Abstract: Design patterns are an appropriate means to code solution knowledge within different areas of science and practice. They cover a description of the problem with the problem context, a description of a possible solution to the problem, and ancillary conditions of this solution. Initially design patterns were invented in the building architecture sciences but were quickly applied to information science and other disciplines. This paper deals with the possible application of design patterns within industrial control. It describes how design patterns can be used by the example of design patterns for industrial field control systems. Therefore, initially the paper presents requirements for field control systems, maps the design pattern approach to them, and describes three basic design patterns for distributed field control systems.

1 Introduction

Control applications have been applied since a few tens of centuries. From simple mechanical controllers in water systems and clocks to mechanical controllers for steam engines or completely hard-wired relay control systems up to the current programmable logic controllers, different technologies for control systems have been developed and applied. In many cases these systems have been based on recurring structures that can be observed as fundamental design patterns for control systems. These design patterns can be used as design guidance in the case of the design of a new control architecture or at least in the case of the design of a new control application.
Design patterns are a currently used widely design aid emerging from building architecture but made more widespread by information sciences. The term design pattern first appeared in the initial work of Alexander in 1979 [1]. Alexander worked as an architect for building design and construction. He recognized that building design always follows the same basic rules but was adapted to the real projects following cultural and geographic implementation specialties. Hence, in his basic work he describes some basic design patterns for building design.

With the emergence of the programming paradigm of object orientation and the software engineering processes and technologies based on it, the idea of design patterns as application-independent basic design principles has been adopted to support engineering process efficiency and quality. The main research activity initiating this trend was the work of Gamma et al. [2].

The main intention behind the application of design patterns in computer science was to provide means to describe and reuse known problem solutions with proven quality and usability and enabling the description of structures behind the original problem solution and dependencies among system components and behaviors. Since 1995 several design patterns have been developed addressing general and special problems [3, 4].

Within the last few years the use of the design pattern concept has been integrated into several scientific disciplines, including the design of control applications. Initially, it was integrated within the design processes of control application software. But it was quickly extended to the engineering of complete control systems including control software, control hardware, and the controlled plant itself. In this field valuable progress has been made. Overviews on the attained results are given in [5, 6, 7].

The application of design patterns was supported by the increasing trend toward the application of mechatronical units within manufacturing system design. Mechatronical units are seen as manufacturing system components comprised of hardware, electrical systems, and electronics including control software designed with the aim of providing special functionality to the overall system [8]. With respect to the application of design patterns within control system engineering, mechatronical units enable the combination of controlled processes with its control device and its controlling software structures. In this way, completely new engineering processes are enabled exploiting libraries of mechatronical units and, hence, libraries of control application building blocks [9].

The description and application of design patterns in control design is based on the principle of describing a solution to a problem within a special context. Therefore, a design pattern includes the description of the basic problem, the context of the problem (i.e., the field where the problem has occurred), drivers important for the solution, the solution of the problem itself, and the background of the origin, and, finally, the possible application fields of the solution. The descriptions within this paper will follow this structure. The particular structure used to document
The design pattern – as that suggested previously – constitutes what is referred-to as a pattern schema. Many pattern schemata have been proposed so far for many concrete domains including control systems [10, 11].

2 Requirements for Field Control Systems

The design of field control systems is driven by a set of requirements emerging from technology and economics. Generally, in the control architecture the Field Control Level aims to control individual manufacturing processes, i.e., to control the physical execution of manufacturing processes and, therefore, to drive them via actuators and measure them via sensors. Hence, the core individual entities of this level to be considered are the manufacturing process, sensors, actuators, and control devices. Following the idea of mechatronical units they are combined with control software running within control devices, sensors, and actuators realizing the required manufacturing process functionality.

All in all, the basic entity structure of field control systems is oriented towards the most economic execution of manufacturing processes against the background of a turbulent surrounding of product variants, manufacturing volume, and exploited technologies. Hence, the design of field level control systems forces special requirements on the design process, on the control system architecture, and on the integrated devices. The most recent are listed below.

**Low building and maintenance costs:** Building and maintenance costs influence significantly competitiveness and customer satisfaction and, hence, the profit margins and market share of the company using the manufacturing system. Moreover, productivity of manufacturing systems is always a ratio between the production obtained and the production costs where the production costs depend on the costs of the design and application (including maintenance) of the manufacturing system. So in order to increase productivity, we have to improve performance but also reduce overall costs.

**Robustness with respect to operability under malfunctions:** Manufacturing systems need to operate in a predictable way under all circumstances. This includes also proper behavior in the case of malfunctions. Independently of its size and complexity, a manufacturing system has to be continuously available or at least prevent dangerous behavior in the case of faults. This requires overall reliability and maintainability, but also some degree of fault tolerance.

**Flexibility with product types, product volume, and equipment:** Flexibility in manufacturing is a very broad concept covering varying types of flexibility. With respect to manufacturing systems most recent types of flexibility are as follows:
– Product type flexibility describing the ability to manufacture different kinds of products and different versions of similar products or the same product,

– Product volume flexibility describing the ability to manufacture different lot sizes of products, and

– Equipment flexibility describing the ability to add, delete, and change equipment within the manufacturing system without additional efforts during system runtime.

Hence, flexibility is the capacity of a system to adapt to new manufacturing requirements as well as to new manufacturing devices and technologies.

**Application-dependent control system development:** The development of the control system should be guided by the application it is intended for, i.e., by the functions and functionalities it has to allow in the controlled system and, thereby, by the manufacturing process the control system has to control. The control engineer should be guided in his work by the desired behavior expected from the manufacturing system and, hence, the controlled system that has to be mapped to available mechatronical units with its inherent manufacturing-process-related capabilities and control code fragments.

**Component-based development approach:** Usually manufacturing processes are structured in a modular/hierarchical way consisting of manufacturing process steps, substeps, subsubsteps, etc. Each of the steps within this hierarchy is based on the execution of a physical process provided by a mechatronical unit. In addition, manufacturing systems are today large and complex real-time systems. They can benefit significantly from a component-based development approach where new systems are constructed by composing reusable, documented, and previously tested concurrent objects. The re-use of components is a very powerful instrument for reducing development time and related costs, but also for increasing the control system reliability.

**Human integration and friendliness:** In automation systems, human integration is more than just a user interface. Control systems usually require human integration for supervisory control. Hence, it covers aspects related to the active cooperation of human and semiautonomous systems to execute coordinated tasks required to successfully execute manufacturing processes. Here, problems of clear representation of current manufacturing system states and possible supervisory control activities have to be solved.

**Compliance with existing system and standards:** Standardization of modules and integration protocols can actively support the long lifecycle of re-configurable automation systems and can provide a means to reduce building costs. Compliance with existing standards is also required by the need of an enterprise to integrate all its automation systems in a larger factory organization.
Integration with existing control devices and legacy systems: Inside an organization an automation system is not an island but a component of a larger system. Control applications must be easy to integrate in more complex systems, providing and receiving a flow of data toward the legacy enterprise management system. On the other end the control architecture must be able to easily interface existing control devices.

This set of requirements needs to be properly addressed within control system design, resulting in an overall increase in design and construction complexity. Thus, it is obvious that it is more necessary than ever to provide some basic design patterns to assist control system designers and end users to design, build, and operate a safe, efficient, and economical control system.

3 Rationale of Design Patterns

The basic problem of control is the necessity of controlling the physical values influencing the controlled system based on the current state of the controlled system. By economic, social, legal, and other reasons the controlled system has to follow a specified sequence of states over time. Which system structure of controlled system and controller enables the most efficient way of control following the specified state sequence? This problem arises in all fields of economy and society including process automation, building automation, and factory automation.

In this light, the complete history of control technology can be seen as the search for a general solution to the named problem and the development of concrete, practical solutions to more tangible and restricted problems. One of the best known general solutions is that of the closed-loop, feedback-based control system. In general, we can say that the usability of this solution for the general problem is not limited to a particular case; hence the feedback control design pattern can be applied – in principle – to any automation system.

As can be seen in this basic example, the main aim of a design pattern is the description of basic structural and behavioral principles of a problem solution that can be reused in a wide domain. Nevertheless, the description of the solution itself is not sufficient [5]. It has to be accompanied by:

- A description of the problem the solution is targeting on,
- The surrounding and environmental conditions the design pattern is valid within, and
- The drivers and ancillary conditions that imposed this solution.

In addition to this directly pattern-related information, the integration of the design pattern in the overall “world of design pattern” must also be characterized,
i.e., other design patterns that are related to the considered design pattern must be identified, as well as the type of relationship must be given.

If design patterns are described in this way, they can be easily used to catalog and document design knowledge. In this way, knowledge exchange among scientists and practitioners, system integrators and end users, and people of different manufacturing sciences can be improved by the common knowledge base and the common description framework that pattern languages provide. This communication problem is very relevant within the automation world. Here specialists from information sciences, mechanical engineering, process industry, electrical engineering, and other fields have to work together towards a common goal.

4 Existing Design Patterns for Field Control Systems

There have been several developments in the field of application of design patterns in the control domain. Obviously there are two main sources of patterns specifically focused on the two sides of the technological spectrum that goes from control theory to computer science.

On one side control patterns are the basic knowledge that all control engineers learn in any control textbook. There are plenty of examples from the feedback control pattern representing the commonly used closed-loop control cycle or the simple Proportional integral differential (PID) pattern describing the most used continuous controller structure exploiting proportional, integral, and differential control reactions on measured controlled system states, to the more complex model-based adaptive-predictive control or expert-fuzzy patterns. These control textbooks, however, lack an important aspect that is vital for the effective use of the pattern at large: the systematic method of the presentation that a pattern schema provides.

On the other side of the spectrum, pattern schemas are usually the very well investigated structures of design patterns in the computer science community. The books [3, 12, 13] offer valuable material for the implementation of computer-based control systems related to the field control layer. Computers and software are the building materials of control systems, and software design patterns are very valuable in the efficient implementation of them.

In this last case, however, most of the patterns lack some of the critical information and guidelines that are necessary to be applicable in the control field: those related to the real-time operation of the pattern. There are few examples of design patterns that fully address these issues, and in most cases they are focused on very specific domains; for example, [14] is focused in the avionics software domain and [15] is specifically focused on small-scale embedded systems on microcontrollers. In the newest approaches the aspect of design methodologies and engineering procedure has attracted greater interest. As examples [6] focuses on the methodological aspect of the integration of design patterns in control system engineering, while
[7] is mainly targeted at the application of design patterns in the engineering process of control systems based on mechatronical units.

Nevertheless, it is necessary to extend the systematic application of a unified pattern schema across the entire disciplines of control, computing, and communications to be effective in the capture of the design knowledge that is necessary for the adequate implementation of the complex distributed controllers that are required in plantwide control of manufacturing plants and can follow the needs of the manufacturing systems sketched above. An example of such a schema is presented in [5].

5 Design Pattern for Distributed Field Control Systems

In the following section selected design pattern dedicated for Distributed Field Control Systems will be described. The main aim of these design patterns is the enforcement of the paradigm of distributed control systems based on distributed intelligence.

5.1 Design Pattern - Distributed Control Applications

Distributed control applications are based on the cooperative solution of a control problem by a set of independent but cooperating intelligent and non-intelligent control devices as usually given in the case of interacting mechatronical units within a manufacturing system. Within this context the problem of assignment of control application parts to control devices has to be solved.

In factory automation the aspect of distributed intelligence on different control devices is growing fast. Control applications and the intelligence inherent within these applications are being increasingly distributed among interacting and cooperating control devices. In parallel, this increases the control application execution capabilities of control devices resulting from the increase in computational capabilities.

This distribution problem requires the allocation of different parts of the control application to control devices. Following this distribution the different parts of the control intelligence are distributed to control devices, and the necessary interactions among the intelligent units and, thereby, the necessary relationships between devices are defined. The distribution has to follow the control system requirements and has to enable a safe, efficient, and economic control system design and application.

The main drivers for a solution to the above-mentioned problem are the reusability of control application parts as well as the maintainability and flexibility of the control system itself. In addition, the application of mechatronical units hosting the
control application and executing the controlled process is intended.

The solution has to ensure that the necessary knowledge about products that have to be produced, including knowledge about necessary manufacturing processes that have to be executed and the necessary knowledge about manufacturing resources that have to be applied (production system knowledge) is integrated within the control design process and, ultimately, within the control application. Here, the mechatronical units will play an important role since they usually have to convey the necessary control application parts relevant for the control of the physical manufacturing process in the layer of sensor and actuator access, and the further control application parts related to product manufacturing are distributed among mechatronical units.

The solution of the mentioned problem is based on the definition of control functions containing control system building blocks and their unique allocation to control devices. The combination and interaction of these control functions will establish the control application.

The control functions associated to the control devices can be classified within two main function classes.

The first class consists of process functions. These functions are used to control the manufacturing process and its progress. They provide a complex system of manufacturing process steps that largely enable the manufacture of the desired products. The process functions are sequenced in a so-called half-order relation. Each process function may have predecessor steps, successor steps, and parallel (concurrent) steps. This half-order, together with the individual process functions, represents the overall product knowledge contained in the control application.

The second class of functions are safety functions. These functions contain all control activities necessary to safeguard machines, material, environment, and human beings against dangerous events. Safety functions interact with other safety functions as well as with process functions. They act as observers for process functions to prevent dangerous situations caused by the activities of process functions. These interactions, together with the safety functions and the process functions, will represent the production system knowledge contained in the control application.

Each control function needs to be able to interact with other control functions crossing device borders. Hence, the control system has to provide a device-independent interaction mechanism for control functions enabling a control function interaction without the knowledge of the function location on control devices. This device-independent interaction mechanism has to be provided by the control device interaction system.

The described solution is depicted in Fig. 1.

The described solution enables an easier handling of control functions within
a distributed control application related to the devices hosting the control application. This property is especially important for device-independent designed and automatically distributed control applications as often generated within Model Driven Engineering (MDE)-based control application design processes [16, 17]. In this case a set of control functions will be distributed among a set of devices without user intervention. Here it is necessary to automatically establish the necessary communication paths among functions to enable their interaction. The provided solution will enable this automatic integration of interaction mechanisms.

A main influence on the described solution has been the International Electrotechnical Commission (IEC) 61499 standard [18]. This standard describes a function-block-oriented structure for specification, implementation, and analysis of control systems using event-based interaction mechanisms for control function interaction. It is useful for the implementation of distributed control applications since it enables a direct realization of the above-described solution. The IEC 61499 standard is widely considered in research resulting in various applications and control engineering design methodologies [19].

The solution is based on the experiences made during the realization of distributed control applications and the development of new control design tools for the Field Control Level. The main drivers have been the research activities within the research projects TORERO (Total lifecycle web-integrated control) [20, 21], JAKOBI (Java und komponentenbasierte Industriesteuerung) [22], and PABADIS’PROMISE (PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises) [23, 24] as well as practical experiences within industrial projects.

The application of the described solution is not limited to a certain manufac-
turing area. In addition it is not limited to a certain programming language- but object-oriented or function-block-oriented languages are preferred.

5.2 Design Pattern – Reusability of Control Software Building Blocks

In the same context of distributed control systems based on intelligent and nonintelligent devices solving a common control problem, the problem of control software building block reuse also emerges.

Distributed control applications have to reflect the modular design of manufacturing systems based on mechatronical units. System designers enforce the standardization and the reuse of these mechatronical units. This concerns manufacturing modules as well as other system parts necessary for the functionality of a manufacturing system like communication systems or data management systems.

Following this trend and based on the aim of economical efficiency and correctness of control systems it is useful to design control applications based on control application building blocks and to reuse existing/tested building blocks within other control projects.

But this objective leads to the problem of control application decomposition enabling a most efficient reuse of control application building blocks.

The main driver for the solution of the reuse problem is the possibility of different viewpoints for the decomposition of manufacturing systems and manufacturing system control applications. The decomposition into control application building blocks can be based on the following decomposition rules:

- Assignment of different production functions and safety functions to different building blocks
- Assignment of different mechatronical units to different building blocks
- Assignment of different technologies to different building blocks
- Assignment of different technological parts of a device to different building blocks
- Assignment of different devices to different building blocks.

The intended solution of the decomposition problem has to be as flexible as possible to enable all these decomposition rules and, thereby, enable the automatic decomposition/composition of control applications and its automatic distribution among devices.

In addition, the intended solution has to enable the easy replacement of building blocks to make it easy to upgrade/redesign the system. This will increase the application fields of control building block reuse.
The solution to the above-mentioned decomposition problem is based on the
distinction of equipment (control devices, sensors, actuators, communication sys-
tem, material, human intervention, etc.)-dependent and equipment-independent
control application building blocks. The equipment-dependent control application
building blocks can be seen as the drivers of the physical capabilities of the mecha-
tronical units involved covering the control of the physical manufacturing pro-
cesses as well as the control of the involved devices. The equipment-independent
control application building blocks can be seen as the control application parts re-
sulting from the control of the product-oriented manufacturing process execution.
Together all these building blocks reflect the internal logic of the control applica-
tion.

The set of equipment-independent control application building blocks contains
the control logic that is independent from the technologies applied in the controlled
system. They reflect the technology-independent product and production process
knowledge of the control system. An example of the contents of these building
blocks is the sequence of necessary manufacturing steps to create a certain product
within a manufacturing system.

The set of equipment-dependent control application building blocks contains
three subsets for equipment control functions, for example controlling a special
drill within a drilling machine, communication system access, for example the
Ethernet-network-card-based access to a MODBUS TCP communication system,
and hardware access, for example to direct input/output cards. These three types
are similar in their functionality since they provide access to the plant but they have
different internal and application characteristics in their functionality as proxies for
plant access.

The control logic in the equipment-independent control application building blocks uses the equipment control functions for the realization of their control tasks.
They access the equipment functions via a generic equipment function interface
that will be configured with respect to their cooperative use by more than one con-
tral logic building block. In addition the control logic building blocks will use the
communication building blocks for communication with other control logic build-
ning blocks and with upper-level control systems.

The equipment functions themselves will use the communication system access
building blocks as well as the hardware access building blocks for the completion
of their control tasks. Again the access to the communication and the hardware
function building blocks is made by a generic and configured interface mapping
logic variables and functions to physical access.

The described structure is given in the following class diagram (Fig. 2).

This structure enables to a large extent the decoupling of the different layers and
their interconnection by generic and configurable interfaces. This is depicted in the
following example of a transportation system (Fig. 3). This transportation system
consists of different turntables and conveyers. Within the overall control logic a transport sequence can be implemented independently of the used conveyers and turntables for their control. The control of the turntables can be implemented independently of the access path to the physical devices (drives and position sensors). The actual sequence of turntables and conveyers controlled by the control logic will be configured within the generic interface between control logic and turntables/conveyers, and the association of turntables/conveyers to physical devices will be configured, for example, in the interface to the communication system enabling the physical system access (Fig. 3).

The combination of different building blocks within a control application follows three main application scenarios. These three scenarios are based on the interaction of a control logic building block with a plant, another control logic building block, or an upper-level control system like MES or ERP.

The interaction between control logic building block and plant is carried out by using equipment function control building blocks and its access to hardware or communication building blocks depending on the possible access paths to sensors and actors. This interaction structure results in a decoupling of the data transmission between plant and control application that is carried out by the hardware access and the communication system access building blocks from the measurement value preparation which is carried out by the equipment function control building block.
block and from the application of the gained data from the plant within the control logic for the overall process. This structure is depicted in the left part of the following collaboration diagram (see 4).

The interaction between control logic building block and upper-level control systems like Manufacturing Execution Systems (MESs) and Enterprise Resource Planning (ERP) systems is initiated by the upper-level control systems. The interaction is realized by using the communication building blocks as interface. This structure enables the decoupling of the accomplishment of the required action of the control logic from the data transmission between both system parts that is carried out by the communication system access building block. This is depicted in the middle of the collaboration diagram in Fig. 4.

Finally the interaction between two control logic building blocks is also based on the application of communication building blocks as generic and configurable interface. In this way, the decoupling of data transmission between control logic building blocks (executed by the communication building blocks) from the accomplishment of control actions (executed by the control logic building blocks) is realised. This is depicted in the right hand side part of Fig. 4.

The described solution enables a widely independent development of control system components with respect to the correct control of manufacturing system building blocks, the correct application of communication systems, and the correct interaction of system components. The very generic interfaces of the different building blocks enable an automatic composition of control applications based on these building blocks integrating predefined or vendor-delivered control system components.
Like the design pattern “Distributed Control Applications”, the described solution is based on the experience gained during the realization of distributed control applications and the development of new control design tools for the Field Control Level. The main drivers have been the activities within the research projects TORERO [25], JAKOBI [26], and PABADIS’PROMISE [24], as well as practical experiences within industrial projects. Additionally, experiences in the field of software design specifically the layers pattern [5, 6] had a primary influence on the solution.

Like the pattern “Distributed Control Applications”, the application of the described solution is not limited to a certain control field or to a certain programming language. Its optimal application will come from using object-oriented programming languages.

5.3 Design Pattern – Devices Within Distributed Control Systems

Within the context of distributed control systems based on intelligent and non-intelligent devices solving a common control problem, the problem of the efficient structure of control devices dedicated to the distributed nature of the control system also arises. Additionally, the control device structure has to reflect the integration of control devices within mechatronical units. Here the devices have to be able to interact within the mechatronical unit to control it as well as to interact with other control devices across mechatronical unit borders.
Hence, distributed control systems require control devices that enable a cooperative solution of a common control problem by using a distributed control application based on control system building blocks. Therefore, the control devices have to be prepared from the technological, structural, and architectural points of view. The emerging requirements for control devices are based on the necessity of running control application building blocks in a cooperative and coordinated way and the necessity to enable communication between control building blocks by enabling communication among control devices.

This problem has to be considered in view of the existing control device technology and the current trend for control system device improvements. These devices have to be adapted to the necessities of distributed control applications. Existing generations of devices should be incorporable within distributed control systems without major modifications.

Since the described solution implies the application of software building blocks on the control devices, intelligent devices as intelligent Programmable Logic Controllers (PLCs), industrial PCs, or embedded control devices within sensors and actuators are essential for the solution of the problem. They have to be considered as the basis of the solution described below. The solution of the mentioned problem is based on a bipartite structure of hardware and software entities that are functionally related. The hardware part consists of the following five parts:

- One or more processors providing the necessary computing power,
- One clock for temporal coordination of the interaction and cooperation of the different control devices with respect to necessary temporal interrelated control actions,
- One or more memory building blocks containing all necessary information like control code, data at runtime, documentations, etc.
- One or more communication networks (resp. network cards) used for physical access to communication media, and
- Zero, one, or more physical inputs and outputs (the physical connection).

The software part consists of two main component classes of entities. The first class contains one or more control application building blocks running on the control device. The second one contains

- The operating system,
- One or more communication protocols, and
- Software drivers needed to access the physical Input/Output (I/O) interfaces.
Each control device is uniquely associated to one or more plant building blocks that will be exclusively controlled by this device. In this way, plant building block(s) and control devices comprise a mechatronical unit. To control its plant building blocks the control devices have to contain the software implementation necessary for control logic.

This control logic has to be implemented using control software building blocks as expressed in the design pattern above. These application building blocks will run on top of the operating system using all its functionalities and will use for its interaction with the plant and other control application building blocks running on other devices the communication protocol drivers and the I/O access drivers. These two types of drivers will be governed by the operating system, which will provide an interface to these drivers to the control application building blocks.

The operating system will use for its activities the processor(s), memory block(s), and the clock. Each communication protocol driver will use the communication network attached to it, and the drivers for local I/O access will use the physical I/O interfaces.

The access of the control application building blocks to other devices will be provided using the communication protocol drivers and the communication networks. Access to the plant will be obtained using the communication protocol drivers and the communication networks as one access path and the local I/O access drivers and the physical I/O interfaces as the other access path. This structure is depicted in Fig. 5.

As in the design pattern “Reusability of Control Software Building Blocks”, the described solution will provide a layered structure of hardware and software building blocks establishing control devices within a distributed control system. It extends the pattern “Reusability of Control Software Building Blocks” by hardware and hardware near software building blocks.

Like all previous patterns, the described solution is independent of an application field or a special industry. It is also independent of a special control system implementation language. But it has been developed especially for devices having a PC-like architecture and being integrated into a factory communication network. In this context mainly fieldbusses and Ethernet-based factory communication systems are used. Hence, the solution is dedicated to systems implementing an Ethernet-based industrial communication protocol.

Like the design pattern “Distributed Control Applications”, the described solution is based on the experience gained during the realization of distributed control applications and the development of new control design methodologies for the Field Control Level. The main drivers have been the activities within the research projects JAKOBI [26] and PABADIS’PROMISE [24], as well as practical experiences within industrial projects.
Figure 5: Design pattern - devices within distributed control systems
6 Application of the Design Patterns Within the PABADIS’PROMISE Project

Within the PABADIS’PROMISE project the described design patterns have been used to develop a methodology and an architecture usable for the design and execution of product-related control applications on demand. Here, control applications will be developed only if a product needs to be manufactured.

Based on the design pattern “Reusability of Control Software Building Blocks” the PABADIS’PROMISE field control architecture is based on a layered structure with three main components.

The first component sitting on top of the architecture is the so-called ability application executed by an ability application runtime environment. This ability application constitutes the equipment-independent control logic within so-called Resource Function Blocks coding how the ability application invokes the underlying manufacturing functions as well as equipment-dependent functions within so-called Device Function Proxies (DFP). DFPs implemented as Service Interface Function Blocks (SIFB) that enable access to a Device Proxy (DP) and represent the underlying device functions inside an Ability Application.

These ability applications are IEC61499-compliant function block (FB) networks, which are executed within the Ability Application Runtime. As this runtime is only responsible for the triggering of functionality in field-level devices it is normally not executed under real-time constraints.

The second component of the three-layer structure is constituted by the set of DPs. DPs implement communication system access components of the mentioned design pattern realizing the interaction of the ability application with the underlying manufacturing system controlling (non-)intelligent devices.

These underlying manufacturing system controlling (non-)intelligent devices constitute the third and lowest layer of the structure.

Following the design pattern “Distributed Control Applications” the ability application as well as the underlying manufacturing system controlling (non-)intelligent devices are divided into Process Function Blocks and Safety function Blocks. Especially within the ability application the Resource Function Blocks plays the role of Process Function Blocks while the DFPs take over both roles of Process Function Blocks and Safety Function Blocks.

The design pattern “Devices within Distributed Control Systems” was the main driver for the design and implementation of the ability application runtime environment. It constitutes a type of operating system executing the model-based ability applications and providing communication and access functionalities over the DPs.

The resulting structure is given in Fig. 6.
Figure 6: Application of the design pattern within the PABADIS'PROMISE control architecture
The essential benefit provided by the PABADIS’PROMISE field control architecture is the capability of the system to handle field control applications in a very flexible way. These capabilities include:

- The ability to add, change, replace, and delete manufacturing system controlling (non-)intelligent devices to the system without changing the ability applications during runtime and, thereby, to make the underlying manufacturing flexible;

- The ability to change ability applications during runtime enabling product-related flexibility; and

- The ability to change manufacturing system controlling (non-)intelligent devices-access structures and, thereby, enabling a resource-application- and manufacturing-process-related flexibility, for example, to increase process optimality.

All in all, this field control architecture enables a maximal flexibility based on the application of the three described design patterns.

7 Conclusion

In this chapter we have described a set of fundamental design patterns that can be used in the field of distributed automation. These design patterns cover the field of distributed applications, reusability of control application building blocks, and the device structure and functionality necessary for distributed control systems in light of the recent requirements of manufacturing system control as well as the strong trend toward the application of mechatronical units.

Following this design pattern it will be possible to design control systems and control applications in an efficient way.

The design patterns introduce a layered structure for distributed control systems that enables a structured design for distributed control systems. In a first design step a manufacturing system resource-independent control application part will be designed. This control application part will be combined in a second step with predesigned control application building blocks necessary for the control of the used manufacturing system resources and control application building blocks necessary for access to communication systems and physical I/Os.

A control application that has been designed in this way is much easier to maintain and to change with respect to new communication systems, manufacturing system resources, or products. In addition, it can be used for automatic control application building block distribution among devices, which is a huge benefit for distributed control. Finally, this procedure can be exploited within an approach
designing control applications on demand related to manufacturing orders as successfully developed within the PABADIS’PROMISE project [24].

8 Acknowledgements

This work has been prepared mainly by Lüder and Peschke that are with Otto-von-Guericke University of Magdeburg, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg, Germany (e-mail: arndt.lueder@ovgu.de and joern.peschke@ovgu.de).

9 References


