Vertical integration in distributed automation environment

M. Lobashov, A. Bratukhin, T. Sauter, P. Palensky, D. Dietrich

Over past decades industrial IT evolved separately in the field automation and office automation domains. Nowadays increasing complexity of manufacturing requires more advanced control, which can be achieved only by integrating these, formerly separated, areas. This article focuses on two aspects of such integration: communication-related, interconnecting networks of different kinds, and functional, establishing a link between business and manufacturing processes.

Keywords: industrial automation; distributed systems; fieldbuses; Internet; gateways; manufacturing execution systems

Vertikale Integration in verteilten Automatisierungsanwendungen.

Büro- und industrielle Automation entwickelten sich – die letzten Jahrzehnte betrachtend – weitgehend getrennt von einander. Die steigende Komplexität heutiger Fertigungsprozesse legt höher entwickelte Steuerungen nahe, die nur durch eine sinnvolle Verbindung dieser bis dato getrennten Domänen erreicht werden können. Dieser Artikel behandelt zwei wichtige Aspekte dieser Zusammenführung: die auf Kommunikation bezogene Verbindung unterschiedlicher Netzwerke sowie den Lückenschluss zwischen Geschäfts- und Fertigungsprozessen.

Schlüsselwörter: industrielle Automation; verteilte Systeme; Feldbusse; Internet; Gateways; Fertigungssysteme

1. Introduction

During the last 50 years evolution of automation at the manufacturing floor has proceeded from pneumatic and hydraulic-based systems, used for establishing simple control loops between sensors and actuators, to analogue and, subsequently, digital electric circuits. The current state-of-the-art is represented by fieldbus systems – communication networks, designed specifically for the automation environment – and, more recently, applications of Ethernet on the field level (*Sauter, 2005a*). Unlike systems that dominated in the 1970s and 1980s, where control of a manufacturing process was conducted at a central computing device, fieldbuses distribute the computing power over multiple networking nodes. Apart from simplified wiring (elimination of the need to connect each device to the central station), the decentralization provides many other benefits, such as better scalability of the system, ease of reconfiguration when the underlying process changes, more deterministic behavior, increased reliability and, typically, lower costs (*Heffernan, 1997*).

On the other hand, the approach to automation ''above'' the manufacturing floor (at the enterprise level – i.e., in the office) has also changed through the years. The general tendency was to automate and combine different management functionalities at a single point to provide a full-scale solution for the enterprise – from the material resource planning (MRP), where only the estimation of material usage was done, to manufacturing resource planning (MRP II), which already included capacity calculation and analysis of possible changes. Eventually, MRP II became a basis for complex enterprise solution – enterprise resource planning (ERP), which today provides a comprehensive framework on the top level and serves as an integrating component for other business and enterprise systems, managing production (*Sauter, 2005b*).

Factory and office automation evolved independently from each other. Due to increasing complexity of manufacturing processes, modern plant automation systems need to provide higher flexibility of control, as well as adaptability to a frequently changing demand. These requirements cannot be met unless the manufacturing and office ''worlds'' are integrated together.

Systems on the ''lower'' and ''upper'' side of the automation hierarchy are built upon different principles to solve different problems. Thus, their vertical integration, which at first glance boils down to ''just'' interconnection of networks, becomes a non-trivial task to accomplish. In particular, the following two aspects of integration can be distinguished:

- Networking: establishing a transparent flow of information between the manufacturing floor (fieldbus systems) and the office environment.
- Functional: providing logical connection between field and office levels by ''translating'' orders of the enterprise to the manufacturing level.

2. Network connection between field and office levels

Like fieldbus systems established themselves as a de-facto standard in the manufacturing automation environment, Ethernet and the TCP/IP protocol suite were adopted as a base platform in the office world (local and wide area networks). Nowadays, network connection between field and office levels mainly implies interconnection of fieldbus systems on one side and IP-networks on the other.

In order to achieve such interconnection, several problems need to be solved. As data are transported from fieldbuses to LANs/WANs and back, the employed mechanisms must comply with standards used in the respective networks. In particular, it is not desirable to

Lobashov, Maxim, Dipl.-Ing. Dr., Bratukhin, Aleksey, Dipl.-Ing., Vienna University of Technology, Institute of Computer Technology, Gußhausstraße 27/E384, 1040 Vienna, Austria; **Sauter, Thilo, Dipl.-Ing. Dr.,** Austrian Academy of Sciences, Viktor Kaplan-Straße 2, 2700 Wiener Neustadt, Austria; **Palensky, Peter, Dipl.-Ing. Dr., Dietrich, Dietmar, O. Univ.-Prof. Dipl.-Ing. Dr.,** Vienna University of Technology, Institute of Computer Technology, Gußhausstraße 27/E384, 1040 Vienna, Austria (e-mail: lobashov@ict.tuwien.ac.at)

deal with fieldbus protocol details at the office level. Furthermore, data acquisition and processing should use standardized mechanisms in order to be easily integrable with existing enterprise-level tools. What makes vertical integration even more difficult, is the abundance of different fieldbus systems on the market and the diversity of principles laid in their foundation. Different control networks use different communication protocols and ways of representing automation process data.

2.1 Approaches to interconnection

The interconnection between fieldbus systems and IP networks can be done in three different ways. The first one is to ''tunnel'' the fieldbus protocol over IP networks to a destination point, equipped with a fieldbus protocol stack and able to parse and analyze the received data. The second, opposite to the first one, is to tunnel IP together with appropriate upper layer protocols over the fieldbus and to provide the data in a correct format directly at the field device. The third approach, most widely used in vertical integration, is that of a gateway, providing high-level access to the fieldbus and offering IP-based services to exchange data with it.

2.1.1 Tunneling of fieldbus protocols over IP networks

Within this approach, a special tunneling router device is connected to both fieldbus and IP networks. The tunneling router wraps fieldbus protocol data units (PDUs) into IP or, typically, UDP packets and sends them further to the destination over a LAN or WAN. At the destination, packets are unwrapped (extracted from the UDP or IP frame) and processed accordingly – for example, by a management station with integrated fieldbus protocol stack (Fig. 1).

Tunneling of fieldbus protocols over IP networks poses several challenges. The main one is that the tunnel tends to disturb timing characteristics of the fieldbus. This is due to the absence of quality of service (QoS) mechanisms in typical LANs and, especially, WANs: an IPv4 network guarantees neither bandwidth availability, nor maximum transmission latency.

It is worth noting, that tunneling fieldbus protocols over IP networks to a management, visualization or a human-machine interface system cannot be regarded as a universal solution for vertical integration. The format of transmitted frames is fieldbus-specific, thus the target application must implement the respective fieldbus protocol stack, which often reduces the potential range of such applications to those few, supplied by the manufacturer of the fieldbus itself.

2.1.2 Tunneling of IP through fieldbuses

The growing computational resources at fieldbus nodes allow for integrating the TCP/IP protocol stack directly into the node. IP traffic can be then brought down, over a fieldbus, directly to the node. This is achieved by means of an IP tunneling router, which encapsulates IP packets into fieldbus protocol PDUs. Upon arrival at the destination node, IP packets are extracted and passed to the integrated TCP/IP stack.

IP-over-fieldbus tunneling is rather difficult to implement in many industrial automation networks, based on a master/slave principle. Communication protocols on top of IP (primarily, TCP) are inherently asynchronous: communicating parties can issue messages towards each other at any time, without adherence to the master/slave poll-cycle schedule. The problem of TCP/IP tunneling through master/slave systems is discussed in detail in (*Pacheco et al., 2001*).

IP traffic, transmitted through a fieldbus network, also shall not affect timing of the native fieldbus protocol – especially if the latter provides real-time communication. A certain amount of bandwidth has to be allocated for the IP traffic, outside ''normal'' scheduled transmission cycle, used for real-time data.

Within the vertical integration context, tunneling of IP through fieldbuses has its area of application: remote maintenance of nodes, multimedia services and the like. Still, mainly due to increased costs of node hardware, it cannot be considered as the only method of integration.

2.1.3 Application gateway

Unlike tunneling routers, which wrap packets of one protocol into another for transmission, an application level gateway performs real protocol translation. It parses and interprets communication objects of the fieldbus system, extracts actual data out of them, and makes this data available to IP clients within a framework of some IP-based application protocol (Fig. 2).

The protocol on the IP side of the gateway is fully independent from the protocol on its fieldbus side. This opens a possibility to support different fieldbus types within the same installation: all of them can be represented to IP clients in a uniform manner. The client, in fact, does not need to distinguish between different types of control networks at all; it simply retrieves data from the automation process and modifies process parameters.

Further, since the gateway operates on fieldbus data (contrary to processing unparsed ''black-box'' packets in the case of tunneling),

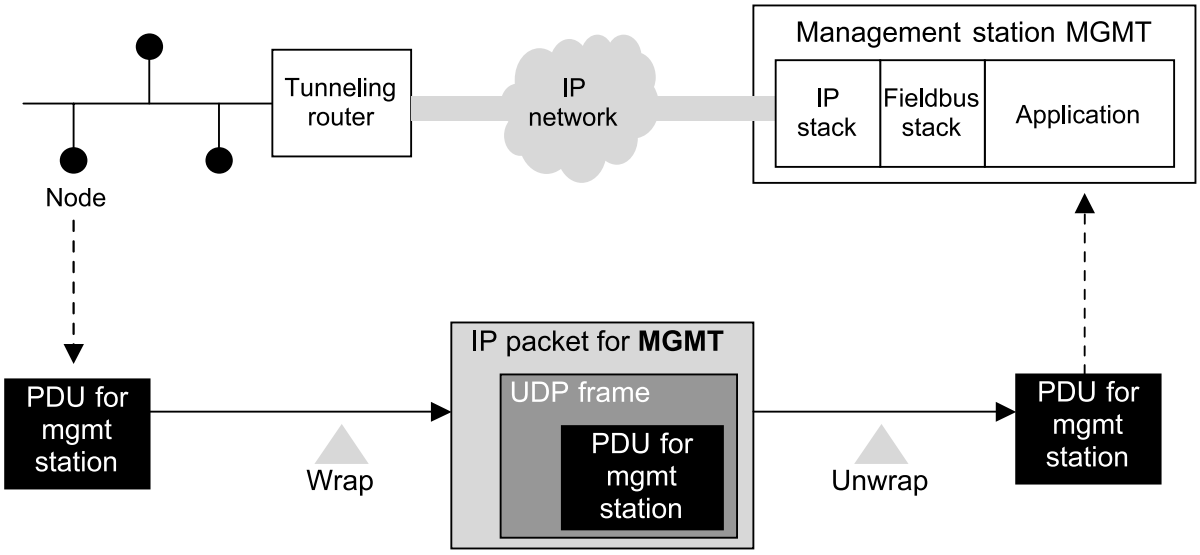

Fig. 1. Tunneling of a fieldbus protocol over IP

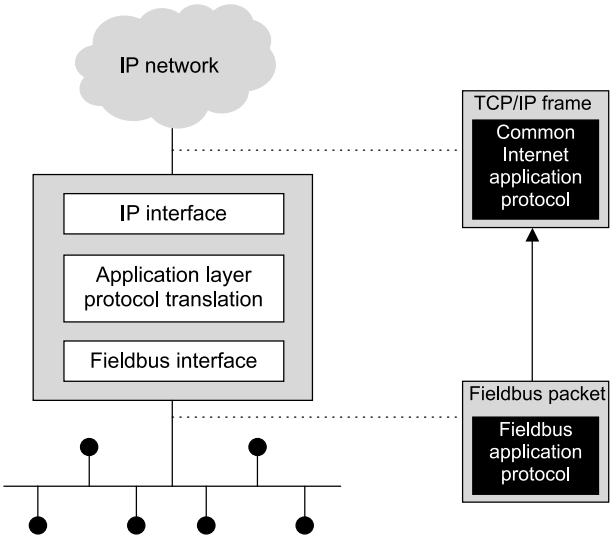

Fig. 2. Application gateway

it can implement a variety of additional services: threshold monitoring and alarming, historical data logging, etc. This makes the application gateway a very functional and elegant solution for vertical integration. The downside is increased implementation complexity and, depending on the protocol at the IP side of the gateway, possible reduction of functionality: a universal gateway, applicable to different fieldbus systems, is not able to equally represent unique features of each of them; it is always ''lossy''.

2.2 Application protocols

The application protocol at the IP side of the gateway is a subject to several requirements. Clearly, the protocol must adequately represent the fieldbus networks and its resources (for example, represent not only data from sensors, but also the installation structure like logical and geographical topology). Any additional services, offered by the gateway, have to be represented as well (*Lobashov, Pratl, Sauter, 2002*).

Another important factor is the protocol's acceptance level. A widely accepted protocol can significantly reduce integration efforts, since many software components and toolkits supporting it will be readily available.

Due to a somewhat sensitive nature of fieldbus data, security often becomes a concern. The data has to be protected from eavesdropping (confidentiality) and in-transit modification (integrity). Authentication services are also necessary to prohibit unauthorized parties from accessing the gateway and fieldbus networks behind it (*Treytl, Sauter, Schwaiger, 2004*).

One other aspect, worth considering, is how well a particular protocol works in the modern Internet infrastructure – for instance, whether it can easily pass through firewalls and proxies, used to isolate Intranet segments from the open Internet.

The following sections give two examples of IP-based application protocols and demonstrate, how particular features of a protocol affect its applicability to vertical integration.

2.2.1 SNMP

The simple network management protocol (SNMP) is an open protocol, designed to control various networking devices (routers, printers, etc.). It can be applied to communicating with fieldbus nodes as well (*Kunes, Sauter, 2001; Kastner, Sauter, 2002*).

SNMP represents data, such as fieldbus or gateway data, as a tree of information objects – the management information base (MIB). Clients can retrieve object values, update them and traverse through the tree. In principle, the tree-like data model is ideal for representing a control network and its hierarchical structure. Other advantages of SNMP are its relatively high efficiency and sophisticated security mechanisms, available within the latest version of the protocol (SNMPv3).

However, applicability of SNMP to representing fieldbus resources is impeded by limitations of the MIB model. SNMP MIBs have to be defined statically; an IP/fieldbus gateway, on the opposite, needs to adapt to an arbitrary installation structure and represent it accordingly. An attempt to work-around these limitations by using non-static table objects causes further problems: operations with tables are non-atomic, and concurrent access to them from different clients cannot be synchronized by means of the protocol.

Finally, the connectionless transport protocol (UDP), used by SNMP, may complicate deployment of the gateway in firewalled IP networks, as firewalls often do not let such traffic pass through.

2.2.2 OPC XML-DA

OPC (OLE for process control, where OLE stands for ''object linking and embedding'') is a set of open standards, developed by the OPC foundation (*OPC Foundation, 2005*). Their goal is to harmonize access to different automation networks by defining a single data representation model, addressing scheme and communication services. One of the latest in the series of OPC standards is the OPC XML-DA (XML data access). It specifies a communication protocol, based on XML and WEB services, which enables IP clients to obtain and update fieldbus data.

OPC represents fieldbus data as items, which are uniquely addressed and can be arranged by server into a tree-like hierarchical structure (''browse space''). Every item can be associated with properties – timestamp, quality, unit and the like. Clients can read and write item values, retrieve properties, subscribe to receiving item values on updates, browse through the tree, and so on.

The transport protocol (TCP) and session layer (HTTP), used by OPC XML-DA, are perfect from the point of ''firewall-friendliness''. TCP is more likely to pass through firewalls than UDP; HTTP can even pass respective application-layer proxies. The choice of WEB services at the application layer ensures interoperability with many existing software packages and development tools. Installations can benefit from standard security mechanisms, offered by HTTP: built-in authentication and external transport layer security (TLS) for encryption and integrity verification.

Unfortunately, the choice of HTTP and XML-based WEB services as base protocols also has a negative impact: the overhead of OPC XML-DA is very high. For example, (*Tian et al., 2003*) mentions a 500 % increase of traffic when using WEB services, compared to transmitting data in ''bare'' HTML. This makes OPC XML-DA inapplicable in IP networks with constrained bandwidth.

3. Functional link: distributed control systems at the office level

The necessity to translate relatively abstract production orders to instructions and information adequate for the manufacturing process emerged in the late 1980s when factory automation became dominant over manual control, and both field and office levels became logically divided, meaning that highly automated systems on both sides talked ''different languages'' and, therefore, were not able to ''understand'' each other.

In order to bridge this gap, the concept of manufacturing execution system (MES) was developed, providing a link between the enterprise and the manufacturing level. Unfortunately, due to the traditional orientation of MES towards the office level, MES failed to provide a close link between highly centralized enterprise

management frameworks and highly distributed automation systems. It thus turned into a data delivery approach rather than a decision making tool.

It became obvious that in order to achieve vertical integration in plant automation, the concept of MES has to be changed. Some of the management and control functions currently performed by the enterprise-level tools have to be delegated to the MES. However, MES functionality presumes direct communication with the field level. With today's complexity of the field level, a centralized MES cannot provide sufficient flexibility to manage a frequently changing distributed manufacturing environment. Therefore, MES functionality has to be distributed at least partly.

Currently, holonic manufacturing systems (HMS) is the best-known concept of distributed plant automation. The paradigm is based on the definition of a holon as a single independent unit which communicates with other components of the system in order to manufacture a product. There are many different implementation approaches of the HMS paradigm, but the PROSA (*VanBrussel et al., 1998*) and MetaMorph architectures (*Maturana, Norrie, 1996*) are probably the most known and have been realized in several industrial applications.

Another concept is PABADIS (plant automation based on distributed systems), which defines an architecture for distributed plant automation using mobile agents (*Klostermeyer, Lüder, 2002*). The following sections describe the main characteristics of the PABADIS and PROSA architectures and compare them, in particular the PABADIS multi-agent system (MAS) and the PROSA holarchy with respect to flexibility and optimization of production.

3.1 PABADIS architecture

In general, a distributed MES consists of three main components: the functional core of the MES and two interfaces to the office and the field levels. In terms of PABADIS, these three components are implemented by the agency (interface to the office level), several co-operative manufacturing units (CMU, interface to the field level) and the multi-agent system (MAS, representing the MES core).

The agency is an interface between the MAS and the upper levels of the control system, mainly the enterprise resource planning (ERP) framework. It is responsible for the interpretation of so-called manufacturing orders (MO) coming from the ''office'' and representing the specification of a complex product. Then, the agency converts the MO into a set of work orders (WO) associated with an actual work piece. One product agent (PA) is created for each WO in order to manage a work piece production.

A co-operative manufacturing unit is a functional instance on the control level which can be associated with an actual machine, a set of machines, or even the whole production. The CMU provides a certain functionality (a robot, a conveyer belt, a database, or even the entire plant) to the agent community.

The multi-agent system (Fig. 3) is the key component for distributing the MES, especially regarding the main functions of the MES: scheduling and dispatching. The distribution of the MES functionality is realized by equipping mobile and non-mobile agents with appropriate mechanisms allowing the agents to perform their tasks independently from any centralized component. Agents negotiate and coordinate their work with each other and other components of the PABADIS system, such as the agency (as an interface to the office level) and CMUs (as field level devices representations). There are essentially two types of agents representing office level and field level.

The product agent (PA) controls the manufacturing process for a particular product and takes care of scheduling, resource allocation as well as reporting to the plant manager. The PA bases its decisions on the WO which describes all tasks to be performed in order of execution. The PA analyzes the WO and finds the best way to manufacture a product. In order to do that, it allocates resources (CMU functions) and migrates to the necessary locations to process its work piece.

A residential agent (RA) resides at a CMU and represents the CMU to the PAs. The RA manages a timetable in order to organize the allocation of the resource it controls. RAs communicate with PAs for scheduling and dispatching purposes.

In addition, there are plant management agents (PMAs) representing the centralized functionality of the MES, such as communication with system operators, like ERP or supervisory control and data acquisition (SCADA), or services like maintenance, report collection, and analysis.

3.2 Holonic manufacturing system

The HMS concept hinges on holons, which are autonomous and cooperative building blocks of a manufacturing system for transforming, transporting, storing, and/or validating information and physical objects. A holon can be part of another holon. By contrast, a PABADIS agent is a single entity and cannot be a part of another agent in the system. HMS also defines the so-called holarchy – a community of holons with certain rules and dependencies. Even if HMS declares autonomy of holons, its holarchy is organized in a hierarchical way, meaning that HMS allows aggregation of holons, where one holon controls another. In contrast to HMS, the PABADIS MAS is a set of completely independent agents.

As said before, HMS is not an architecture in itself, but a set of constraints which the designers of the systems should obey. PROSA follows the rules of HMS on the very basic level, meaning that the architecture defined in PROSA uses the basic holons and introduces specific holons only when it is impossible or not efficient to implement the functionality using just the basic holons, making the architecture generic. The basic PROSA holons are order holons, product holons, and resource holons (Fig. 4).

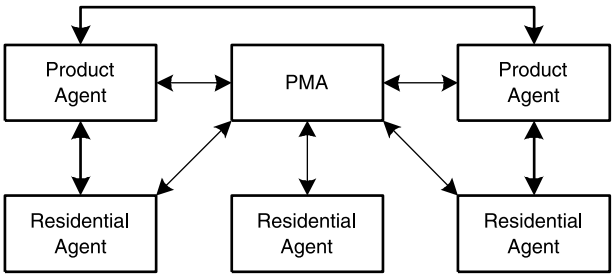

Fig. 3. PABADIS MAS

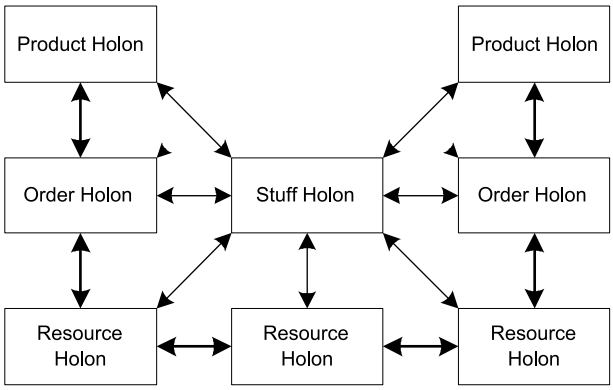

Fig. 4. PROSA architecture

3.2.1 Resource holon

Resources in HMS are represented by resource holons, which are responsible for machine level representation. A resource holon consists of a physical part and an information processing part that controls the resource. It holds the methods to allocate the production resources, as well as the knowledge and procedures to organize, use, and control these resources to drive production (*Bongaerts, 1998*). In terms of PABADIS, a resource holon is a CMU which consists of a logical part responsible for the machine representation to the agent community (RA) and a physical, processing part (function) performing some functionality. PABADIS distinguishes the standard (logical) part of the control level which can be used for each CMU from the ''machine''-specific part which has to be implemented by the customer of the system. This makes the system more flexible and encapsulates the MAS from the control level. That makes adding of new functionality easier, because the (industrial) user has to add a specific plant-dependent resource to the system and the generic RA is responsible for incorporating the resource into the system.

Additionally, resource holons are not just providing resources, but also managing the whole production facilities. This means that they are able to communicate in order to optimize the use of machines. PABADIS does not allow RAs to communicate with each other. This is done in order to make the system product-oriented and not machine-oriented, which is maybe the biggest difference to the HMS concept. The product and its manufacturing are a main goal of the system in PABADIS. HMS is more focused on machine utilization, where an order holon is reduced to a set of parameters of a product that has to be produced.

3.2.2 Order holon

The customer side in HMS is represented by the order holon, which is an analog of the product agent in PABADIS, but differs significantly in functionality. On the one hand, a product agent manages the whole production of a single work piece based on a work order which gives a full specification of the production activities but does not assign actual machines; instead it describes the function which has to be used. This principle gives the product agent freedom to decide what machine to use and a possibility to change the machine during execution. On the other hand, an order holon is more or less the customer request defining the requirements to the product. This has an advantage in mass production, because an order holon is not attached to a single work piece. Note that an order holon does neither scheduling nor resource allocation by itself, because it simply does not have knowledge about the physical layout of the plant and the product specification.

3.2.3 Product holon

Order holons representing the work pieces do not have knowledge about the technological properties of the plant. They have no information on how to manufacture a product. The order holon contacts the respective product holon in order to get information such as ''how to produce this product''.

The combination of the ''blind'' order holon and the layout-centric product holon makes HMS inflexible to changes in the production facility and lacks integration between the office level (order holon) and the field level (product holon). The PABADIS product agent, representing the office level, operates with the abstract term ''functionality''. It does not require knowledge of the physical layout of the field level (rather complicated and changeable). This level of abstraction allows PABADIS to provide vertical integration from the office down to the field. Additionally, it gives, compared to HMS, more flexibility, which is a driving force of vertical integration.

3.2.4 Stuff holon

A stuff holon matches the definition of the PMA in PABADIS. The main purpose of the stuff holon is to give the other components the overall view of the system needed for functions which are centralized by nature. Contrary to PMAs, stuff holons bring centralization into the MES system. PMAs provide functionalities centralized by nature in parallel with distributed ones. By contrast, stuff holons are centralized parts in the distributed system. This results in a lack of flexibility compared to PABADIS. For instance, one of the main MES functions is scheduling. On one hand, scheduling in PABADIS is distributed via product agents. On the other hand, PROSA uses a centralized unit called scheduler stuff holon, which provides a schedule for the whole plant. The first approach makes the systems more flexible, but the second one provides a better solution. For a customer of an MES, it is a basic yet difficult decision to find a balance between flexibility of the automation system and the level of optimization it provides. There is a direct dependency between the size of the system and the required level of optimization (*Hebert, 2003*). The bigger the system, the more flexibility it requires.

4. Conclusions

From the networking point of view, vertical integration between the manufacturing and office level networks can be best achieved by means of a gateway, which translates between two (or more) protocols and offers a unified representation of various fieldbus systems and their resources. The choice of the IP-side application protocol can affect performance and functionality of the interconnection and, thus, is a subject to special considerations, depending on the particular project.

On the functional side, MES is the link between the (centralized) enterprise and (distributed) field levels, even though it usually follows a mostly centralized approach. Both the PABADIS and HMS concepts provide solutions for distributed MES, which fulfill the requirements of vertical integration. Due to its more distributed architecture, PABADIS is better suitable for large-scale installations providing higher flexibility, while HMS guarantees better optimization, because its paradigm is more focused on the centralized office level. This finding is however not a final conclusion, for research in the area continues, and both concepts are evolving.

References

Bongaerts, L. (1998): Integration of scheduling and control in holonic manufacturing systems. Ph.D. Thesis, PMA/K.U.Leuven, Chapter 3.

Hebert, D. (2003): Centralised vs. distributed: is bigger better? Control Magazine, April 2003.

Heffernan, D. (1997): A technical overview of fieldbus developments from origins to present day standards (keynote paper). Proc. of the 1st Annual Fieldbus Conf. (ETCI), 1997.

Kastner, W., Sauter, T. (2002): Network management of fieldbus systems with limited devices: a case study for EIB and Palm OS. Proc. of the 2002 IEEE Int. Symposium on Industrial Electronics (ISIE'2002): 129–134.

Klostermeyer, A., Lüder, A. (2002): A re-configurable multi-agent based architecture to enhance the flexibility of job control in single piece production systems in turbulent environments. Proc. of the 2002 Conf. on Embedded Systems in Mechatronics, (MSY'02): 131–138.

Kunes, M., Sauter, T. (2001): Fieldbus-Internet connectivity: the SNMP approach. IEEE Transactions on Industrial Electronics, Vol. 48, No. 6, 2001: 1248–1256.

Lobashov, M., Pratl, G., Sauter, T. (2002): Applicability of Internet protocols for fieldbus access. Proc. of the 4th IEEE Int. Workshop on Factory Communication Systems (WFCS'2002): 205–213.

Maturana, F., Norrie, D. (1996): Multi-agent mediator architecture for distributed manufacturing. Journal of Intelligent Manufacturing, No. 7, 1996: 257–270.

OPC Foundation (2005), http://www.opcfoundation.org

Pacheco, F., Tovar, E., Kalogeras, A., Pereira, N. (2001): Supporting Internet protocols in master-slave fieldbus systems. Proc. of the 4th IFAC Int. Conf. on Fieldbus Systems and their Applications (FeT'2001): 260–266.

Sauter, T. (2005a): Fieldbus systems: history and evolution. In: R. Zurawski (ed.): The industrial communication technology handbook. CRC Press 7-1.

Sauter, T. (2005b): Integration aspects in automation – a technology survey. Proc. of the 10th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA'2005), Vol. 2: 255–263.

Tian, M., Voigt, T., Naumowicz, T., Ritter, H., Schiller, J. (2003): Performance impact of web services on Internet servers. Proc. of the 2003 Int. Conf. on Parallel and Distributed Computing and Systems (PDCS'2003).

Treytl, A., Sauter, T., Schwaiger, C. (2004): Security measures for industrial fieldbus systems – state-of-the-art and solutions for IP-based approaches. Proc. of the 2004 IEEE Int. Workshop on Factory Communication Systems (WFCS'2004): 201–209.

VanBrussel, H., Wyns, J., Valckenaers, P., Bongaerts, L., Peeters, P. (1998): Reference architecture for holonic manufacturing systems: PROSA. Computers in Industry, No. 37, 1998: 255–274.