Project no. FP7-ICT- 223888

BeyWatch

Project title: Builiding EnergY Watcher

Instrument: CA STREP √ IP NOE

ICT - Information and Communication Technologies Theme

Deliverable D2.2: End-to-End Platform specification
BeyWatch Data Model (Annex)

Start date of project: 1/9/2008 Duration: 30

Organisation name of lead contractor for this deliverable: EDF

Revision [1]

Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013)		
Dissemination Level		
PU	Public	√
PP	Restricted to other programme participants (including the Commission Services)	
RE	Restricted to a group specified by the consortium (including the Commission Services)	
CO	Confidential, only for members of the consortium (including the Commission Services)	

European Commission
Information Society and Media

Disclaimer

The BeyWatch Consortium consists of the following companies:

No	Participant name	Participant short name	Country	Country
1	Telefonica I+D	TID	Co-ordinator	Spain
2	Synelixis Solutions Ltd	Synelixis	Contractor	Greece
3	Germanischer Lloyd (GL) Industrial Services UK Ltd	GL	Contractor	UK
4	Electricité de France S.A.	EDF	Contractor	France
5	Gorenje Gospodinjski Aparati D.D.	Gorenje	Contractor	Slovenia
6	Fagor Electrodomesticos S. Coop.	Fagor	Contractor	Spain
7	Universita Degli Studi Di Palermo	UniPa	Contractor	Italy
8	Sigma Orionis	Sigma	Contractor	France

Document Editors/Contributors:

Anne Le Mouel, Frédéric Silvi, Clément Solau (EDF), Pierre Plaza, Alberto Muñoz Gallego (TID), Menelaos Perdikeas, Theodore Zahariadis (Synelixis), Julen Ugalde, Unai Garro (Fagor), Jerabek Borislav (Gorenje), Costantino Giaconia, Diego La Cascia, Rosario Miceli (UNIPA).

This page has been intentionally left blank

Table of contents

Figures' List

Tables' List

Abbreviations

AES	Advanced Encryption Standard
AMM	Advanced Meter Management
API	Application Programming Interface
BSS	Business Support Systems
BWA	BeyWatch Agent
CAP	Compact Application Protocol
CPS	Combined Photovoltaic – Solar panel
DB	DataBase
DW	Dishwasher
EJP	Energie Jour de Pointe
FZ	Freezer
GUI	Graphical User Interface
HAN	Home Area Network
HGI	Home Gateway Initiative
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
MAC	Message Authentification Code
MPPT	Maximum Power Point Tracking
M2M	Machine To Machine
OSGi	Open Services Gateway initiative
PDA	Personal Digital Assistant
PLC	Power Line Communication
PV	Photovoltaic
RF	Radio Frequency
RG	Residential Gateway
SP	Solar Panel
ST	Solar Thermal
UI	User Interface
UPnP	Universal Plug and Play
VCC	Variable Capacity Compressor
WC	Working Condition
WM	Washing Machine
WPAN	Wireless Personal Area Network
6loWPAN	IPv6 over Low power Wireless Personal Area Networks

Annex

This is an annex that provides the BeyWatch Data Model and interface specification. It should be considered as a Quick Reference. For more information please refer to the BeyWatch deliverable D2.2. "End-to-end Platform Specification".

1.1. BSS - General technical specifications

1.1.1. Data Model

The following tables show the initial set of data that the BSS and Energy Management Service would require in order to start working, having in mind the architecture shown in previous figures. The key entities are identified: Residential Gateway, Customer, User (there might be some users per utility customer), Operator, Node, Device or Energy Profile.

This data model will be amplified and modified through the evolution of the project, but can serve as a starting point for work and initial specification.

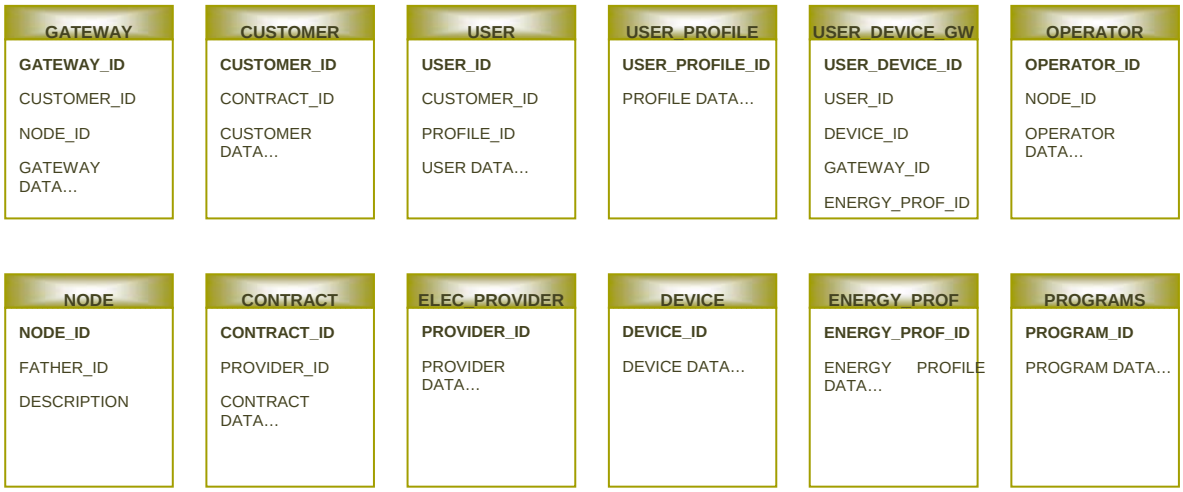

Figure 1 - Draft Data model for Energy Management Service

1.1.2. Electricity provider APIs

This **API** offers the different electricity companies the possibility to monitor power consumption figures, trends and past historic information supplied by BeyWatch Supervisor. Apart from that, this API allows them also to initiate actions on the Supervisor that will populate these actions to the corresponding BeyWatch Agents.

The idea is to publish this API and to let each Electricity Company to develop its own portal, so it will have the look&feel of the company and will make available to the electricity company operator the functionality needed in the appropriate format for him.

The expected methods available for the electricity providers are the following:

For Electricity Providers Management

- createProvider(providerData, password): this method will add a new provider to the database.
- modifyProviderData(providerData): will modify the provider data.
- deleteProvider(providerId): will delete the provider reference and associated data.
- authenticate(login, password): will provide user validation for Provider portal.

- modifyPassword(providerId, newPassword): will modify the Provider password.
- getProviderData(providerId): will retrieve the data associated to the identified provider.

For BSS Specific Functionality

- addContractType(contractData): this method will include a new type of contract in the database. The type of contracts will determine the user alternatives for tariff optimization .
- deleteContractType(contractTypeId): will delete a contract type from the database.
- getContractTypes(providerId): will retrieve al the contract types offered by a specific provider.
- getAllSubscribers(providerId): will retrieve a list of users subscribed to the service and the specific provider.
- getSubscribersPerContractType(providerId, contractTypeId): will retrieve a list of users subscribed to a specific contract and provider.

For Supervisor functionality Access

- getInstantConsumption(nodeId): will retrieve the current consumption in one specific node controlled by the Supervisor addressed.
- getInstantConsumption(nodeId[]): will retrieve several instant consumptions for data comparison.
- getAccumulateConsumption(nodeId, timePeriod): will retrieve accumulated consumption in a specific period from a specific node.
- getAccumulateConsumption(nodeId[],timePeriod): will retrieve accumulated consumption in a specific period from a set of nodes.
- getPeakPowerDemand(nodeId): will retrieve the limits for power demand in a specific node.
- getPeakPowerDemand(nodeId[]): will retrieve the limits for power demand in a set of nodes.
- notifyPowerCut(nodeId, startTime, stopTime) : will automatically notify a power cut to a specific node (via Supervisor).
- decreasePowerDemand(nodeId, percentage): will send the petition of lowering power demand to a specific node.
- decreasePowerDemand(nodeId, percentage, time): will send the petition of lowering power demand to a specific node at a specific time.

1.1.3. End User APIs

The customer is the person that subscribes the service (pays for it). He is the owner of the residential gateway (RG) (or rents the RG to the telco operator or service provider, depending on the business model). There is one customer per home that can register as many users as he wants. Energy related preferences will vary depending on the user. The data model allows the possibility of associating each user to some or all appliances in the home, and to set the energy profile that will apply for white appliance depending on the user. The expected methods available for users are:

User Management

- createUser(userData, password): will add a user to the database and give it access to the Graphic User Interface.
- modifyUserData(userData): will modify the user data once the session in established.

- deleteUser(userId): will delete a user from the database.
- authenticate(login, password): will allow the user to enter the application.
- modifyPassword(userId, newPassword): will modify the user password in the database.
- getUserData(userId): will retrieve the user data associated to userId.
- isCustomer(userId): will determine if the user is the utility customer.

BSS Functionality

- getEnergyProviders(): will retrieve the available Energy Providers in the DB.
- getEnergyProviderContractTypes(providerId): will retrieve the available contracts and tariffs from the Provider.
- getAvailableContractTypes(): will retrieve the different contract types available.
- getContractTypeDetails(contractId): will retrieve the tariff details from a specific contract.
- subscribeContractType(contractId, customerId): the user subscribes to a specific Contract type.
- unsubscribeContractType(contractId, customerId): the user unsuscribes from a the contract.

Appliance Management

- getAppliances(gatewayId): the GUI will retrieve the appliances available in the DB associated to a specific RG.
- getApplianceFunctionality(deviceId): will return the profile for inputs/outputs in the appliance.
- getApplianceStatus(gatewayId, deviceId): will return the status of a specific appliance;
- setApplianceStatus(gatewayId, deviceId, params[]): will set an action depending on the preferences.
- getApplianceInstantConsumption(gatewayId, deviceId): will get the appliance consumption at a specific moment.
- getApplianceAccumulateConsumption(gatewayId, deviceId, timePeriod): will get the aggregated consumption of a specific device.
- getElectricityConsumptionStats(gatewayId, deviceId, startDate,finishDate, timeInterval): advanced electricity consumption information retrieval.
- getPrograms(gatewayId): possible programs for devices management.
- getAppliancePrograms(gatewayId, deviceId): possible programs from a specific appliance.
- setApplianceProgram(gatewayId, deviceId, programParams[]): will trigger a specific program in a specific appliance.
- modifyApplianceProgram(programId, programParams[]): will modify a stored program.
- deleteApplianceProgram(programId): will delete an appliance program.

APIs will be published as Web Services that will feed content to the different Front Ends. These APIs would give flexibility for the utilities and service providers in order to build adapted and specific intefaces for the different users.

1.2. State items and Interface Messages

1.2.1. CPS

STATE ITEMS

The following table presents the BeyWatch Agent's view of the CPS system in terms of "read only" and "read & act" state items.

Table 1 - BeyWatch Agent's view of the CPS system in terms of its state items

#	state item	kind of state item	S/D	possible values	
1	run level	read only	D	1, 2 or 3	
2	water temperature top of tank (current)	read only	D	values in degrees Celsius to the decimal point	
3	water temperature middle of tank (current)	read only	D	values in degrees Celsius to the decimal point	
4	water temperature bottom of tank (current)	read only	D	values in degrees Celsius to the decimal point	
5	target water temperature for the coldest segment of the tank (goal)	read & act	D	values in degrees Celsius to the decimal point	
6	CPS profile (to be elaborated)	read only	S	practically infinite	
7	power available from PV	read only	D	practically infinite	
8	time since last flip of PV power switch	read only	D	practically infinite	
9	PV power switch	read & act	D	to the house loads	to the grid
10	electrical power available (the human user may toggle-off during the summer months the manual electrical switch at the control panel that sends power to the hot water tank).	read only	D	yes	no
11	internal electrical switch. This is not the manual electrical switch discussed in item #9 above but an internal switch, connected with the electrical resistor within the water tank that is used to directly heat the water tank using house grid electricity.	read only	D	on	off

Table 1 above summarizes the conception the Agent has of the CPS system. Note that the reverse is not applicable: the CPS does not send any messages to the Agent and has conception of it. It simply replies to messages it receives.

For each of the "read-only" state items, there is a single message prefixed with "get" that reads the value of that state item. For the "read & act" state items there are two messages: one prefixed with "get" that reads the value of that state item and one prefixed with "set" that sets the value. It could be argued that since the BeyWatch Agent has toggled the value before, it can "remember" that value and need not query the CPS system. However, this assumption is false since the CPS system is designed to be able to run in three different modes (run levels), two of which involve no communication with the BeyWatch Agent. Under these modes the system can change its state without receiving explicit instructions from BeyWatch Agent. These changes in the state are therefore neither ordered nor directly perceived by the BeyWatch Agent. When, eventually the system reverts back to the normal mode of operation (under the BeyWatch Agent control), the Agent cannot assume that the state of the system is according to the last issued commands and therefore needs a means to interrogate the system of the state of the various state items, including those which are normally under its control (the "actionable" ones).

For the non "actionable" state items (the "read only") ones, the Agent obviously needs a way to query their values since it does not control them. Moreover, one of them (the supply of grid electrical power to the water tank - item #7 in Table 1 above) is under external, manual control.

MESSAGES

Based on Table 1 and the discussion around it, the messages exchanged between the Agent and the CPS system are identified below:

Table 2 - Agent / CPS interface messages

#	Message	Description
1	**getRunLevel**	this message is sent from the Agent to the CPS system. There are no input parameters. The single output parameter returns the run level (one of "0", "1' or "2"). There is no side effect of this message. That is, the CPS system simply replies with the run level and initiates no other action. This is the case for all 'read only" state items so this observation will not be repeated again.
2	**getWaterTemperature**	this message is sent from the Agent to the CPS system. There is a single input parameter to the message: the segment of the tank for which the reading is requested ("top", "middle" or "bottom"). The (single) output parameter is the temperature of the water in the indicated section of the tank in Celsius degrees (given to the decimal point). Alternatively, a different format can be envisaged. The Agent can simply ask for the temperature vector (without indicating segment) and the CPS can respond with the vector of the above three values (or as many values as are available). The difference between the formats is a matter of implementation and does not affect the semantics or the functionality. Similar comments can be made with respect to other messages identified below and in the sections that follow and for the same reason (i.e. that they are implementation details) will not be addressed or further considered on this deliverable.
3	**getTargetWaterTemperature**	sent by the Agent to get the reading of the target water temperature at the coldest section of the tank. This target temperature has been set either by the Agent or by the user through an interface. No input parameters.

		The single output parameter is the goal temperature in degrees Celsius (decimal point precision).
4	**setTargetWaterTemperature**	sent by the Agent to set the target temperature for the coldest section of the tank. The internal control logic of the tank will then try to use that temperature using solar power (if available) or by flipping the internal electric switch (if solar power is reduced). There is obviously an input parameter which is the temperature to be reached (in degrees Celsius with decimal point precision) but there may be another parameter: the "urgency" to reach that target temperature. If set to "fast" the tank will attempt to reach that temperature as soon as possible and when that temperature is lowered, it will again seek to restore it as soon as possible (e.g. even using the internal electric switch). If set to "normal" the tank (on its own) will only rely on available solar power to reach and maintain that temperature. Note that this only affects the tank's logic –for safety reasons it is useful for the Agent to be able to set a target temperature and rely on hot water tank's control electronics to reach it, maintain it, and not exceed it. Note that even if communications with the Agent are disrupted (e.g. the CPS system not working at run level 3), the CPS should still be counted on to reach and maintain the target temperature that was last communicated. In other words, once a target temperature has been communicated it stays in effect for ever or until explicitly superseded by another "*setTargetWaterTemperature*" message that indicates a different temperature.
5	**getCPSProfile**	this message is sent from the Agent to the CPS system. There are no input parameters. The output parameters provide the following information to the Agent: amount of water the tank can hold, relationship of the measured water temperature to the average temperature (e.g. if a value of 104 is returned, this means that the measured water temperature is 104% of the average water temperature - i.e. the temperature is measured at the top of the tank where the water is hotter).
6	**getAvailablePVPower**	this message is sent from the Agent to the CPS. No input parameters. The (single) output parameter is the power (in Watts) that is presently available (generated) at the solar panels.
7	**getTimeSinceLastPVPowerFlip**	this message is sent from the Agent to the CPS. It inquires how much time has elapsed since the PV power switch was last flipped either to the "grid" position (send power to the grid) or to the "water tank" position (use PV power to heat the water tank). There is a single output parameter that returns the time in minutes since the last flip.

8	**getPVPowerSwitch**	this message is sent from the Agent to the CPS. No input parameter. There is a single output parameter: the present state of the switch. It can take two possible values: o "house_loads": PV power is used locally, sent to the house loads (can heat the water tank indirectly) o "grid": PV power is sold to the grid.
9	**setPVPowerSwitch**	this message is sent from the Agent to the CPS. There is a single input parameter: "grid" or "house_loads". No output parameters. Effect: the CPS on receipt of this message will flip an internal switch and will use the available PV power to: o Supply electricity to the house loads if the input parameter is "house_loads" (can heat water in the tank if requested) o connect PV power output to the grid if the input parameter is "grid" The new setting will remain in effect until the explicit sending of another **setPVPowerSwitch** with the opposite parameter or the activation of one of the two lower run levels (operation modes) of the CPS system that may override these instructions according to their internal logic.
10	**getElectricalPower**	this message is sent from the Agent to the CPS. There is no input parameter. A single output parameter ("true" or "false") indicates whether external electrical power is available to heat the water tank or whether the human user has toggled off the hot water tank electrical switch from the control panel.
11	**getInternalElectricalSwitch**	this message is sent from the Agent to the CPS. There is no input parameter. A single output parameter ("on" or "off") indicates whether electricity from the house grid is currently being used to heat the hot water tank.

With respect to the messages described above and the state items of Table 1, kindly note that the BeyWatch Agent has no conception of the Solar Thermal module of the CPS system. There is no explicit state connected with the operation of the Solar Thermal module, nor any action to take with respect to it. Put in other words, there are no decisions to take with respect to this module, Solar Thermal panels are always used to heat the water in the tank and their effect is felt only indirectly through the temperature of the water tank. If further analysis demonstrates that this assumption is not correct, we may need to add a further state item and message in the Agent - CPS interface.

One other important point to note is that there are some interface-related issues to be elaborated with respect to the semantics of the various run levels and the switching between them. These issues affect the logic of the BeyWatch agent and may necessitate additional messages. Consider the following scenario which highlights some points that need a complementary analysis: in short, the semantics and the implications of CPS run level transitions need to be examined more closely and likely in conjunction with the actual communications protocol used to implement the interaction. The visibility of these transitions

affects the Agent logic and the definition of the messages that need to be exchanged. For example these differ if:

- we assume that run level transitions are not visible to the Agent; as in that case the Agent will have to inquire by polling
- we assume that run level transitions become immediately visible to the Agent as part of the functionality of the protocol or platform supporting the communications; in this case there is no need for an explicit **getRunLevel** message
- we assume that run level transitions are somehow communicated to the Agent not by polling but by a callback method (e.g. maybe by a proxy that's handling CPS communications with the Agent).

If, we assume that, as part of the communications protocol or other platform features used to support the interaction (e.g. OSGi platform), the Agent becomes immediately aware of any lapses of communication between the Agent and the CPS system then the Agent can assume that no such lapses have been detected and that the CPS system is still running on level 2 and any previously issued instructions continue to apply and be in effect. This leads to fewer interactions but also to a more statefull communications protocol which has its own disadvantages. Moreover, not all run level transitions are due to communication failures. For instance, run level 0 is described in the Minutes of the WP2 Workshop of 2009-03-03 as a "backup mode [to] support a CPS system failure (i.e. no in-CPS intelligence)".

1.2.2. Washing Machine

STATE ITEMS

The following table presents the BeyWatch Agent's view of the Washing Machine in terms of "read only" or "read & act" state items. Fully similar with the CPS state items description, these are closely related to the physical process or states of the appliance but exclusively intend to define the significant information, offered by the appliance to the benefit of the BeyWatch Agent(or reciprocally).

Those information may be "passive" information, notified or made available to the Agent (useful as input of its own decision algorithms), or "active" information, offering a way to the Agent to modify the nominal process of the appliance, compliantly to the goals of the BeyWatch services.

Table 3 - BeyWatch Agent's view of the Washing Machine in terms of its state items.

#	state item	kind of state item	S/D	possible values
0	System Status	Read only	D	One of either : **[SYSTEM_OFF]**: Absolutely no power. Means general cut off of the electric power **[SYSTEM_ON]*:*** The system is working in the different states mentioned below (see ***General State*** item).
1	Communication Interface Status	Read only	D	One of either : **[ISOLATED]*:*** No communication exists between the washing machine and the BeyWatch agent, due to either a communication failure or simply the connection has not been established There is no communications between the BeyWatch agent and the home appliance. There are two actions to develop : • Trying to re-establish communication with the BeyWatch agent sending repeatedly **Advertise** / ***login*** request (specified timeout and period) • Change the behaviour of the appliance in the following way: if the appliance is executing a program it continues in autonomous mode (not asking for power to start new phase). If the appliance is waiting for the start signal, it will continue waiting and the situation is resolved manually for the user or automatically reestablishing the communications. **[CONNECTED]*:*** There is communication between the washing machine and the BeyWatch agent. Bi-directional communication is established between BeyWatch Agent and the appliance. When the appliance is "isolated", this state item is

#	state item	kind of state item	S/D	possible values
				not instantly available to the Agent. As a solution, the appliance keeps the network statistics data (energy and power consumption) by its own means, and provide these information when it is connected again.
2	Slave State	Read & act	D	One of either : **[SLAVE_ON**, Alternative program, Alternative profile**]** The washing machine has delegated the control of the beginning of the phases of a particular washing program to the BeyWatch Agent and allows monitoring of consumption. **[SLAVE_OFF]** The washing machine controls the execution of the washing program, *asking for permission at the start of each new program and each new phase* and allows monitoring of consumption for the BeyWatch Agent. This requirement is for security reasons. The Agent is responsible of evaluating the current situation and if the working of the washing machine can create a net overloading situation, postpone and interrupt the execution even when the appliance is in the state of SLAVE_OFF
3	General State	Read only	D	One of either : **[WORKING**, Appliance id, program identifier, profile identifier, phase identifier, time since program start (in seconds), phase finish, program finish, electricity consumption, remote command, last failure, date, time, communication interface stats] The washing machine is executing *or programmed for later executing a washing program* (the program to be executed or executing is identified by the program identifier). The communication established during this state, depends on the state of the Slave state item (set by the means of the BeyWatch button), but the working cycle is always divided in a three-steps protocol for each program execution: pre-init protocol, executing, finish (see below for more details). **[NOTWORKING**, CommunicationInterfaceStats**]**

#	state item	kind of state item	S/D	possible values
				The washing machine is neither executing nor programmed for later executing of a washing program **[FAILURE]** The washing machine has detected a failure and the execution of the program must be postponed till the failure has been resolved. The appliance has a failure situation that not allows it to continue with the execution of the program. The appliance will notify this situation to the BeyWatch Agent. The evolution of the situation will be reported periodically to the agent in the ***status*** information. **[INTERRUPTED,** program identifier, next phase identifier, discretionary delay remaining before start of next phase (in seconds)**]** A remote demand has been received or security needs require the interruption of the program execution. The appliance receives a remote command to STOP or the BeyWatch Agent decides to interrupt it because a dangerous situation has been reached. In both situations, the appliance interrupts the execution of the program immediately.
4	Energy Consumption	Read only	D	A continuously incrementing counter (until roll-over)
5	Remote Command	Read & Act		The possible parameters ("Command") are: • ON: Start or Continue a program phase • STOP: Temporal interruption of the program. • OFF: Cancel the program.
6	Time	Read&Act	D	"read" and "act" state item so that the Agent is able to synchronize the Appliance
7	Date	Read&Act	D	"read" and "act" state item so that the Agent is able to synchronize the Appliance

Table 3 above summarizes the conception the Agent has of the Washing Machine system. In any case (whatever the BeyWatch slave status), the working cycle is divided in a three steps protocol for each program execution:

Pre-init protocol

- Notification to the BeyWatch Agent (***program_startup_req)*** that a new program is demanded to start. Sending information about the program, profile and the **status** it is demanded.

If the washing machine is in slave mode ([SLAVE_ON]), the BeyWatch Agent may request information from the appliance like: alternative programs and their profiles.

In order to define a uniform message for the startprogram_req valid in all the situations, a two-phase protocol is defined to decide the alternative program to use. First: the appliance communicates the program selected by the user, indicating the existence of alternative programs. (valid in all situations). Second: The appliance sends (if there are) the alternative programs under demand of the Agent. (valid only in SLAVE_ON)

- Answers from BeyWatch Agent: it could be "***continue"*** that means the appliance can start or "***delay"*** that means the appliance must delay the start of the program till the Agent sends a "***continue"*** message.
  If the washing machine is in slave mode ([SLAVE_ON]), there is one more possible answer: "***change program"*** asks to the appliance to use an alternative program.

Executing protocol

The information shared between the BeyWatch Agent and the appliance is:

- The electric power consumption
- The status of the machine
- The information of finalization of the current phase **(phase_finish)** and the starting of the next phase (**phase_start_req)** of the program. The BeyWatch Agent answers "***continue"*** or it can send a "***delay"*** request postponing the start of the new phase
- The reception of a signal for interruption: in this case the appliance must interrupt the execution changing to the **Interrupted State**
- The existence of a failure: in this case the appliance interrupts the execution changing to the **Failure State**.

Finish protocol

- The appliance sends a "***program_finish"*** message to the BeyWatch Agent and after receiving the acknowledge from the BeyWatch Agent makes the transition to the [**NOTWORKING]** state.

The detailed description of transition between states of the appliance is available as an appendix.

Note the following evolution and simplifications implied in with respect to document "Requirements for Home Devices", released by Fagor, 2009-02-13:

– The responsibility of the Agent for the **security** of the system (taking care for electrical net overload) establishes a protocol of communication with the Agent, based on:
  - The notification of the program to execute (startProgram_req)
  - The notification of the finish of each phase and
  - The asking for permission to start a new phase (startPhase_req)

  This way, the Agent, has all the information about the program and knows continuously the status of the WM (getStatus). The moment for the Agent to take a decision is the start_req message sent by the WM, and the answer can be:

  Delay: Setting the WM in a waiting state till

  Continue: Allowing the WM to execute the washing program

– There is no need for the Washing Machine to explicitly issue a "Hot Water Request". The Agent can poll the Washing Machine for the profiles of the various supported programs and each phase indicates the required water temperature. By polling for the status value the Agent keeps track of the phase of each program and so can be relied to know in advance about any hot water needs.

MESSAGES

Now, it is important to specify the content of the information shared between the Agent and the appliance and associate it to concrete message. In order to show this, the format used is "Remote Calls", but it does not pretend that this will be the way it would be implemented. We are going to leave this definition to a further phase.

So, this communication has been specified in two blocks, depending in which is the actor of the system that takes the initiative (doing the call/sending a message). Parameters are the information sent in the call and Return Value, which is the information sent back from the other actor.

It is not the aim of this document to specify the actual format (method signatures) or protocol that must be used in the final design. Its intention is specify the information that must be interchanged between the Agent and the appliance, leaving the task of design and simplification to a forward stage.

Table 4 - Agent / Washing Machine interface messages

#	Message	Description
1	**getStatus**	This is a call done periodically in a way of ALIVE signal in order to know communications are working well and check any difference between the state of the appliance and the appliance state registered by the Agent. The appliance will **return** information about: ***Status*:** • *General State*: Operation Cycle, Delay Function, Execution, Domestic Control, Power Waiting, Failure State, Situation, Children Protection • *Program Selected* • *BeyWatch On/Off* • *Spin Speed* • *Program phase* • *Program duration/Time remaining* • *Finalisation hour/Init delay*
2	**getAlternativeProgram**	This call is done to negotiate a better option for doing the program, demanding information about alternative programs if they exists. The sent **parameter** is the idProgram which programs is looking for. Information **returned** is a sequence of programs profiles: ***Program Profile*:** • *idProgram* • *phaseNumber* • *Time* needed for the execution of the phase • *maxDelay* after ending phase • *waterTemp*
3	**changeProgram**	this call is done when the Agent decides the convenience of an alternative program instead of the programmed program.

		So it sends (**parameters**): idProgram,idAlternativeProgram The appliance will send (**return**) an acknowledge message when the change is done.
4	**getPowerConsumption**	this call is done to know the current consumption of the appliance. The **returned** value is: *powerConsumption)*
5	**setRemoteCommand**	this call is done to translate to the appliance a remote order sent by the user. The possible parameters ("Command") are: • ON: Start or Continue a program phase • STOP: Temporal interruption of the program. • OFF: Cancel the programm
6	**setDate**	call to synchronize the date between the BeyWatchAgent and the appliance
7	**setTime**	call to synchronize the time between the BeyWatchAgent and the appliance
8	**getAvailablePrograms**	Returns the list of supported programs according to the format identified in state item #5: "available programs".
9	**Pause**	The BeyWatch Agent requests the appliance that it should pause the current washing process. This could be associated with the concept of interruption. For security reasons, the Agent needs to pause the consumption of the washing machine and sends this message. The washing machine must pause.
10	**Continue**	The BeyWatch Agent requests the appliances to continue the washing process.
11	**Change Program**	If and only if the program has not yet started, the configured program can be changed. Once started, this command is rejected since it would affect the overall washing process.
12	**programStartReq**	This call is done when the appliance wants to start the execution of a new program. Program information that is sent (**parameters**): ***Program Profile*** • *idProgram* • *phaseNumber* • *Time* needed for the execution of the phase • *maxDelay* after ending phase • *waterTemp* ***Status****:* • *General State*: Operation Cycle, Delay Function, Execution, Domestic Control, Power Waiting, Failure State, Situation, Children Protection • *Program Selected* • *BeyWatch On/Off* • *Spin Speed*

		• *Program phase* • *Program duration/Time remaining* • *Finalisation hour/Init delay* • *AlternativeProgram* The Agent will reply sending a message for continue, delay or a changeProgram(programId).
13	**phaseStartReq**	this call sends information to the BeyWatch Agent about the new phase the appliance has to execute. These are the sent **parameters**: • *IdProgram* • *phaseNumber* • *maximumDelay* The Agent will answer (**return**) if there *is power available* or not (continue/delay). The Washing machine or dishwasher should also specify the maximum amount of time that the start of the next phase can be delayed. The value could obviously be zero (0) where a pause is not possible.
14	**phaseFinish**	This call sends information about the completion of a phase (**parameters**): *IdProgram, phaseNumber* The Agent will answer (**return**) with an acknowledge signal.
15	**programFinish**	This call sends information about the completion of a phase (**parameters**): *IdProgram* The Agent will answer (**return**) with an acknowledge signal
16	**getFailureInfo**	This call sends information about a failure situation in the appliance (**parameters**) : *codFailure, stateFailure* The Agent will answer (**return**) with an acknowledge signal

1.2.3. Dishwasher

As it is explained above, the Washing Machine and Dish Washer are interchangeable for the purposes of the BeyWatch system's functional specification. Thus § 1.2.2 also describes the Agent - Dish Washer interface.

1.2.4. Refrigerator - Freezer

STATE ITEMS

The following table presents the BeyWatch Agent's view of the Refrigerator in terms of "read only" and "read & act" state items.

Table 5 - BeyWatch Agent's view of the Refrigerator / Freezer in terms of its state items.

#	state item	kind of state item	S/D	possible values
1	fridge temperature	read only	D	the current fridge temperature (in Celsius degrees with precision 0.5 degrees)
2	freezer temperature	read only	D	the current freezer temperature (in Celsius degrees with precision 0.5 degrees)
3	maximum fridge temperature	read only	S	the manufacturer's recommended maximum fridge temperature for preserving food. The Agent is responsible for ensuring that the fridge temperature exceeds the maximum fridge temperature only for limited durations. Configurable limits for the duration of these off-limits periods will be set as part of the Agent software configuration. Optionally, the maximum fridge temperature could be part of the Agent software configuration and not be reported by the manufacturer (if we assume that this value is not manufacturer- or model-specific).
4	maximum freezer temperature	read only	S	the manufacturer's recommended maximum freezer temperature for preserving food. Comments on maximum fridge temperature (state item #3) above apply.
5	minimum fridge temperature	read only	S	the minimum fridge temperature (the Agent cannot set a target fridge temperature below the minimum fridge temperature for energy storing purposes)
6	minimum freezer temperature	read only	S	the minimum freezer temperature (the Agent cannot set a target freezer temperature below the minimum freezer temperature for energy storing purposes)
7	fridge temperature target	read & act	D	the fridge temperature set as a target (in Celsius degrees with precision 0.5 degrees)
8	freezer temperature target	read & act	D	the freezer temperature set as a target (in Celsius degrees with precision 0.5 degrees)

MESSAGES

Based on Table 5, the messages send by the Agent to the Refrigerator can be readily derived, with obvious semantics, and so are not explicitly defined.

1.2.5. Electricity Meter

STATE ITEMS

As seen by the BeyWatch Agent, the Electricity meter system therefore consists of component providing "read only" state items. "Read only" state items provide information related to the Electricty meter system detailed status, which the BeyWatch Agent can read but cannot directly set. "Actionable" or "read & act" are thus discarded as previously discussed, except for setting up the internal parameters of the "listener".

The following table presents the BeyWatch Agent's view of the Electricity meter system in terms of state items and focusing exclusively on data related to the "Tempo" tariff. All the state items denoting an index are coded in numerical characters and expressed in the relevant unit.

Table 6 - BeyWatch Agent's view of the Electricity Meter in terms of its state items

#	State item	Kind of state item	S/D	Possible values
1	Electricity Meter Identifier **ADCO**	Read only	S	12 characters No unit
2	Subscribed tariff **OPTARIF**	Read only	S (D)	4 characters No unit Statically set to "TEMPO" for BeyWatch and denoted by the "BBRx" string; The "x" character can take several values. To get more details, please refer to the appendix
3	Subscribed intensity **ISOUSC**	Read only	S	2 numerical characters unit = Ampere
4	Blue days Low hours meter index (if "Tempo" tariff option chosen) **BBR-HC-JB**	Read only	D	9 numerical characters unit = Wh *All indexes contain the energy consumed during the corresponding tariff period (here it is Blue days Low Hours for instance). A counter is incremented as soon as energy is consumed during the corresponding period and the index is reset once its maximal possible value is reached. To get the energy consumption of the time interval $[t_1,t_2]$ with $t_1<t_2$, it is required to compute Energy(t_2)-Energy(t_1).*
5	Blue days High hours meter index (if "Tempo" tariff option chosen) **BBR-HP-JB**	Read only	D	9 numerical characters unit = Wh
6	White days Low hours meter index (if	Read only	D	9 numerical characters

	"Tempo" tariff option chosen) **BBR-HC-JW**			unit = Wh
7	White days High hours meter index (if "Tempo" tariff option chosen) **BBR-HP-JW**	Read only	D	9 numerical characters unit = Wh
8	Red days Low hours meter index (if "Tempo" tariff option chosen) **BBR-HC-JR**	Read only	D	9 numerical characters unit = Wh
9	Red days High hours meter index (if "Tempo" tariff option chosen) **BBR-HP-JR**	Read only	D	9 numerical characters unit = Wh
10	Current tariff period **PTEC**	Read only	D	4 characters No unit Possible values are: • "HCJB" for blue days low hours • "HCJW" for white days low hours • "HCJR" for red days low hours • "HPJB" for blue days high hours • "HPJW" for white days high hours • "HPJR" for red days low hours
11	Tomorrow's colour **DEMAIN**	Read only	D	4 characters No unit Possible values are: • "BLEU": tomorrow is day blue • "BLAN": tomorrow is day white • "ROUG": tomorrow is day red
12	Instantaneous intensity **IINST**	Read only	D	3 numerical characters unit = Ampere
13	"Over Subscribed Power" Notice **ADPS**	Read only	D	3 numerical characters unit = Ampere Message emitted only in case of an effective

				over-consumption (immediate broadcast in this case)
14	Maximal intensity **IMAX**	Read only	S	3 numerical characters unit = Ampere
15	Apparent power **PAPP**	Read only	D	5 characters unit = VA
16	Low Hours Code **H HP HC**	Read only	D	1 character No unit Possible values are: A, C, D, E or Y corresponding to the metering device programming
17	Meter Status Word: **MOTDETAT**	Read only	D	6 alphanumeric characters No unit Its usage is only reserved to the distributor
18	All meter indexes.	Read only	D	This state item can be requested by the Agent to the "listener", that stores the data into the embedded database (this is not a "label" of the native date frame emitted by the electricity meter). Data frame made up of 6*9=54 numerical characters. The frame contains all the values related to : **BBR-HC-JB, BBR-HP-JB, BBR-HC-JW, BBR-HP-JW, BBR-HC-JR, BBR-HP-JR**. It could be used to compute the whole energy consumption of the house by ignoring the time period. Each component is expressed in Wh.
19	Data collection periodicity of the "listener"	Read & act	D	This item can be requested by the Agent to set up the frequency of the "listener".
20	Time range data collection of the « listener »	Read & act	D	The listener can store data to work out statistics for a given period (3 days, 1 week, 1 month). The time range data collection is set up by the Agent, thanks to this state item.

(the detailed possible values are available as an appendix to the current document)

It has to be noted that item 18, 19 and 20 are more related to the "listener". These items can show the conception the BeyWatch Agent has of the "listener" software. All others items (from 1 to 17) summarize the conception the BeyWatch Agent has of the electricity metering system. The BeyWatch Agent simply receives periodically data, stores or processes them: this storage is direct if the "listener" is included in the Agent, and indirect if the "listener" is outside the Agent (beside the electricity meter). Last, it is crucial to highlight again that the electricity meter can not be requested by any external devices involving the BeyWatch Agent.

MESSAGES : AGENT TO ELECTRICITY METER

This table explicits at a "semantic level" the type of messages that will be circulating between the BeyWatch Agent and the EDF intelligent metering device. On one hand, for each "read only" item, there is a single message prefixed with "get" that ask for the value of the specific item. The answer will obviously contain the relevant and desired value. On the other hand, for each "read & act" item the message will be prefixed with "set" that sets the value of a specific state item.

Table 7 - Agent / Electricity Meter interface messages

#	Message	Description
1	**getElectricityMeterIdentifier**	Sent by the Agent to the "listener" to get the identity (physical address) of electricity meter. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending the answer back to the Agent.
2	**getSubscribedTariff**	Sent by the Agent to the "tariff management" module to obtain the subscribed tariff option of the customer. The "tariff management" module will answer by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending the subscribed tariff back to the Agent.
3	**setSubscribedTariff**	Sent by the Agent to the "tariff management" module, to notify (message transmitted to the BSS) a change of tariff option. This kind of message is only useful in the case of an "alternative" price signal architecture, where tariffs are managed by the BSS (Retailer) and not the electricity Distributor (as it is currently in France).
4	**getSubscribedIntensity**	Sent by the Agent to the "listener" to retrieve the subscribed intensity. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database, or by requesting such information from the BSS) and transmitting the subscribed intensity back to the Agent.
5	**getBlueDaysLowHoursMeterIndex**	Sent by the Agent to the "listener" to get the Blue Days Low Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database) and transmitting the relevant index back to the Agent.

6	**getBlueDaysHighHoursMeterIndex**	Sent by the Agent to the "listener" to get the Blue Days High Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database) and transmitting the relevant index back to the Agent.
7	**getWhiteDaysLowHoursMeterIndex**	Sent by the Agent to the "listener" to get the White Days Low Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database and transmitting the relevant index back to the Agent.
8	**getWhiteDaysHighHoursMeterIndex**	Sent by the Agent to the "listener" to get the Blue Days High Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database and transmitting the relevant index back to the Agent.
9	**getRedDaysLowHoursMeterIndex**	Sent by the Agent to the "listener" to get the Red Days Low Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database and transmitting the relevant index back to the Agent.
10	**getRedDaysHighHoursMeterIndex**	Sent by the Agent to the "listener" to get the Blue Days High Hours Meter Index. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database and transmitting the relevant index back to the Agent.
11	**getCurrentTariffPeriod**	Emitted by the Agent to get the reading of the current tariff period. The "tariff management" module responds by internally reading its memory and sending back the current tariff period to the Agent.
12	**getTomorrowsColor**	Sent by the Agent to the "listener" to obtain tomorrow's color. The "tariff management" module will answer by reading the last emitted data frame (or by extracting related information available in its embedded memory) and transmitting the tomorrow's color back to the Agent.

13	**getInstantaneousIntensity**	Emitted by the Agent to the "listener" to get the instant intensity. The "listener" will respond by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending instant intensity back to the Agent.
14	**getOverSubscribedPower**	Emitted by the Agent in order to get the last record of the over subscribed power. The "listener" answers by transmitting back the relevant data to the Agent.
15	**getMaximalIntensity**	Sent by the Agent to the "listener" to get the reading of the maximal intensity. The "listener" will answer by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending the maximal intensity back to the Agent.
16	**getApparentPower**	Emitted by the Agent to the "listener" to get the value of the apparent power. The "listener" will respond by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending the apparent power back to the Agent.
17	**getLowHoursCode**	Emitted by the Agent to the "listener" to get value of the "low hours code". The "listener" answers by reading the last emitted data frame (or by extracting related information available in its embedded database) and sending the low hours code back to the Agent.
18	**getAllMeterIndexes**	Emitted by the Agent to the "listener" to retrieve all the 8 meter indexes (TEMPO tariff). The "listener" will answer by concatenating all the indexes and by sending them back to the Agent in one data frame.
19	**setDataCollectionFrequency**	Emitted by the Agent to set the data collection frequency of the "listener". Indeed, a large amount of frames (several per second) are emitted by EDF intelligent metering device. This functionality will allow the "listener" to collect a fewer number of incoming frames. The frames that will not be read are discarded.

20	**setTimeRangeDataCollection**	Sent by the Agent to set time range data collection. Indeed, the time range data collection can vary from one day to perhaps a few months in order to provide statistics to users on the corresponding time period. As soon as the period is over, the first data are erased and the process can be repeated (in a "round robin" way).

MESSAGES : ELECTRICITY METER TO AGENT

Table 8 - Electricity Meter notification messages

#	Name	Functionality
1	**sendOverSubscribedPowerNotice**	Emitted by the smart meter to notifiy that an "over subscribed power" event has occurred. This notification is immediately transmitted by the meter to the "listener". After then, the "listener" notifies the Agent core of this event.

The mecanism (polling or other) used by the "listener" to notify – as soon as possible - this event to the Agent core is not described here. This is relevant to the implementation steps.

1.2.6. Watcher

STATE ITEMS

The following table presents the BeyWatch Agent's view of the Watcher in terms of "read only" and "read & act" state items according to the following tabular notation.

Table 9 - BeyWatch Agent's view of a Watcher in terms of its state items.

#	state item	kind of state item	S/D	possible values
1	energy consumed (in Wh)	read only	D	an integer in the range [0, ... max supported number] - see state item #2
2	max supported number (maximum reading that can be reported - before overflowing and folding up)	read only	S	a large integer
3	current instantaneous power (in W)	read only	D	a decimal number in W (precision to be decided but likely to be at most to the tenth of a Watt).
4	On/Off output power	read and act	D	A two possibilities value indicating if the output power of the watcher is feeding (when 'On') or not ('when off') the appliance to be supervised.

Table 9 above summarizes the conception the Agent has of a Watcher. Note that the reverse is not applicable: the Watcher does not send any messages to the Agent and has no conception of it. It simply replies to messages it receives. Also note that implicit in the descriptions provided in Table 9 is that it is the Agent's responsibility to diagnose when a folding up has taken place in the values reported. This simplifies the job of the Watcher.

A point to be decided with regard to Watchers is whether their use will be universal or restricted to otherwise dumb appliances. The approach in this document is to treat all devices (either dumb or smart) as embedding a Watcher. Accordingly the messages and the reading implied in Table 9 above will be considered available on all appliances and will not be repeated on each case.

MESSAGES : AGENT TO WATCHERS

This table defines at a semantic level the messages that will be exchanged between the BeyWatch Agent and the different Watchers. On one hand, the prefix "get" is put before every "read only" item. On the other hand, for each "read & act" item the message will be prefixed with "set" that sets the value of a specific state item.

Table 10 - Agent / Watcher interface messages

#	Message	Description
1	**GetEnergyConsumed**	Sent by the Agent to get the reading of the energy consumed since last reset.
2	**GetMaxSupportedNumber**	Emitted by the Agent to get the maximum supported number. Since Watchers embedded memory is limited, a maximum value has been defined above which power consumption will not be measured anymore.
3	**GetCurrentPower**	Sent by the BeyWatch Agent to get the value of the current power consumption.
4	**GetLog**	Sent by the Agent to get the log of the Watcher. The log will contain several statistical data such as maximum stored value of power, phasing angle, frequency, reactive power.
5	**SetDeviceOn**	Sent by the Agent to the Watcher to switch the connected appliance off.
6	**SetDeviceOff**	Sent by the Agent to the Watcher to turn the connected device off.

2. Electricity Meter Physical Interface

The customer meter data link is achieved by a modulated serial communication support which permanently broadcasts information in the meter internal memory. These electricity meters offer to customers the possibility to be notified in real-time of its own consumption, tariff period information and also to manage load by itself through a numerical information output. These considerations can be achieved by connecting systems to the electricity meter such as:

- a remote display unit
- a load management system

These systems obviously need to be connected to the numerical output of the electricity meter. In the following part, an overview of how consumption pattern can be extract from transmitted data will be provided.

Data packets are periodically transmitted as real-time series data frame over the communication channel (i.e. the customer meter data link). On one hand, each data is preceded by a label that allows data packet identification. On the other hand, each data is inserted into a block called information group. The set of transmitted data relies on the way by which the meter has been programmed. The useless information groups are not emitted.

2.1. Physical feature of the link

The physical output of the customer meter data link is at the bottom of the meter, and marked “I1-I2”. Another output, left to this one and that looks similar, is also located at the bottom of the meter: it is related to the controllable output circuit, marked “C1-C2”, also presented in this document, and not to be confused with the customer meter data link. This can be explained through the below depicted architecture of the electricity meter:

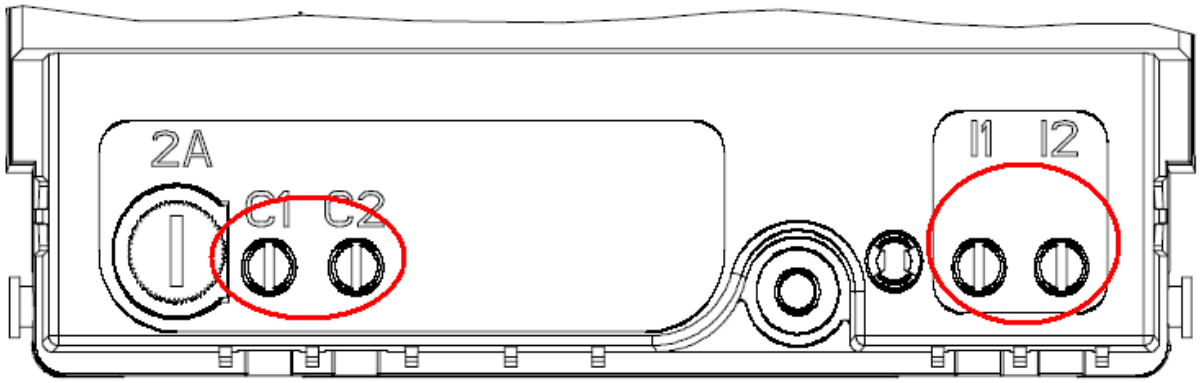

Figure 2 - Smart Meter Retailer interfaces

The system achieve data transmission through a serial communication link (modulated in Amplitude Shift Key modulation) by a carrier wave of 50kHz frequency. The coding logic is negative meaning that :

- a “0” emitted bit does correspond to the presence of the carrier wave during the corresponding time.
- a “1” emitted bit does correspond to the absence of the carrier wave during the corresponding time.

After demodulating, the link becomes again a classical asynchronous link associated with the following features:

- Bandwidth: 1200 bits per second.
- ASCII data-encoding scheme has been chosen for all data.

- 7 bits used for matching an ASCII character.
- 1 bit used for the parity bit.
- Each character is preceded by a start bit which corresponds to “0”.
- Each character is followed by a stop which corresponds to a “1”.

The different ASCII characters are encoded into the following data frame:

7	6	5	4	3	2	1	0	Bit number
128	64	32	16	8	4	2	1	Weight

MSB LSB

When emitting a byte, bits are transmitted one by one and the Least Significant Bit (i.e. L.S.B.) is sent first whereas the Most Significant Bit (i.e. M.S.B.) is transmitted last. The data transmitted data frame can be depicted as follows:

Start bit	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	Parity bit	Stop bit

As mentioned earlier, data are gathered into entities called information groups. As an instance, when emitting the ADCO information group (information group will be explained in 7.6.2.4 part of the document), “LF” character is transmitted first, then the ASCII character “A”, the “D” and so on until the checksum character followed by CR.

Since the link is unidirectional, there is no data packet transmitted from the receiver to the metering device.

2.2. RS 232 conversion module

A conversion module allow to convert the physical link into a serial RS 232 link format

2.2.1. Frame structure emitted by the meter

Data frames are continuously sent one after the others. Between the end of a frame and the beginning of the following one, a delay without any emission must be observed. Its duration ranges between 16.7 ms and 33.4 ms. Similarly, an inter-group delay must also be respected and ranges between 0 and 33.4 ms. In addition, and average inter-byte (computed on the entire data frame) time has been established and does not last beyond 2 ms. Frame length can vary and also relies on the chosen contract.

The frames are split up into three parts:

- The “Start Text” STX (002h) character denotes the beginning of the frame.
- The frame core is made up of several information groups.
- The “End Text” ETX (003h) character denotes the end of the frame.

It is admitted that emission process can be temporally interrupted on the customer meter data link. As an instance, it might occur during a distributor meter data session. In this case:

- The ASCII “EOT” character (004h) is generated before the interruption.
- Emission restarting is achieved at the beginning of the frame by emitting “STX” character.

All data coming from the metering system are delivered by information groups which make a set of coherent data associated with a label that eases to distinguish them from the others. The line-up of information group is as follows:

- A “Line Feed” LF (00Ah) character that denotes the beginning of a group[1].
- The label field whose length ranges from four to eight characters.
- A “SPace” SP (020h) character separating the label field from the data field.
- The data field whose length ranges from one to twelve characters.
- A control field made up of one character containing the “CheckSum” (checksum calculation will be explained later) character.
- A “Carriage Return” CR (00Dh) indicating the end of an information group.

The “CheckSum” is computed on the entire set of characters starting from the label field and ending at the end of the data field, “SP” included. The “CheckSum” calculation process starts with summing the ASCII codes of all the characters. However, the calculation only keeps the six least significant bits from the obtained result (this operation can be defined as a logical conjunction AND between the previously computed sum and 03Fh i.e. 0x00111111 in base-2 number system). At last, 20 (i.e. 0x00100000 in base-2) is added in base-16 (i.e. hexadecimal). The result will always be in printable ASCII characters (sign, number, capital letter) ranging from 20 to 5F in base-16.

2.2.2. Coding of different information groups

The following table gives for each possible information group its name, label, number of characters and the data unit :

Name	Label	Number of characters	Unit
Meter adress	ADCO	12	
Customer tariff option	OPTARIF	4	
Subscribed intensity	ISOUSC	2	A
Meter Index for « Base » tariff option	BASE	9	Wh
Meter Index for « Low 8 hours » tariff option			
Low hours meter index	HCHC	9	Wh
High hours meter index	HCHP	9	Wh
Meter Index for « Peak days » tariff option (EJP)			
Normal Hours meter index	EJPHN	9	Wh
Mobile Peak hours meter index	EJPHPM	9	Wh
Meter Index for « Tempo » tariff option			
Blue days Low hours meter index	BBRHCJB	9	Wh
Blue days High hours meter index	BBRHPJB	9	Wh
White days Low hours meter index	BBRHCJW	9	Wh
White days High hours meter index	BBRHPJW	9	Wh
Red days Low hours meter index	BBRHCJR	9	Wh

[1] Remark: “h” denotes the hexadecimal numerical system.

Red days High hours meter index	BBRHPJR	9	Wh
EJP Start Notice (30 min)	PEJP	2	min
Current Tariff Period	PTEC	4	
Tomorrow's colour	DEMAIN	4	
Instant intensity	IINST	3	A
Over Subscribed Power Notice	ADPS	3	A
Maximal intensity	IMAX	3	A
Apparent power	PAPP	5	VA
Low Hours Code	HHPHC	1	
Meter Status Word	MOTDETAT	6	

2.2.3. Data encoding:

- The meter identification address (Group « ADCO ») is coded on 12 characters.
- The Customer Tariff Option (Group "OPTARIF") is encoded on 4 characters as follows:
  - "BASE": for the BASE tariff option
  - "HC..": for the low hours tariff option
  - "EJP.": for the "EJP" peak days tariff option
  - "BBRx": for the "TEMPO" tariff option

  x is an ASCII printable character that reflects the control programs of the controllable output circuit of the meter. The x byte is defined as following :

7	6	5	4	3	2	1	0

- Bit 7: even parity bit,
- Bit 0 to 6 coding the characters,
  - Bit 6: always set to 0,
  - Bit 5: always set to 1,
  - Bits 4 and 3: circuit program:
    - 01: for program A,
    - 10: for program B,
    - 11: for program C
    - To get more details about A, B and C programs please refer to the part entitled Controllable Circuit Output and related to the "Tempo" Tariff Option.
  - Bits 2 to 0 : no more used and always set to 000
- The "ISOUSC" group is coded on two numerical characters. The "ISOUSC" group denotes the subscribed intensity.
- All the index are coded on nine numerical characters.
- The "PEJP" group (respectively "DEMAIN") is only emitted when the meter is set up in EJP (respectively Tempo) Tariff Option. Its value is fixed and set to "30".
- The "PTEC" group denotes the current tariff period and is coded as follow:

- "TH..": for all hours
- "HC..": for low hours
- "HP..": for high hours
- "HN..": for normal hours
- "PM..": for mobile peak hours
- "HCJB": for blue days low hours
- "HCJW": for white days low hours
- "HCJR": for red days low hours
- "HPJB": for blue days high hours
- "HPJW": for white days high hours
- "HPJR" : for red days high hours

- Tomorrow's color is given in the group of data labeled "DEMAIN". It is only emitted for the "Tempo" tariff option. The data field, encoded on four characters, has one of the following values:
  - "----": unknown tomorrow's color ("-" is the character whose ASCII code is equal to 45 in decimal)
  - "BLEU": tomorrow's color is blue
  - "BLAN": tomorrow's color is white
  - "ROUG" : tomorrow's color is red
- Data coming from "IINST" and "ADPS" information group is equal to the respectively instantaneous intensity and power root mean square value. Both are encoded on numerical string of length three characters. The "IINST" is expressed in ampere whereas "ADPS" is expressed in watt.

  N.B. The "ADPS" information group is only emitted when the consumed power does exceed the subscribed power. Otherwise, "traditional" data frames coming from the meter do not contain this information group.
- The "IMAX" group denotes the maximal intensity and is encoded on three numerical characters.
- Data from "PAPP" information group denotes the value of the instant apparent power expressed in VA (i.e. Volt-Ampere) and rounded up to the closest ten or so VA and encoded on five numerical digits.
- The coding of Low Hours Code (Group "HHPHC ") is coded by the A, C, D, E, Y, 1, 2, 3, 4, 5, 6, 7, 8, 9 or X digit and comes from PLC TCC.
- The coding of the meter state word (Group "MOTDETAT") is worked out from the hexadecimal value (one digit per 4-bits groups) of 3 bytes from the state word. The goal of this value is to give the energy supplier a mean to identify a meter failure or a possible fraud. The normal status of this data is all bits to 0. Its usage is only reserved to the distributor.

2.3. Controllable output circuit:

The way the controllable output circuit is working depends on the customer tariff option.

« Base » Tariff Option:

In this case, the output circuit is not used.

Low 8 hours Tariff option:

In this case, the output circuit is used as the following :

- High hours => circuit open
- Low hours => circuit closed

EJP Tariff Option:

In this case, the output circuit is used as the following :

- Normal Hours => circuit open
- Mobile Peak hours => circuit closed

Tempo Tariff Option

In this case, the output circuit is primarily intended for controlling Electric Domestic Hot Water Tank, according to 3 different strategies among which the customer may choose and select on his meter :

Output Circuit Program	BLEU		BLANC		ROUGE	
	HC	HP	HC	HP	HC	HP
A	X		X		X	
B	X	X	X		X	
C	X	X	X	X	X	

2.3.1. Emission schedule of information groups:

The data frame are emitted one after the others in a continuous way. Each data frame is made up by the set of information groups explained in the document, and useful or significant when at emission time:

- Among the set of information groups related to consumption index, only those corresponding to the customer Tariff Option are sent.
- The EJP Start Notice Group (30 minutes) is only emitted during the notice and related mobile peak period, if the Customer Tariff Option is set to EJP.
- The information group related to the Over Subscribed Power Notice is only emitted when Instant Intensity is exceeding Subscribed Intensity.