An Agent-Oriented Approach to Industrial Automation Systems

Thomas Wagner

Institute of Industrial Automation and Software Engineering
University of Stuttgart
Pfaffenwaldring 47, D-70550 Stuttgart – Germany
Fax: +49 711 685 7302
Phone: +49 711 685 7295
E-Mail: wagner@ias.uni-stuttgart.de

Abstract. Multi-agent systems have been successfully used in a number of industrial applications. However, in the domain of industrial automation systems, little practical experience about the use of agents in such a demanding environment exists. Nowadays automation systems face new challenges and therefore new concepts are needed to meet them. This paper presents a vision of using agents within industrial automation systems from the point of view of automation engineering. To this end, the characteristic structures of automation systems are analyzed and opportunities, as well as advantages, of applying agent-oriented concepts are investigated. A possible approach for the integration of a multi-agent system into existing automation systems is introduced and illustrated by an application example.

Keywords: agent applications, automation systems, process control software, plant design, plant engineering.

1 Introduction

Automation systems such as those deployed in industrial plants, are complex, large and persistent hardware software systems being stamped by the characteristics of the technical processes they are designed to control. The software applied today in industrial automation systems has grown incrementally and therefore distinguishes itself by static hierarchical structures with precisely separated tasks, specialized functions and information structures. The applied concepts are based on different functional software levels according to the hardware structure, and hardware devices are modeled as predefined software components that have to be configured and connected into complete applications. These concepts have been successful during the last 20 years.
However, the development of automation systems software faces new challenges. The rapid development of high-capacity hardware components such as microcontrollers

leads to more decentralized systems structures. Besides that, a strong need for integration of new functionality from enterprise management and business levels into the automation system arises. This trend leads to an increasing complexity[1] in automation systems, appearing in a more extensive functionality of the used hardware devices in the plant (the so called field devices) and consequently in more complex interactions between hardware and software entities. This evolution effects an expanded effort for development, plant design[2], plant engineering[3] and maintenance. In order to decrease this effort and reduce the tasks of engineers, future software for automation systems has to be able to handle this complexity. Comparing these challenges to static structures and functional characteristics of existing software in automation systems, it is doubtful if actual software concepts are qualified to match the increasing complexity.
An interesting approach to handle complexity in automation systems could be to map the increased "intelligence"[4] of hardware components into the software system. However, therefore new concepts and methods for automation software engineering are needed. The paradigm of autonomous agents is a promising concept for engineering complex, distributed systems. In some fields of software systems (e.g. the automation of business processes or information management), they have already found their way into real-world applications. This paper deals with the application of agent-oriented approaches to enhance the software of automation systems for simplification of both plant design, engineering and maintenance as well as the integration of functionality from external systems. Towards this objective, a clear understanding of the target area is necessary and two central questions have to be answered:

- *Is the agent-oriented approach well suited to design and build software for automation systems? (Or conversely: Do the characteristics of automation systems indicate the suitability of an agent-oriented approach?)*
- *How can agent-oriented concepts be integrated into the existing and persistent software / hardware structures of automation systems?*

In seeking to answer these questions, this paper is structured as follows. In section 2 the target area is analyzed with regard to the relevance of agent-oriented approaches. For this purpose a detailed view at the software structures in automation systems is introduced, together with a general rating of the existing concepts compared to the challenges on future software development. In section 3 the relevance of the agent-oriented paradigm to meet these challenges is examined and the fundamental possibility of applying agent-oriented concepts to legacy automation systems is investigated.

---

[1] In this context, the term "complexity" is used in a general manner; not in the specific technical sense of algorithmic or computational complexity [1].

[2] Plant design contains activities of technical realization of a plant with design phase, implementation phase and plant commissioning phase.

[3] Plant engineering contains activities during plant operation with the aim to maintain or improve the technical operativeness of a plant. This includes condition monitoring and estimation of plant components, decision on measures and implementation of measures in the plant.

[4] The term "intelligence", as widely used in the application domain of automation engineering refers to the extended functionality of field devices, like self-diagnosis and failure prediction [12]; in this article it is not used in the sense of artificial intelligence.

The basic ideas for the introduction of multi-agent systems are described in section 4. In section 5 an example of using agents in asset management systems shows the advantages of an agent-oriented approach in this area.

2 Problems in engineering and operation of automation systems

In this section the target area of automation systems in industrial plants will be introduced in detail to make clear the characteristics of the problem and compare them to the merits of an agent-based approach.

2.1 What is an automation system?

Industrial plants are technical facilities, in which technical (e.g. energy-related, chemical, production or material-handling) processes are controlled automatically. Examples are pharmaceutical plants, automotive production plants or power plants. The automation system used within a plant consists of three different elements (see Fig.1):

- The technical process running in a technical system
- The control system and the communication system for information flow linkage of hardware and software entities
- The process personnel for operating & managing

More detailed information on the automation system's elements is provided in [2]. There are also other types of automation systems, e.g. in the domain of product automation, where the technical process takes place within a single device. Within the scope of this paper only automation systems used in plant automation are regarded.

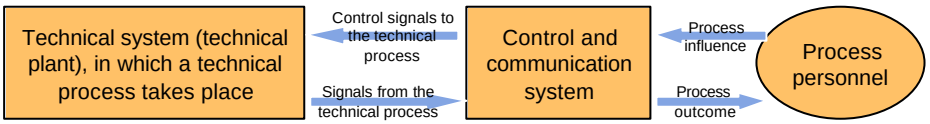


Fig. 1. Elements and information flows of an automation system.

Nowadays deployed automation systems feature three outstanding characteristics:

1. High degree of automation, i.e. the operations to control the technical process are predominantly performed automatically. The tasks of process operators are limited to the superordinated process control and intervention in exceptional situations.
2. The automation system has to accomplish extensive and complex automation functions [2]. Automation functions are for instance: Control of the technical process, handling and observation of operating sequences, monitoring, diagnosis and maintenance. These automation functions are nearly solely fulfilled by software.

3. Plant development and design (including the configuration of hardware entities and communication systems) are often considered separately from the operation of automation systems. This leads to a lack of transmissibility between design phase and plant engineering phase (during operation) and therefore causes problems in data consistency together with a high manual engineering effort.

Additionally it should be considered that, from a technical point of view, an automation system often consists of hard and software entities of different manufactures and their different technologies (e.g. bus protocols or field device interfaces). These different entities have to be integrated into the complete system (including the software system) with manual engineering effort.

In order to determine the fundamental problems and challenges in plant design, installation, commissioning and engineering, a more detailed investigation of the inherent structure of automation systems will be given in the next section.

2.2 Characteristics of modern automation systems

In the course of the development of automation technology more and more functions which take place within technical systems were performed by automated equipment, i.e. the degree of automation increased. At first automation functions were developed at the operational level (like supervision of process values, control of sequences, etc.) for the respective measurement and control systems. With the application of automation computer systems comprehensive functions from the „higher“ levels were integrated into the automation system (like sequence planning and optimization) [2].
As a consequence of this evolution, modern automation systems have a hierarchical structure with different automation functions at each hierarchical level. This hierarchical structure is called the automation pyramid and represents a canonical abstract model for the structure of complex multilayered plants [2],[3]. Within this model the several organizational levels of an automation system are shown (see Fig. 2). The typical automation pyramid for automation systems in a plant consists of the following levels (from the bottom up):
The *actuator-sensor level*, the *field level*, the *process control level* and the *management level*. These levels are networked between each other with different bus systems, which are adapted for the specific requirements of data flows at each level.

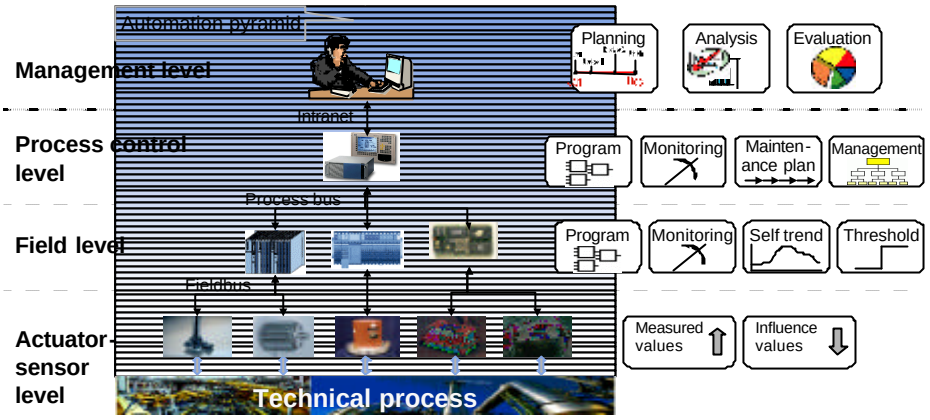


Fig. 2. Canonical structure of modern automation systems.

At the *actuator-sensor level* (also called process level) measured values, i.e. physical process variables like temperature, pressure and distance, are captured via sensors and influence values for controlling the technical process are output via actuators (controlling elements) like valves, pumps, heatings and servo-motors.

The *field level* accommodates all tasks needed for processing measured values from the process and directly influencing of the process [3]. At the field level, typical automation computers like programmable logic controllers and microcontrollers are used. In addition the process can be influenced using monitoring and operating consoles.

At the *process control level* monitoring and process management of the overall process is performed, as well as production control, start up and shut down processes and superordinate malfunction handling. Due to the high number of plant hardware components, the management and engineering of the plant (e.g. setting of field devices and communication parameters) as well as maintenance planning has to be done.

Within the *management level* all higher-ranking enterprise levels are summarized. At this level tasks such as production planning, evaluation and analysis of process results and optimization of operational sequences are executed.

Regarding the software in the automation system, a clear separation of information processing can be identified: at each level different data is processed and different views on the technical process exist.

During the last 20 years the implementation of software in automation systems has been performed in the same way [2],[6]: The overall functionality is decomposed into single functions, of which a lot of are available in the form of prefabricated universal program building blocks (i.e. standard blocks for control, monitoring, operating, supervision and recording tasks). These single functions are connected together with regard to data flows and temporal dependencies. This process is called configuration.

The quite different goals and specific conditions at the several levels and the resulting requirements to the software (e.g. amount of data, temporal restrictions) have caused a

strong orientation of the means of description[5] used for implementation towards the automation functions and views that are realized at each level.
The prevalently applied means of description are some kind of visual net-based languages. Examples are Function Building Blocks (FBB), Continuous Function Charts (CFC) and Sequential Function Charts (SFC) [4],[6]. The abstraction level of these means of description is very close to implementation and focused on the hardware- and machine-related properties of the system [5].
The same characteristics apply accordingly to application models[6] that are the basis for the means of description. All application models have in common to describe a system in the form of a composite (net) of passive entities and provide a function or sequence oriented view on the automation system.
Today the software systems in automation systems with their specific characteristics face new requirements as explained in the next section.

2.3 Challenges of future automation systems

Two main factors have a strong influence on the evolution of automation systems:

1. Due to the rapid development of information technology, high-capacity hardware components like fast processors, large memories and wide-band networks are available today. This leads to an increasing decentralization within automation systems and a distribution of functionality into field devices. Resulting from this is an increasing amount of communication and increasing effort for the configuration of system entities as well as of the complete system. The dynamic development of hardware components in automation systems is in contradiction to the long life-cycle of a plant (up to 30 years). Hence, automation systems are subject to an ongoing partial modification. This affects particularly the automation system software which is intensely stamped by the structural aspects of the plant: several software entities are optimized or adapted to changes in the hardware structure or in processes. This results in strong requirements on flexibility and adaptability of the software during the whole life-cycle [8].

2. The requirements of the process control technology market are a driving force for future development directions and affect automation systems in several aspects:

- The trend to shorter time-to-market cycles and higher productivity require a flexible and dynamic production [9].

---

[5] Means of description picture certain issues in graphical or textual manner for visual perception and storage. They include alphanumeric characters, symbols or other graphical description elements (semiotics) as well as conventions on their combinations (syntax). The several description elements, their combinations or assignments are correlated to specific conditions or concepts of a certain technical context (semantics) [5].

[6] An application model is a representation of a certain part of the real world which is subject to modeling and realization. An application model normally consists of a data model and a sequence model [7].

- For further increase of the degree of automation, autonomous and adaptive automation systems are required. This includes support of human operators in complex tasks like condition monitoring of plant components and decision on measures.
- The vertical integration of additional functionality form higher enterprise levels (e.g. enterprise resource planning and management information systems) requires a stronger interconnection of the machine level with all business levels. Therefore, the interoperability of the different systems has to be accomplished. That means a higher coupling and transmissibility of information flows between the different, strictly separated levels of the automation system [17].
- A better support of users in plant design and engineering, especially configuration and parameterization[7] of system entities is needed (because of the increasing functionality in field devices, the efforts for configuration are even higher). The integration of new system components has to be supported and automated. The vision is a "plug'n'play" scenario comparable to the MS Windows world [10].

The two main factors mentioned above, that drive the evolution of automation systems, can be abstracted to the following consequence: Due to the extension of the former solely hierarchical communication within an automation system by additional services and the demand for a flexible communication to more "intelligent" hardware components (up to n:n communication correlations), more and complex interactions within the automation system become necessary. As automation systems cannot be built form scratch but are subject to ongoing change, these influences have to be compared to the static and hierarchical structure of contemporary systems. It seems obvious that as a result of these influences, both the scale and the complexity of automation software will strongly increase to fulfill all necessary automation functions. This complexity can be characterized and classified in two aspects:

- **Functional aspects.** Organization and control of a distributed system at runtime (i.e. including the dynamic allocation of functionality).
- **Design and Engineering aspects.** Development and design of a decentralized plant structure as well as configuration and management of single components.

The capabilities of existing software to manage the evolving complexity in comparison with the merits of an agent-oriented approach are presented in the next section.

3 Why are agent concepts relevant to automation systems?

In section 2.2 and 2.3 the characteristics of modern automation systems and requirements for further evolution were shown. They can be summarized in properties that are intrinsic of all automation systems and their software used in industrial plants:

- Automation systems are complex and distributed systems.
- Automation systems require different views. Engineers, process personnel and managers have a individual view on structure, data and functionality.

[7] Choice of variable values to characterize the system's behavior within a given structure [4].

- Automation systems require flexible and adaptive software. Changes affecting the software can refer to data, structure or operating sequences of the system.

These properties are comparable to the general properties of complex decentralized systems as described by Jennings in [11]. Fig. 3 shows, on the left side, an overview of the inherent properties of complex systems: A complex system has a large number of parts that have many interactions. Complexity frequently takes the form of an organizational structure of interrelated subsystems and their components, relations between subsystems vary over time. Generally viewing the development of software these days, a wide range of software models and techniques have been devised to make it easier to handle this complexity. However, they fall short in two main ways [11]:

1. The interactions between the various computational entities are too rigidly defined.
2. There are insufficient mechanisms available for representing the system's inherent organizational structure.

Picking up the analysis aspects in section 2.2, this statement also applies to software of automation systems: This software is hardware- and implementation-oriented and has strict restrictions to structure and interactions within clearly separated functional levels. Although there are means to represent organizational structures, these structures have to be defined at design time and are not adaptive to dynamic changes later at runtime. Also these structures implement a determinate view on functional levels within an automation system and rigid interactions between system entities. They do not provide the means for a flexible structural reorganization and new flexible interactions as needed to match the challenges mentioned in section 2.3.
According to [1] the agent-oriented approach is a natural way of system decomposition and a reasonable alternative to contemporary approaches in software engineering. Fig. 3 illustrates how the inherent properties of a complex system are reproduced in an agent-based system: Subsystems and subsystem components are mapped to agents and agent organizations; interactions between subsystems and subsystem components are mapped to cooperation, coordination and negotiation mechanisms; and relations between them are mapped to explicit mechanisms for representing organizational relationships. For further details, see [1],[11].

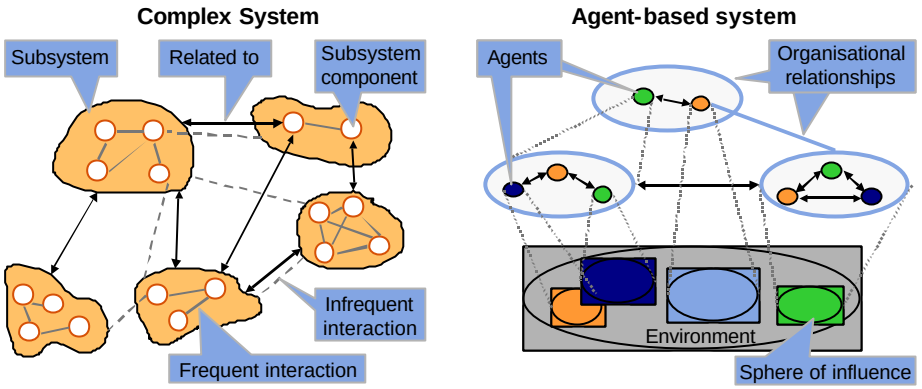


Fig. 3. Canonical view of a complex system compared to an agent-based system [11].

Two facets of the comparison between complex systems and agent oriented systems are of particular significance:

1. In an agent-oriented software system, multiple agents represent the decentralized nature of the problem, the multiple locations of control, the multiple perspectives or the competing interests. This abstraction reproduces the inherent properties of real-world systems and leads to a better handling of complexity within the software.
2. By high-level interactions, agent systems provide mechanisms for building flexible organizational structures. This effects dynamic bottom up coordination and enhances possibilities for dynamic adaptation of software to a changing environment.

For these reasons, it can be concluded that agent-oriented concepts are suitable to meet the requirements of modern automation systems. The agent-oriented approach seems applicable for the development of automation system software.
However, as automation systems and process control software have a long life-cycle, they can be subject to only slow and incremental changes. Hence, the question arises, can agent-oriented concepts be integrated in existing automation system structures and are they able to deal with legacy software or its underlying application models? As explained in section 2.2, the means of description for the development of automation system software typically are specializations of some kind of visual net-based languages. With object oriented methods these languages can be described as networks of interconnected instances. In [5] the possibility is shown, that these models can always be mapped to an object-oriented reference model. In a reference model objects, encapsulate processes and functionality, but also can represent their dynamic input or output information [8].

There are certain similarities and conceptional correlations between object- and agent-oriented approaches. Both adhere the principles of encapsulation and information hiding and recognize the importance of interactions. Agents extend these concepts by autonomy (control over internal state and action choice), own thread of control and persuasion of objectives within themselves [1]. Hence, the ability of object-

oriented concepts to reproduce application models of automation system software in principle can be assigned to agent-oriented concepts as well. In short: agent-oriented approaches basically can be integrated in existing automation systems.
In the next section the fundamental ideas of integrating agent-oriented approaches into existing automation systems are presented.

4 Application of the agent-oriented approach to existing automation systems

4.1 Constraints

Within the domain of information technologies, new software concepts and technologies are deployed in real-world applications very rapidly. However, the innovation of automation system software structures is characterized by an explicitly slower progress than those in office automation or internet applications. This for several reasons:

- Software in automation systems evolves incrementally due to its long life-cylce.
- In contrast to the information technology domain, where mostly standard computers or servers are used, hardware components used in the automation domain are specific to the application area. For this reason software and hardware components have a much closer association (e.g. programmable logic controllers), changes in software often affect hardware and vice versa.
- Industrial plants are large heterogeneous hardware-software systems causing an enormous effort for development and operation and therefore they have a proportionate investment volume. Herein lies the reason for a significant skepticalness of plant operating managers against radical technical innovation.

Since the early nineties, cooperative multi-agent systems and intelligent agents are of increasing concern with respect to the software engineering of large scale distributed systems [13]. For an efficient introduction and a wide acceptance in the automation domain, new software concepts and software systems based on them have to support legacy systems to a large extent. Thus a number of constraints have to be considered:

- The new approach has to be adopted in a pragmatic manner. This means that the design of the new software system introduced has to be oriented to the basic structures of existing application models in order to meet the requirements of process control technology performantly and to be understandable for the user [10].
- Application models already used in the application domain have to be integrated while keeping interoperability. Existing implementations should be expandable [4].
- Adding new performance features has to be possible without deep changes to existing software and hardware architectures.

Considering these conditions leads to the approach for the integration of an agent-based system into the automation system structure explained in the next two sections.

4.2 The automation system seen as a role system

The fundamental approach is based on the principle of a comprehensive complete representation of real entities by agents. This idea follows the insight that every real entity including its functional range can be regarded as a role within a role system. The notion of a role focuses on the position and responsibilities of an entity within an overall structure or system [14]. The role results from the task a certain entity has to fulfill within the system, in order to provide the system's overall functionality. Further it is important that a certain entity can - depending on its functional range - possibly obtain different roles, depending on the individual view (partial functional aspect) the system is regarded with. The analogy of roles within a system is very close to the human mental model[8] when decomposing, developing and realizing a system.
Due to the modeling concepts and implementation techniques applied so far, an automation system is viewed only as a composite of passive entities. That is, the activities necessary for the realization of several roles in a executable system have to be performed by humans during plant design and engineering phase. These activities include the customization of a system entity's functionality (both hardware and software entities) to the required role within the system and its environment (configuration). But further, the mentioned system entity fulfills the expected role in a passive manner. On changes within the system or its environment, the provided functionality of the entity has to be customized again by human activity according to the changes of the role.
If, in contrast, the role is realized by agents, substantial activities can be automated. The agent - aware of its own objectives - can dynamically adapt its behavior to changes and re-customize the represented entity. Beyond configuration, further activities can be automated such as condition monitoring of entities, overall management of system components and their configurations, or integration of external systems.
Below the basics for modeling existing automation systems with agents are explained.

4.3 Integration of an agent system

Based on the abstraction of roles, the automation system is extended by an agent system in the form of an orthogonal communication level within the hierarchy (Fig. 4). The cooperating agents are designed to act very closely with the automation system and can be divided in two basic categories: *resource agents* and *service agents*.

[8] Mental models are representations of the real or virtual world in the human brain, built by the cognitive abilities of human [5].

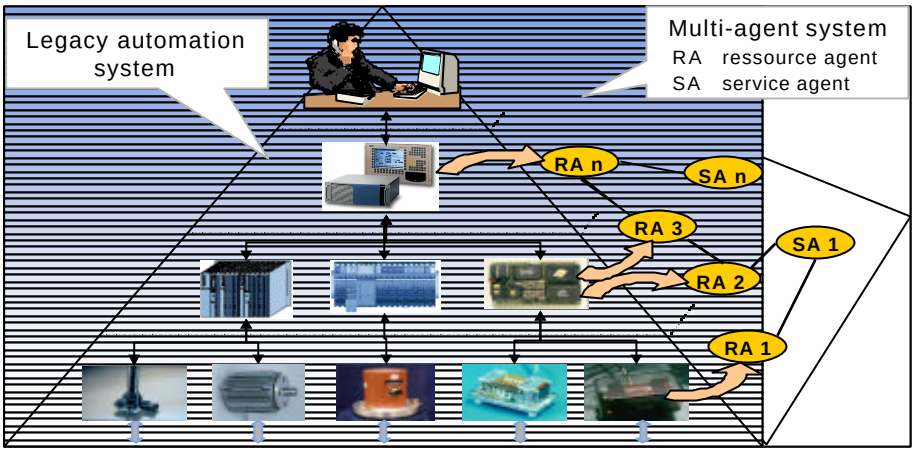


Fig. 4. Multi-agent system within the automation system hierarchy.

- Within the agent system all of the real existing entities addressing the problem of the automation system are represented by resource agents. The entity represents the sphere of perception and influence of the agent. The granularity of this sphere depends from the goals to achieve (e.g. a single device, a device interface, a control program, a control element such as function building blocks or a set of variables).
- Resource agents encapsulate the information flow of the entity they represent. That is, they are able to communicate with the entity's (e.g. device's) specific interfaces and provide an abstract view on the entity within the agent system.
- Resource agents can be of different types in order to represent different views (roles) on the entity: *design agent, commissioning agent, condition monitoring agent, diagnosis agent*. These agents may have to cooperate in performing a role.
- Resource agents act autonomously in reference to the entity they represent. For example: agents representing a field device are capable of reacting to changing environmental conditions by self-adaptation of communication or physical parameters of the device. Device-specific status and diagnosis signals (as provided by modern field devices) can be recorded and interpreted by resource agents in order to generate diagnosis and maintenance information (e.g. life and failure prediction).
- Resource agents can autonomously build organizational structures reflecting the physical or logical plant structures, e.g. a compound of field devices within a certain machine. That way "plug'n'play" of devices can be realized. Within the compound of agents diagnosis and maintenance information can be used to determine the functional reserve[9] of single devices or of plant components. Pro-active acting can be applied for failure prediction or preventive maintenance planning.
- Service agents do not represent a certain real existing entity, but a certain task or a view on the automation system or part of it. They are designed to realize superordi-

[9] The functional reserve characterizes the instantaneous availability of an entity and allows conclusions on future plant or process behavior.

nate functionality that is not bound to a certain device or control element. Examples of such a functionality are management of devices, management of configurations or inspection of the overall system in order to generate maintenance information.
- The existing communication within the automation system (e.g. block-to-block communication via status signals) is maintained. This is necessary to accomplish the generally strict requirements for control of the technical process (e.g. real-time requirements) without restrictions on the flexibility within the agent system.

This way additional functionality and services like automation of configuration sequences or flexible management of devices, can be realized within the agent system. This functionality is enabled as a result of the high-level flexible interactions in an agent system compared to the signal oriented communication of the automation system. Operations and sequences that were restricted by the structure of the existing hardware and software system are reproduced in the orthogonal agent layer. By doing so, a closer coupling of information flows of the different levels in the automation system can be obtained without fundamental changes to existing hardware and software structures - an important aspect for the plant operating managers.
An interesting aspect for the manufacturers of plants and plant components is the effect of implementation protection: The resource agents of a device encapsulate all functionality and data specific to the device. The automation and engineering systems solely have access to the software interface of the agents, not to their implementation. Using this "black-box" principle, in-house knowledge can be hidden, which already today is of great interest to plant manufacturers.
The next section illustrates, how to build applications for automation systems and which advantages can be achieved. As an application example the area of asset management is used, which is an upcoming trend in today's automation engineering world.

5. Application example: agent-based asset management

In automation engineering, functions for administration and optimization of devices are gaining importance in comparison with pure process control functions. These functions are summarized in the term "asset management[10]" or "asset optimization". The ongoing developments in automation engineering concerning this area are targeted at supporting extensively plant commissioning and engineering (online asset management). That is, to utilize all plant equipment such as PLCs, machines, or field devices for adding value and provide the required availability with optimal effort.
An asset management system is a data processing system that collects, concentrates and rates heterogeneous information to conclude the condition of several plant components. Additional access exists to external systems like (electronic) plant documen-

[10] In a broader sense the term "asset" is used in a more comprehensive way, describing all value added by entities of an enterprise; in this context "asset" refers to the plant equipment only.

tation or business administration systems [15]. The NAMUR[11] [16] recommendation for asset management systems lists the functional requirements of these systems:

- Parameterization and configuration of field devices
- Documentation of field device history
- Condition monitoring of field devices and other plant components
- Access to components of the process control software
- Access to plant documentation, CAE-systems and business administration systems

This makes clear that asset management not only targets the maintenance of the existing plant, but starts with design and engineering of process control components and includes access to plant environment. To fulfill this functionality, the asset management system has to access process operating information. Sources of information are measured values from the technical process, status signals & diagnostic information of field devices and special monitoring facilities. Further sources of information that are not directly needed for process operation but for planning, design and engineering are:

- Engineering tools, CAE (computer-aided engineering) systems
- Parameterization tools for field devices
- Documentation of the plant and plant components
- Business administration systems

The data required for optimization of plant engineering must be completely extracted from the existing systems. Finally different user groups who have individual views on the automation systems have to be supported:

- Plant designers and plant commissioning experts
- Plant operating managers e.g. factory managers and plant operators
- Plant maintenance staff e.g. engineering and system experts

At present, enterprises have to bring up high costs for engineering their asset management applications to increase the availability and productivity of plants. The reasons are, on one side, the higher configuration effort due to the increased functionality of the field devices. On the other side, there is an increasing specialization of device data for the same reason. Thus, a divergent variety of control concepts and tools for field devices results. Another substantial problem is the access to existing engineering, CAE and business administration systems, as they are based on historically evolved "de facto standards" and have little in common regarding their data representation. In addition, there are problems of information protection that occur when information has to be accessible across the boundaries of different network infrastructure.
In short, two fundamental challenges exist in realizing an asset management system: 1. Support for varying views of different users groups, depending on their objectives and 2. Integration of heterogeneous distributed information sources.
In an agent-oriented approach, the different information sources, views, tasks and system abstractions are represented by resource and service agents. These agents

[11] European association of users of process control technology - see http://www.namur.de

encapsulate external data sources within the agent system and cooperate on a common semantic level. This way, the highly decentralized character of the asset management system can be abstracted to a flexible software structure. Fig. 5 shows the basis structure of an agent-based asset management system: The blocks left and right show the system's environment, the different views of the system are illustrated on top.

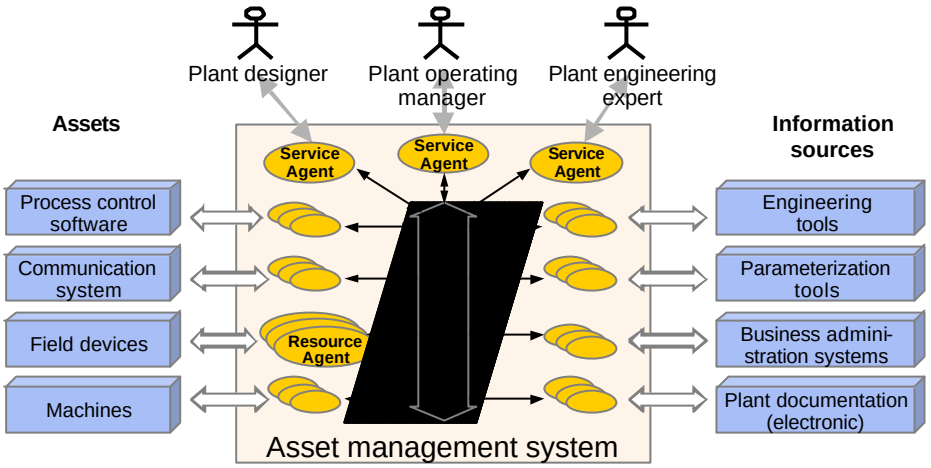


Fig. 5. Agent-based asset management system.

The application of the agent-based approach, results in an integrated overall system that can also be introduced in an existing environment. Its properties are:

- The agent-based asset management system is a self-contained data processing system. The core functions of the automation system are not impaired.
- The heterogeneity of information sources has no impact on the entire asset management system, but is hidden by several agents.
- Changes in parts of the system have only local effects, new system parts or tasks can be added dynamically to the overall system.
- Every implementation of an asset management system shows a high degree of individualization, due to the particular environment. The use of agents facilitates the reuse of single system components or even partial systems.

The realization of the presented approach will be a significant enhancement towards the goal of an extensive and effective asset management system.

6. Conclusion

In this paper the opportunities as well as the advantages, of introducing of agent-based concepts into automation systems have been investigated. The focus was on legacy, contemporary automation systems used in industrial plants. Due to the limits

of current software concepts new approaches are needed to handle the increasing complexity in automation systems. It was argued that agents are well suited to solve the problems of contemporary automation systems and a basic approach for the introduction of a multi-agent system into existing systems was presented. Compared to current software concepts for automation systems, the merits of the presented approach are a transparent software structure, and a dynamic, adaptive application software.

For a successful application of the presented approach, the existing constraints regarding legacy automation system structures and applications have to be considered. The implementation of the approach is subject to work-in-progress. Therefore, technical difficulties have to be overcome. These difficulties arise due to the limits of existing communication structures that restrict the access to field devices and process control data. Also challenges have to be met when introducing a new orthogonal agent-based communication layer. These challenges concern e.g. the data consistency between automation system and multi-agent system and the coordination of both systems without affecting existing automation functions.

References

1. Jennings, N.R.: On agent-based software engineering. Artificial Intelligence 117, 2000
2. Lauber, Rudolf; Göhner, Peter: Prozessautomatisierung I, 3. Vollst. Überarb. Aufl. - 1999, Springer-Verlag Berlin - Heidelberg - New York, 1999
3. Scherf, B.; Haese, E. und Wenzek, H.R.: Feldbussysteme in der Praxis: ein Leitfaden für den Anwender. Springer-Verlag Berlin - Heidelberg - New York, 1999
4. Meyer, Dirk: Innovation leittechnischer Softwarestrukturen. Fachkonferenz "Verteilte Automation 2000" (VA2000), Magdeburg, 2000
5. Mchouikha, M. et. al.: Klassifikation und Bewertung von Beschreibungsmitteln für die Automatisierungstechnik. at - Automatisierungstechnik, Nr.46 (Heft12), 1998
6. Fay, A.: Methoden zur Unterstützung der Migration von Prozessleitsystem-Software. atp - Automatisierungstechnische Praxis, Nr.44 (Heft 6), 2002, S.39 - 44
7. Hesse, W. et. al.: Terminologie in der Softwaretechnik - Ein Begriffssystem für die Analyse und Modellierung von Anwendungssystemen. Teil 2: Tätigkeits- und ergebnisbezogene Elemente In: Informatik-Spektrum 17, pp. 96-105, Springer-Verlag 1994
8. Albrecht, H.; Meyer, D.: Ein Metamodell für den operativen Betrieb automatisierungs- und prozessleittechnischer Komponenten. at - Automatisierungstechnik, Heft 50, 3/2002
9. Tonshoff, H.; Woelk, P.-O.; Timm, I.; Herzog, O.: Flexible Process Planning and Production Control Using Co-operative Agent Systems. International Conference on Competitive Manufacturing (COMA '01), Stellenbosch, South Africa, 2001
10. Epple, U.: Agentensysteme in der Leittechnik. atp - Automatisierungstechnische Praxis, Nr.42 (Heft 8), 2000, S.45 - 51
11. Jennings, N.R.: An agent-based approach for building complex software systems. Communications of the ACM, 44 (4) 35-41, 2001

12. Döbel, U; Heidel, R: Strukturen künftiger verteilter leittechnischer Systeme am Beispiel der Feldtechnik. atp - Automatisierungs-technische Praxis, Heft 9 (42), 2001
13. Jennings, N.R. and Woolridge, M.J.: Agent Technology: Foundation, Applications, and Markets. Springer, New York, 1998
14. Kendall, E.: Role Modeling for Agent System Analysis, Design, and Implementation. First Internat. Symposium on Agent Systems and Applications (ASA'99), Palm Springs, 1999.
15. Nicklaus, E.; Zöller, M.: Anlagennahes Asset Management - Namur spricht Empfehlungen aus. atp - Automatisierungstechnische Praxis Messekompass, 2001
16. NAMUR-Recommendation: Requirements for Online Plant Asset Management Sytems, erstellt durch NAMUR-Arbeitskreis 2.7. "Asset Management", Leverkusen, 2001
17. Shen, W.; Norrie, D.: Agent-based systems for intelligent manufacturing: a state-of-the-art survey. Knowledge and Information Systems, 1(2), S. 129-156, 1999