2SE、Servlet/JSP、Apache、Tomcat、Firebird、MySQL

その他

OPEN LIB に関する技術的なご質問は、E-Mail により以下まで送信して下さい。

info@ossbn.co.jp

弊社の知的財産権に含まれない規格・技術の記述や情報の更新・バグに関し、弊社では一切の責任を負いませんのでご了承下さい。

目次

第1章　システム概要

1.1.　システム概要

OPEN LIB（OPEN LIBrary product series）は、業界標準の技術規格・仕様に基づく、様々な業務システムへの応用が可能なソフトウェア開発基盤製品シリーズです。OPEN LIB EA（Embedded Agent。以降、「EA」）は、Java/C 言語で開発されたプログラムを実行する常駐型のエージェントプログラムです。対象機器への EA 組み込みにより、業界標準プロトコルに準拠したプロビジョニング、ソフトウェア自動更新、エンド～エンド間の認証・暗号化、情報収集・遠隔制御等、様々なバックエンド通信機能を、シンプルかつ標準的な方式で対象機器に追加できます。

EA の組み込み対象は、ホームゲートウェイ・STB・スマート TV・USB ドングル等の情報端末、スマートメーター・コンセントレーター等の電力計測用機器、および、火災センサー・防犯センサー等の各種センサー機器です。

EA は、ヘッドエンド側に配置される各種上位システムとの組み合わせにより、システムとして動作します。

EA のシステム構成例を図 1.1 に示します。

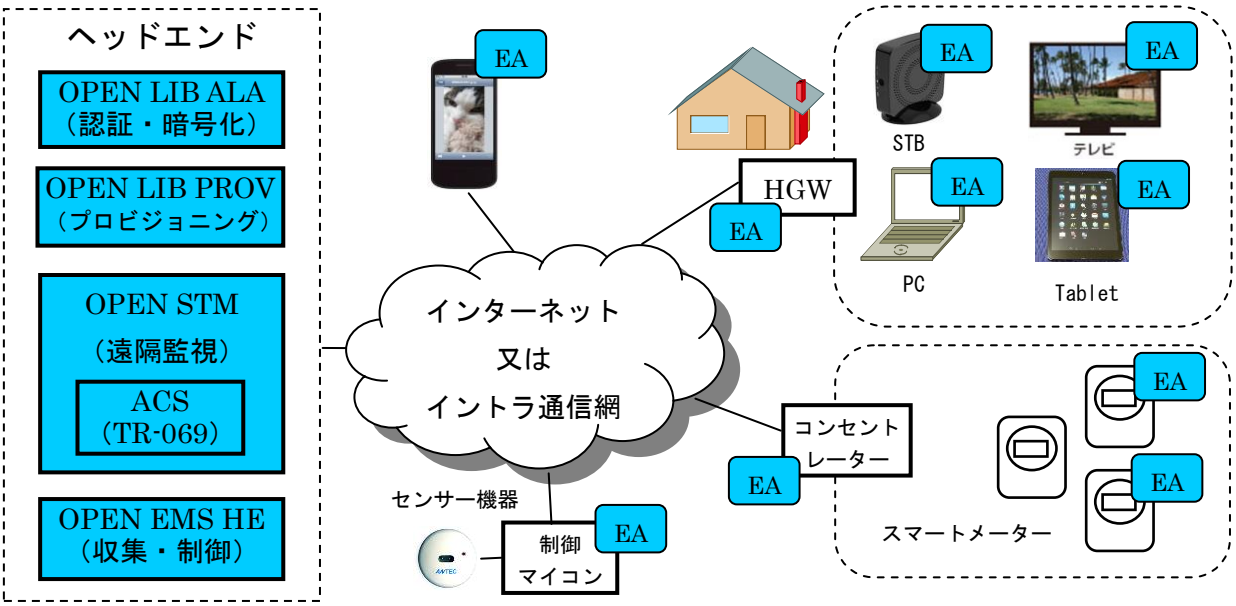

図 1.1　EA のシステム構成例

認証・暗号化ソフトウェアである OPEN LIB ALA（Application Level Authentication。以降、「ALA」）のセッションマネージャー機能と EA 暗号化オプションの組み合わせにより、ヘッドエンドと対象機器間の通信を暗号化できます。更に、ALA の CA 機能との組み合わせにより、プライベート CA 局による X.509 証明書の運用や、公開鍵暗号方式を利用した暗号鍵更新によるセキュリティ強化が可能になります。

文書番号：OSSBN-EGMN-13-11-001

プロビジョニングソフトウェアである OPEN LIB PROV（PROVisioning。以降、「PROV」）のプロビジョニング機能と EA の各プロトコル対応オプションの組み合わせにより、対象機器の設置・設定コストの低減と、トラックロールの短縮を図れます。

TR-069 オプションにより、対象端末を CPE エージェントとして管理できます。TR-069 の特長である NAT 越え方式により、インターネットを介して接続される OTT-STB やスマート TV、タブレット等の端末機器や各種のセンサー機器をヘッドエンドから管理できます。また、監視系製品シリーズである OPEN STM との組み合わせにより、TR-069 ACS 機能を利用したヘッドエンドからの対象機器の監視・制御が可能になります。

IEC62056 オプションにより、国際標準規格の DLMS/COSEM 通信プロトコルを、スマートメーター・コンセントレーターに組み込めます。HE・ALA・PROV との組み合わせにより、認証・暗号化やプロビジョニングを含めたスマートメーター情報収集＆運用管理の包括的なシステム基盤を構築できます。

ACS や HE・ALA・PROV の詳細については、関連文書を参照して下さい。

1.2. システム構成

EA の構造を図 1.22 に示します。

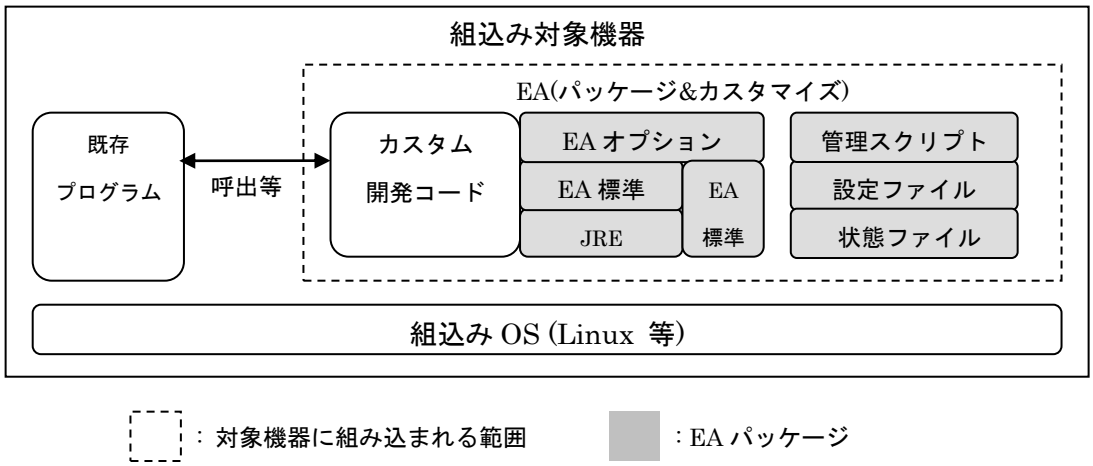

図 1.22 EA の構造

EA は、本体の「EA 標準」ライブラリ、拡張機能の「EA オプション」ライブラリ、起動処理と動作監視処理を記述する「管理スクリプト」、動作設定を記述する「設定ファイル」、動作状態=プロセス ID を保持する「状態ファイル」および、組込み対象機器の既存プログラムや API を呼び出す「カスタム開発コード」により構成されます。

EA オプションとカスタム開発コードの開発言語は、Java または C から選択できます。開発者は、対象機器の構成・仕様、開発アプリケーションの機能・性能要件に基づき、適切な開発言語を選択します。

1.3. 機能一覧

EA の機能一覧を表 1.3 に示します。

機能	分類	概要
動作設定	標準	EA の動作を設定します。
常駐エージェント		常駐プロセスとして各種の動作を実行します。 EA のリファレンス実装である DHCP/SNTP/TFTP によるプロビジョニング動作を含みます。プロビジョニング動作は、オプションまたはカスタム開発コードの追加による別方式への置換も可能です。
動作ログ出力		EA の動作履歴をログファイルに出力します。
動作監視・自動復旧		EA の動作状態を OS から定期的にチェックし、EA 常駐エージェントプロセス消失の検出時、EA を再起動します。
暗号化	オプション	SSL/TLS 認証・暗号化、SSH ポートフォワード、および OSSBN 独自の認証・暗号化フレームワークである OPEN LIB ALA への対応です。
TR-069		BBF TR-069 の定める CWMP の CPE エージェントです。ホームゲートウェイ・STB・スマート TV・スマホ等の情報端末上で動作します。
IEC62056		IEC62056 の定める DLMS/COSEM 通信仕様への対応です。 スマートメーター・コンセントレーター上で動作します。
SNMP		対応時期未定。
ANSIC.12		対応時期未定。
SEP2.0		対応時期未定。
ECHONET Lite		対応時期未定。

表 1.3　機能一覧

1.4. 動作環境

EA の動作環境を表 1.4 に示します。

項目	仕様
CPU	200MHz 以上
RAM	10MB 以上の空き領域
HDD	50MB 以上の空き領域
NIC	Ethernet MAC アドレス×1 以上
O/S	Linux (Linux 以外の場合、ターゲット OS へのポーティングが必要)
必須ソフト	JRE

表 1.4　動作環境

1.5. 制限事項

EA の制限事項を表 1.5 に示します。

No.	制限内容	制限量	備考
1	通信対象上位サーバ数	1	ALA, PROV, ACS, HE が対象。
2	動作ログサイズ	10MB	2 ファイル構成。
3	IP アドレスバージョン	V4	V6 は対応時期未定。
4	対応 STUN 仕様	RFC3489	RFC5389 は対応時期未定。

表 1.5　制限事項

1.6. 性能積算

1.6.1. 起動時間

EA の起動時間は、EA 標準の常駐エージェント起動時間と EA オプション各機能の起動時間の総和になります。EA の起動時間は、選択した各機能の処理オーバーヘッドと EA の動作環境により変わります。

本項では参考に、BeagleBone Black の場合の起動時間（概算）を示します。

BeagleBone Black は、2013 年 4 月発売開始の BeagleBoard ファミリ製品であり、テキサインスツルメンツ社の Sitara XAM3359AZCZ100 Cortex A8 ARM のプロセッサー（1GHz, 2000MIPS)、512MB DDR RAM, 2GB EMCC Flash を搭載しています。

起動時間例を表 1.36.1 に示します。

機能	分類	起動時間	備考
常駐エージェント	標準	100msec ～ 1sec	JVM の起動オーバーヘッドを含む
暗号化	オプション	100msec ～ 4sec	RSA 公開鍵暗号のオーバーヘッドを含む
TR-069		2sec	Boot Inform のオーバーヘッドを含む
IEC62056		1sec	TCP 動作時

表 1.6.1 起動時間例（Beagle Bone Black）

なお BeagleBone Black の場合、表 1.6.1 に示す起動時間以外に、Linux OS 起動に約 10sec ほど掛かります。

1.6.2. 暗号・復号化処理のオーバーヘッド

暗号化オプションで RSA 公開鍵認証を使用時、AES のような秘密鍵による暗号化方式に比べるとオーバーヘッドが桁違いに大きい（数秒単位）ため、処理の実行タイミング・間隔の選択に注意が必要です。

例えば 30 分周期メータリングの場合、RSA による処理時間の間延びはデータエラーの発生確率も副次的に増加させるため、特にマルチホップ無線のような低速・不安定な通信環境で RSA を毎回実行するとタイムアウト・リトライが頻発し、通信が更に不安定となる恐れがあります。

暗号化オプションの組込み時、伝送メディアの性質により、RSA による秘密鍵の生成を起動時の 1 回のみとする、ないしは 1 日一回程度の頻度で定常処理の空き時間に実行する等、通信ネットワークの実態に合わせた調整を実施して下さい。

第2章　内部構造

2.1.　処理構成

2.1.1.　プログラム構造

EA は、起動スクリプトからの起動により、OS 上にエージェントとして常駐します。

EA 常駐エージェントは、本体プログラム・オプションおよびその他のライブラリ・カスタム開発コードにより構成される、単一の Java VM プロセスです。

EA のプログラム構造を図 2.2.1 に示します。

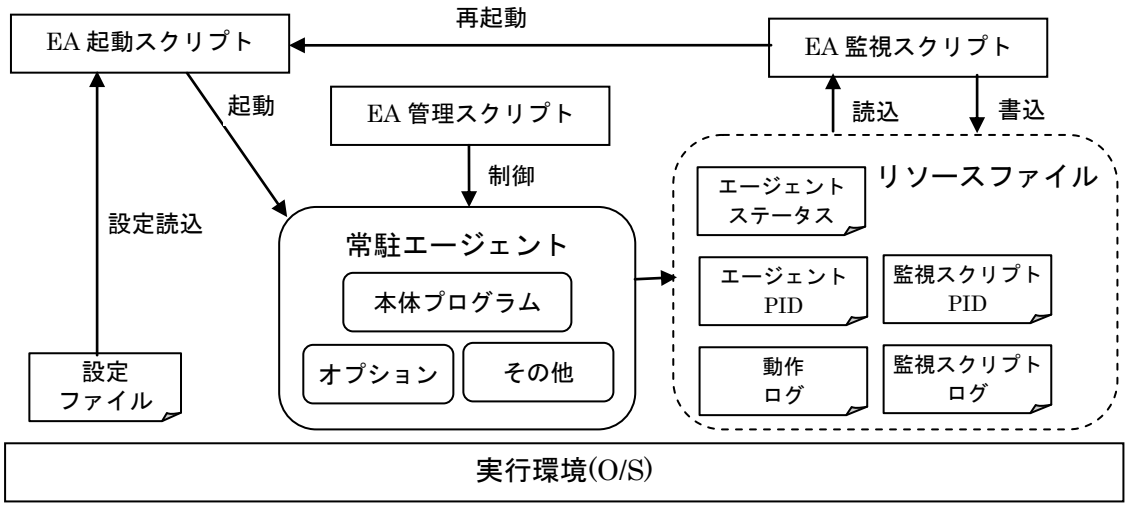

図 2.11.1　EA のプログラム構造

EA の動作は、設定ファイルにより設定します。

EA 起動スクリプトは、設定ファイルの内容を読み込み、常駐エージェントを起動します。

EA 管理スクリプトは、システム管理者が手動実行する停止・再開等のメンテナンスコマンドを発行します。

EA 監視スクリプトは、cron 等から定期的に呼び出される監視機能であり、リソースファイルに記録された常駐エージェントの PID 等の状態情報を参照し、常駐エージェントに障害が発生時、プロセス kill & 再起動等、各種のメンテナンス処理を自動実行します。

オプションの追加時、各オプションに付随する物理ファイル（実行ライブラリや暗号鍵ファイルなど）が追加されます。

各スクリプト、プログラム、設定ファイル、リソースファイルの配置先は、実行環境に合わせ、任意に変更可能です。

2.1.2. 開発用言語の選択基準

EA へのオプション機能やカスタム機能の追加時、Java または C 言語を使用します。

EA の本体プログラムは Java により開発されています。常駐エージェントの実態は単一の JVM プロセスです。EA に追加される各種のオプションやカスタム機能は、仕様上の必要性に基づく一部の例外ケースを除き、基本的には全て単独・同一の JVM 内で動作します。

このため EA の開発言語は、Java を使用できる限り、Java を使用する方が合理的です。

しかしながら、EA のような組み込みエージェントのプログラム開発では、Java よりも C の方が開発言語として適切なケースが存在します。

具体的には、2 層以下のプログラム制御が必要な通信機能の実装（ドライバやセンサーとの I/O 処理等）、実行性能重視の処理、O/S の直接呼出し時、及び、利用可能な Java のライブラリに不具合・制限があり C 言語による開発が必要な場合　等々です。

また、C 言語により開発された既存プログラムを EA のオプションないしはカスタム開発コードとして組み込みたい場合、既存 C コードの Java 移植には大きな開発工数が掛かり、かつデバッグや機能追加時には C と Java 双方のメンテが必要となるなど、保守性も低下します。

EA では上述の考え方に基づき、開発言語として Java よりも C の方が適切な場合、開発言語として C を積極的に採用する方針を採っており、具体的な Java と C の接続技術として、JNA（Java Native Access）ライブラリを利用しています。

Java から C/C++のネイティブライブラリ API を呼び出す技術としては、以前より JNI（Java Native Interface）が仕様と実装を提供しています。

しかしながら JNI による開発は、メソッドのマッピングやデータのマッピングが煩雑でコーディング作法も特殊なため開発者へのハードルが高く、生産性や保守性が低下しがちです。

JNA は、直接 JNI を利用せず、Java プログラムが簡単にネイティブライブラリ（.dll、あるいは、.so）を共有する方法を提供するライブラリです。特別処理が必要となる配列などの複雑なデータ型を使う一部の API を除き、C のコードを書かずに Java プログラムからネイティブライブラリ API を呼び出せるので、システム開発の生産性が向上し、保守性が低下しません。

JNA により、C で開発されたプログラムを、コンパイルしてネイティブライブラリ化後に Java から JNA 経由で呼び出せるため、Java/C の開発言語によらず、全機能を単一の JVM プロセスから実行するシンプルな構成が可能となります。

EA 標準機能では、リファレンス実装である DHCP/SNTP/TFTP によるプロビジョニング動作の組込みに JNA/JNI が使われています。DHCP では 2 層のプログラム制御が必要なため、一部の動作を Java で開発できません。このため EA では、ISC のオープンソース DHCP クライアントである C アプリの dhclient を Java から呼び出します。SNTP, TFTP も同様です。

TR-069 オプション（CWMP 通信処理）にも、既存プログラムを組み込む目的で、C 言語が使われています。

JNA は便利な手法ですが、OS のネイティブライブラリを呼び出すため、Java の特長である OS 非依存性が失われます。このため EA では、組み込み対象の OS が変わる場合、ポーティング開発が発生する点に注意して下さい。

2.1.3. プロビジョニング動作

プロビジョニング動作は、組み込み対象機器の仕様・構成や、対象機器を収容するシステム全体の構成・ポリシーにより様々に変化します。EA のリファレンス実装は、バリエーションに合わせた設定・チューニング・カスタマイズのベース概念を提供します。

EA ではリファレンス実装として、DHCP/SNTP/TFTP によるプロビジョニングを提供しています。各プロトコルは、設定ファイルにより使用・不使用を選択できます。また、オプションやカスタム開発コードの追加による別プロトコルへの置換が可能です。

本項では説明用の具体例として、EA に IEC62056 オプションを組み合わせたスマートメーターが、EA のプロビジョニングリファレンス実装で動作する場合に基づき説明します。

EA の起動時処理フロー例を図 2.2.13 に示します。

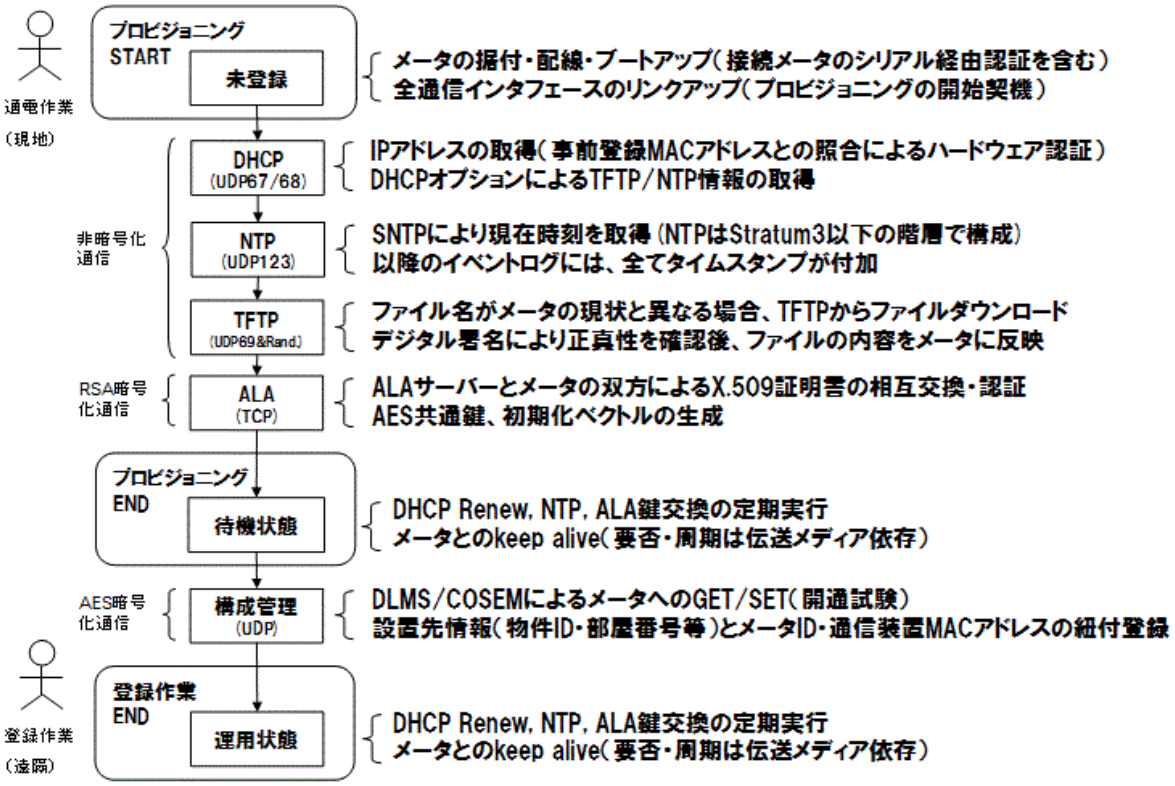

図 2.11.3 EA の起動時処理フロー例

EA は、起動後に DHCP ディスカバーをブロードキャスト後、受信した DHCP パケットのオプションから続く SNTP, TFTP の IP アドレス(または URL)を抽出します。

DHCP には、ISC のオープンソース DHCP クライアントである C アプリの dhclient を使用します。同様に SNTP/NTP/TFTP には OS の標準コマンドを使用します。

ALA への接続は、TFTP により取得した設定ファイルの情報に基づき、暗号化オプションの Java プログラムにより実行されます。

DHCP の使用・不使用は、設定ファイルに設定します。不使用時、各 IP アドレスを設定ファイルに静的に設定するか、ないしは代替プロトコルを指定して必要なオプション・カスタム開発コード EA に組み込み、必要なプログラムライブラリを OS にインストールします。
SNTP/NTP、TFTP、ALA の使用・不使用についても同様です。

TR-069 の場合、インターネットを経由したプロビジョニング動作が前提となりますが、端末を収容するネットワークのポリシーによっては DHCP が使えないため、DHCP 以外の方法として、工場出荷時または手動での ACS-URL 設定と、HTTP/SOAP によるプロビジョニングがプロトコル上に定義されています。
EA への TR-069 オプション追加時、収容ネットワークのポリシーに基づき、リファレンス実装のデフォルトである DHCP を、適切な手法＆代替プロトコルに置換して下さい。

ヘッドエンドとスマートメーター間の通信が、電力事業者のイントラネットでなく、3G/LTE 等、他のネットワーク事業者のネットワークを経由するケースでは、TR-069 同様、DHCP 以外の手法・プロトコルへの置換が必要な場合がある点に注意して下さい。

2.1.4. 状態遷移

EA に IEC62056 オプションを組み合わせ、スマートメーターに実装した場合の状態遷移例を図 2.2.14 に示します。

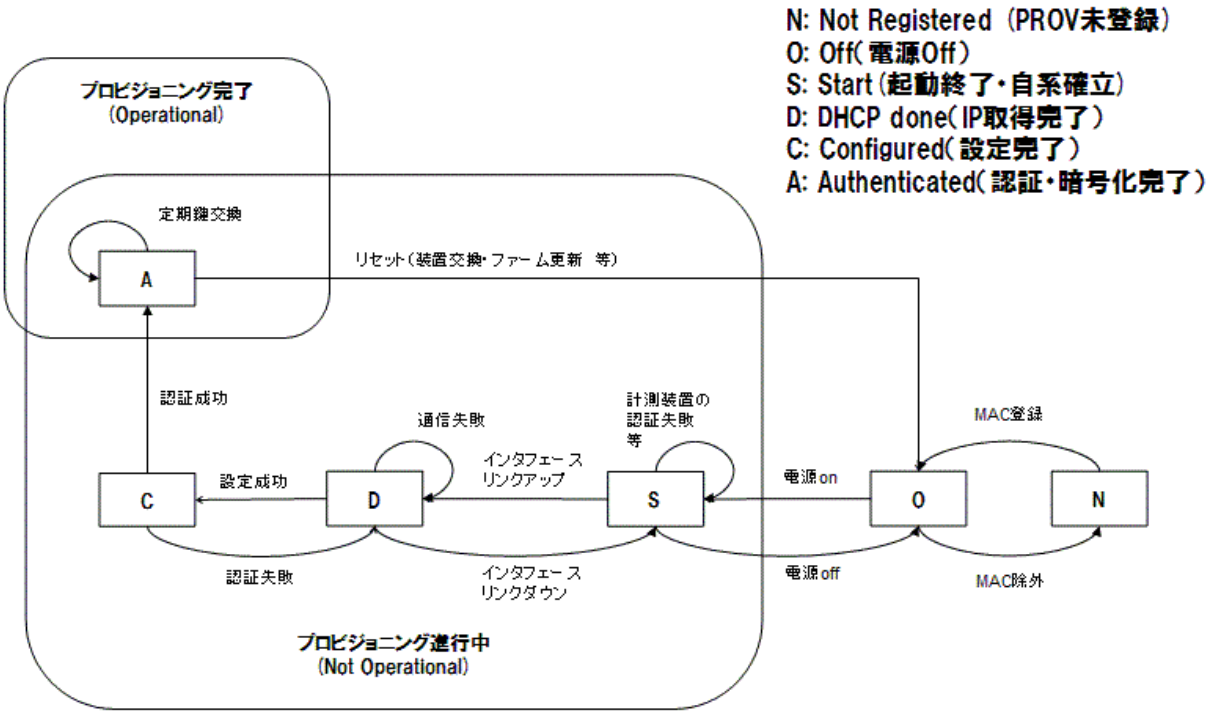

図 2.11.4 EA の状態遷移例

2.1.5. 基本通信シーケンス

EA に IEC62056 オプションを組み合わせ、スマートメーターに実装した場合の基本通信シーケンス例を図 2.2.15 に示します。

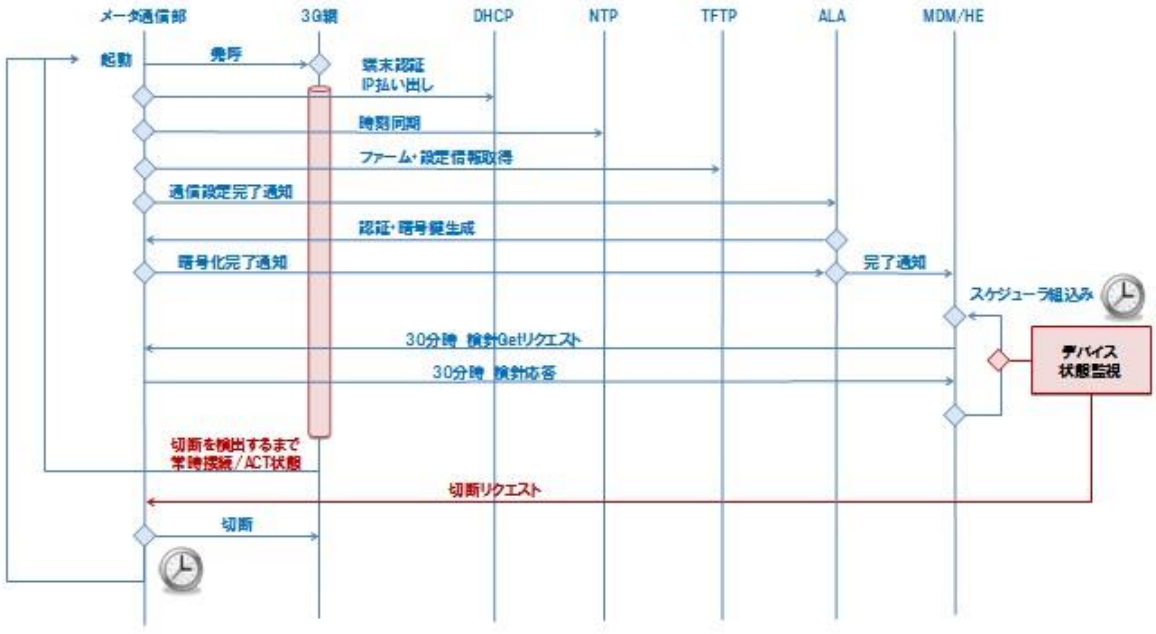

図 2.11.5 EA の基本通信シーケンス例

2.2. オブジェクト構成

2.2.1. Java パッケージ構成

EA の Java パッケージツリー構成を図 2.2.1 に示します。

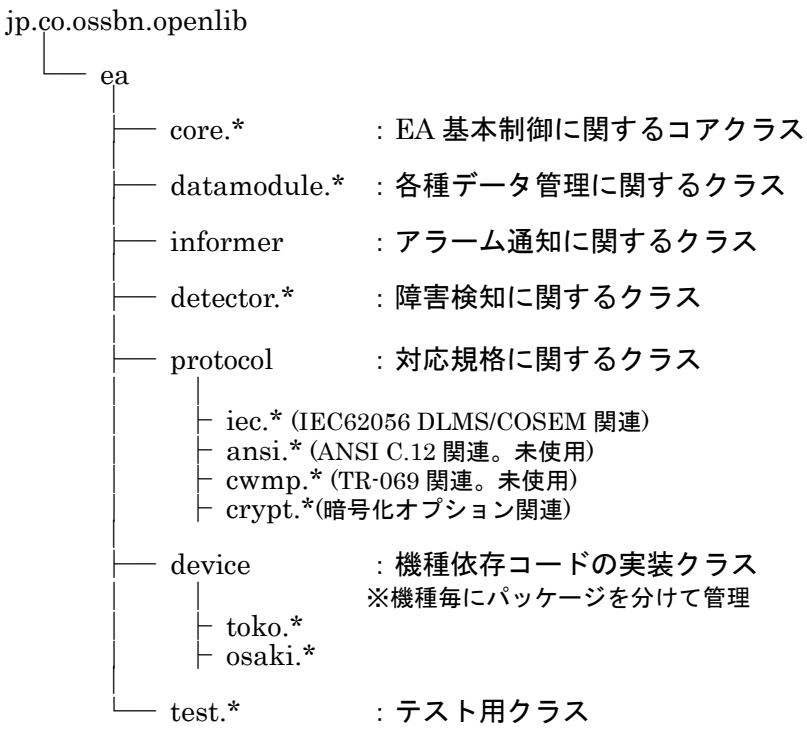

図 2.2.1 EA の Java パッケージツリー構成

IEC62056 オプションのプログラム実装には、jDLMS が使われています。
jDLMS の開発元情報、パッケージ概要、ライセンス上の制限事項（LGPL）等については、以下の URL を参照して下さい。
http://openmuc.org/

暗号化機能の使用時、EA に 128 ビット以上の AES 暗号をサポートした JCE（Java Cryptgraphy Extention。以降「JCE」）プロバイダが必要です。実行環境が JRE 1.6 以降の場合、JCE プロバイダは標準でサポートされていますが、AES 暗号については、JRE のバージョン・種類によっては拡張クラス扱いのケースがあります。通常は拡張クラスでも問題なく利用できますが、Java 環境のシステムプロパティ java.ext.dirs が変更されていると、拡張クラスが利用できず、暗号化オプションが機能しない事があります。拡張クラスは Java ホームディレクトリの./lib/ext に配置されるので、java.ext.dirs を独自に指定する場合、./lib/ext を忘れずにパスに加えて下さい。

2.2.2. C パッケージ構成

T.B.D.

第3章　運用・保守

3.1.　ファイル構成

ファイル一覧を表 3.1 に示します。

No.	ファイル	内容	備考
1	/etc/init.d/S99eaagent	管理スクリプト	
2	/opt/ea/bin/eaagent.sh	エージェント起動スクリプト	
3	/opt/ea/bin/eawatch.sh	監視スクリプト	
4	/opt/ea/etc/eashared.key	AES 共通鍵	
5	/opt/ea/etc/openlibea.conf	スクリプト用設定ファイル	
6	/opt/ea/lib/openlibea.jar	本体プログラム	
7	/opt/ea/lib/	Java クラスライブラリ及び設定ファイル	
8	/opt/ea/log/eaagent.log	動作 ログファイル	
9	/opt/ea/log/eawatch.log	監視スクリプト ログファイル	
10	/opt/ea/var/eaagent.pid	エージェント PID ファイル	
11	/opt/ea/var/eaagent.stat	エージェント ステータスファイル	
12	/opt/ea/var/eawatch.pid	監視スクリプト PID ファイル	
13	/opt/ea/var/eawatch.ready	監視準備ファイル	

表 3.1　ファイル一覧

C 言語関連のファイルを追加する事。

3.2. 動作設定

3.2.1. EA 設定ファイル

EA 設定ファイルの項目一覧を表 3.2.1 に示します。

変数名	内容
EA_JAVA	JAVA 実行コマンド パス
EA_SERVER_OPT	EA サーバ パラメータ -server <バインド アドレス>:<バインド ポート>:<転送先アドレス>:<転送先ポート> に形式で指定する。
EA_STAT_INTERVAL	-stat_interval オプションに渡すステータスファイルへの出力間隔(秒)
EA_CRYPT	-crypt オプションに渡す暗号モード.
EA_AUTH_DIGEST	-authDigest オプションに渡す認証モード.
EA_NETWORK_TIMEOUT	-network_timeout オプションに渡すタイムアウト ミリ秒.
EA_CONNECT_TIMEOUT	-connect_timeout オプションに渡すタイムアウト ミリ秒.
EA_MAXTHREAD	-maxthread オプションに渡す最大生成スレッド数
EA_LOG_LEVEL	-loglevel オプションに渡すログ出力レベル
EA_ETC_DIR	設定ファイルの格納ディレクトリ
EA_LOG_DIR	ログファイル格納ディレクトリ
EA_VAR_DIR	PID/ステータス出力ディレクトリ
EA_LIB_DIR	ライブラリ格納ディレクトリ
EA_BIN_DIR	プログラム格納ディレクトリ
EA_SHARED_KEY	暗号鍵ファイル
EA_WATCH_RUN	0 の場合は.監視スクリプトを起動しない.cron で起動する場合は 0 を指定しておく. 1 で監視スクリプトを常駐プログラムとして起動.
EA_WATCH_UPDATE_TIME	ステータス更新間隔(秒). この設定値よりも長時間ステータスファイルの更新がない場合にはプログラムが異常により停止したと見なす.
EA_WATCH_STAT_RETRY	プログラム停止状態の再確認回数. 停止状態がこの設定値を超えた場合に、エージェントを再起動する. ※ EA_WATCH_RUN=0 の場合は使用されない
EA_WATCH_SLEEP	監視スクリプト常駐時の監視間隔(秒)
EA_WATCH_MARGIN	監視スクリプトのエージェント再起動にかかる猶予時間. エージェントの再起動にはある程度時間がかかるため、5 秒以上を設定してくことを推奨.
PROV_PROTOCOL	プロビジョニングプロトコル。 0: なし 1: DHCP,（デフォルト） 2: CWMP(TR-069) 3: mail
PROV_DISCOVER_METHOD	自動検出手法。 0: DHCP, 1: 出荷時設定 2: 端末への手動設定
PROV_REPORTING_METHOD	プロビジョニング完了（自系確立）通知。 0: なし, 1: Syslog 2: CWMP Inform 3: mail

表 3.21 設定ファイルの項目一覧

設定ファイルには、EA 標準の基本動作（プロビジョニング等）および、暗号化オプションに使用する設定が格納されます。TR-069 オプションやカスタム開発コード、及び、EA 標準が呼び出す OS ネイティブ API（dhclient 等）の設定項目は、各アプリケーションの設定ファイルに個別に記述されるため、本設定ファイルには含まれません。

EA 設定ファイルの設定例を示します。

```
EA_JAVA=/home/java/bin/java

EA_BASE_DIR=/opt/ea

EA_SERVER_OPT=0.0.0.0:11234:127.0.0.1:12345
EA_STAT_INTERVAL=1
EA_CRYPT=AES-CBC
EA_AUTH_DIGEST=MD5
EA_NETWORK_TIMEOUT=5000
EA_CONNECT_TIMEOUT=5000
EA_DISCONNECT_TIME=5000
EA_MAXTHREAD=10
EA_CONN_OPT="-conn_limit_time 60 -conn_limit_count 100 ¥
          -conn_limit_keeptime 30 ¥
          -reject_time 1800 ¥
          -auth_limit_count 10"
EA_LOG_OPT="-logtype ea -logcount 10 -logsize 1000000 -logappend"
EA_LOG_LEVEL=3
EA_MSG_LEVEL=3
EA_DEBUG_OPT=

EA_ETC_DIR=$EA_BASE_DIR/etc
EA_LOG_DIR=$EA_BASE_DIR/log
EA_VAR_DIR=$EA_BASE_DIR/var
EA_LIB_DIR=$EA_BASE_DIR/lib
EA_BIN_DIR=$EA_BASE_DIR/bin

EA_OPENLIB_JAR=$EA_LIB_DIR/openlibea.jar
EA_OPENLIB_CLASS=jp.co.ossbn.openlib.ea.custom.ses.tk.TkServer

EA_CLASSPATH=.:$EA_OPENLIB_JAR
for jar in ${EA_LIB_DIR}/*.jar
do
  if [ "$EA_OPENLIB_JAR" = "$jar" ]; then
    continue
  fi
  EA_CLASSPATH=${EA_CLASSPATH}:$jar
```

```
done

EA_SHARED_KEY=$EA_ETC_DIR/aes_key

EA_AGENT_LOG=$EA_LOG_DIR/eaagent.log
EA_WATCH_LOG=$EA_LOG_DIR/eawatch.log

EA_AGENT_PID=$EA_VAR_DIR/eaagent.pid
EA_WATCH_PID=$EA_VAR_DIR/eawatch.pid
EA_AGENT_STAT=$EA_VAR_DIR/eaagent.stat

#
# eawatch parameter
#
# EA_WATCH_RUN = 1 - resident
# EA_WATCH_RUN = 0 - non-resident (for cron EA_WATCH_STAT_RETRY=0)
EA_WATCH_RUN=0
EA_WATCH_UPDATE_TIME=5
EA_WATCH_STAT_RETRY=1
EA_WATCH_SLEEP=1
EA_WATCH_MARGIN=5
```

3.3. EA の操作

3.3.1. 管理スクリプトの使用

EA の管理スクリプトである S99eaagent の実行例は、以下の通りです。

【実行例】

/etc/init.d/S99eaagent <パラメータ>

＜パラメータ＞には以下を指定できます。

パラメータ	内容
start	エージェントプログラムの起動
stop	エージェントプログラムの停止
restart	エージェントプログラムの再起動
status	エージェントプログラムの状態表示

手動で起動したい場合、起動スクリプトである/opt/ea/bin/eaagent.sh による起動も可能です。
eaagent.sh はサービス(デーモン)として起動させるプログラムではないので、実行するとフォアグラウンド・アプリケーションとして動作します。
終了させたい場合、CTRL+C を入力します。

3.3.2. 監視スクリプトの使用

EA では、Java 実装の監視スクリプトとして、/opt/ea/bin/eawatch.sh が用意されています。監視スクリプトは、常駐エージェントの動作監視・自動復旧用に使われます。

設定ファイル/opt/ea/conf/openlibea.conf 内の環境変数 EA_WATCH_RUN=1 を指定すると、/etc/init.d/S99eaagent サービス起動スクリプトからの起動時に監視スクリプトも自動的に実行されます。この場合、監視スクリプトは O/S 上に常駐して動作します。

cron 等のスケジュール機能を利用して実行したい場合、設定ファイル内の環境変数に EA_WATCH_RUN=0 を指定します。

監視スクリプト eawatch.sh の設定例は、以下の通りです。

【crontab 設定例】

```
#毎分起動
* * * * *       /opt/ea/bin/eawatsh.sh 1>/dev/null 2>/dev/null
```

監視は/opt/ea/var/eawatch.ready ファイルが作成されている状態で行われます。本ファイルが存在しない状態では、監視スクリプトを実行しても動作監視は行われません。

/opt/ea/var/eawatch.ready ファイルは/etc/init.d/S99eaagent での開始時に作成され、停止時に削除されます。

3.3.3. 参考情報・本体プログラムの直接起動

EA では管理スクリプト（Linux 用）として/etc/init.d/S99eaagent が用意されているため、本体プログラムを直接起動する Java 実行コマンドをコンソールから発行する事はありませんが、開発段階では EA のデバッグ目的で本体プログラムを直接起動する場合があるので、この場合に指定可能なパラメータについて説明します。

（１） 本体プログラムで使用可能なパラメータの種類・概念

本体プログラムで使用可能なパラメータは、コマンドとオプションの 2 種類に大別できます。コマンドはエージェントの基本動作を指定するもので、ひとつしか指定できません。コマンドを複数指定した場合、最後に指定したものだけが有効となります。

コマンドやパラメータでは、大文字・小文字は原則として区別しません。ただし、ファイル名やディレクトリ、パス名は OS やファイルシステムに依存します。

コマンドやオプションは、付随するパラメータを除けば順不同で指定できます。ただし、ログ出力関連のオプションだけは、順番が意味を持ちます。

（２） コマンド一覧

使用可能なコマンドの一覧を以下に示します。

コマンド	内容
-server <バインド アドレス>:<バインド ポート> :<転送先アドレス>:<転送先ポート>	サーバモード
-client <バインド アドレス>:<バインド ポート> :<転送先アドレス>:<転送先ポート>	クライアントモード
-genkey	暗号鍵生成
-ssh <ssh 用オプション>	SSH ポートフォワード

（３） オプション一覧

使用可能なオプションの一覧を以下に示します。

オプション	内容
-crypt <暗号モード>	暗号モード
-random_algname	乱数アルゴリズム
-authDigest <認証モード>	認証モード
-keyfile <パス>	暗号鍵ファイル
-logtype <ログタイプ>	ログ形式
-logcount <n>	ログローテート数
-logsize <バイト数>	ログサイズ
-logappend	ログファイル追加モード
-lognoappend	ログファイル上書モード
-logfile <ログファイル>	ログ出力ファイル
-loglevel <n>	ログ出力レベル
-msglevel <n>	標準出力レベル
-pid <pid ファイル>	PID ファイル
-stat_file <ステータスファイル>	ステータス出力ファイル
-stat_interval <秒>	ステータス出力間隔
-maxthread <n>	最大生成スレッド数
-timeout <ミリ秒>	次のタイムアウト設定を設定する. -network_timeout -connect_timeout -disconnect_time
-network_timeout <ミリ秒>	通信タイムアウト
-connect_timeout <ミリ秒>	接続タイムアウト
-disconnect_time <ミリ秒>	無通信 切断時間
-conn_limit_time <秒>	接続制限時間
-conn_limt_count <n>	連続接続制限数 –conn_limit_time で指定した時間内で制限する
-reject_time <秒>	接続拒否時間
-auth_limit_count <n>	認証失敗制限数

（４） 書式例

コマンド・オプションの指定書式例を以下に示します。

■server コマンド

【書式】

-server <バインドアドレス>:<バインドポート>:<転送先アドレス>:<転送先ポート>

・バインド アドレス

・バインド ポート

・転送先アドレス

・転送先ポート

【説明】

サーバモード エージェントを指定します。

バインドアドレスとバインドポートから暗号通信を受信し、転送先アドレス／転送先ポートに平文として転送します。

【使用例】

TCP 110 ポートで受信した暗号通信を 110 ポートに転送します。

-server 0.0.0.0:10110:127.0.0.1:110

■client コマンド

【書式】

-client <バインドアドレス>:<バインドポート>:<転送先アドレス>:<転送先ポート>

・バインド アドレス

・バインド ポート

・転送先アドレス

・転送先ポート

【説明】

クライアントモード エージェントを指定します。

バインドアドレスとバインドポートから平文通信を受信し、転送先アドレス／転送先ポートに暗号化して転送します。

【使用例】

TCP 110 ポートで受信した平文通信をアドレス 192.168.1.1:110 ポートに暗号化して転送します。

-client 0.0.0.0:10110:192.168.1.1:110

■genkey コマンド

【書式】

-genkey

【説明】

乱数により暗号鍵ファイルを生成します。

生成する鍵ファイルは-keyfile オプションにより指定します。

【関連】

-keyfile オプション

-random_algname オプション

【使用例】

-genkey –keyfile /opt/etc/easahred.key

■crypt オプション

【書式】

-crypt <暗号モード>

・暗号モード

暗号モード	内容
OFF	暗号なし。
AESCBC	AES/CBC モードを指定。

【説明】

通信に暗号モードを指定します。

OFF を指定した場合、暗号化は行いません。

【使用例】

-crypt AESCBC

■ssh コマンド

T.B.D

エージェント オプション

■authDigest オプション

【書式】

-authDigest　<認証モード>

・認証モード

認証モード	内容
MD5	MD5（デフォルト）
SHA1	SHA-1
SHA256	SHA -256
SHA512	SHA -512

【説明】

暗号化通信のユーザー認証に使用するハッシュ用ダイジェストアルゴリズムを指定します。クライアントとサーバで認証モードを一致させておかないと、認証失敗となり正しく通信できません。

このオプションの指定がない場合は、MD5 が指定されたものとみなします。

【使用例】

-authDigest MD5

■keyfile オプション

【書式】

-keyfile　<鍵ファイルパス>

・鍵ファイルパス

【説明】

暗号通信に使用する鍵ファイルのパスを指定します。

【使用例】

-keyfile /opt/etc/easahred.key

■logtype オプション

【書式】

-logtype <ログ形式>

・ログ形式

ログ形式	内容
SIMPLE	ロガー標準簡易形式
EA	EA モード（デフォルト）

【説明】

ログの出力形式を指定する。

このオプションは-logfile オプションよりも先に指定する必要があります。

このオプションの指定がない場合は EA が指定されたものとみなします。

【使用例】

-logtype SIMPLE

■logcount オプション

【書式】

-logcount <ローテート数>

・ローテート数

0 を指定した場合、ローテーションしません。

【説明】

ログファイルのローテーション数を指定します。

0 を指定した場合はローテーションしません。

このオプションは-logfile オプションよりも先に指定する必要があります。

ローテーションしたログ出力ファイルは、ファイル名の最後にピリオド「.」+数字が追加されます。

【使用例】

-logcount 19

■logappend オプション

【書式】

-logappend

【説明】

ログファイルに追加モードで出力します。

これは規定の動作で、上書きする場合には-lognoappend オプションを使用します。

このオプションは-logfile オプションよりも先に指定する必要があります。

【使用例】

-logappend

■lognoappend オプション

【書式】

-lognoappend

【説明】

ログファイルに上書モードで出力します。

追加モードで出力したい場合は-logappend オプションを使用します。

このオプションは-logfile オプションよりも先に指定する必要があります。

【使用例】

-lognoappend

■logfile オプション

【書式】

-logfile <ログファイル パス>

・ログファイル パス

【説明】

ログを出力するファイルパスを指定します。

-logcount, -logsize, -logtype, -logappend, -lognoappend オプションは、このオプションよりも先に指定しないと正しく動作しません。

コマンドライン パラメータを解析する処理中、このオプションを見つけると直ちに有効となりログ出力が開始されます。このためログ関連のオプションパラメータは、他のコマンドやオプションよりも先に指定することを推奨します。

【使用例】

-logfile /var/log/eaagent.log

■loglevel オプション

【書式】

-loglevel <出力レベル>

・出力レベル

指定された出力レベル以下のメッセージをログとして出力します。

出力レベル	内容
0	出力なし
1	エラー
2	警告（デフォルト）
3	情報
4	デバッグ
5	詳細デバッグ
6	リザーブ
7	リザーブ
8	リザーブ
9	すべて表示

【説明】

ログを出力するレベルを 0～9 の数値で指定します。

【使用例】

-loglevel 3

■msglevel オプション

【書式】

-msglevel <出力レベル>

・出力レベル

指定された出力レベル以下のメッセージを標準出力に出力します。

出力レベルについては-loglvel オプションを参照して下さい。

【説明】

標準出力に出力するメッセージレベルを 0～9 の数値で指定します。

【使用例】

-msglevel 4

■pid オプション

【書式】

-pid <PID ファイル>

・PID ファイル

プロセス ID を出力するファイルへのパス。

【説明】

指定したファイルにプロセス ID を出力します。

【使用例】

-pid /var/run/eaagent.pid

■stat_file プション

【書式】

-stat_file <ステータスファイル>

・ステータスファイル
　ステータスを出力するファイルへのパス。

【説明】

指定したファイルに動作状態をステータスとして出力します。
出力間隔は-stat_interval オプションで指定します。

【使用例】

-stat_file /var/eaagent.stat

■stat_interval オプション

【書式】

-stat_interval <出力間隔(秒)>

・出力間隔(秒)

【説明】

-stat_file オプションで指定したステータスファイルへ出力する間隔を秒単位で指定します。

【使用例】

-stat_file /var/eaagent.stat –stat_interval 10

■maxthread オプション

【書式】

-maxthread <スレッド数>

・スレッド数
　最大スレッド数。0 を指定した場合は無制限。

【説明】

通信処理に使用する最大スレッド数を指定します。
エージェントはこのオプションで指定したスレッド数の通信を同時に処理できます。0 を指定した場合はスレッド数を制限しません。

このオプションを指定していない場合、スレッド数を制限しません。

【使用例】

-maxthread 10

■timeout オプション

【書式】

-timeout <タイムアウト(ミリ秒)>

・タイムアウト(ミリ秒)

通信用のタイムアウトをミリ秒単位で指定します。

【説明】

-network_timeout, -connect_timeout,-disconnect_time を同時に指定します。
つまり、-timeout A は、-network_timeout A –connect_timeout A -disconnect_time A と同じ意味となります。

詳細は各オプションの説明を参照。

【使用例】

-timeout 5000

■network_timeout オプション

【書式】

-network_timeout <タイムアウト(ミリ秒)>

・タイムアウト(ミリ秒)

通信用のタイムアウトをミリ秒単位で指定します。

【説明】

受信用のタイムアウトをミリ秒単位で指定する。0 を指定した場合、無限のタイムアウトとなります。

この設定は通信用ソケットに設定する SO_TIMEOUT に相当します。

規定の設定は 5,000 ミリ秒。

【使用例】

-network_timeout 5000

■connect_timeout オプション

【書式】

-connect_timeout <タイムアウト(ミリ秒)>

・タイムアウト(ミリ秒)

接続タイムアウトをミリ秒単位で指定します。

【説明】

接続時のタイムアウトをミリ秒単位で指定する。0 を指定した場合、無限のタイムアウトとなります。

規定の設定は 5,000 ミリ秒。

【使用例】

-connect_timeout 5000

■disconnect_time オプション

【書式】

-disconnect_time <切断判定時間(ミリ秒)>

・切断判定時間(ミリ秒)

接続判定時間をミリ秒単位で指定します。

【説明】

このオプションで指定した時間内に受信処理が行われなかった場合、通信を切断します。

規定の設定は 30,000 ミリ秒。

【使用例】

-disconnect_time 30000

■conn_limit_time オプション

【書式】

-conn_limit_time <接続制限時間(秒)>

・接続制限時間(秒)

【説明】

接続制限を行う際の、判定期間を秒単位で指定します。-conn_limit_count オプションとセットで指定します。

規定の設定は 60 秒。

このオプションで指定した時間内に、同一 IP アドレスからの接続数が-conn_limit_count オプションで指定した制限数を見えた場合、接続を拒否します。

【使用例】

60 秒間に 100 回以上の接続があった場合、接続を拒否します。

-conn_limit_time 60 –conn_limit_count 100

■conn_limit_count オプション

【書式】

-conn_limit_count <接続制限数>

・接続制限数

【説明】

接続制限を行う際の、制限数を指定します。-conn_limit_time オプションとセットで指定します。

規定の設定は 100。

詳しくは-conn_limit_time オプションの説明を参照。

■reject_time オプション

【書式】

-reject_time <接続拒否時間(秒)>

・接続拒否(秒)

【説明】

-conn_limit_count や-auth_limit_count によって接続を制限した場合、このオプションで指定した時間が経過すると接続拒否を解除します。

指定の設定は 1800 秒（30 分）。

【使用例】

-reject_time 3600

■auth_limit_count オプション

【書式】

-auth_limit_count <認証失敗制限数>

・認証失敗制限数

【説明】

暗号通信の際、このオプションで指定した回数、認証処理に失敗すると以後その IP アドレスからの接続を拒否します。

規定の設定は 10 回。

【使用例】

-auth_limit_count 10

第4章　システム拡張

4.1.　オプションの追加

オプションの追加により、様々な機能拡張が行えます。

T.B.D.
Java によるオプションコードの開発作法を説明
JNA を使った既存 C アプリのマージ方法を説明
JNA を使った OS ネイティブ API の呼出方法を説明
JNA を使った OS ネイティブアプリの呼出方法を説明

文書番号：OSSBN-EGMN-13-11-001

4.1.1. 暗号化オプション

EA への暗号化オプション追加により、EA とヘッドエンド間の通信を暗号化できます。

暗号化オプションは、JCE により Java で開発された機能です。

暗号化オプションに ALA を組み合わせずに単独で使用する場合、ヘッドエンドと EA 間の通信は、平文または、恒久的な共有秘密鍵による AES 暗号化通信のみ可能です。AES 暗号化通信で使用する秘密鍵は、出荷時にメーカーにより設定されるか、ないしは端末設置時にユーザーまたはインストーラーにより手動設定されます。

暗号化オプションに ALA を組み合わせて使用する場合、RSA 公開鍵暗号と X.509 証明書認証方式により、ヘッドエンドと EA 間の通信で使用される共有秘密鍵を更新できます。

AES 暗号化通信で使用する秘密鍵は、ALA により初回インストール時、ないしは設定間隔で定期的に更新されます。

暗号化オプションの動作設定は、EA 標準の設定ファイルにより行います。

設定可能な内容の詳細については、3.3.3 節を参照して下さい。

4.1.2. IEC62056 オプション

EA への IEC62056 オプション追加により、DLMS/COSEM 通信を実行できます。

IEC62056 オプションは、jDLMS により Java で開発された機能です。

EA への IEC62056 オプション追加により、DLMS/COSEM が定義する物理デバイス・論理デバイスとしての動作機能がスマートメーターに組み込まれます。

EA では、DLMS ユーザー協会の定める各標準 COSEM クラス定義に基づき設計した仮想スマートメーターを想定し、APDU コマンドおよび収集対象のアトリビュート群を、デフォルトのメータリングポリシーグループとして予め定義しています。実際に使用するスマートメーターの構造や事業者の運用管理ポリシーがデフォルトと異なる場合、カスタム開発コードを追加し、必要な項目を設定します。

仮想スマートメーターの構成を図 4.1.2-a に示します。

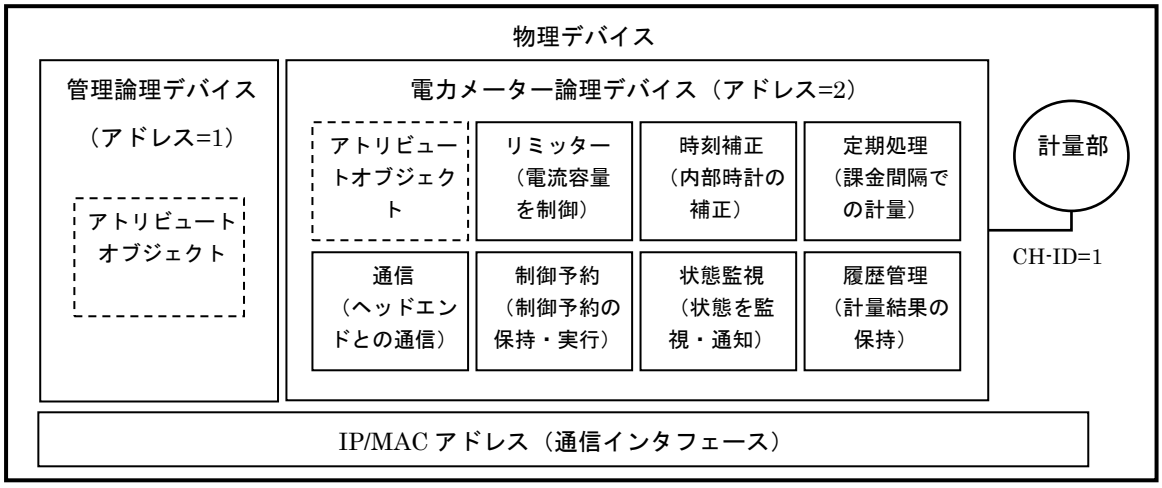

図 4.1.2-a 仮想スマートメーターの構成

仮想スマートメーターは、電力向けの論理デバイス×1、単相メーター×1 のシンプルな構成であり、時刻補正、状態監視機能を備え、計量履歴は直近 2 日分を保持、IP によりヘッドエンドと通信します。DR 制御用に制御命令予約を保持・実行するスケジュール管理機能を持ち、リミッターによりヘッドエンドから電流容量を制御します。

計量指標は電流・電圧・電力の 3 つであり、それぞれ順・逆潮流、正時・瞬時値を対象とします。

瞬時値は、DR 制御に先立ち、ヘッドエンドからリアルタイムに取得します。

デフォルトのメータリングポリシーグループ項目については、「HE エンジニアリングガイド」を参照して下さい。

EA は、OSSBN の独自 ADR 概念である「メーターADR」に対応しています。

メーターADR では、需要家が過去の電力量使用履歴に基づき設定した判断閾値と、OpenADR の DRAS から電力事業者および需要家に通知される節電要請メッセージに基づき、ヘッドエンドがスマートメーターの契約アンペア数を変更するコマンドを EA に送信します。

EA はヘッドエンドから受信した DLMS/COSEM コマンドに基づき、メーターの契約アンペア数＝電流容量を指定日時に切り替える制御予約を追加し、時刻到来時に契約アンペア数を切り替えます。

メーターADR で使用する DLMS/COSEM コマンド一覧を図 4.1.2-a に示します。

IC 名 （class_id）	アトリビュート/ メソッド名（id）	説明・用途 （含単位・値例）	LN （OBIS）	APDU （Get/Set/Action）
リミッター				
Limmiter (71)	実装依存	電流容量の変更	実装依存	Set
制御予約				
Single Action Schedule (22)	実装依存	制御を行う日付時刻の指定	実装依存	Set

表 4.1.2-b メータリングポリシーグループ項目一覧（デフォルト）

各 OBIS コードは、DLMS User Association が公開する OBIS List に基づきます。

4.1.3. TR-069 オプション

EA への TR-069 オプション追加により、EA とヘッドエンド間の通信に TR-069 の CWMP プロトコルを使用できます。

TR-069 オプションは、C で開発された機能です。

T.B.D.

TR-069 CPE エージェントの構造の概要、設定方法について説明。

EA の設定ファイルに TR-069 オプションの使用・不使用の設定項目を追加。

設定ファイルは既存の CPE エージェントのままとし、EA の設定ファイルにマージせず。

JNA により EA 本体の JVM から呼び出す構造にコードを修正。

4.2. カスタム開発コードの追加

カスタム開発コードの追加により、組み込み対象機器メーカーの独自仕様への対応等、柔軟な機能拡張が行えます。

柔軟な拡張が可能となる反面、カスタムコードは OSSBN によるプログラム保守のサポート対象外であり、EA 標準や各オプションのバージョンアップの動作保証対象外となる点に注意して下さい。

カスタムコードの開発時、カスタムプログラムの呼出処理と標準プログラムとの接続またはオーバーライドコードを記述した機種依存クラスを作成し、「図 2.2.1　EA の Java パッケージツリー構成」に示す「機種依存コード」を格納するパッケージに格納します。

Java によるカスタム開発コードの開発作法、JNA を使った既存 C アプリのマージ方法、JNA を使った OS ネイティブ API の呼出方法、JNA を使った OS ネイティブアプリの呼出方法については、4.1 節を参照して下さい。

付録A　　結果コードと障害番号

EA が出力する結果コードと障害番号の一覧を表 A に示します。

番号	内容	Facility	Priority	備考
30001				
T.B.D.				

表 A　結果コードと障害番号一覧

追記要。

付録B　　組み込み対象機器

EA の組み込み対象機器の一覧を表 B に示します。

No.	ベンダー	製品	対応機能	備考
1	東光電気	STiNC-II	スマートメーター通信	
2	TI(設計)	Beagle Bone Black	EA 標準	
3	Pixela	OTT-STB	TR-069	

表 B　組み込み対象機器一覧

付録C　　通信方式

HE が対応する通信方式の一覧を表 C に示します。

No.	通信方式	規格標準化団体	仕様	備考
1	東光電文	東光電気		TCP
2	ECOP	富士電機		TCP
3	DLMS/COSEM	IEC62056		UDP/TCP
4	CWMP	BBF TR-069		TCP(https)

表 C　通信方式一覧

OPEN LIB EA　エンジニアリングガイド

2014 年 5 月 7 日 第 0.6 版 発行

著　　者　　宮副　英治、小林　太郎
発　　行　　オーエスエスブロードネット株式会社
〒 213-0011 神奈川県川崎市高津区久本 3-5-7 新溝ノロビル 5F
電子メール: info@ossbn.co.jp
