Enabling the business-based
Internet of Things and Services

(FP7 257852)

D8.4 Integration of physical world in manufacturing

Published by the ebbits Consortium

Dissemination Level: Public

European Commission
Information Society and Media

Project co-funded by the European Commission within the 7th Framework Programme
Objective ICT-2009.1.3: Internet of Things and Enterprise environments

Document version: 1.0

Submission date: June 2011

Document control page

Document file: D8 4 Integration of physical world in manufacturing v1.0.docx
Document version: 1.0
Document owner: COMAU

Work package: WP8–Physical sensors and Networks
Task: T8.2 – Integration of physical world in manufacturing
Deliverable type: R

Document status: ☒ approved by the document owner for internal review
☒ approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Roberto Checcozzo	2011-01-14	TOC definition
0.2	All	2011-05-11	TOC revision
0.3	Gonzalo Alcaraz, Mauricio Caceres Duran,Claudio Pastrone,Maurizio Spirito,Riccardo Tomasi(ISMB)	2011-06-10	Subsections 4.1/4.3/5.3 contribution
0.4	Marek Paralic [IS]	2011-06-12	Added contributions in chapter 5
0.5	Gonzalo Alcaraz	2011-06-20	Added contributions
0.55	Peeter Kool	2011-06-23	Added contributions in section 4
0.9	Pietro A. Cultrona	2011-06-23	Release for internal review
0.92	Pietro A. Cultrona	2011-06-29	Minor updates according to the reviewers' comments
1.0	Pietro A. Cultrona	2011-06-30	Final version submitted to the European Commission

Internal review history:

Reviewed by	Date	Summary of comments
Michael Jacobsen [TNM]	2011-06-28	Internal review with comments.
Mark Vinkovits [FIT]	2011-06-28	Internal review with some comments

Index:

1. Executive summary

The ebbits project aims to develop architecture, technologies and processes, which allow businesses to semantically integrate the Internet of Things into mainstream enterprise systems and support interoperable real-world, on-line end-to-end business applications. It will provide semantic resolution to the Internet of Things and hence present a new bridge between backend enterprise applications, people, services and the physical world, using information generated by tags, sensors, and other devices and performing actions on the real-world. ebbits opens possibilities to offer a wide range of new business services based on choreography of physical devices, software services, and people that we introduced as Internet of People, Things, and Services (IoPTS).

For this aim, WP8 will develop a secure network and communication infrastructure and semantic integration of physical world objects into the ebbits platform.
The main objectives of this work package will be the research and development of the Network Management subset responsible for the physical communication among the nodes that compose the ebbits cloud , and the research and development of the ebbits Security Framework.
It will also Identify and analyze the available solutions and network technologies adopted in manufacturing environment as well as in consumption use case and it will integrate the Physical World into the ebbits platform in both environments.

In this deliverable it is described the integration of Physical World in manufacturing, through an explanation of the global architecture and the devices involved in the production process overview.
It also describes the connection and the technologies used for the communication among the devices taking into account also the main constrains in wired and wireless connection.
In the last part of the document it is possible to find the description of the integration of the devices and subsystems classified into three different groups: powerful devices, constrained devices and sub-systems.

2. Introduction

The ebbits project main goal is to research and integrate *Internet of Things* (IoT) technologies into the business domain. Business applications developed within ebbits will be able to incorporate physical objects, services and people into mainstream enterprise systems supporting real world as well as online end-to-end interoperability.

The challenging environment containing a massive amount of heterogeneous devices and business applications from the IoT will be managed by using semantic technology that allows automatic processing of information and autonomous collaboration among devices.

ebbits will open new possibilities to offer a wide range of novel business services based on the orchestration of physical devices, software services, and people. The latter concept is introduced in the ebbits project as the *Internet of People, Things, and Services* (IoPTS).

2.1 Purpose, context and scope of this deliverable

The main objective of this document is to describe the physical world in manufacturing, with an overview of the global architecture and the devices involved in the production process. It also describes the connection and the technologies used for the communication among the devices taking into account also the main constrains in wired and wireless connection.

2.2 Deliverable organization

This deliverable is organized in the following chapters:

- Chapter 3 describes the industrial environments and the components and devices used in the manufacturing plant.
- Chapter 4 describes the devices classification distinguishing among powerful devices, constrained devices and sub-systems and the exposure of information coming from the physical world through an adaptation layer (PWAL).
- Chapter 5 includes the description of the integration of the devices classified in the section 4.1.

3. Physical World in manufacturing

3.1 Global architecture

A welding plant is made by several operation stations, in each of them a particular step of the production process is made. A Line Supervisor PLC coordinates each PLC that manages a single station.

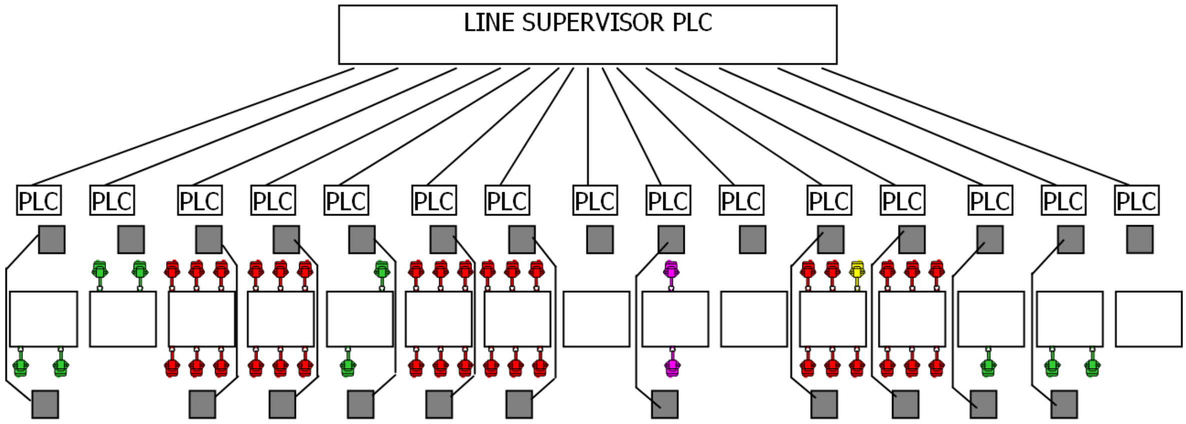

Figure 1: Welding line

In order to complete a particular step of the production process, a single station is composed by several devices like valve packs, proximity and light sensors, safety devices, I/O interfaces between the sensors and actuators and the PLC, and welding robots. As described before a local PLC plays different roles within the production plant that are:

- Managing and coordinating devices in the field.
- Preventing the risk of crashes.
- Managing the safety part of the plant avoiding safety risks for the humans.
- Interacting with the central PLC giving information about the production steps and acquiring important data for the productive process from the neighbors PLCs.

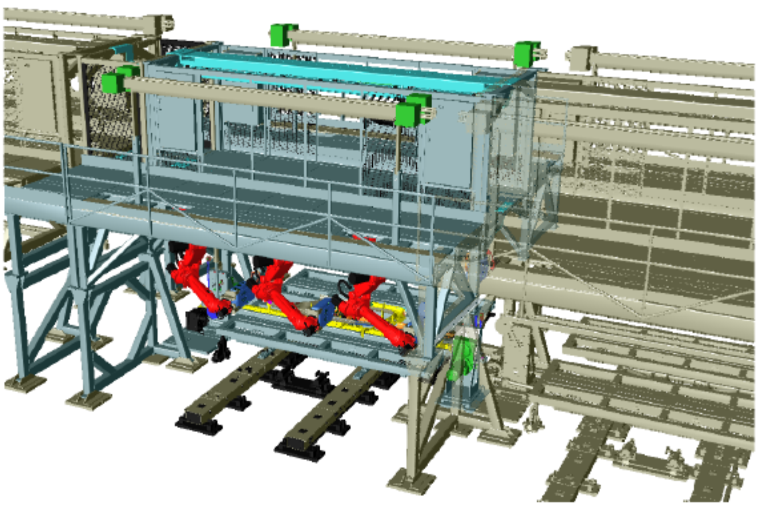

Figure 2: Welding station

3.2 Components in the manufacturing scenario

Deliverable D8.2 "Survey of Physical World in manufacturing and traceability scenarios" surveys components and sensors in the manufacturing scenario. Here we summarize the main devices normally used in a manufacturing plant:

- **The PLC (programmable logic controller):** that is a digital computer intended for the automation of electromechanical processes. Unlike general-purpose computers, the PLC is designed for extended temperature ranges, immunity to electrical noise, and resistance to vibration and impact. It can arrange multiple inputs and output, using expansion cards, and through dedicated field buses it can interface with various field devices.

  The PLC can integrate safety functions or they can be implemented in an external dedicated PLC with a dedicated fieldbus.

Figure 3: PLC Siemens S7-400 especially suitable for data-intensive tasks

- **The robot:** as stated by the Robot Institute of America, the robot is "A reprogrammable, multifunctional manipulator designed to move material, parts, tools, or specialized devices through various programmed motions for the performance of a variety of tasks". The main task of the robots is to help people in the works that would be difficult, unsafe, or boring for a human to do.

  The main applications in the manufacturing plants, as shown in the picture below, are assembling, handling, welding, sealing, painting and inspecting.

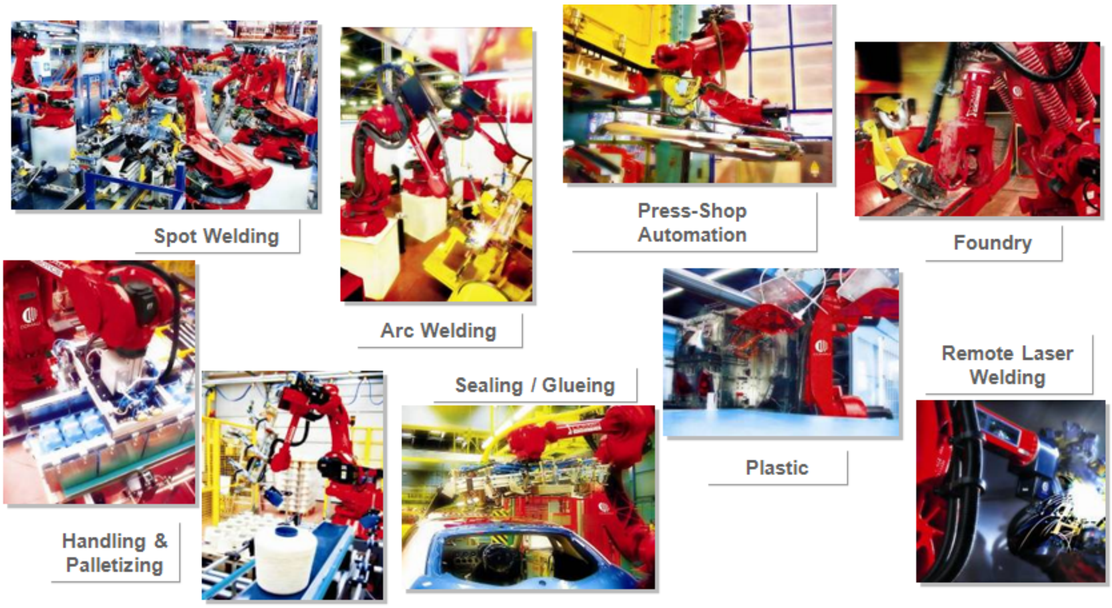

Figure 4: Main robot applications

- **The drive:** inside a manufacturing plant it is usual to find rollers, elevators and turn tables used for the handling of the products during the several steps of the production process. These equipments are moved through servo motors that are not directly connected to the PLCs, but they use electronic amplifiers which continually adjust the output parameters according to the deviation from the expected behaviour. The servo are nodes of the fieldbus network, can integrate safety parts allowing emergency stops according to the European norms.

Figure 5: Lenze Fu 9400 Servo Drive

- **The fieldbus:** some years ago each device was connected to the PLC directly with a cable, all this meant a big quantity of cables coming to the cabinets and running though the plants. For an easy installation and a better standardization the field-bus has born: a cable connecting all the devices and arriving to the PLC that took the place of all the old cables, using only one protocol and just one kind of connection. The mostly used fieldbuses in automotive manufacturing plants are Interbus, Profibus, PROFINET and Devicenet.

- **The valve packs:** are packets of electromechanical valves used typically for compressed air water or sealing material, the first is used for moving lock pins and clamps used to center and hold the parts to be machined, the second is used as coolant in the welding systems.

Figure 6: Festo valve pack

- **PC-based HMI (Human Machine Interface):** computers used for monitoring the industrial process. Each HMI runs a customized program and connects to the Programmable Logic Controller (PLC) via Ethernet TCP/IP and OPC protocols. They convert the real-time signals status proceeding from the PLC into easy-to-understand visual pages depicting the process status. On an average automatic line one HMI per station is typically present.

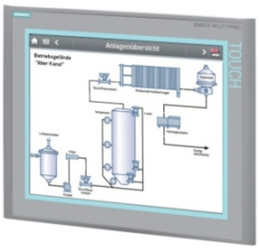

Figure 7: Siemens HMI Touch panel

- **The Scada PC (Supervisory Control And Data Acquisition):** consists of a PC connected to the plant (usually via Ethernet or serial communication interfaces) and executes specific program used in industrial process control applications for centralised monitoring and recording the status of switches, temperatures, information about production. It provides the operator with a user interface to quickly visualize a full real-time overview of the production process. Usually there is one Scada PC per line providing the whole overview of the manufacturing operations.

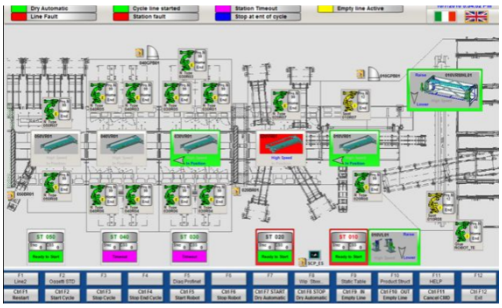

Figure 8: SCADA

- **The Sensors:** during the production cycle it is extremely important for the PLC collect data from the field about the position of the parts worked and all the components that can affect the process. In order to acquire this data several sensors must be connected to the PLC. The most used are:
    - **Inductive**: they are normally used for object detection of ferrous metals. With this kind of sensor the PLC is able to identify if the part into the station is correctly positioned in order to be worked.
    - **Capacitive**: used for detection of any material, even non-conducting, they are rarely used in the "body in white" production, normally they are used to detect liquids and in plastic parts in the assembly.
    - **Photoelectric:** used when it is necessary to check thickness, distance, shape or just presence of a part in a particular position. They are normally used on the grippers of the handling robot.

Figure 9: Balluff Sensors

3.3 Industrial environment

The industrial environment is a hard field that highly stresses the components, normally most of them are powered on 24h per day and are always connected to the network.

Inside a manufacturing plant is usual to find several constraints to face: it is possible to divide them in two main groups depending on the type of connection used:

Wireless networks:

- **Coexistence of networks/ possible interferences:** inside a plant it is usual to find several wireless networks. Generally, they are classified in production plant networks (used for communicate devices related to the production process) and corporate networks (used for internet within offices). In addition, devices normally available from the retailers operate in the same range of frequency, like for example mobile phones (GSM and UMTS), WSNs, Bluetooth devices and Cordless phones. Some considerations should be taken into account in order to avoid possible interferences between these systems (i.e.; spectrum management).
- **Density of devices:** the number of devices involved in the production process and in general present in a manufacturing plant in increasing significantly: this means that it is very important to reduce the bandwidth usage and the possibility of collisions between the packets crossing the network. In order to increase the efficiency it is important to adopt some minor tricks like using switches, dividing the network into sub-networks.
- **EMC pollution:** also called "electro-smog", it is a problem that is focusing the attention on it only recently. The environment is already full of EMF due to the presence of welding guns and motors. Through the increasing number of wireless devices the amount of EMC pollution

increases which makes long cables act as transformers. This phenomenon could create high voltage shocks damaging the devices.

Wired networks:

- **Routing in hostile environments:** not always is possible to use wired connections in particular when there is not enough space for the routing of the cables or when the maintenance of them exceeds the cost of a wireless connection. This can be applied for instance to the welding plant involved in the production of more than a model and with mobile tools where is not possible to use cables to put them in communication.
- **Cables characteristics:** another important aspect to analyse in the connection of different devices in the manufacturing plants is cables: they must be selected in relation to the environment of use.

  As stated in the Specification CEI EN60204-1 and the IEC 332-1, they can have particular characteristics like resistance to oils, to chemical, resistant to high temperature and/or extra flexible, or with a stronger insulation.

  Moreover all the cables connecting to drives or that are used for data exchange must be shielded, like for example cables connecting motors, encoders and proximities.

Figure 10: Shielded cable

Another technique of cabling two conductors for reducing the effect of electromagnetic interference from external sources is twisting them together. This technique is widely used for the network wired connection, in fact the network cables are made by 4 couples of twisted cables.

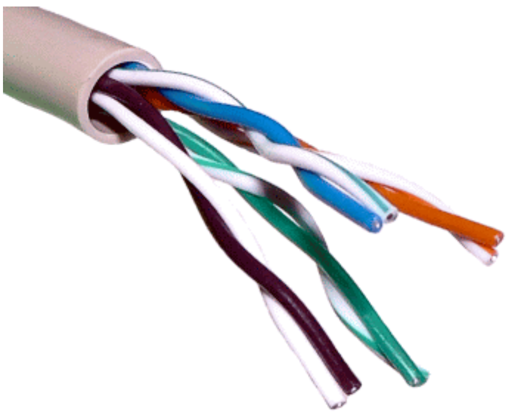

Figure 11: Twisted cable

Performance required: this is a common requirement of wired and wireless connections. It is very important to ensure a high level of performance of the network to allow the devices to get real-time data coming from the field. It is important to consider latency that is a "dead time" that must be reduced to increase the efficiency of the network. It can depend on possible devices between the transmitter and the receiver, like switches and router, or by the transmission medium (air or copper).

3.4 Available solutions and network technologies

Manufacturing environment is extremely wide and even the available solutions and the network technologies offered by the market are endless, the choice must be annotated according to the compatibility offered with the industrial devices.

As already described in the document D5.1.1 "Concepts and technologies in intelligent service structures 1", the available industrial communication standards and messaging formats for the High-Level Communication are plentiful. As a reminder we describe them briefly as following:

3.4.1 Existing Formats for High level Ethernet protocol

Wired and Wireless Sensors and Actuators networks interoperability is one of the main issues: devices produced by different manufacturers must interact together in a common network through standard protocols. The most used in manufacturing based on TCP/IP protocol are Ethernet/IP and PROFINET:

- **Ethernet/IP[1]:** it stands for Ethernet Industrial Protocol (to not be confused with the widely known IP that stands for Internet Protocol) and it is the name given to the **Common Industrial Protocol** (CIP), implemented over standard Ethernet (IEEE 802.3 and the TCP/IP protocol suite). This format was developed in 2001 by Rockwell Automation, managed by ODVA (Open Devicenet Vendors Association).
  It is a media independent, connection-based, object-oriented protocol designed for automation applications. It provides a unified communication architecture including a set of communication services for automation applications and it uses all the seven layers of the OSI reference model. This solution allows the integration of user's applications with the Internet and Ethernet networks. It also defines *devices profiles* and *application objects* that describe behaviours and interfaces enabling an end-to-end communication between devices.

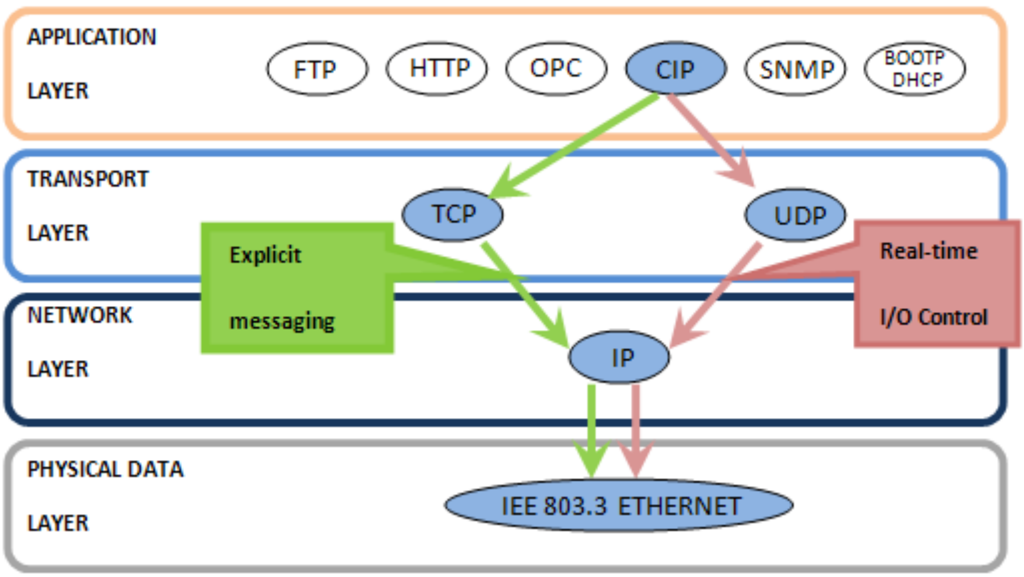

Figure 12: CIP Communication

[1] http://www.anybus.com

- **PROFINET[2]:** it is another open standard developed by Siemens and the PNO (Profibus User Organization). It is based on Ethernet and uses TCP/IP standards and it ensures a very flexible topology with connection in line, bus, star or tree, depending on the physical media. It can be connected to a controller that uses also profibus connection, differing from generic fieldbuses that need a specific proxy to be interfaced, like shown in the picture above.The information on the field devices are contained in GSD (General Station Description) files based on XML.

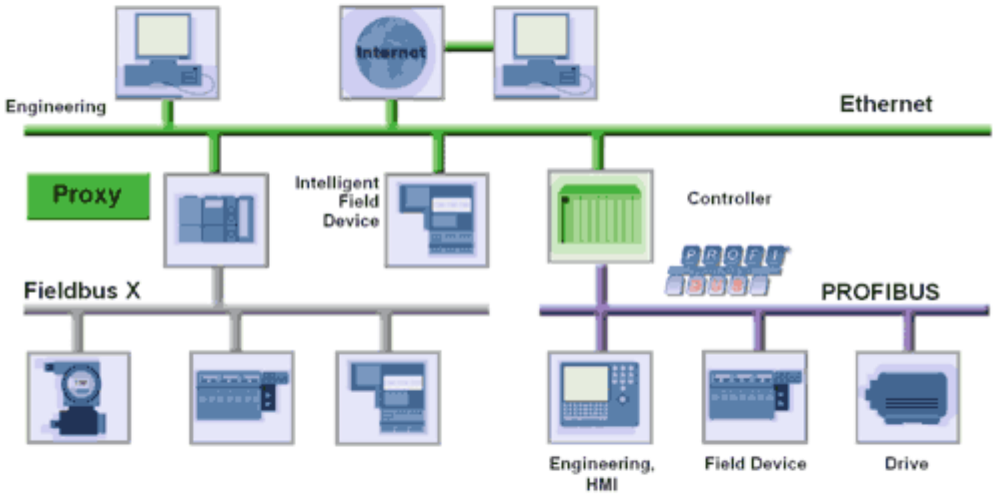

Figure 13: Profibus and PROFINET connection

3.4.2 Messaging for High-Level Communication

OPC: the term OPC stands for **O**LE (Objects Linking and Embedding) for **P**rocess **C**ontrol specific for automation devices. It was developed to ensure the communication of real-time data between control devices coming from different suppliers, in order to provide a common bridge for Windows based software applications and process control hardware.

3.5 Relevant information from manufacturing field

The quantity of information coming from a manufacturing plant varies and can be classified as stated below:

- **Production data:** here it is possible to list all those information coming from the physical machine production plant like parts produced, destructive tests and not destructive tests, scrap parts, parts produced for measurements e their results. Another important data usually collected in a manufacturing plant is the cycle time, this parameter is important to plan the production and for the knowledge of the maximum production per shift.

  In order to improve the manufacturing plant and to reduce the waste it is important to collect information about the stops of the machine: part of PLC Program sends to a central remote server all data about the stops distinguishing non-loads, non-discharges, fault and safety stops, etc.

  It is also possible to include in this section all the information about the status of the production plant collected through for example ultrasound sensors. They are able to detect abnormal vibrations and they can also give a sort of X-ray about the internal structure of the machine in order to prevent possible faults due to the wear.

[2] http://www.siemens.com

- **Energy data:** in order to have a plant with low environmental impacts it is important to collect all the data on the consumption, like the whole power consumption of the machine or simply the power consumption of the welding circuit and the water used by the cooling system.

  Every manufacturing plant uses the compressed air for many features. During the production cycle of the body of a car compressed air is used, for instance, to move the lock-pins and clamps to center and hold the parts during the production process. The electrical consumption for the production of the compressed air impacts on the overall need for the whole plant, for an average of 11%. Of all the energy used during the process of air compression, approximately the 75% of energy is wasted in heat generated (that can be used to heat the rooms and /or produce hot water for sanitary use). In the theme of energy saving, thus compressed air plays an important role, for all the energetic, economic and environmental aspects associated with it.

  In this section it is also possible to include the data about the power consumption of the air conditioning and the lighting, limiting the wastes during non-production times (breaks and night or weekend for example).

  The two groups of information can be merged together, in this way it is possible to understand for example the wear of the machine in connection with the parts produced or the energy consumption in relation with the parts produced.

4. Physical World Adaptation layer

Integration of different devices or sub systems in the physical world to the ebbits platform will require a physical world adaption layer (PWAL) that acts as the glue between the ebbits system and the physical world. One can see this layer as the hardware drivers needed for ebbits in order to communicate with the devices.

The ebbits platform uses virtual device representations to represent the physical world devices in the ebbits system. These virtual devices expose the devices services and also carry the ebbits device specific services that handle metadata, etc.

The typical considerations that are necessary to take into account when deciding on which strategy to use when integrating the physical world systems will include:

- The desired level of integration.
- Which services offered by the physical world devices should be actually exposed
- The type of interaction between the ebbits platform and device/sub-system i.e. a two way communication or one-way communication from device/sub-system to ebbits.
- The addressing schemes adopted in ebbits and the physical world addresses translation.
- The APIs exposed from the physical world devices/sub-systems.

4.1 Devices classification

Within the project, several heterogeneous devices will be integrated into the ebbits platform. Heterogeneity refers to different capabilities, communication technologies, functionalities and complexity of devices within the ebbits physical world. In this context, a device classification is essential to group devices with similar characteristics, purposes or work fields allowing a simpler view of the physical world. Nevertheless, it is worth mentioning that this classification does not affect directly the way the different devices are integrated within the ebbits platform. Details concerning the integration of all devices presented in this section will be described in chapter 5.

Three main categories of devices have been identified to describe the ebbits physical world: *powerful devices*, *constrained devices* and *subsystems*.

4.1.1 Powerful devices

Devices tagged as powerful refer to objects having significant amount of resources in terms of energy, computational power and memory and communication capabilities. These resources would possibly allow hosting in a native way the ebbits middleware.

The main powerful devices examples within a manufacturing plant are PC-based HMIs and SCADA PCs.

4.1.2 Constrained devices

Constrained devices are objects mainly characterized by limitations concerning energy, computational and memory resources. By definition, they could not able to run the ebbits middleware by themselves.

An example of this type of devices is a network of sensors and actuators.

4.1.3 Sub-systems

A sub-system is defined as a group of independent but interconnected elements usually controlled by a specific device which exposes the sub-system functionalities as a whole. The controller device is also able to expose all information contained within the elements connected to it.

An example of sub-system is a group of PLCs coordinated by a PLC supervisor which exposes all the information from the PLCs and the sensors or actuators attached to them.

4.2 Architecture

This section covers the different considerations that should be made when selecting the integration architecture for a specific physical world device.

It should be noted that creating the PWAL alone will not be enough for bringing in a device in to the ebbits platform. In order to create the virtual devices there are several information and functionalities that need to be added for each of the device types:

- *Device Discovery Manager*: Manages device discovery. Needs to be created for each protocol that is supported.
- *Device Service Manager*: Acts as the driver, i.e. encapsulates all device communication with the platform. Needs to be created for each protocol or device type depending on chosen model.
- *Virtual Device Services*: Creates the WS interface for the virtual device that represents the "physical device". Needs to be created for each device type.

As a general approach, all the functionalities that depend on the specific physical world device/sub-system being integrated should be included in the PWAL. In fact, the PWAL will be the "device driver" for the ebbits platform and it should take into account the ebbits platform with its baseline technology LinkSmart in order to ease the creation of the virtual devices.

The following sections will propose an architecture and discuss some issues that need to be addresses for successfully integrate the PWAL with the ebbits platform.

It is worth observing that this is an initial view on the minimum functionalities needed to integrate physical world devices into ebbits platform. The same view will be subject to further specification and evolution through the development of the project.

4.2.1 ebbits device proxy architecture for the PWAL

The ebbits device proxy architecture contains the same basic components in the runtime architecture as the baseline LinkSmart. The two most important managers with regards to the PWAL are the Device Discovery Manager and the Device Service Manager. These device managers are the only parts of the ebbits platform that directly interface with the actual devices and perform communication with them. All applications within the ebbits platform will only communicate with the device proxies and never address the physical devices directly.

Of course there are more managers involved in the platform, for instance *Application Device Manager* etc, but these are generic and not dependent on the PWAL. Descriptions of all managers can be found in the Hydra project deliverable "Final System Architecture" (HydraD315, 2010).

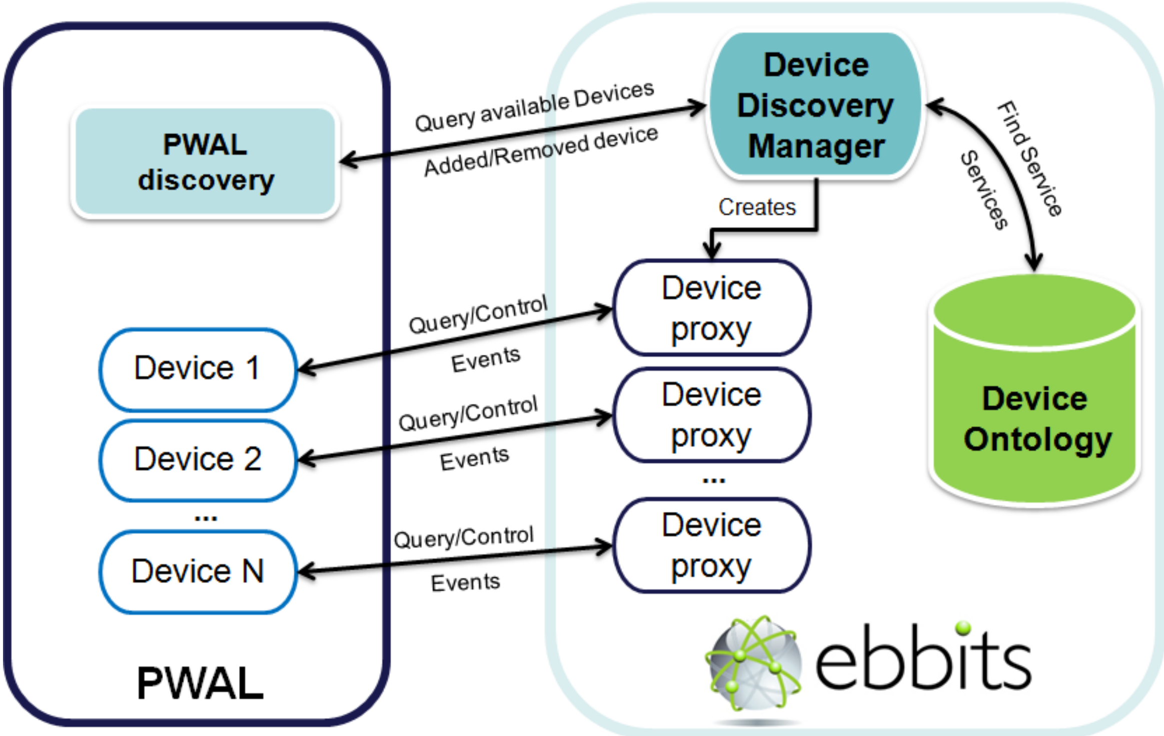

Figure 14: The basic operations in-between PWAL and ebbits platform

The PWAL needs to support a basic set of operations in order to be properly integrated into the ebbits platform (See Figure 14). There are three groups of operations:

- Discovery: provide the means for ebbits platform to find the devices and determine their services. Discovery operations are also responsible for detecting new or disappeared devices in run-time.
- Query/Control: provide the interface for retrieving values, configuring of devices as well as controlling different types of actuators (part of the Device Service Manager).
- Events: provide the means for relaying events into the ebbits platform (part of the Device Service Manager).

A good general rule is to try and integrate in the PWAL most of the available functionalities offered by the physical world devices/sub-systems. This will enable the Device proxies to expose all the "physical world" functionalities and thus make it possible to write applications in the ebbits platform that take advantage of these capabilities.

The following sections will go into more detail of the operation groups/interfaces. It is worth considering that some of these operations may not be performed in the PWAL side but rather in the ebbits virtual devices, depending on the capabilities of physical device and requirements/constraints of the ebbits application.

4.2.2 Discovery

An essential part of the integration is the discovery of devices, i.e. enabling the ebbits platform to detect that there are devices available and discover the capabilities of the same devices. The ebbits platform processes the discovery information to create the proper virtual proxy devices with their corresponding device services.

The discovery information should contain the information needed to be able to communicate with the device. In addition, it should contain enough metadata so that the type of device can be determined. A very simple example of the discovery information could contain just the device address (used for communication) and the type of the device. The discovery information is always represented as an Xml-document.

```
<?xmlversion="1.0"encoding="utf-8"?>
<device>
<address>0xF3E6</address>
<typeOfService>OutdoorTemperature</typeOfService>
</device>
```

Figure 15: Simple discovery information

The interpretation of the discovery information is managed in the ebbits platform at two levels:

- Service Discovery: it is done by the *Ontology Manager* that selects the best matching device based on the discovery information. When a successful match is made the device is "resolved".
- Physical/Communication: the collected information is used by the Device Discovery Manager to set up the Device Service Manager with the correct parameters in order for it to be able to communicate with the device.

An important aspect of the discovery is the handling of IDs and whether the IDs are persistent over time. If there are persistent IDs available it will enable the ebbits platform to uniquely identify the sensors/devices and it will be able to maintain persistent identifiers for the virtual devices as well. The other option that is available when there are no persistent IDs, is that the ebbits platform will create the persistent IDs depending on the ontology information and discovery information. This however will not guarantee that the same device will get the same persistent ID over time.

When adding a new protocol or means of communication, such as OPC, to the ebbits platform it is necessary to create the specific discovery manager that can manage this protocol. This "physical world" dependent capability is possibly integrated within the PWAL.

In the case of a dynamic system where devices can be created and removed in run-time, the Discovery Manager is the main responsible for removing existing and creating new device proxies. The resulting virtual devices will actually represent the state of the "physical world" being integrated in the ebbits platform.

4.2.3 Query/Control (Device Service Manager)

The role of the Device Service Manager is to act as the driver for the device, i.e. exposing the functionality of the device but hiding the complexity of the actual communication. *This functionality is then used to realise the virtual device services*.

The Device Service Manager is created by the Device Discovery Manager when a device has been discovered and resolved by the ontology. The discovery information is sent as a parameter for the newly created Device Service Manager in order for it to be able to set up communication to the device.

In most common cases, all devices belonging to a certain Device Discovery Manager share the same Device Service Manager Class but have individual instances of it. However there are situations where it is necessary to have a singleton Device Service Manager. For instance if the communication with the devices/sensors cannot be done in parallel because of the communication link they will all share the same instance of the Device Service Manager. Note that choosing this solution requires the developer of the Device Service Manager to manage the threading problems that will occur, i.e. making the necessary critical sections etc.

4.2.4 Events

Events are integral part of the ebbits platform and are also part of the interface to the PWAL. The Device Service Manager creates an *event sink* for the device/sensor events. This sink will parse (if necessary) the event received from PWAL and will normally forward the event to the ebbits platform using the ebbits event mechanism. Note that the forwarded event will be enhanced with metadata regarding the device/sensor such as the ontology device type etc.

It should be noted that the Device Service Manager event sink needs to be created by the developer when creating the Device Service Manager. In order to simplify the implementation of the event sink it is preferable if the Device/Sensor can be configured with the event sink address in run time. The event from the PWAL should possibly contain enough information to determine which physical device generated it.

For devices/sensors that do not support eventing, e.g., a switch device that will not inform outside world that its state changed to ON, the only real way of integrating with events is to have a polling solution. In such a case, the Device Service Manager could query the device/sensor state at regular intervals and generate the corresponding ebbits events.

4.2.5 Metadata and Service Description (Virtual Device Services)

Service descriptions are necessary for creating the ebbits platform proxy devices. In the ebbits platform the service descriptions are created using UPnP (Universal Plug and Play) Service Control Protocol Description documents (SCPD). The SCPD is used as a basis for creating the ebbits platform device proxy Web Services as well as the information in the Device Ontology.

```
<?xmlversion="1.0"encoding="utf-8"?>
<scpdxmlns="urn:schemas-upnp-org:service-1-0">
<specVersion>
<major>1</major>
<minor>0</minor>
</specVersion>
<actionList>
<action>
<name>GetStatus</name>
<argumentList>
<argument>
<name>ResultStatus</name>
<direction>out</direction>
<relatedStateVariable>Status</relatedStateVariable>
</argument>
</argumentList>
</action>
<action>
<name>SetTarget</name>
<argumentList>
<argument>
<name>newTargetValue</name>
<direction>in</direction>
<relatedStateVariable>Target</relatedStateVariable>
</argument>
</argumentList>
</action>
</actionList>
<serviceStateTable>
<stateVariablesendEvents="yes">
<name>Status</name>
<dataType>boolean</dataType>
</stateVariable>
<stateVariablesendEvents="no">
<name>Target</name>
<dataType>boolean</dataType>
</stateVariable>
</serviceStateTable>
</scpd>
```

Figure 16: A simple SCPD

In the example SCPD above defines two operations GetStatus and SetTarget. This SCPD would be used to create (possibly, in a dynamic way) the stub for the *virtual device service* (i.e. the Web Service that will represent the device proxy service) for the device/sensor. Additionally this SCPD together with its corresponding discovery information will be stored in the ontology for the service discovery process.

Additionally, the discovery information for the device type is also stored together with the service definition in the device ontology. This information creates the necessary knowledge base to enable resolution of device services based on the provided discovery information.

There are a number of tools developed in the Hydra project for creating the necessary SCPDs and code stubs for Device proxies, Discovery Managers and for entering the ontology information. These tools are described in the hydra deliverable " Final External Developers Workshops Teaching Materials" [HydraD219, 2010].

4.3 Physical World information exposure

As it was discussed in section 4.2, the PWAL is in charge of enabling adaptation of functionalities provided by the so-called *physical world* to the *virtual world* of ebbits. Therefore the PWAL must be able to enrich raw data coming from the physical world with context information linked to parameters of the processes of interest. The resulting information from the physical world (data + meta-data) must then be **exposed** to a virtual representation of the device i.e., the ebbits virtual device, and from there made available to the ebbits platform.

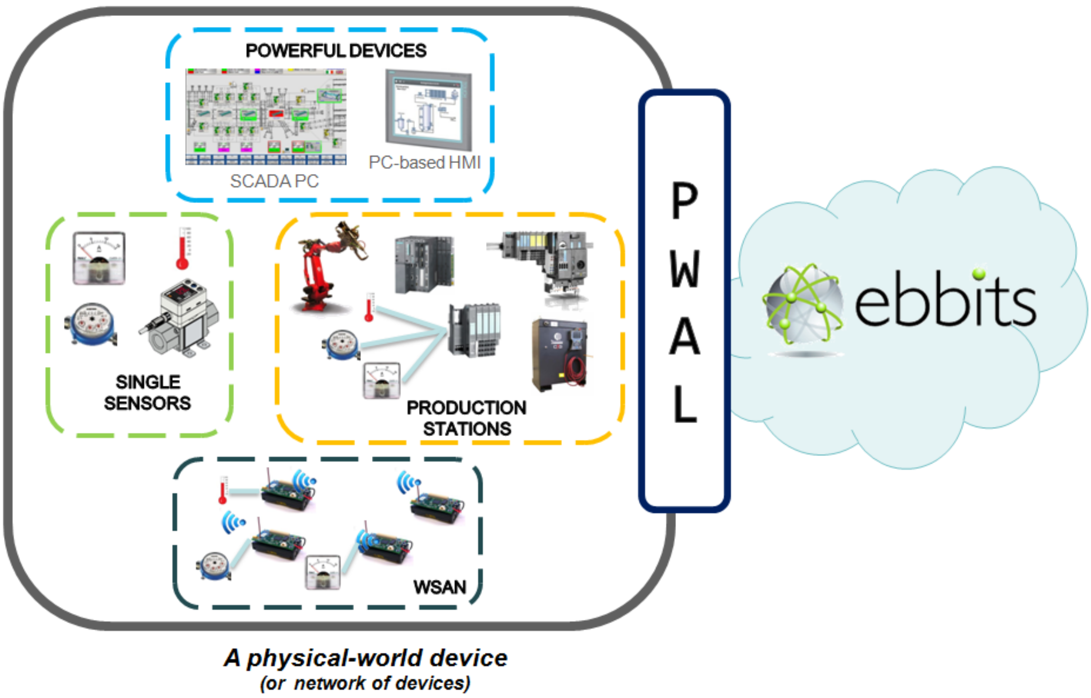

Figure 17: Physical World Adaptation Layer concept

Therefore the physical world information exposure consists in the specification of a set of data structures, request conventions and communication protocols used for the interaction between the PWAL and the ebbits middleware. Since a middleware is, from the practical point of view, a software application (typically written in Java or C#), the most natural way to achieve such exposure is through an Application Programming Interface (API).

Due to the wide variety of physical devices that the PWAL would interact with, the paradigms of the ebbits middleware, and the semantic data modeling devised for the ebbits platform, the API should have the following characteristics:

- Support to generation/consumption of events (to be coordinated with WP7)

- Support to the exposure of services used to model functionalities of devices/systems (e.g. controls, queries, configurations)
- Support to additional features (e.g. tagging) to enrich collected data with meta-information which can be used to:
  1. Match devices with ontology/semantic models (to be coordinated with WP4, WP6)
  2. Allow context management/data fusion (to be coordinated with WP5, WP6)
  3. Support opportunistic communication

A functional diagram of the API and the whole PWAL is presented in Figure 18:

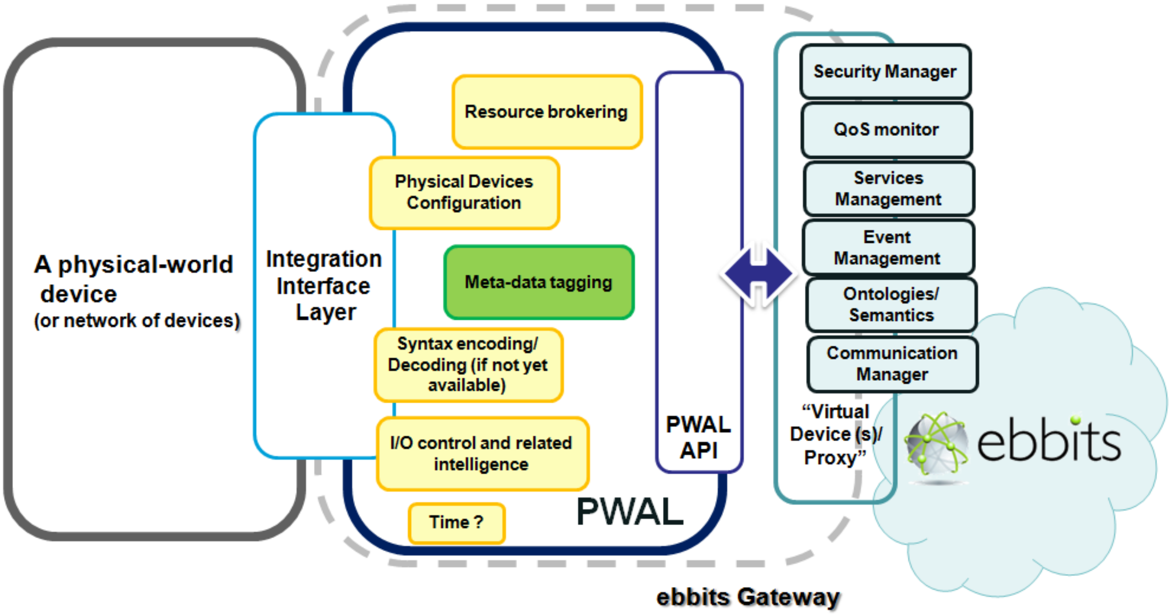

Figure 18: Functional components of the PWAL

In order to specify the PWAL API, two main approaches can be followed:

- Adopt a fixed API able to describe all the possible data structures, functionalities and events that can occur in the physical world: this would ease the integration in ebbits applications/devices, but defining a vast and generic API it is quite challenging and could lead to not optimized solutions.
- Use a dynamic API defined by some description language: this approach would increase the flexibility of the proposed solution and ease the reconfiguration and inclusion of new components, data structures, functionalities and events, but would increase complexity on both PWAL and ebbits devices.

The second approach seems the most appealing, since one of the objectives of the PWAL is in fact the flexibility to add heterogeneous systems and new devices in production phase. Potential candidates for the PWAL API include UPnP and service-oriented technologies like remote procedure calls (RPC), representational state transfer (REST) or simple object access protocol (SOAP).

It is worth mentioning that currently in LinkSmart solutions, actual generation and exposure of events, services, data, etc. is performed by virtual devices (i.e. LinkSmart applications) in an ad-hoc fashion. The PWAL API would provide a more structured support to access them, abstracting the underlying physical issues and specific intelligence from ebbits.

5. Integration of devices and subsystems

In order to enable the participation of different devices in ebbits network, the architecture for device integration should be defined. By device integration we mean the ability to make projection from the business and production processes level to the device level (devices able to impact on physical phenomena), or to be able to take the real world devices as information source for higher-level process control (devices as information extractors about physical phenomena).

Regarding the interaction with a physical device we can distinguish three different layers. First one is the lowest one and the closest to the device – the physical layer. The communication with the device over the physical layer is highly specific – according to the section 3 it could be Ethernet with TCP/IP for PC based HMI or a fieldbus connection (PROFINET) for the welding robot and the PLC connection. Second layer is an application layer, where all ebbits devices should support unified scheme for addressing and messages exchange. This is supported via ebbits "virtual" devices – proxies, which represent the device in the ebbits platform. The last layer is the semantic one, where the meta-data according to the utilised ontologies can be used for the device discovery and interaction. In this section, we examine mainly the first and second layer.

Considering the basic operations with devices – discovery of a device and its service interface(s), description, event notification and consumption, control and presentation – we can identify the following basic components of the integration architecture:

1. *Device Service Manager*. It handles the low level communication with the device. It maps services to the physical device operations. It also produces device specific events that could be enriched by meta-information either for semantic processing or context information processing purposes. Device Service Manager is used only by the Device Managers.
2. *Device Manager.* It receives the service requests and generates the appropriate responses. Moreover, it advertises the device and its service interfaces (e.g. for controls, queries, configurations...) in ebbits platform.
3. *Application Device Manager*. It discovers new or existing semantically described devices, assigns a device type to the device based on the device ontology, returns service interface for the device, creates device proxy during the device virtualization process and manages the *Device Application Catalogue*. Application Device Manager offers a unified view to the devices supported by the ebbits platform and maintains information about devices that are discovered and active in ebbits platform. The main subcomponents are protocol specific discovery managers (e.g. Bluetooth, RFID, ZigBee).

Other important components that enable service orchestration (whereby the basic activity is a device operation), device ontology management and device monitoring, will not analysed in this deliverable since they belong to the semantic level.

Thus, taking into account the software architecture layers of the ebbits middleware (see Figure 19), the PWAL can be seen as a new software layer that sits between the device elements and the physical world elements, and serves as a link between these two. The integration layer is in charge of interacting with the underlying operative system, physical layer, etc.; while the application programming interface exposes the physical world features of interest to ebbits devices.

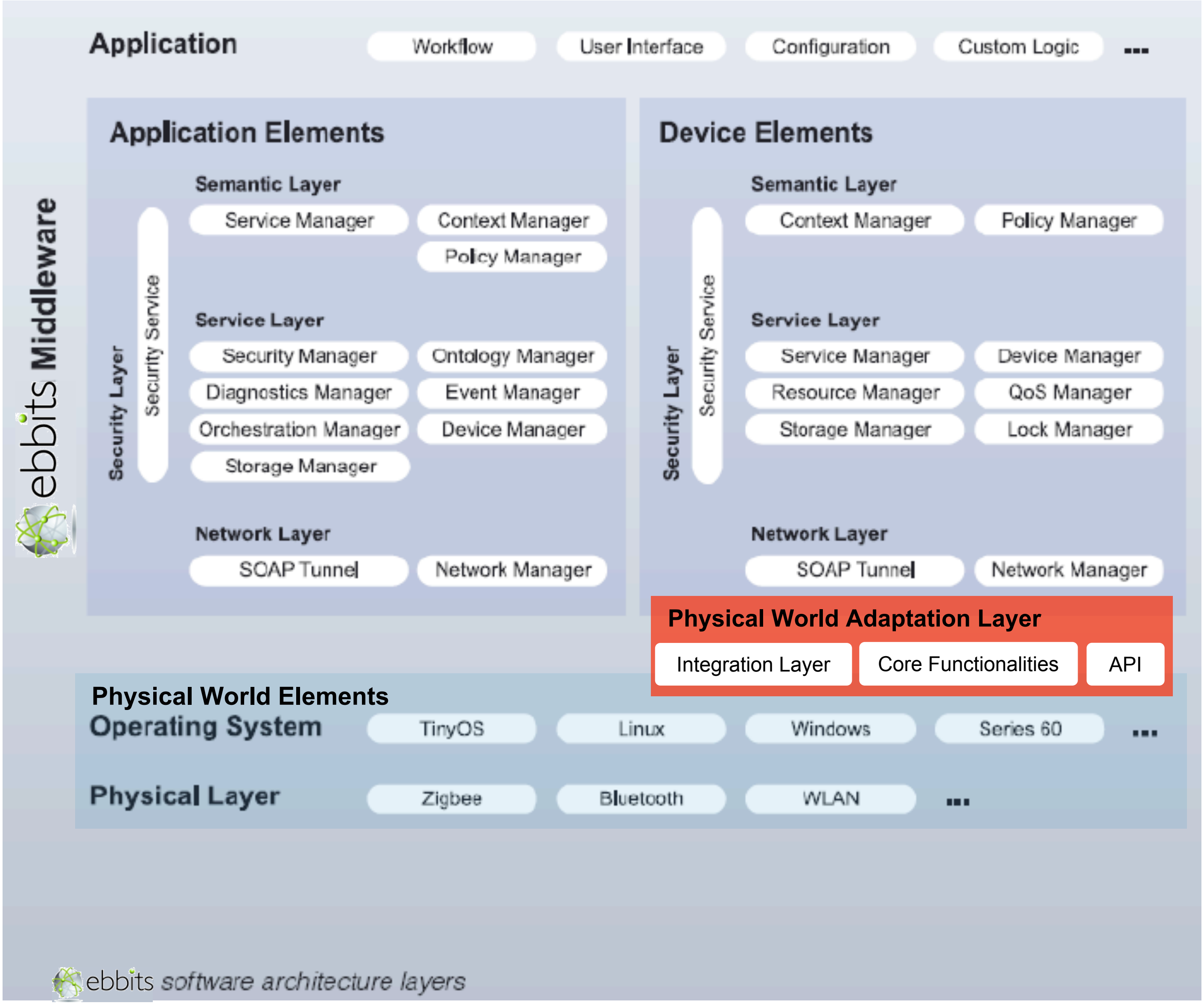

Figure 19: New ebbits software architecture layers

For the integration of devices and subsystems the Device Service Manager and Device Manager functionalities are the most relevant. PWAL could implement part of these functionalities. The following subsections relate to how powerful device, constrained devices and sub-systems could be integrated within the ebbits platform.

5.1 Powerful devices

Powerful devices (see section 4.1) have significant amount of resources in terms of energy, computational power, memory and communication capabilities. Examples from the manufacturing scenario are PC-based HMI (Human Machine Interface), Scada PC (Supervisory Control and Data Acquisition PC), C5G Robot controller or Vision systems.

Powerful devices run own applications and communicate by means of proprietary message-oriented protocols. In order to integrate such device in the ebbits platform, the *Device Service interface* should be defined, annotated and stored in the *Device Application Catalogue*. Some services can be of generic nature (e.g. energy service) and enable the definition of appropriate profiles, which are again part of the Device Application Catalogue. The Device Service Manager of a powerful device then implements defined services interface and supports defined profiles. Moreover the device specific rules for service specific behaviour can be defined in a form of a policy file (e.g. energy policy).

Events for powerful devices may be generated by applying rule assessment derived from associated physical device and performed at local level. Thus each device proxy will have a rule engine intended to run a set of device specific rules. The low-level device-specific events can be enriched by semantic data – for further usage or for context creation.

Supported functionalities are exposed via Device Manager that enables control, configuration and query of the device status. Device Manager communicates with the rest of the ebbits platform in ebbits specific higher level communication protocol, maps requests to device services, generates responses, advertising device descriptions and advertises device services.

5.2 Constrained devices

Constrained devices (see section 4.1) are mainly characterized by limitations concerning energy, computational and memory resources. Examples from the manufacturing scenario are wired/wireless sensors and actuators networks, local PLCs, frequency inverters, custom hardware HMI and welding controllers.

In order to integrate constrained devices in the ebbits platform, again the Device Service interface should be defined, annotated and stored in the Device Application Catalogue. Device Service Manager of a constrained device then implements defined services interface and supports defined profiles.

From the event generation and processing point of view, in distinction to powerful devices, these devices can be e.g. sensors or actuators spread over the ebbits physical environment. For example, it can be a temperature sensor, a Radio Frequency Identification (RFID) reader or some sensor measuring mechanic movement (e.g. pig feeder or roller rotate). Therefore the Device Service Manager has to detect events or poll sensors in order to supply the events in ebbits platform.

If we consider a Wireless Sensor and Actuator Network (e.g. 6LoWPAN, ZigBee) that is used to collect data from the environment, whereby WSN devices are heterogeneous and collect different types of data, the number and identity of each node is not known a priory, each device has its own identity (address) and has some type of meta-data "embedded", then device service manager should also be able to configure and control gateway I/O, support network discovery process, as well as already mentioned to poll sensors and detect events.

Again the supported functionalities are exposed via Device Manager that enable control, configuration and query of the device status. Device Manager communicates with the rest of the ebbits platform in ebbits specific higher level communication protocol, maps requests to device services, generates responses, advertising device descriptions and advertises device services.

The following example should be of help in order to understand how a set of constrained devices in the manufacturing scenario is exposed in the ebbits platform through the PWAL.

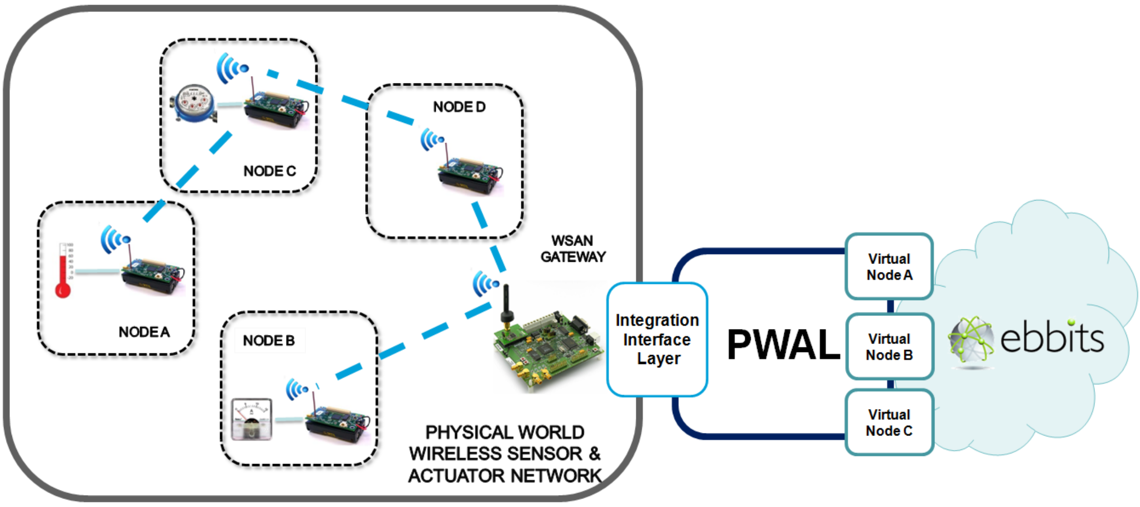

Figure 20: Physical world constrained devices general view integration

The example shown in Figure 20, illustrates a group of wireless sensor nodes coordinated by a WSAN gateway, while the *integration interface layer* of the PWAL is in charge of extracting the information from the WSAN gateway. The information related to the group of devices inside the network is collected within the WSAN gateway internal memory according to a predefined reporting schedule or by polling nodes on request. All relevant devices are connected to the supervisor e.g., using an IEEE 802.15.4 interface in a mesh, multi-hop and delay-tolerant network. Moreover, each one of the wireless nodes may be linked to heterogeneous sensors, actuators and/or possibly other production equipment, or serve simply as relay nodes.

5.3 Sub-systems

Sub-systems, as described in section 4.1.3, are groups of independent but interconnected elements usually controlled by a specific device which exposes the sub-system functionalities as a whole. Within the physical world of the manufacturing scenario, sub-systems regard to collections of PLCs in charge of the automation and control of the production processes.

A PLC network is basically composed by a PLC supervisor (acting as a controller), which coordinates the work of several interconnected PLCs and its production equipment (i.e., welding gun, robot), by using the PROFINET protocol. These PLC networks are normally referred as *production stations*. In addition, diverse sensors (i.e., temperature, water flow, power consumption, etc.) are connected to the PLCs inside production stations, providing updated information of the environment or relevant data from equipments that these PCLs are working with.

All data generated from a production station are transmitted and stored within specific memory addresses inside the PLC supervisor. Afterwards, this information could be accessed through an OPC client working at the *Integration interface layer* between the physical world and the PWAL, as shown in Figure 18. These memory addresses are user-defined and are assigned during the configuration process of the PLC.

The mapping between the address within the memory and the actual information contained should be gathered from the PLC, possibly in an automatic way. In such case, the OPC client would be aware about which information is available and where is located.

The following example should be of help in order to understand how a sub-system from the manufacturing scenario is conformed and how is its relation with the PWAL.

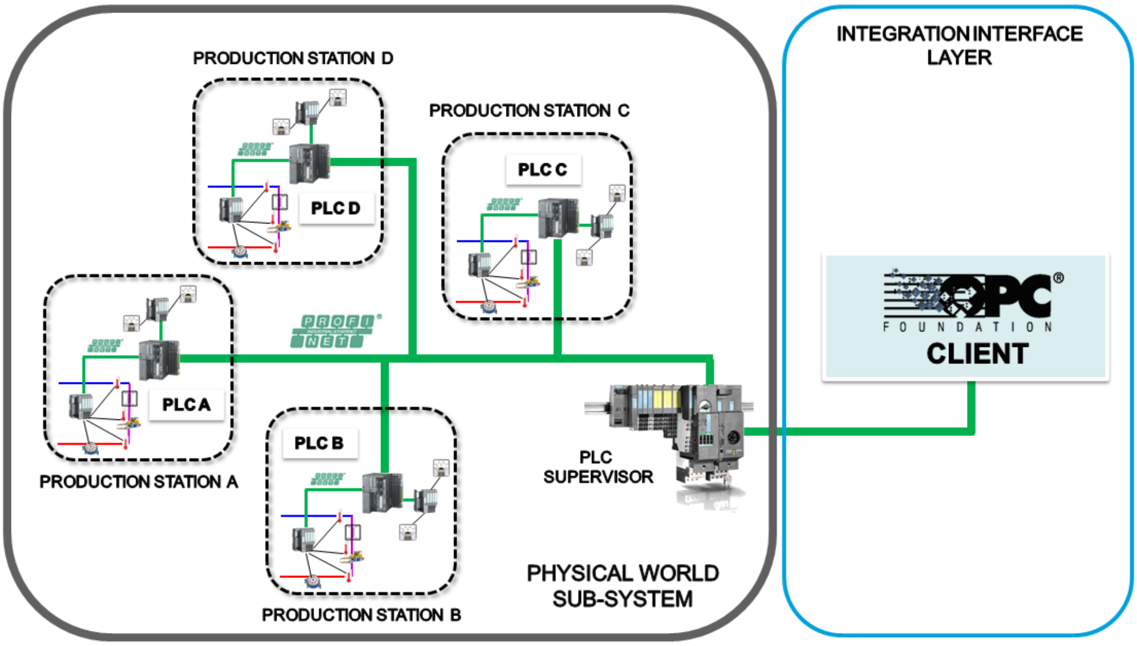

Figure 21: Physical world sub-systems general view integration

The example shown in Figure 21, illustrates a group of production stations coordinated by a PLC supervisor which is connected to an OPC client using a unique bus called PROFINET. One PLC is responsible for every production station. Data related to each production station are allocated within the PLC supervisor memory.

Moreover, within each production station, PLCs are linked to sensors, actuators and other production equipment. As example, a detailed view is shown in Figure 22 regarding some of the possible components included in a production station. This example presents a set of sensors connected to a PLC, in charge of providing information related to temperature, water flow and power consumption of the production equipment attached to the PLC. Essentially, the data related to sensors within Production Station A considered in the example is the following:

- Robot and welding power supply consumption
- Water Out consumption
- Water In/Out temperature
- Welding gun temperature

It is worth mentioning that a production station may contain other devices and equipments as well. They were not included in the figure in order to make the scenario the most simple as possible.

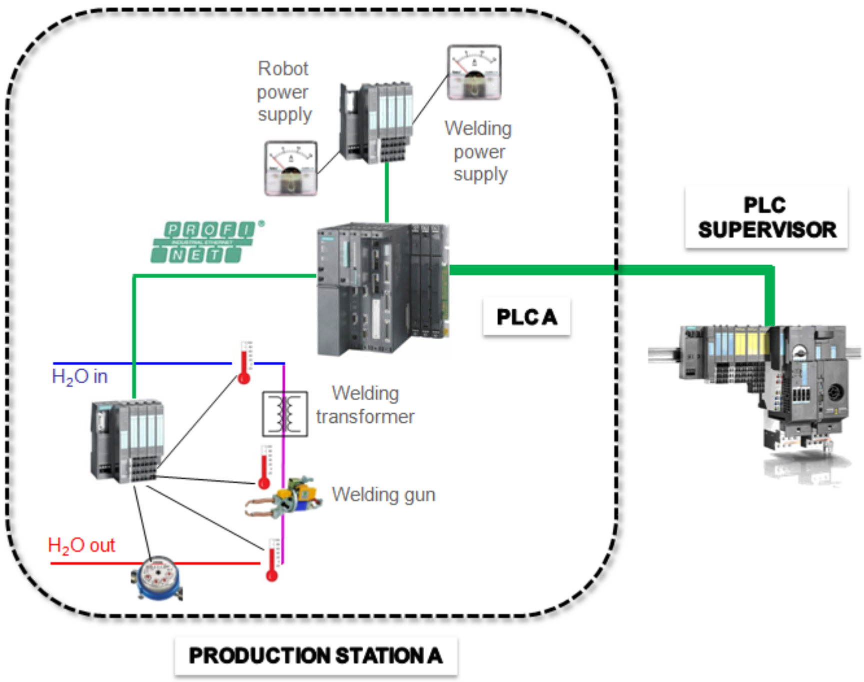

Figure 22: Production station components

In order to better understand how information is organized within the PLC supervisor, Table 1 shows a global view. Each production station provides information to the supervisor which saves the data into its own memory. This memory is accessed by the OPC client which extracts the relevant data and make it available to the PWAL.

For example, the sensor data mentioned before and included in Figure 22 should be allocated in the addresses reserved for *Sensors, Actuators, and Equipment* from the Production Station A within the PLC supervisor memory, as shown in the table below:

Device	**PLC Supervisor**	**Production Station A**		**Production Station B**		**Production Station …**	
Data	Specific information	**PLC A**	Sensors, Actuators and Equipment	**PLC B**	Sensors, Actuators and Equipment	**PLC …**	Sensors, Actuators and Equipment

Table 1 - PLC supervisor memory organization

6. Conclusions

A manufacturing plant is composed by several heterogeneous devices, cooperating in a strongly synchronized way to perform process activities on the products. In an automotive manufacturing plant this means machining of engines, assembly of particulars, welding of body in white, and so on. These devices feature heterogeneous network protocols and media to communicate among them and to the business layer. Today they need a new communication infrastructure to provide a standardized way to retrieve information from the field, integrating them in manufacturing system.
In this document it has been provided a categorization of the devices that participate to this innovative communication infrastructure. They are able to integrate the manufacturing plant from the lower level, composed by sensors and actuators, to the higher level that interfaces with the MES of the plant.

7. List of Figures

8. List of tables

9. References

(HydraD315, 2010) Deliverable, D3.15 "Final System Architecture"

(HydraD219, 2010) Deliverable, D12.9 - Final External Developers Workshops Teaching Materials

http://www.anybus.com/ (Profinet and Ethernet/IP)

http://www.siemens.com (Profinet)