



DLMS based smart meter

- Introduction to smart meter
 - Trends in metering
 - Liberalized energy market requirements
- DLMS based smart meter over view
 - Key features of DLMS
 - DLMS User Association
- Data modeling in DLMS
 - Basic data models
 - Data models to suit smart metering requirements

Introduction to smart meter

Energy meter - introduction

- Energy Meters - key player in power system
- Record the consumption
- Uses
 - Billing
 - Identify technical and commercial losses
 - Understand load patterns
 - Tamper detection

Trends in metering



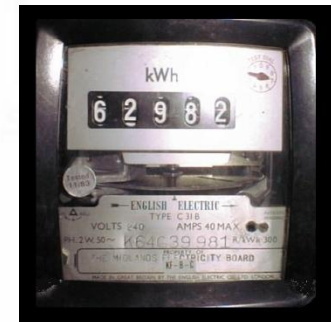
Trends in metering

Functional	Electro mechanical	Electronic	Smart meter
Measurement	Coil, Rotating Discs and Counters	ADC's, DSP-Micro-Processor	Metering and Communication ASIC
Storage	Nil	EPROM,RAM, Flash	EPROM, RAM, Flash
Communication	Nil	Optical/RS232/RS485	PLC/GPRS/CDMA/RF Mesh/Wi-Max ..
Protocols	Nil	Proprietary/Open Protocol	Open Protocols DLMS(IEC-62056)/ANSI C12 /M-Bus
Other functions	Nil	Multi tariff, billing schedules	Remote Connection/Disconnection, Demand Response/Real-time pricing/Sub-Meter/HAN

Why smart meter

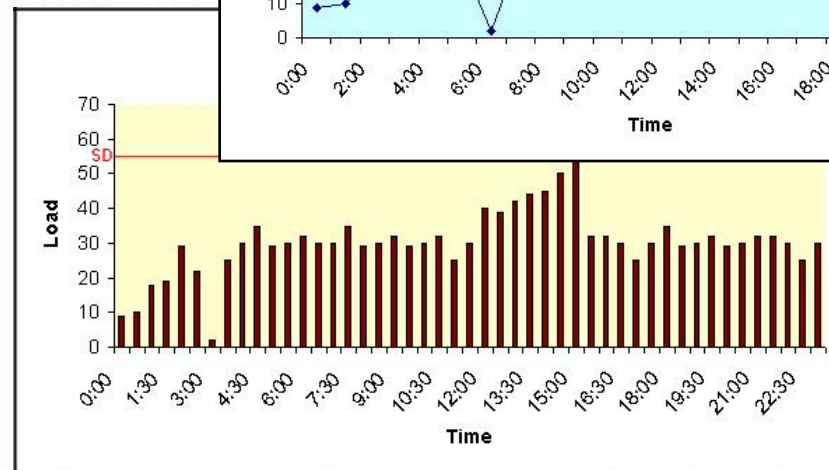
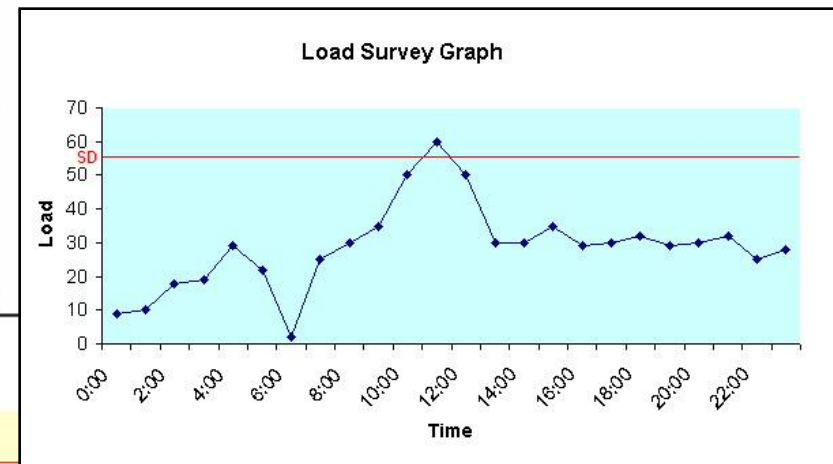
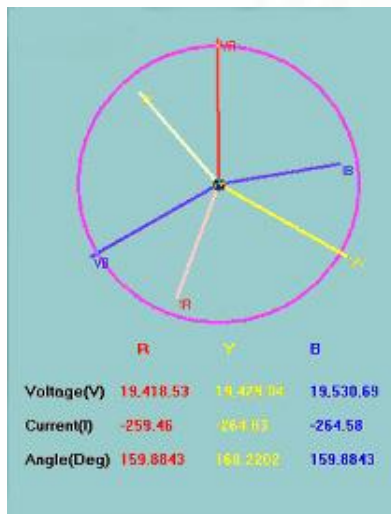
- To meet liberalized energy market requirement

Manual reading
outdated



AMI requirements

- Periodic meter reading(example – hourly) for AMI to analyze power quality, consumption trend



AMI requirements

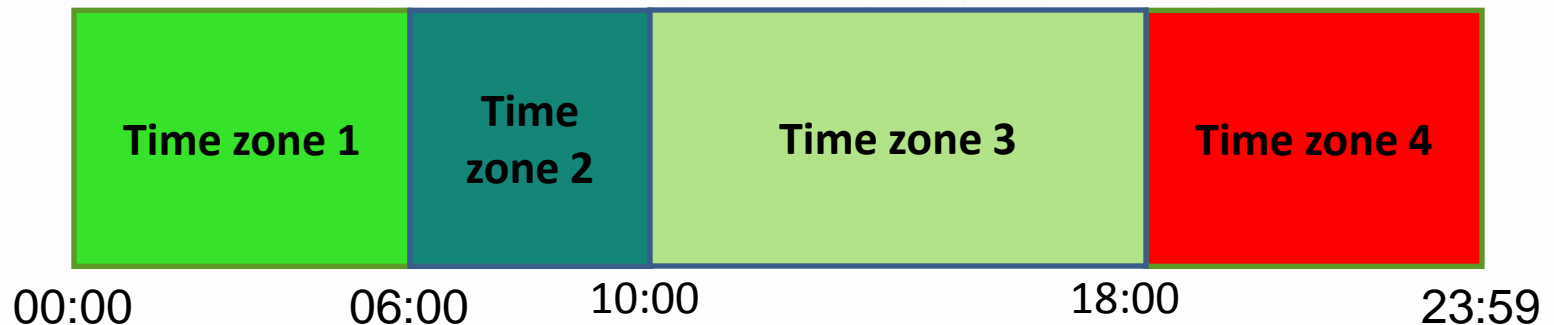
- All billing data stored in meter
- Programmable billing dates
- Multi tariff data inside meter



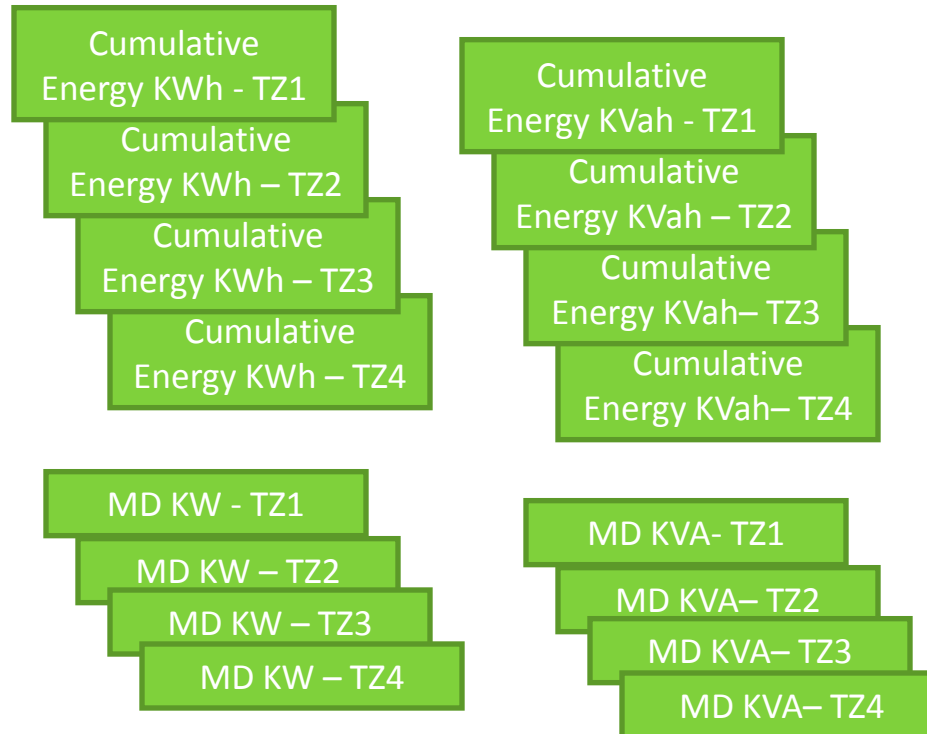
Multi tariff - illustration

Time zone 1– Tariff1
Time zone 2– Tariff2
Time zone 3– Tariff3
Time zone 4– Tariff4

Tariff4 > Tariff2 > Tariff3 > Tariff1



Smart meter with separate tariff registers

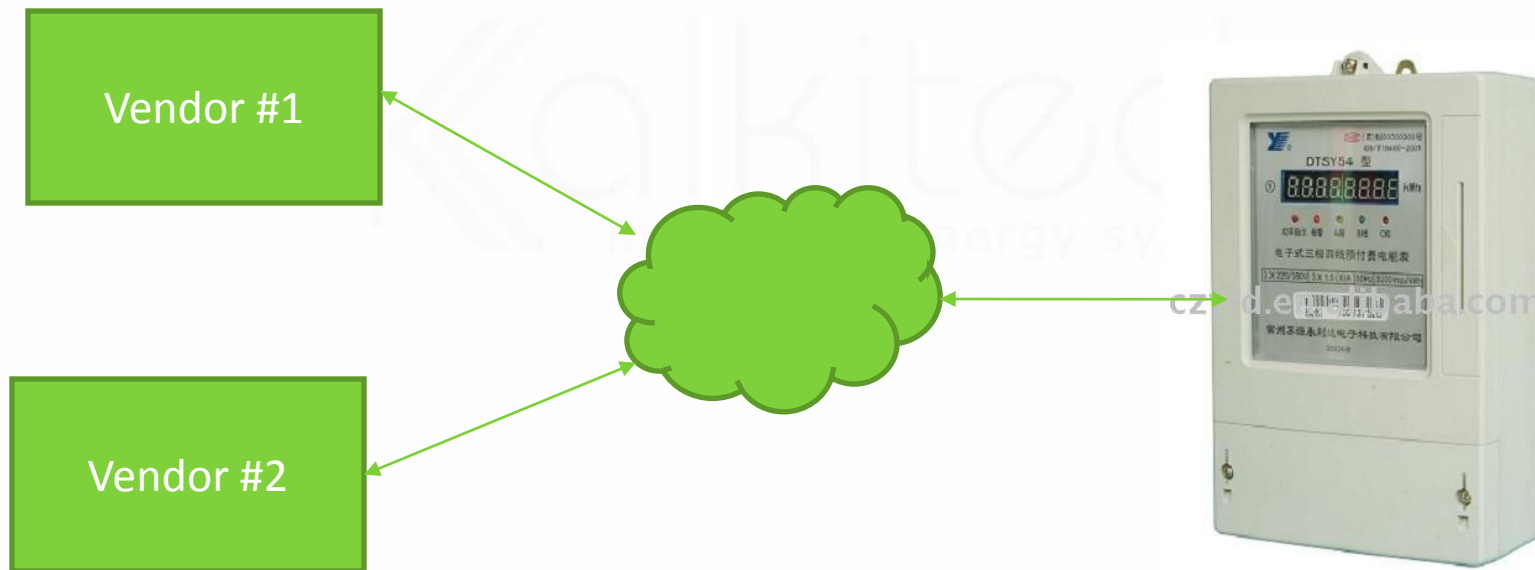


Demand response



Futuristic options

- Intelligent meter can decide out of multiple vendors(distribution companies)



A smart meter

- Enhanced electronic meter
- Vital component for a smart grid based on AMR/AMI



- Reliable and efficient data communication interface
- Demand response
- Multi tariff
- Historic storage
- Programmable tariff, billing schedules
- Firmware download

DLMS based smart meter overview

Liberalized energy market requirements

- Unbundling of monopolistic utilities
- Introduction of competition in all activities: – generation – transport – supply – customer management - meter operation – meter reading – meter data management
- Geographical dispersion, volatile customer base

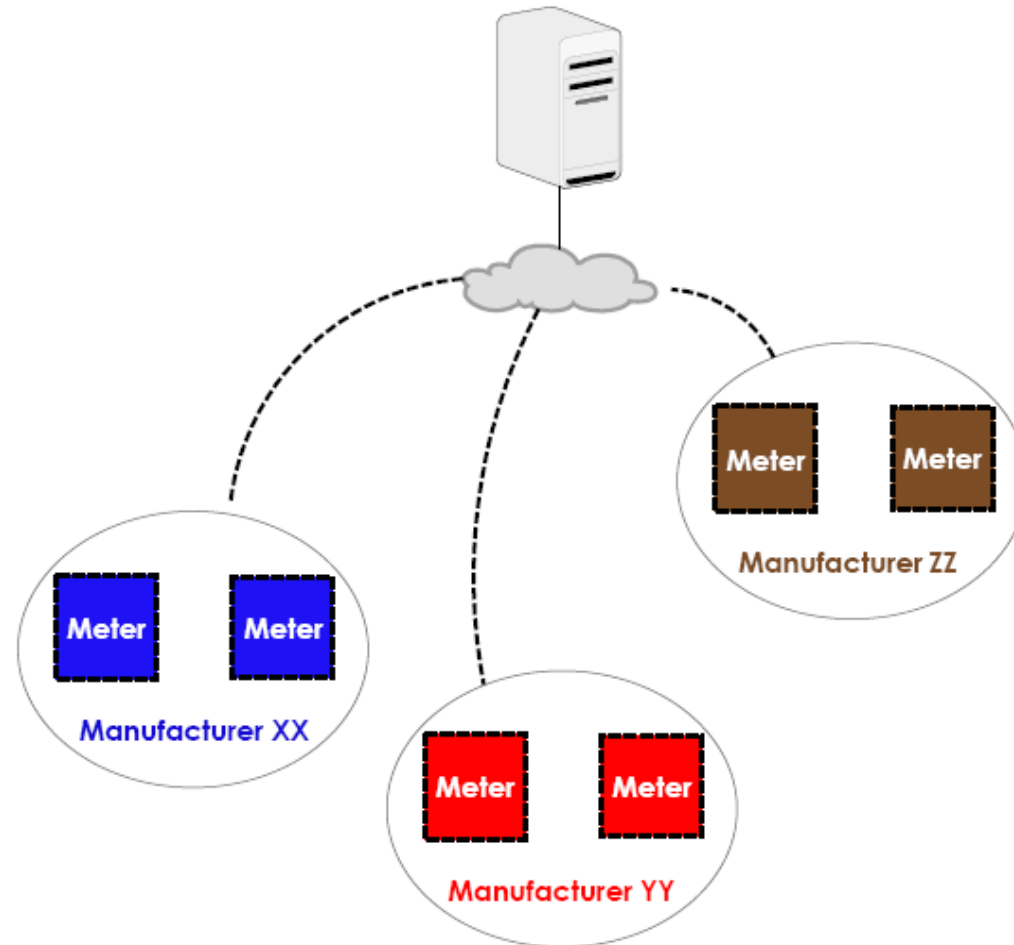
Liberalized energy market requirements

- Multi-energy - multi-user - multi-vendor environment
- Need selective and secure access to data
- **Need interoperability**

- The ability of a system or a product to work with other systems or products without special effort on the part of the customer
 - Any system can read any meter
 - Any meter can be read by any system
 - No special involvement of vendors
- To achieve interoperability we **need standards**

DLMS is the most popular metering standard

Interoperability



Interoperability testing

- DLMS User Association's Conformance Test tool(CTT) to ensure interoperability
 - A must with modern communication standards
 - Verify that standard is properly implemented:
good / bad / marginal cases
 - Simple self-testing system



DLMS logo

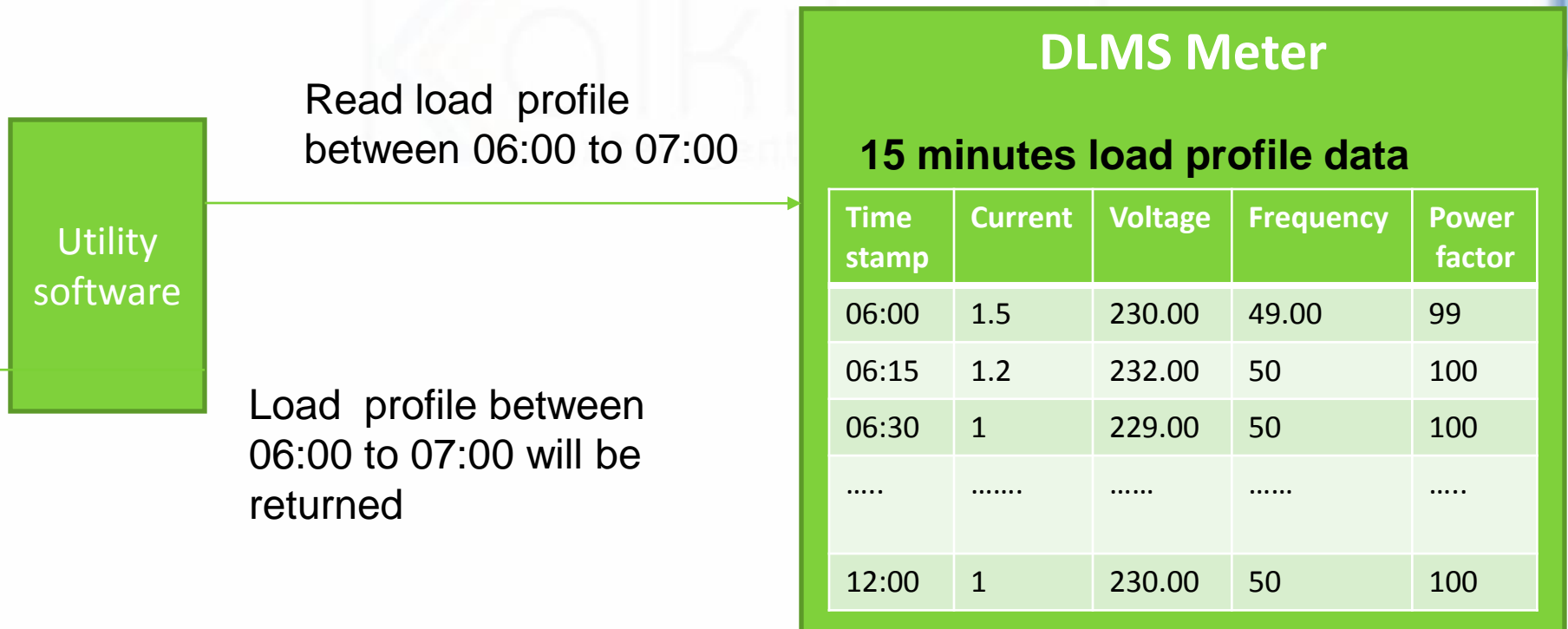
Why DLMS

- DLMS is comparable to a set of rules or a common language, on which the various operators have agreed.
- The DLMS-protocol enables the integration of energy meters with data management systems from other manufacturers. This secures that the energy supplier gets the full advantage of the meter functions.

- The utility that has invested in a smart metering solution pulls an enormous amount of information out of the meters – information that can be used for a lot more than billing purposes such as
 - Load control,
 - Development of tariff models for special customer segments
 - Energy trade

Selective access

- Low communication overhead
- Only necessary data reaches utility software



Data presentation – ASN1

```
GET-Request-Normal ::= SEQUENCE
{
    invoke_id_and_priority ::= bit string
    cosem_attribure_descriptor
    {
        class_id ::= unsigned16
        instance_id ::= octet-string
        attribute_id ::= Integer8
    }
}
```

Clear data structure, whatever the communication technology

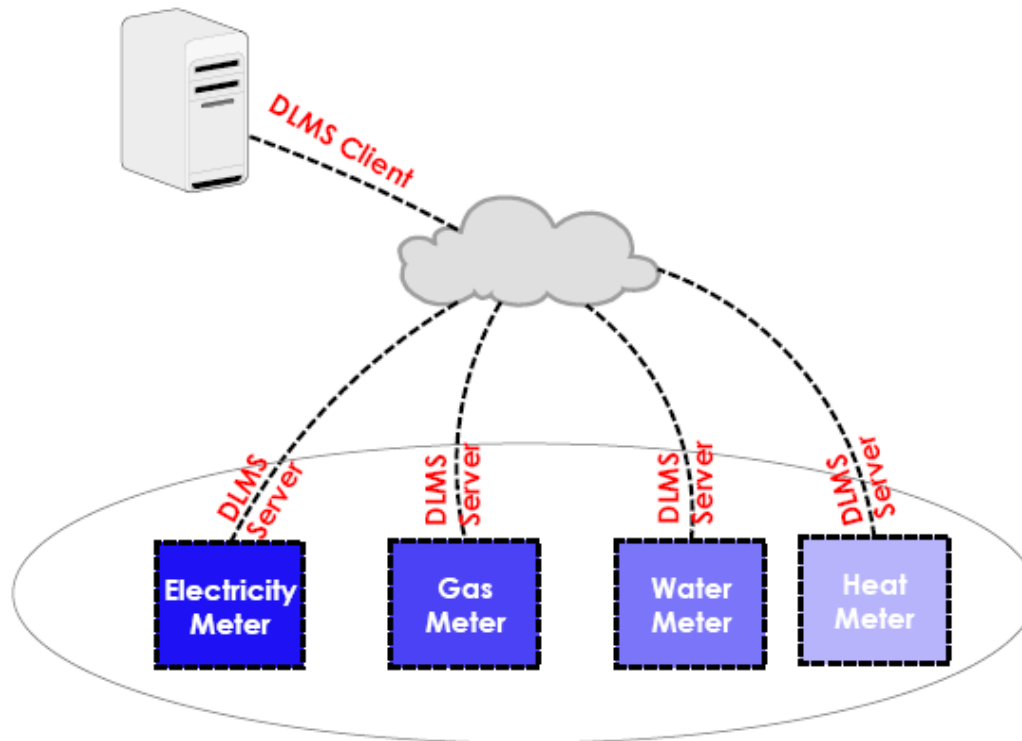
- Access control
 - Lowest level
 - Low level
 - High level(4 pass authentication)
- Data security
 - AES GCM

Standard data identifiers and data models

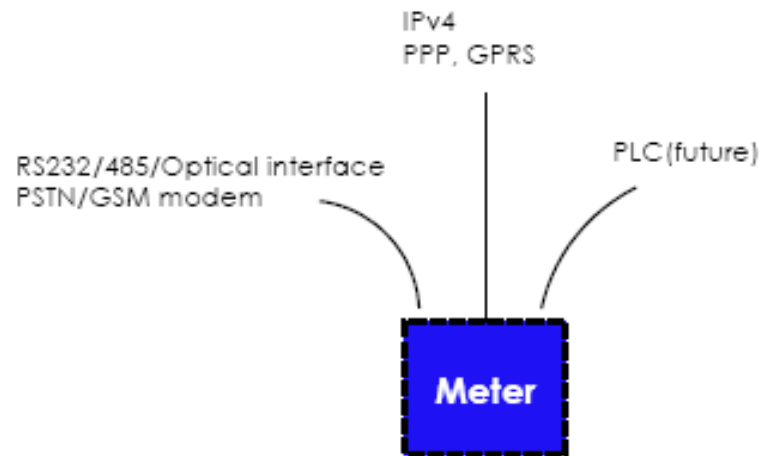
- Unambiguous data identification using standard OBIS codes
- Interface classes



Multi energy support



Multi communication media support



DLMS User Association

- Formed in 1997
- 160+ members
- Presence in 5 continents, 40 countries
- From all branches of industry such as utilities, meter and system providers
- 123 product certificates

DLMS popularity

- Started from Europe
- Most popular metering protocol in the world today with strong presence in Europe, Asia, Africa
- Important smart metering projects based on DLMS/COSEM - China, France, India, the Netherlands, Middle East, Scandinavia, Spain, South Korea...

DLMS UA key roles

- Development and enhancement of standards(Working Group) – releases standards in the form of colored books
- Conformance Test Tool – development, maintenance and upgrades

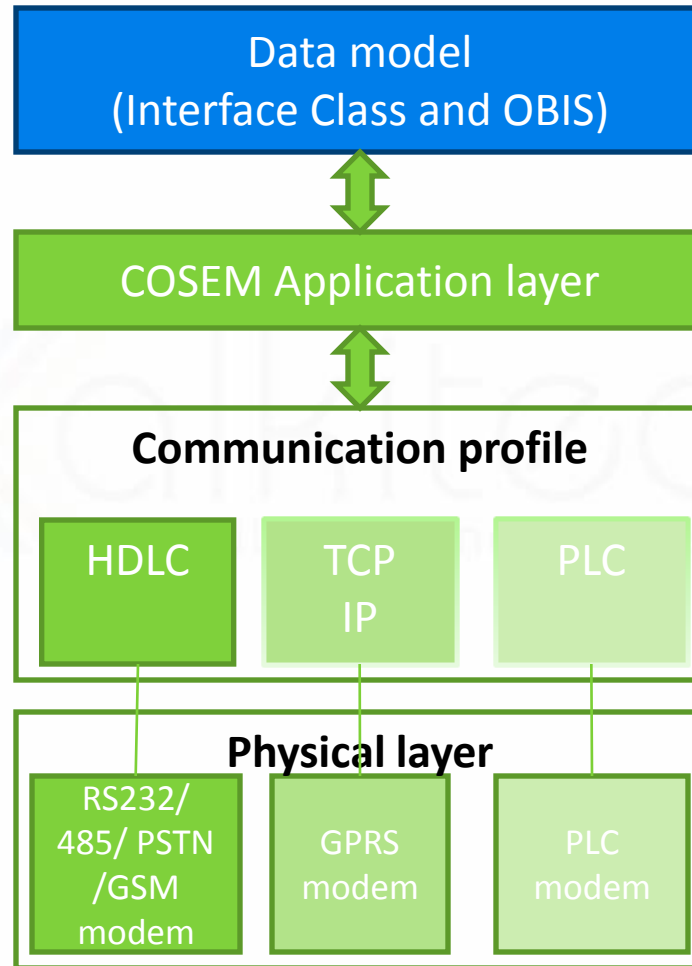
Evolution of DLMS based AMR standards

- 1992 - IEC 61107 “FLAG”: simple protocol for (local) reading
- 1996 - DIN 43863-3 “EDIS”: Identification system. IEC 61334-4-41 “DLMS”: Application layer protocol
- 2002 - IEC 62056 “COSEM”: Interface model for e-meters and “DLMS” based OSI protocol. EN 13757-1: IEC 62056 adapted for gas, water, heat..
- 2005 - IEC 62056 Ed. 2: TCP-UDP/IP profile added
- Added connect/disconnect and PLC setup classes

DLMS COSEM specification

<u>IEC specification</u>	<u>Description</u>	<u>DLMS Colored book</u>
62056-62	Interface Class	BLUE book
62056-61	OBIS code	
62056-53	COSEM Application layer	GREEN book
62056-47	COSEM transport layer for IPv4 n/w	
62056-46	HDLC	
62056-42	Physical	
62056-21 Mode E	Direct local data exchange	
	Conformance test procedure	YELLOW book
	Glossary of terms	WHITE book

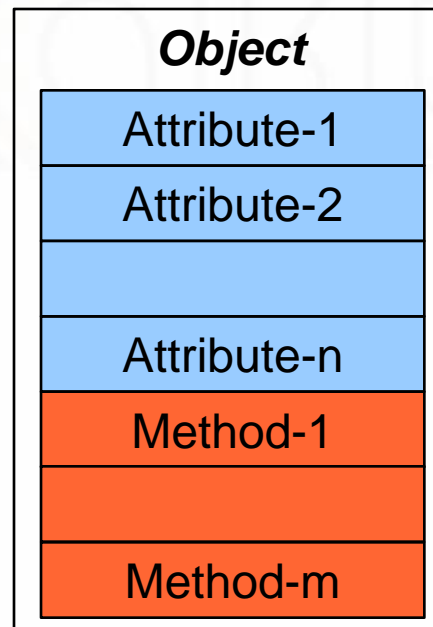
DLMS stack overview



DLMS data modeling

COSEM Object

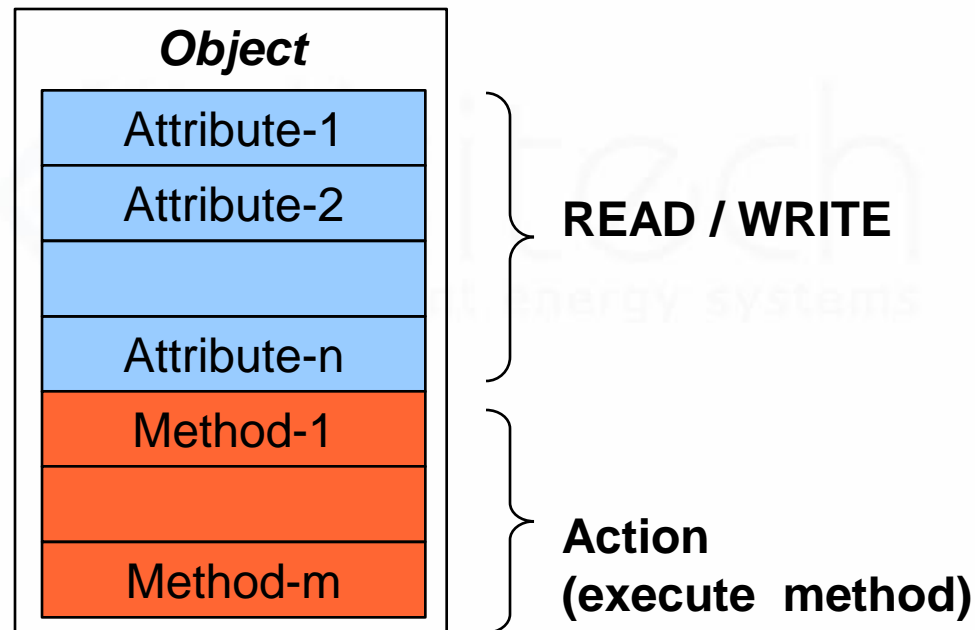
- DLMS models all meter data as objects
- Abstraction of real world things
- Collection of attributes and methods



COSEM Object

- The information of an object is organized in attributes. They represent the characteristics of an object by means of attribute values
- An object may offer a number of methods to either examine or modify the values of the attributes.

COSEM object



Interface class

- Objects that share common characteristics are generalized as an interface class with a class_id
- Within a specific class, the common characteristics (attributes and methods) are described once for all objects
- Instantiations of an interface class are called COSEM interface objects

IC and objects

IC = 1	
Attributes	Data type
Attribute-1	Octet string
Attribute-2	Choice

Object
 Attribute 1 :
 Attribute 2 :

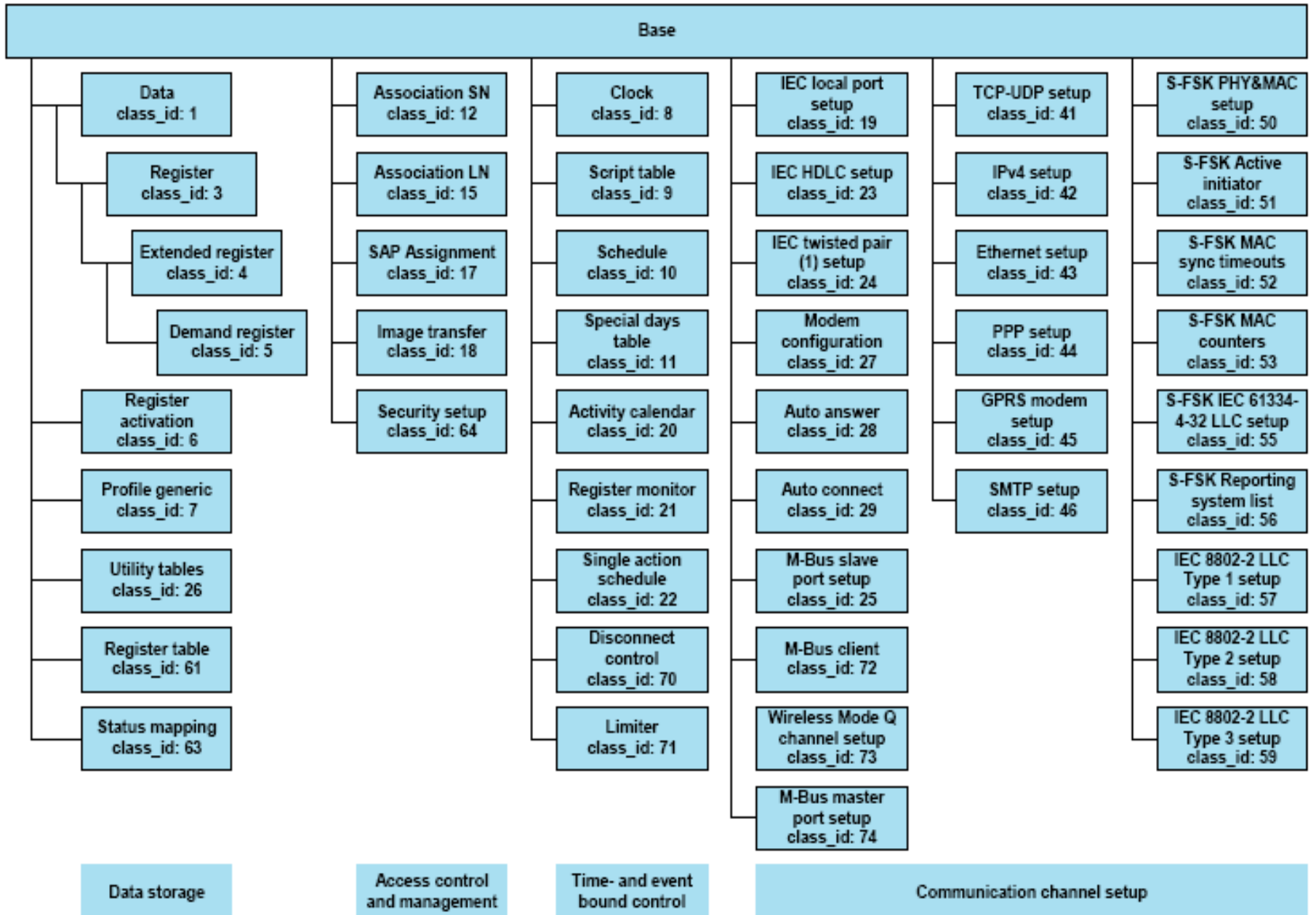
Object
 Attribute 1 :
 Attribute 2 :

IC = 3	
Attributes	Data type
Attribute-1	Octet string
Attribute-2	Choice
Attribute-2	Structure
Methods	
Method-1	

Object
 Attribute 1 :
 Attribute 2 :
 Attribute 3 :
 Method 1 :

Object
 Attribute 1 :
 Attribute 2 :
 Attribute 3 :
 Method 1 :

Overview of interface classes



- Used to store configuration data and parameters.

Data	0...n	class_id = 1, version = 0			
Attribute(s)	Data type	Min.	Max.	Def.	Short name
1. logical_name (static)	octet-string				x
2. value	CHOICE				x + 0x08
Specific methods (if required)	m/o				

Register (IC:3)

- Used to store a process value or a status value with its associated unit.

Register	0...n	class_id = 3, version = 0			
Attribute(s)	Data type	Min.	Max.	Def.	Short name
1. logical_name (static)	octet-string				x
2. value (dyn.)	CHOICE				x + 0x08
3. scaler_unit (static)	scal_unit_type				x + 0x10
<i>Specific methods (if required)</i>	<i>m/o</i>				
1. reset (data)	o				x + 0x28

Register (IC:3)

Attributes

- Value : process or status value
- Scaler unit : qualifies “value”
 - Scaler : exponent (to the base of 10) of the multiplication factor.
 - Unit : Enumeration defining the physical unit
 - 64 units defined in DLMS

Units for “value”

- Time
- Phase angle
- Temperature
- Currency
- Length
- Speed
- Volume
- Corrected volume
- Volume flux
- Corrected volume flux
- Mass
- Force
- Energy
- Pressure
- Thermal power
- Active power
- Apparent power
- Reactive power
- Active energy
- Apparent energy
- Reactive energy
- Current
- Electrical charge
- Voltage
- Electrical field strength
- Capacitance
- Resistance
- Resistivity
- Volt-squared hour
- Ampere-squared hour
- Mass flux
- Conductance
- Temperature
- Dynamic viscosity
- Mass density
- Magnetic flux
- Magnetic flux density
- Magnetic field strength
- Inductance
- Frequency
- R_w
- R_s

Register (IC:3)

Illustration

Attribute		Data type	Value
Logical Name		Octet string	1.1.72.7.0.255
Value		Double long unsigned	2309
Scaler Unit	Scaler	Integer	-1
	Unit	Enum	35

Volts

L3 voltage

Thus, L3 voltage = 230.9 volts

Register (IC:3)

- Reset method forces a reset of the object. By invoking this method, the value is set to the default value.



Extended Register (IC:4)

- “Extended register” class store a process value with its associated status, unit, and time information.

Extended register		0...n	class_id = 4, version = 0			
Attribute(s)		Data type	Min.	Max.	Def.	Short name
1. logical_name	(static)	octet-string				x
2. value	(dyn.)	CHOICE				x + 0x08
3. scaler_unit	(static)	scal_unit_type				x + 0x10
4. status	(dyn.)	CHOICE				x + 0x18
5. capture_time	(dyn.)	octet-string				x + 0x20
<i>Specific methods (if required)</i>		<i>m/o</i>				
1. reset (data)		o				x + 0x38

Extended Register (IC:4)

Attributes

- In addition to all attributes and methods of “Register class”, “Extended register” has the following additional attributes
 - Status : Extended register specific status information
 - Capture time : Provides an “Extended register” specific date and time information showing when the value of the attribute "value" has been captured.

Demand Register (IC:5)

Demand register		0...n	class_id = 5, version = 0			
<i>Attribute(s)</i>		<i>Data type</i>	<i>Min.</i>	<i>Max.</i>	<i>Def.</i>	<i>Short name</i>
1.	logical_name (static)	octet-string				x
2.	current_average_value (dyn.)	CHOICE			0	x + 0x08
3.	last_average_value (dyn.)	CHOICE			0	x + 0x10
4.	scaler_unit (static)	scal_unit_type				x + 0x18
5.	status (dyn.)	CHOICE				x + 0x20
6.	capture_time (dyn.)	octet-string				x + 0x28
7.	start_time_current (dyn.)	octet-string				x + 0x30
8.	period (static)	double-long-unsigned	1			x + 0x38
9.	number_of_periods (static)	long-unsigned	1		1	x + 0x40
<i>Specific methods (if required)</i>		<i>m/o</i>				
1.	reset (data)	o				x + 0x48
2.	next_period (data)	o				x + 0x50

Demand Register (IC:5)

- Instances of a “Demand register” class store a demand value with its associated status, unit, and time information.

Demand Register (IC:5)

- Current average value : Provides the current value (running demand) of the energy accumulated since start_time divided by number_of_periods*period
- Last average value : Provides the value of the energy accumulated (over the last number_of_periods*period) divided by number_of_periods*period
- Capture time : Provides the date and time when the last_average_value has been calculated

Demand Register (IC:5)

- Start time current : Provides the date and time when the measurement of the current_average_value has been started
- Period : Period is the interval between two successive updates of the last_average_value
- Number of periods : The number of periods used to calculate the last_average_value.

Demand Register (IC:5)

- Reset : Activating this method provokes the following actions
 - the current period is terminated
 - the current_average_value and the last_average_value are set to their default values
 - the capture_time and the start_time_current are set to the time of the execution of reset(data).

Demand Register (IC:5)

- Next period : This method is used to trigger the regular termination (and restart) of a period.
 - Closes (terminates) the current measuring period
 - Updates capture_time and start_time and copies current_average_value to last_average_value
 - Sets current_average_value to its default value
 - Starts the next measuring period

Profile (IC:7)

- Generalized concept to store dynamic process values of capture objects

<i>Attribute(s)</i>	<i>Specific methods (if required)</i>
1. logical_name (static)	1. reset (data)
2. buffer (dyn.)	2. capture (data)
3. capture_objects (static)	3. <i>reserved from previous versions</i>
4. capture_period (static)	4. <i>reserved from previous versions</i>
5. sort_method (static)	
6. sort_object (static)	
7. entries_in_use (dyn.)	
8. profile_entries (static)	

Profile (IC:7)

- Buffer: stores dynamic process values as capture objects. This attribute contains a sequence of entries. Each entry contains values of the captured objects.
- Capture object: Specifies the list of capture objects that are assigned to the object instance. Upon a call of the capture (data) method or automatically in defined intervals, the selected attributes are copied into the buffer of the profile.

Profile (IC:7)

- Capture period
 - ≥ 1 : automatic capture
 - 0 : Capturing is triggered externally or capture events occur asynchronously

Profile (IC:7)

- Sort method
 - FIFO
 - LIFO
 - Largest
 - Smallest
 - Nearest to zero
 - Farthest from zero

Profile (IC:7)

- Sort method: Unsorted buffer works as “first in first out ” buffer
- If the profile is sorted, a call to capture () will store the new entry at the appropriate position in the buffer, moving all following entries and probably losing the least interesting entry.

Profile (IC:7)

- Sort method : specifies the object that the sorting is based upon (only for sort methods other than FIFO and LIFO)
- Entries in use: number of entries in buffer
- Profile entries: specifies maximum number of entries to be retained in buffer

Profile (IC:7)

Logical Name -----> 1.0.99.1.0.255

Buffer ----->

X1	Y1	01-01-07 06:00:00
X2	Y2	01-01-07 06:15:00
.	.	.
.	.	.
X16	Y16	01-01-07 10:00:00

Capture Objects ----->

IC	OBIS	Attr ID	Data indx
0003	01 00 00 06 00 FF	02	00
0003	01 00 00 06 01 FF	02	00
0008	00 00 01 00 00 FF	02	00

Capture period -----> 15 mnts

Sort method -----> FIFO

Sort object -----> Default

Entries in use -----> 16

Profile entries -----> 1000

Profile (IC:7)

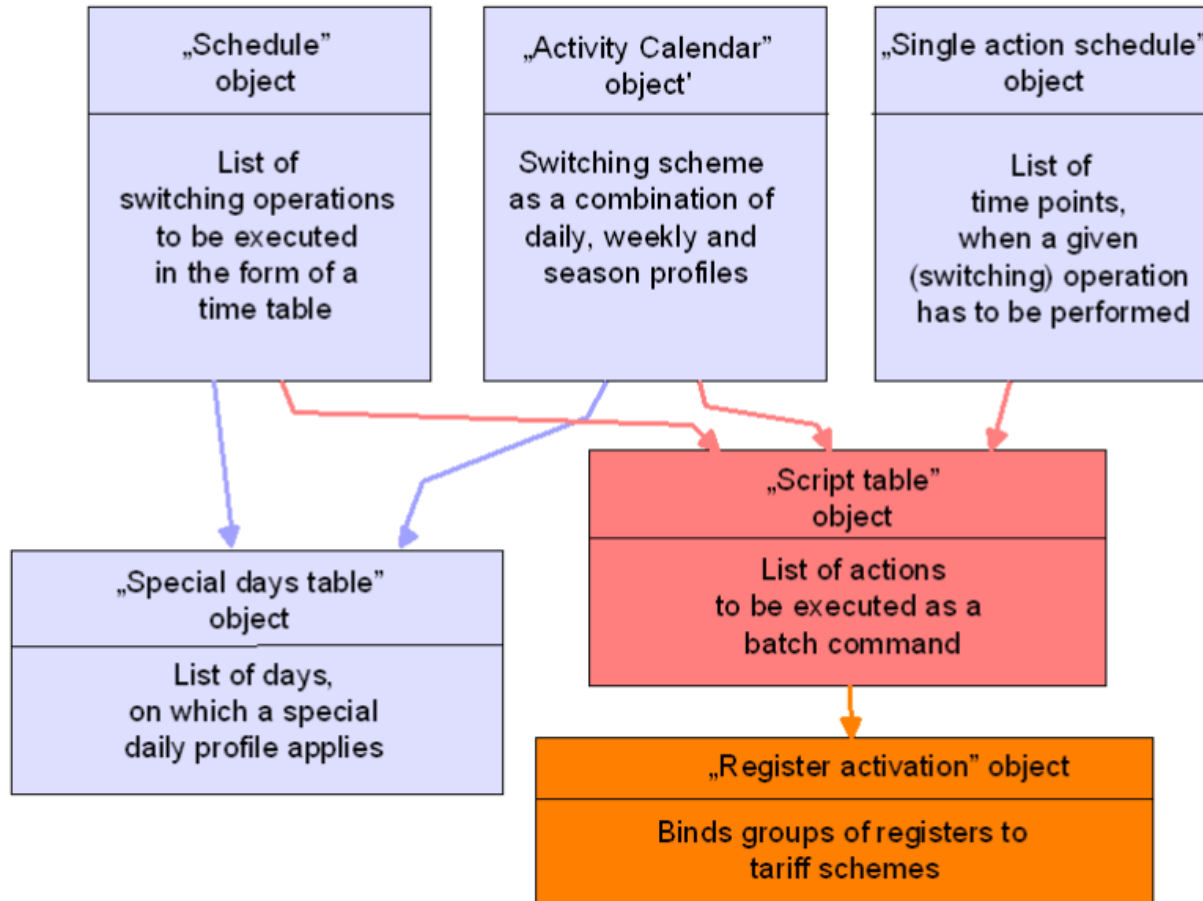
- Selective access : allows reading selected portion of attribute rather than the normal practice of accessing entire attribute
- Profile buffer supports selective access
- Types of selective access
 - Selective access by range
 - Selective access by entry

Clock (IC:8)

- Handles all information that is related to date and time, including leap years and the deviation of the local time to a generalized time reference (Greenwich Mean Time, GMT).

<i>Attribute(s)</i>	<i>Specific methods (if required)</i>
1. logical_name (static)	1. adjust_to_quarter (data)
2. time (dyn.)	2. adjust_to_measuring_period (data)
3. time_zone (static)	3. adjust_to_minute (data)
4. status (dyn.)	4. adjust_to_preset_time (data)
5. daylight_savings_begin (static)	5. preset_adjusting_time (data)
6. daylight_savings_end (static)	6. shift_time (data)
7. daylight_savings_deviation (static)	
8. daylight_savings_enabled (static)	
9. clock_base (static)	

Schedule and Calendar



Register monitor (IC:21)

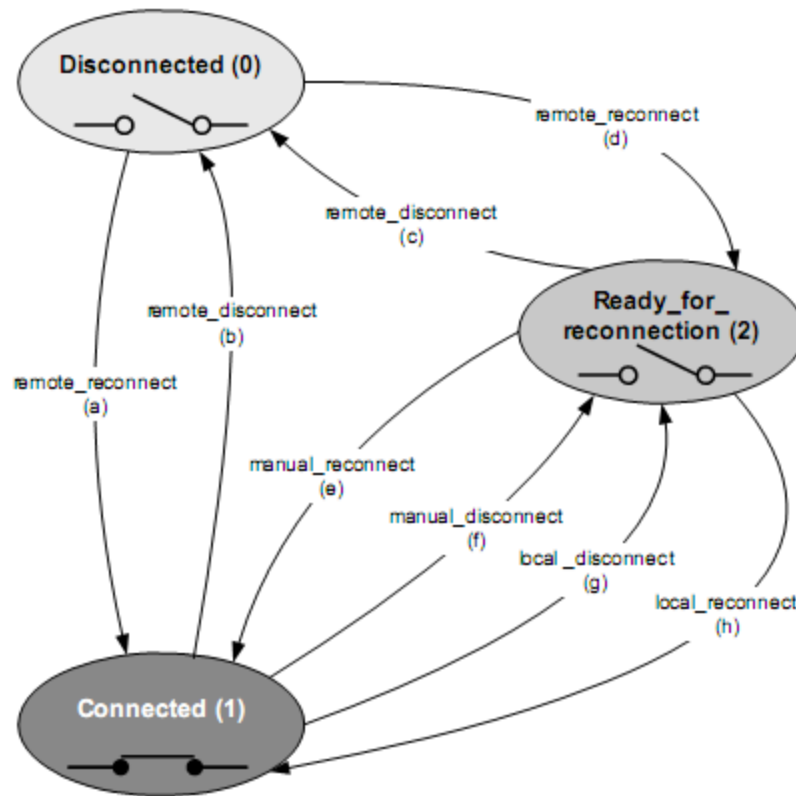
- This interface class allows defining a set of scripts that are executed when the value of an attribute of a monitored register type object “Data”, “Register”, “Extended register”, Demand register, etc. crosses a set of threshold values.

<i>Attribute(s)</i>	
1. logical_name	(static)
2. thresholds	(static)
3. monitored_value	(static)
4. actions	(static)

Disconnect Control(IC:70)

Instances of the Disconnect control IC manage an internal or external disconnect unit of the meter (e.g electricity breaker, gas valve) in order to connect or disconnect – partly or entirely – the premises of the consumer to / from the supply.

State diagram of Disconnect control IC



Disconnect Control(IC:70)

Disconnect control		0...n	class_id = 70, version = 0			
<i>Attributes</i>		<i>Data type</i>	<i>Min.</i>	<i>Max.</i>	<i>Def.</i>	<i>Short name</i>
1.	logical_name (static)	octet-string				x
2.	output_state (dyn.)	boolean				x + 0x08
3.	control_state (dyn.)	enum				x + 0x10
4.	control_mode (static)	enum				x + 0x18
<i>Specific methods</i>		<i>m/o</i>				
1.	remote_disconnect()	m				x + 0x20
2.	remote_reconnect()	m				x + 0x28

Image transfer (IC:18)

- Instances of the Image transfer IC model the mechanism of transferring binary files, called firmware Images to COSEM servers.

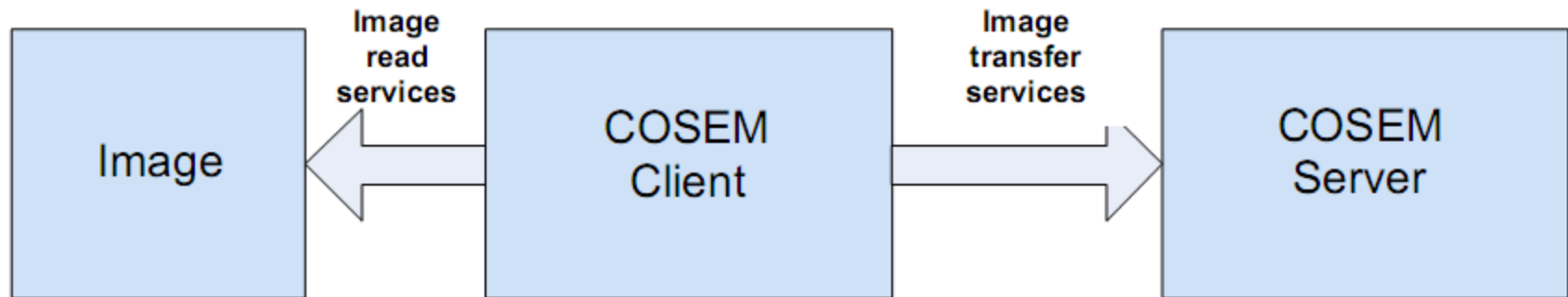


Image transfer (IC:18)

Image transfer		0...n	class_id = 18, version = 0			
<i>Attributes</i>		<i>Data type</i>	<i>Min.</i>	<i>Max.</i>	<i>Def.</i>	<i>Short name</i>
1.	logical_name (static)	octet-string				x
2.	image_block_size (static)	double-long-unsigned				x + 0x08
3.	image_transferred_blocks_status (dyn.)	bit-string				x + 0x10
4.	image_first_not_transferred_block_number (dyn.)	double-long-unsigned				x + 0x18
5.	image_transfer_enabled (static)	boolean				x + 0x20
6.	image_transfer_status (dyn.)	enumerated				x + 0x28
7.	image_to_activate_info (dyn.)	array				x + 0x30
<i>Specific methods</i>		<i>m/o</i>				
1.	image_transfer_initiate	m				x + 0x40
2.	image_block_transfer	m				x + 0x48
3.	image_verify	m				x + 0x50
4.	image_activate	m				x + 0x58