

基于 DLMS/COSEM 协议的智能电表固件升级方案研究

高 华，陈方周，王小芬

(国电南瑞科技股份有限公司，南京市高新路 20 号 210061)

摘 要：本文针对目前电能表通讯协议不统一、固件升级机制同样标准各异现状，提出了一种基于设备语言报文规范（DLMS）/能源计量配套规范（COSEM）协议的智能电表固件升级的方案。其中对升级中启动升级任务、固件包的传输、电能表对固件包的验证、固件包的激活方式、升级过程中的异常处理等方面进行了详细的阐述与解释。实际应用表明，通过文中的方案可以成功地对智能电表进行固件升级，方便了智能电表的后期维护，有力地支撑了高级计量架构（AMI）的建设。

关键词：DLMS/COSEM；智能电表；固件升级；AMI

0 引言

在当前计量计费系统中，选用的电能表生产厂家不同、型号各异、尤其是通信协议不统一，从而导致当用户提出升级需求时，如需要在现有的硬件基础上新增功能、当前软件版本存在功能缺陷或出现兼容性问题、固件不完整或已过时等，升级维护将相当困难并且费用高昂。

随着国家智能电网以及高级计量架构建设的不断深入，智能电表市场容量逐年加大，降低升级维护成本费用的重要性不言而喻。因此，研究一种基于 DLMS/COSEM 这种互操作性强的协议的固件升级方案显得势在必行。

1 基本原理

按照能源计量配套规范（COSEM），主站与智能电表在信息交互的过程中，主站系统作为客户端（COSEM Client），智能电表作为服务器端（COSEM Server）。固件升级即为客户端将二进制格式的固件包传输至服务器端并完成验证、激活、应用等升级流程。如图 1 所示。

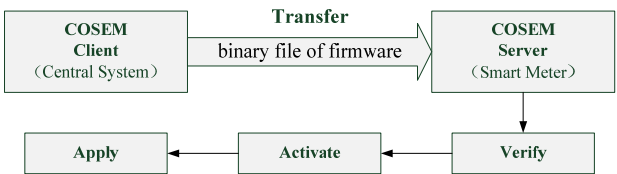


图 1 固件升级基本原理

2 传输对象说明

2.1 固件包

固件包(Image)即为升级中需要从客户端传送至服务器端的一个二进制信息文件，它可以分成更小的元素，称为固件块(Image Block)，其序号将从 0 开始。Image 和 Image Block 的大小分别称为 Image Size 和 Image Block Size，它们都以字节(bytes)来标称。它们的关系如图 2 所示。

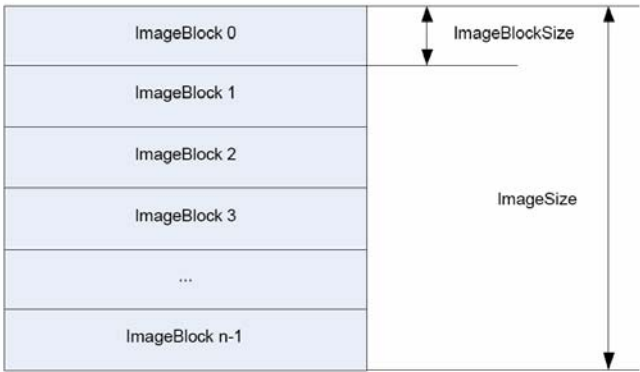


图 2 固件包的结构

ImageBlockSize 可以被设定为 32、64、128 字节或更高，智能电表根据自身传输能力在出厂时已被预先设定，但根据 DLMS 协议，此值不可超过链路协商报文中的 ServerMaxReceivePduSize（服务器最大接收协议数据单元大小）。

2.2 固件包标识符

固件包标识符(Image Identifier)包括固件包的循环冗余校验码(CRC: Cyclic Redundancy Check)，也可以根据需要定制一些可以标识固件包的内容，如固件包序列号ISN(Image Serial Number)。

对于 CRC 校验码，客户端在获取到固件包时

会对其进行 CRC 计算，并将得出的值在进行固件包传输初始化操作时随固件包标识符下发给智能电表；在整个传输结束后，智能电表也将按照同样的计算方式计算 CRC 校验码，来完成验证。关于 CRC 校验码的计算方式请读者查阅相关资料，本文不作赘述。

2.3 固件包允许传输标志

固件包允许传输标志(Image Transfer Enabled)的数据格式为布尔值，智能电表根据自身的运行状态决定当前状态下是否允许升级，若允许将此属性值置为“TRUE”，反之置为“FALSE”，客户端在升级前需首先查询此属性值，只有在服务器端返回“TRUE”时才可继续进行后续升级流程。

2.4 固件包传输状态

固件包传输状态(Image Transfer Status)表示在升级流程中，智能电表将根据当前的实际传输状态在相应的寄存器中写入对应状态枚举值以供客户端查询，从而判断后续升级流程，固件包传输状态包含以下 8 种：

- (0) Image transfer not initiated
- (1) Image transfer initiated
- (2) Image verification initiated
- (3) Image verification successful
- (4) Image verification failed
- (5) Image activation initiated
- (6) Image activation successful
- (7) Image activation failed

在特殊情况下，客户端操作者也可以对此属性进行写操作，但只可被写为(0)号状态，具体内容将在本文章节 4.2 中介绍。

3 固件包传输

在固件包传输流程中，我们定了一个固件包传输的 COSEM 对象(Image Transfer)，即 Class Id=18 的接口类的范例，其对象标识系统代码(OBIS code)定义为 0-0:44.0.0.255。此 COSEM 对象包含 7 个属性和 4 个方法，显然 logic_name（属性 1，逻辑名）为 OBIS 的十六进制体现，即为 0x00002C0000FF。上文传输对象说明中也已经提到了其中两个属性，分别是 image_block_size（属性 2）和 image_transfer_status（属性 6），下文还将在具体升级流程中详细介绍其他属性和方法。固件包传输流程如图 3 所示。

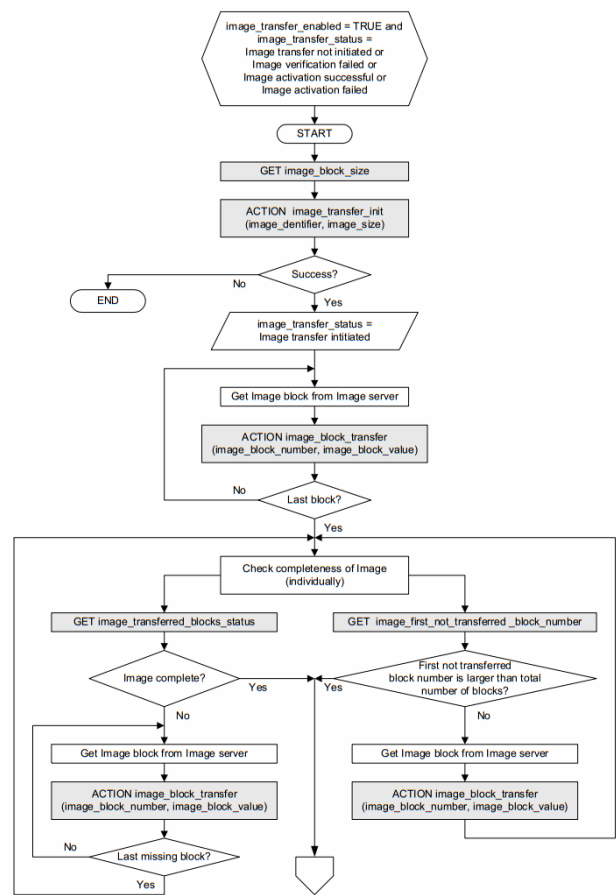


图 3 固件包传输流程

3.1 传输准备

首先，客户端查询服务器端的固件包允许传输标志，属性值必须为“TRUE”才能开启升级任务。其次，查询固件块大小，以供客户端根据此值以及固件包大小计算出所需传输的固件块数量(Image Block Number)。

3.2 固件包传输初始化

固件包传输初始化使用 Image Transfer 对象的第一个方法 image_transfer_initiate(data)实现，此方法的参数为 data,格式为 structure（结构体），内容包括固件包标识符和固件包大小。

```
data ::= structure
{
    image_identifier    : octet-string
    image_size          : double-long-unsigned
}
```

通过此操作客户端向服务器端传达了此次升级所要传输的固件包标识符（包含 CRC）和固件包大小的信息。完成此操作后查询固件包传输状态，初始化成功则为(1)号状态，失败则为(0)号状态且升

级终止。

3.3 开始固件包传输

固件包传输使用 Image Transfer 对象的第二个方法 `image_block_transfer(data)` 实现，此方法的参数为 `data`，格式为结构体，内容包括固件块序号和固件块内容。

```
data ::= structure
{
    image_block_number :double-long-unsigned
    image_block_value   :octet-string
}
```

客户端根据之前步骤已经确定了升级块的数量并且将固件包的内容分割成固件块，通过此操作客户端将固件块从序号 0 开始依次传输给服务器端，直到最后一个固件块传输完成。

3.4 检查固件包完整性

在完成固件块传输后，服务器端将根据固件块序号检查有无丢失的固件块。客户端将根据丢失固件块信息进行重新传输，现提供如下两种检查策略。

3.4.1 检查已传输固件块状态

`image_transferred_blocks_status`: 已传输固件块状态, 作为 Image Transfer 对象的第三个属性, 其数据格式为比特串(bit-string), 即字节的每一个比特位表示每个 ImageBlock 的传输状态, 一一对应。第一比特位(第一个字节的最高有效位)表示第一个固件块的传输状态(序号为 0), 相邻的第二比特位表示第二个固件块的传输状态(序号为 1), 依次类推。“1”表示此比特位对应的固件块已完成传输,“0”表示此比特位对应的固件块没有被传输。显然, 比特位的个数即为固件块的个数。

若所有固件块对应的比特位均置“1”，则表示所有固件块均完成传输，升级包完整。若存在某些固件块对应的比特位置“0”，则返回值固件包传输流程重新传输这些丢失的固件块。传输完成再次进入此流程进行检查，直到所有固件块成功传输至服务端。

3.4.2 查询第一个未完成传输的固件块序号

`image_first_not_transferred_block_number`: 第一个未完成传输的固件块序号，为 Image Transfer 对象的第四个属性。服务器端在固件包传输完成后，将对未完成传输的固件块进行统计，客户端将根据查询到的此属性值进行如下判断。

若此值大于最后一个固件块的序号，则表明所有固件块均成功完成传输；若此值和某一个固件块

的序号相同，则客户端将重新传输从此固件块开始到最后一个固件块的部分，此策略很好地体现了断点续传的功能。

4 固件包验证与激活

固件包传输完成后智能电表将根据客户端的指令对固件包进行验证与激活，具体流程如图4所示。

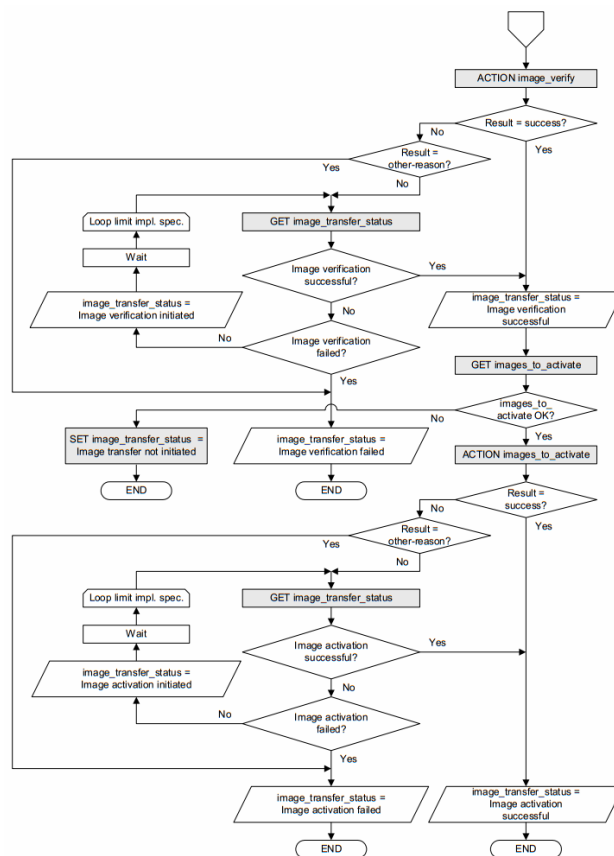


图 4 固件包验证与激活流程

4.1 固件包验证

当服务器端成功地接收到固件包所有内容后，将对此固件包进行验证，即计算 CRC 校验码。上文中提到客户端在对服务器端进行固件包传输初始化操作中，包含了此固件包的 CRC 校验码信息，则此时智能电表将自身计算的 CRC 校验码和预先从固件包传输初始化命令中接收到的 CRC 校验码进行验证。客户端通过 Image Transfer 对象的第三个方法 `image_verify(data)` 查询服务器端的验证结果，包括以下几种类型(前面括号内的数值表示结果代码，详见 DLMS UA 1000 - 2 Ed. 7.0):

- (0) success: 验证成功完成;
- (2) temporary-failure: 正在验证, 尚未完成;

(250) other-reason: 验证失败。

若结果为 success, 则查询固件包传输状态为(3)号状态, 且进入后续准备激活流程。

若结果为 temporary-failure, 则继续通过查询固件包传输状态判断下一步流程, 状态号为(3)则将进入后续准备激活流程; 状态号为(2)则在相应等待后重新查询固件包传输状态; 状态号为(4)则表明验证失败, 升级终止。

4.2 准备激活的固件包信息查询

image_to_activate_info: 准备激活的固件包信息, 为 Image Transfer 对象的第七个属性。当服务器端成功接收完所有的固件块并验证成功后, 将处于等待激活的状态, 客户端在激活固件之前可通过查询此属性得到将要激活的固件包的大小、标识符、签名等信息, 组织结构如下:

```
array image_to_activate_info_element
image_to_activate_info_element ::= structure
{
    image_size      : double-long-unsigned
    image_identification : octet-string
    image_signature  : octet-string
}
```

客户端操作者在获取到此准备激活的固件包相关信息后, 将自行判断此固件包是否为需要激活的对象且相关信息正确。若此固件包不正确或是不需要激活的, 则将固件包传输状态设置为(0)号状态 Image transfer not initiated, 且立即终止升级。若此固件包正是将要升级的, 则进入后续激活流程。

4.3 固件包激活

在固件包验证成功, 且获取到的准备激活固件包的信息正确时, 升级流程就将进入激活操作。固件包激活有两种方式: 立即激活和计划激活。如图 5 所示。

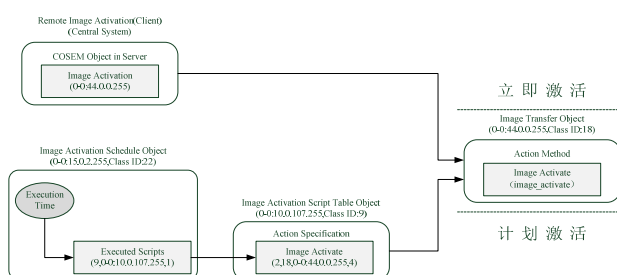


图 5 固件激活两种方式

4.3.1 立即激活

立即激活命令通过 Image Transfer 对象的第四

个方法 image_activate 实现, 服务器端收到此命令, 则开始执行固件包激活流程, 并在激活结束后在相应寄存器写入激活结果。与固件包验证相同, 固件包激活同样有 3 种结果, 且处理流程也类似。

4.3.2 计划激活

计划激活即为客户端给服务器端下发一个计划表, 让智能电表按照计划表中的执行时间来计划进行, 其中计划表属性内容中包含一个操作脚本, 此脚本直接将直接映射到立即激活命令所使用的 COSEM 对象范例的方法。

预先定义一个脚本表 (Script table, Class Id=9) 的范例作为计划激活脚本, OBIS code 为 0-0:10.0.107.255, 其 script (属性 2, 脚本内容) 内容设置如下:

```
array script
script structure
{
    script_identifier: 1
    actions: array action_specification
}
action_specification structure
{
    service_id      : 2
    class_id        : 18
    logical_name    : 0-0:44.0.0.255
    index          : 4
    parameter       : 0
}
```

上述属性值表示此脚本标识为 1, 映射到 class id=18 且逻辑名为 0-0:44.0.0.255 的范例的第 4 个方法 (即固件激活操作), 参数为 0 (无效)。

其次定义一个单时间操作表 (Single action schedule, Class Id=22) 的范例作为计划表, OBIS code 为 0-0:15.0.2.255, 其 executed_script (属性 2, 运行脚本) 内容设置如下:

```
script structure
{
    script_logical_name : 0-0:10.0.107.255
    script_selector     : 1
}
```

execution_time (属性 4, 脚本执行时间) 内容按照日期和时间的数据格式设置。上述计划表的属性表示智能电表将在 execution_time 指定的日期和

时间执行逻辑名为 0-0:10.0.107.255 范例的第 1 个 script_selector (对应脚本表中的 script_identifier)。

5 新固件应用

智能电表接收客户端激活命令后自行将新固件激活并替换旧版本固件启用, 但旧版本固件并非立即删除。

我们预先定义一个寄存器(Register, Class Id=3)的范例, 称为 New Firmware Safety Time (新固件安全期), 表示新固件启用后的安全过渡时期。新固件启用后, 智能电表将在这个安全期内对自身的各个组件进行自我检查(SelfCheck), 自我检查机制可由供应商自行定义。若在新固件安全期内, 自我检查不通过导致智能电表运行不正常, 则需将固件版本退回至老版本并丢弃新固件; 若在新固件安全期内, 智能电表通过自我检查并运行正常, 则在安全期到期时丢弃旧固件。

当智能电表成功度过新固件安全期并正常运行, 则表明此次固件升级任务圆满成功。

6 结论

本文提出了一种基于 DLMS/COSEM 协议的智能电表升级方案, 详细的阐述了各个升级流程中的操作步骤以及异常处理机制, 并研究了升级中的固件传输、验证、激活、新固件启用等流程的协议实现。此方案的应用将对 AMI 系统中智能电表的升级

维护提供极大的便利, 使用 DLMS/COSEM 协议也保证了互操作性和可靠性。

参考文献:

- [1] IEC62056-61:2006 Electricity Metering-Data Exchange for Meter Reading, Tariff and Load Control-Part 61: Object Identification System(OBIS)[S].
- [2] IEC62056-62:2006 Electricity Metering-Data Exchange for Meter Reading, Tariff and Load Control-Part 62: Interface Classes[S].
- [3] IEC62056-53:2006 Electricity Metering-Data Exchange for Meter Reading, Tariff and Load Control-Part 53: COSEM Application Layer[S].
- [4] DLMS UA 1000-2 Ed.7.0:DLMS/COSEM Architecture and Protocols[Z]. DLMS User Association. 2009-12-22.
- [5] DLMS UA 1000-1 Ed.10.0:Identification System and Interface Classes[Z]. DLMS User Association. 2010-08-26.

作者简介:

高 华 (1987—), 男, 江苏大丰人, 助理工程师, 研究方向: 高级计量架构(AMI), 用电信息采集系统, E-mail: gaohuateo@163.com;

陈方周 (1986—), 男, 陕西蓝田人, 助理工程师, 研究方向: 用电信息采集系统;

王小芬 (1989—), 男, 江苏徐州人, 本科, 研究方向: 用电信息采集系统。