Efficiency of DLMS/COSEM for large systems with constrained resources

A White Paper by the DLMS User Association

Authors:
Győző Kmethy, DLMS UA Executive Director and President, Gnarus Engineering
Milan Kozole, Convenor of the DLMS UA Technical Standing Committee, Iskraemeco

Preamble

DLMS/COSEM is the world leading standard (IEC/EN 62056, EN 13757) for data exchange with smart devices, currently used predominantly in smart metering systems, the "original IoT application". Such systems generally comprise a Head End System (HES) that collects data from and controls millions of devices deployed in the field, transporting data over various communication media.

DLMS/COSEM uses a three-tier approach: the COSEM object model allows modelling the functions of the end device. The DLMS Application layer specifies the services to access the COSEM objects. Communication profiles specify how those services can be transported over various communication media. DLMS/COSEM is based on the client – server paradigm: the HES acts as a client sending requests to the end devices. End devices act as servers sending responses to the requests.

Some communication networks and devices implementing DLMS/COSEM have ample resources, but DLMS/COSEM is increasingly used in constrained devices communicating over constrained networks. Device resources may be constrained in that they provide limited processing and storage capabilities or may be powered by batteries throughout the service life of the device. Communication networks may be constrained in terms of the number and length of the data packets that can be transmitted. Systems may be constrained in that they have to meet stringent requested service levels.

To meet the challenges of such environments DLMS/COSEM has been designed with efficiency in mind. The efficiency mechanisms evolved over the years as the application of DLMS/COSEM spreads to new domains.

The DLMS/COSEM efficiency toolbox

Efficiency mechanisms are available both on the COSEM object model level and on the DLMS application layer messaging level. They are summarized in Table 1.

The various mechanisms can be freely combined for an optimum performance tailored to the specific characteristics of the devices and the communication networks. They are briefly discussed in the remaining sections of this White Paper. Detailed information is available in the DLMS UA Blue Book specifying the COSEM object model and the Green Book specifying DLMS application layer messaging.

Table 1 – DLMS/COSEM Efficiency Mechanisms

On the COSEM object model level	On the DLMS application layer messaging level
Separation of data and metadata	
Aggregation of data	Aggregation of services
Selective access	Composable messages
Null-data compression	Pre-established and persistent Application Associations
Compact array encoding	Push operation
Compact data	Broadcasting and multicasting

Separation of data and metadata

COSEM objects represent data with several attributes. A COSEM object generally comprises:

- the logical_name attribute that – together with other attributes – provides the semantical meaning of each data element;
- the value attribute that represents a process value or a parameter;
- further attributes that provide metadata like scaler, unit, a time stamp, status etc.

Some metadata may also be held in other COSEM objects, e.g. the serial number of the meter, contract identifiers, unit prices etc.

Process values that are measured by the device are generally dynamic, so they have to be transferred frequently. Metadata are generally static, so it is enough to transfer them only once or infrequently. The separation of process values from metadata greatly enhances efficiency by reducing the amount of information to be transferred.

Aggregation of data

This mechanism allows aggregating any combination of data – COSEM attribute values – into a single attribute, so that they can be accessed with a single DLMS service thus reducing the number of message exchanges. This mechanism is available with:

- "Profile generic" objects that allow capturing a number of attribute values into the buffer attribute at regular intervals or upon events, so that the buffer has several columns and lines / entries. Examples are load profiles, event logs, series of instantaneous data;
- "Data protection" objects that allow applying cryptographic protection on aggregated attributes;
- "Register table" objects that allow capturing similar values in a tabular format. Examples are capturing data related to harmonics, phase angles, gas composition;
- "Compact data" objects that allow capturing raw data. This data can be loaded then to a pre-defined template; see below.

Selective access

When data are aggregated, it may be necessary to access only parts of it e.g. in order to retrieve just the data of current interest or to recover lost data. *Selective access* may be absolute, e.g. in a specified interval of time or range of entries, or relative, e.g. related to the current point of time or entry. Selective access is available with "Profile generic", "Data protection" and "Compact data" objects.

Null-data compression

Null-data compression is useful for transferring arrays of data, like "Profile generic" buffers holding a load profile or an event log. With this mechanism, null-data encoded as a single byte replaces the data when the value can be determined from the previous value, either because it did not change (e.g. a register reading or a status is the same as the previous one) or because the change is known (e.g. in the case of time stamps).

A more advanced mechanism, *delta-array compression* is under development. This mechanism will also be used with transferring data arrays. It allows dynamically using the shortest possible data type depending on the change in the value.

For example, a long-64-unsigned value that requires 9 bytes (including the tag) may be replaced by a delta-unsigned value – that requires only two bytes – if the change of the value is small.

Both mechanisms greatly reduce the size of the messages.

Compact array encoding

Compact array encoding is applicable when an array of homogenous data, for example a load profile is transferred. With this mechanism, the data types that are common to each element of the array are transferred only once. This mechanism greatly reduces the encoding overhead.

Compact data

The "Compact data" interface class, already mentioned above, is designed to capture raw data in its compact_buffer. The data captured may be simple or complex; selective access is available. For example, it is possible to capture "Profile generic" buffer attributes into "Compact data" objects.

The data type and length of each raw data element are described in the template_description attribute. Each template is identified with a template_id. The compact_buffer – that also contains the template_id – can be then read by or pushed to the client. This mechanism almost completely eliminates the encoding overhead, reducing it to a single byte.

Aggregation of DLMS services

DLMS services are the tools for accessing COSEM object attributes or methods. Service requests include the attribute or method reference and may include data. Service responses include the result and may include data.

Aggregation of services means that with a single request/response several attributes and/or methods can be accessed. This mechanism reduces the number of message exchanges necessary to perform all required operations.

"With-list" type services allow aggregation of services of the same kind. They are available with the GET, SET, ACTION, READ, WRITE, and UnconfirmedWrite services. With-list type service requests carry a list of attribute / method references and the related list of data. With-list type service responses carry the list of results and the list of data as applicable.

The ACCESS service is a unified GET – SET – ACTION service. It allows reading and writing several attributes, and invoking several methods with a single request / response.

Composable messages

The concept of composable messages is the most powerful efficiency tool in the DLMS/COSEM application layer. It combines several process steps that allow reducing both the size of the message and the number of exchanges for an optimal use of communication channel capacity. The concept is shown in Figure 1.

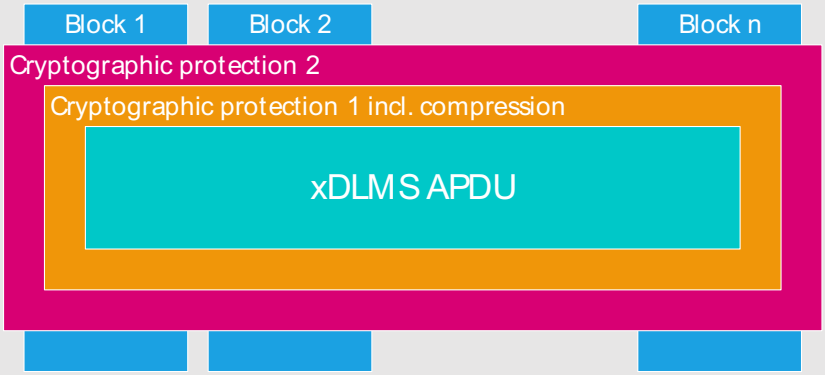

Figure 1 – Composable messages

First, the xDLMS APDU – that may represent a single service primitive or several service primitives aggregated – is encoded. Notice that the data carried by the ADPU may be already optimized by applying one or more mechanisms on the COSEM object model level discussed above.

In the second step, data may be compressed to remove redundancy. DLMS/COSEM uses V.44 compression.

In the third step, one or more layers of cryptographic protection may be applied to ensure confidentiality, authenticity, integrity and non-repudiation as needed.
The resulting APDU may be transferred in blocks using the general block transfer mechanism. This mechanism allows simultaneous bi-directional transfer, streaming and also supports lost block recovery.

Pre-established and persistent Application Associations

DLMS/COSEM data exchange takes place in Application Associations (AAs) that identify the partners and define the context within a session including the use of ciphering, the list of services available and message lengths. At the beginning of a communication session, the AA has to be established and the contexts are negotiated. This may comprise authentication of the peers. At the end of the session, the AA has to be released. This requires several message exchanges. See Figure 2.

.

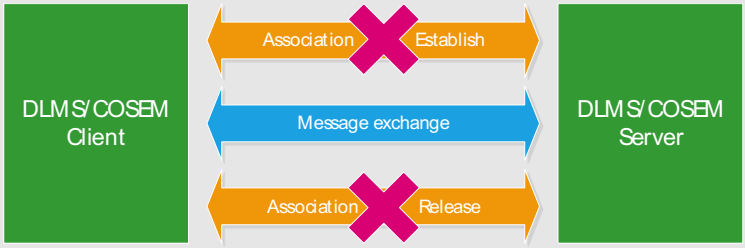

Figure 2 – DLMS communication session with explicitly established and pre-established AAs

On lossy networks, connection may be intermittent. An explicitly established AA is released when the lower layer connection is lost. Persistent AAs allow continuing data exchange when the connection is back, without the need to re-establish the AA. Persistent AAs can be released and established again if it is necessary to change the contexts.

Pre-established AAs can also be used if the parties have already agreed on the contexts in advance, e.g. through reference to a companion specification. Pre-established AAs cannot be released.

With these mechanisms, the exchanges to establish and release the AAs are spared.

Pull and Push operation

DLMS specifies two kinds of operations, as shown in Figure 3:

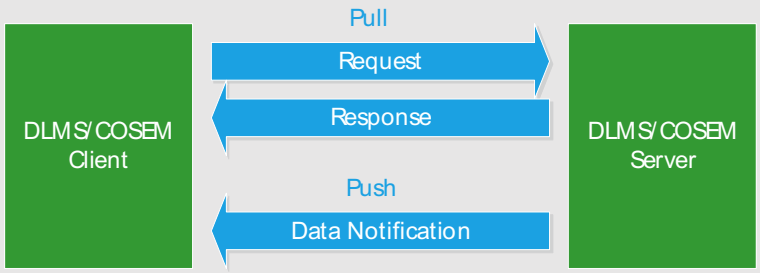

Figure 3 – Pull and Push operation

- Pull operation, when the client (e.g. the Head End System) sends a request and the server (e.g. a meter) sends the response. The services are GET, SET, ACTION, ACCESS, and Read / Write;

- Push operation, when the server sends – "Pushes" – a pre-defined set of data on pre-defined conditions to pre-defined destinations. The client may be able to change all three elements whenever needed. The push operation is generally triggered locally multiple times. With this, the service requests can be spared. However, it is also possible to trigger to push remotely by the client, for example to recover missing data.

The service available for this purpose is DataNotification. The mechanisms described under *Composable messages* apply.

Broadcasting and multicasting

Broadcast and multicast messages – when supported by the communication media – can be used to send information from the HES to multiple end devices with a single request. This mechanism – shown in Figure 4 – can be used to send tariff programs, synchronize clocks and other similar tasks.

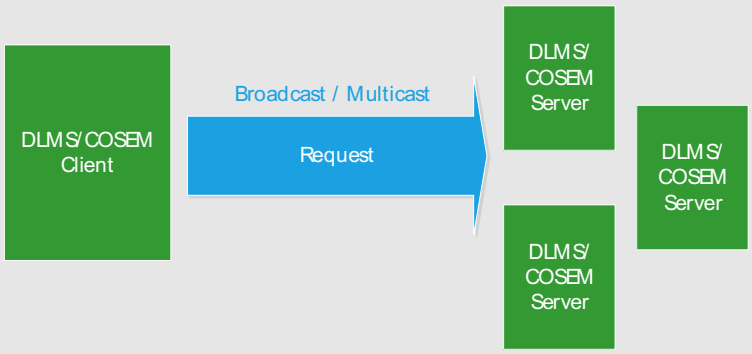

Figure 4 – DLMS Broadcast / multicast

Another very useful application is the *Image transfer* mechanism, which allows downloading and activating a new firmware to groups of devices. With this mechanism, lost firmware blocks can be recovered in a one-to-one relationship.

Example use cases

Example use cases for reading large data include "Association LN" object_list, reading or pushing of a load profile buffer for multiple channels with status and timestamp, firmware upgrade with unconfirmed services using broadcast or multicast, show up to ten times reduction in the number of bytes transferred, and the number of messages exchanged when the mechanisms presented are suitably used.

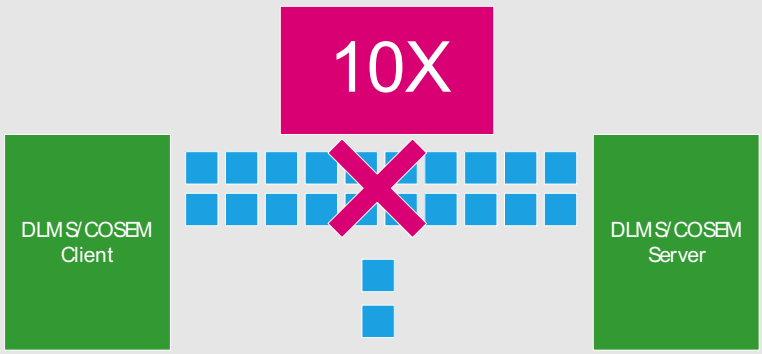

Figure 5 – Efficiency potential in DLMS/COSEM

If there are any questions or suggestions, please contact:
ed@dlms.com, technical@dlms.com