Agent-based Concepts for Manufacturing Automation

Peter Göhner and Michael Weyrich

University of Stuttgart, Institute for Industrial Automation and Software Engineering,
Pfaffenwaldring 47, 70550 Stuttgart, Germany
{peter.goehner, michael.weyrich}@
ias.uni-stuttgart.de
http://www.ias.uni-stuttgart.de

Manufacturing automation is an industrial field of application in which agent-based concepts are of a high relevance. Flexibility in engineering, quick and easy adaptation and evaluation of the systems are key issues in the domain and can significantly be improved using the paradigm of multi-agent systems. The approach of agent technology is very suitable to conceive assistant systems or control architectures which can be used throughout the engineering and the operation of automated manufacturing systems. This article presents an overview of the research undertaken the past years which has yielded workable concepts for typical use cases of the industry.

Keywords: manufacturing automation, multi-agent systems in automation, software agents, agent-based planning, agent-based engineering, decentralized systems, test and operation of manufacturing

1 Introduction

Manufacturing automation has been a key driver to increasing productivity of factories for many years. So-called Smart Factories of the future might have capabilities to react to changes in products or product variants. These Smart factories can produce newly developed products quickly as they allow for an easy adjustment of the automated manufacturing equipment, which reduces time to market of new products.

Utilizing manufacturing automation provides high quality processes such as machining, assembly, packaging etc.

A key success factor of automated manufacturing systems is the capability to dynamically adjust to the requirements of production for new products or variants and production volumes. However manually adjusting the automation is time consuming if conducted by human operators and results in high costs.

New architectures of manufacturing automation systems are required to implement a Smart Factory. Though there is a wide variety of automation research, which can be deployed, there is a need for efficient automation architectures in engineering, commissioning and test processes.

One of the promising technologies is the multi-agent Technology which is becoming increasingly attractive for the software and hardware developments in manufacturing automation. Multi-agent systems promise to solve architectural problems that are difficult or impossible for monolithic systems to cope with. Agents are an approach for intelligence and decentralized problem solving leading to superior system architectures. In comparison to the conventional control methods used in automation, the multi-agent technology can be more effective, flexible and easier to adjust.

The concept of multi-agent systems dates back to about four decades and many approaches have been made to deploy the technology in automation of manufacturing. First traces of the concept of agent systems were outlined in the Science fiction tales on Artificial Intelligence around the 1960s. A prime example is “A Space Odyssey” written by Arthur C. Clarke with concise visions of men-machine interaction, intelligence and sentience.

Research was conducted from the 1980s onwards, in the newly rising field of computer science and artificial intelligence where multi-agent systems were viewed as concepts for problem solving.

The past 20 years has seen agents and multi-agent systems in research yielding solutions which are deployed in industrial applications such as supply chain management, supervisory control of plants or manufacturing automation. Due to the broad nature of the agent concept, there are many interpretations of what exactly a multi-agent system is and what the core questions are.

Agent technology evolves from the relevant research conferences, technical committees of IEEE and VDI/VDE and multiple university textbooks that shape the domain.

1.1 The Concepts of Agent Technology

The agents act independently within their scope of actions to pursue prescribed goals. They can interact with each other and cooperate through negotiations in order to achieve these goals.

According to the VDI/VDE Guideline 2653 [12] the terminology is as following: “An agent is an encapsulated (hardware / software) entity with specified objectives. An agent endeavors to reach these objectives through its autonomous behavior, in interaction with its environment and with other agents. A multi-agent system is the interaction of a set of agents to fulfil one or more tasks. Runtime environments and platforms for technical agents are a possible basis for realizing multi-agent systems. However, they themselves are not multi-agent systems”.

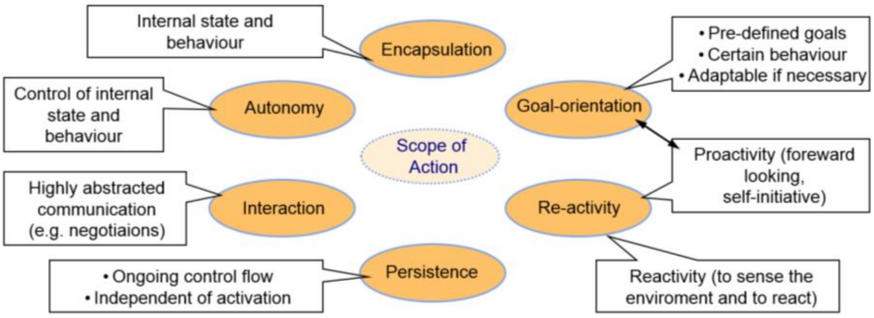

Fig. 1: Attributes of multi-agent systems

The behavior of a multi-agent system is determined at run-time by the dynamic co-operation of the individual agents [1]. The systems are often deploying different types of agents and those systems are referred to as multi-agent systems.

Fig. 1 presents the attributes of a multi-agent system which can be considered as an autonomous software entity.

According to the VDI/VDE Guideline 2653 [12] the properties of a multi-agent system are:

"**Scope of action:** The scope of action limits the application of the capabilities of an agent. Accordingly, the degree of flexibility of a technical agent is determined by a given scope of action.

Autonomy: Attribute of an agent which allows it to control its internal state and its behavior. Through autonomy, an agent acts and makes decisions based on its local knowledge and activities.

Encapsulation: Attribute of an agent which allows it to control the access to its individual constituents which are not visible externally. State, behavior, strategies and objective represent the individual constituents that are encapsulated within an agent.

Goal-orientation: Attribute of an agent which allows it to orientate its behavior to one or more objectives, which it attempts to accomplish.

Reactivity: The capability of an agent to sense the environment and to react accordingly.

Persistence: The capability of an agent to keep its internal state during its lifecycle.

Interaction: The capability of an agent to interact with other agents, in order to accomplish individual objectives or to manage dependencies among each other. The basis for the interactions among agents is a shared semantic, an underlying organizational context and a common terminology model, which are together referred to as ontology."

The basic concepts of agents are embossed by the autonomy of the agent and can encapsulate itself. Furthermore an agent retains its internal state during its lifecycle.

The communication schema of agents is sophisticated. They can for example negotiate with each other and are interactive.

In addition, agents are persistent as they have an ongoing control flow and are active and reactive. They can proactively demonstrate self-initiative, sense their environment and react to it in a goal oriented manner.

Agents can act according to predefined goals and adapt appropriately.

The systematic distribution of functionalities and knowledge in autonomous units controlled by agents, leads to a low structural coupling between system elements and multi-agent systems.

1.2 Architectures of Multi-Agent Systems

Agents require interaction and communication with other agents. Therefore they require knowledge about the existence of other agents in their neighborhood; they need a communication infrastructure, a messaging protocol and some sort of a dictionary of services provided by all agents.

There are different ways on how such a structure could be implemented in a multi-agent system for industrial automation. The commonly used structure is the FIPA reference model described in Fig. 2, a standard architecture of multi-agents systems.

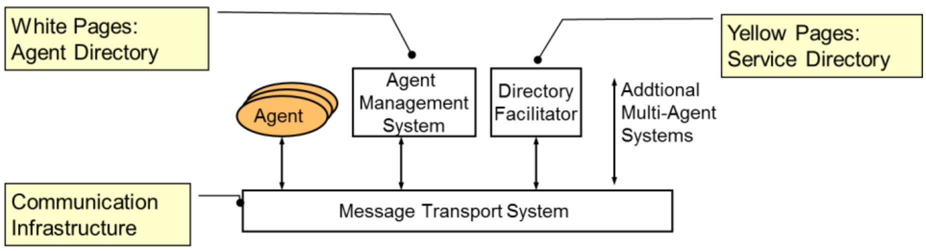

Fig. 2: Reference model of agent systems

This standard was conceived by the FIPA, the Foundation of Intelligent Physical Agents, Geneva, Switzerland in 1996 to 2002 (see www.fipa.org), this standard is heavily deployed in the community. An overview to the application impact and agent technologies in use can be found in [14].

According to that reference model, multi-agent systems have an interface to the message transport system to communicate with other agents. The communication infrastructure of the message transport system is standardized. With the help of an agent management system, agents are able to locate other agents to interact. For that purpose agents are listed in so-called “white pages” with their addresses. This administration is done by the agents themselves. They register or deregister themselves in the “white pages” depending on whether they are activated or deactivated.

The so-called directory facilitator provides information of services offered by the agents similar to the “yellow pages”.
The communication infrastructure is standardized for the Message Transport System which aids agents in interacting among themselves.
The architecture of the multi-agent system can be based on the FIPA standard or any other concept as there is a lot of flexibility in the definition of the architecture due to the inherent flexibility of agent technology. However, the major weakness of the agent technology lies in the multiple task control. The runtime behavior is partially unpredictable due to the parallel nature of the executed tasks. This means that the behavior of a multi-agent system cannot be determined a priori as the decision making process is distributed and a result of the interaction of autonomous agents. In manufacturing automation this uncertainty is not acceptable for many industrial applications for reasons of safety and security. For such applications multi-agent systems are used in the engineering phase to conceive a control paradigm which is thereafter analyzed and implemented using conventional technologies.

2 Areas of Application in Manufacturing Automation

The sector of manufacturing automation is huge and entails a large number of application domains in various industries. This means agent technology is not limited to one field but has multiple means of usages.

Basically there are three identifiable areas:

- the engineering and planning of automated systems, focusing on the creation of the manufacturing systems;
- the manufacturing operations where an automation system produces goods;
- the phase of evaluation of systems in which the functions of an automated manufacturing system are tested.

These areas are elucidated below, highlighting the various cases of agent technology in manufacturing automation.

2.1 Engineering and Planning of Automated Systems assisted by Agents

Manufacturing systems of today are exposed to volatile markets with changing requirements for the products produced. Automatic manufacturing systems need to be created quickly with limited effort or retrofitted based on existing installations.
The required flexibility is attainable with the help of modular design, adaptive automation and intelligent planning systems. Methodologies of engineering and planning are required to accommodate complexity and flexibility.

An approach to achieving flexible engineering and planning processes is a structured approach with processes which can be decoupled as much as possible and thereafter, executed concurrently.

The agent-based concept allows mechanisms to improve the planning process and provide tools to make it faster, more efficient and flexible due to the decentralized nature of the agent paradigm.
This paradigm is based on the concept of agents representing sub-systems or parts to support the planning of an automated system. Each agent has encapsulated engineering knowledge, follows planning goals and negotiates with the neighboring component agents, thereby automatically generating planning proposals.

In the following, some examples for agent-based engineering systems are given:

- Rauscher et al [2] uses a multi-agent system in the early design phase of a mechatronic design. Each model is represented by an agent holding specific knowledge about that model. So-called consistency rules are expressed in ontologies where attributes between the models during the design are monitored and violations alerted.
- Kratzer et al. [3] presents a similar approach to support the design of mechanic components by validating the current design against requirements and boundary conditions. Non-conformance designs are revealed and solutions are proposed to the designers.
- Tompkin [6] describes the deployment of multi-agent systems in a knowledge-based engineering process resulting in the reduction of the planning complexity.

The value of agent technology in engineering is to assist the expert by taking over the tedious and time consuming steps of the planning procedure as well as computer aided proposals of solutions or alternatives for decision making.

Example: Engineering based on Sub-systems supported by Agents

A concept of designing complex closed-loop control systems is shown in Figure 3. The independent design of decentralized open-loop control systems with multiple inputs and outputs and strong linkages between them is difficult to execute. This agent-based approach allows for the creation of simple controllers for the sub-processes. Each of these controllers is part of an agent with its own objectives and a set of actions. The agents interact with each other to get an optimal solution for the desired feedback-loop control problem [2].

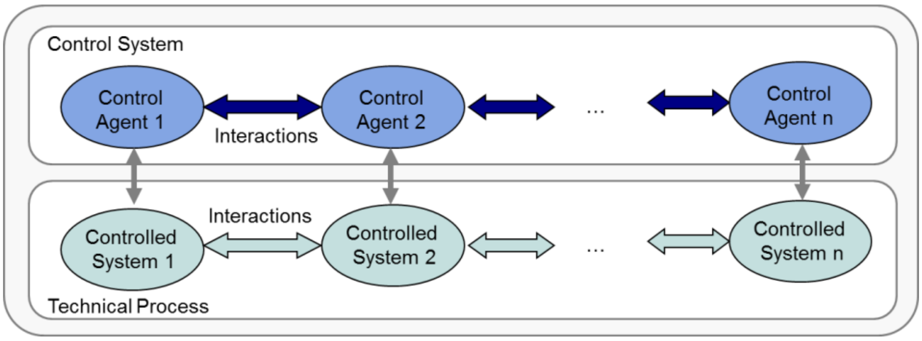

Fig. 3: Example of an agent-based design of a complex feedback control system

Consider a warehouse in which different parcels have to be stored. The manufacturing automation system has to take care of various system elements such as the parcels, conveyor belts, cages, separation devices, storage units, operator units etc.

A conventional logic control system would require a complex central program to control all the individual units. An approach based on agent technology would look into required services and define a “service agent” and “coordination agents”. For instance a service agent would be defined for each of the major systems functions of a warehouse which are: initialization, store goods, retrieve goods, separate parts, check occupancy.

The resulting structure of the agent system is based on those services and decouples the overall control and reduces the complexity of the software.

Example of Agent based Product Design using Components

Consider a scenario where the engineering is based on a set of pre-engineered components or modules. Here the blue-print of the manufacturing automation system can be decomposed. Each of these components or modules can be represented by an agent in a computer aided planning system which administrate the design kit.

The agents can utilize their autonomy acting as virtual surrogates to form the networks of modules dynamically as well as the overall system. Once certain engineering or planning rules are implemented into the knowledge base of the system, the agents can take over the planning partially, by interacting between each other, selecting, configuring and connecting system components by proposing new alternatives and solutions.

Figure 4 illustrates an agent-based concept of an automated consistency check in a computer aided design process. Each part of the assembly which has to be designed is represented by one so-called “part agent”. This agent acquires any changes made to the part by its designer and shares this information with other agents.

These “device agents” identify newly designed parts and their dependencies on each other. Thereafter the so-called “aspect agents” checks for consistency with regards to issues, such as costs, geometry or materials. If the consistency check is declined, potential alternatives will be investigated and presented to the designer.

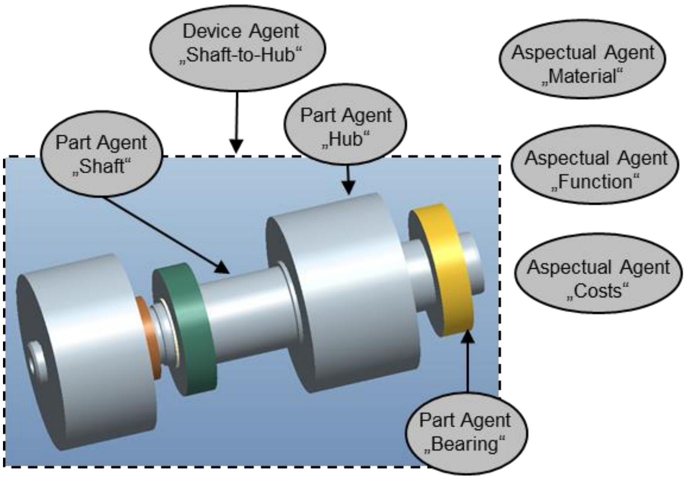

Fig. 4: Design support based on agent technology

The deployment of predefined rules and the modeling of resources allow a bottom-up planning based on dialog to form a design. See [3] for further reading.

More complex design tasks such as mechatronic systems consisting of mechanics, electronics and software need an automated consistency check. This is of particular interest in order to define changes, identify the impact of changes and automatically check the design.

2.2 Agents in Manufacturing Operation

During the operation phase, the automated manufacturing systems produce goods and materials. There are various use-cases which can typically benefit from the agent technology e.g. the process planning over the course of time can be optimized by agents making the system more efficient. This becomes even more relevant once unforeseen events such as failure of machine or shortage of supply materials occur and the plan needs to be updated on the fly during production.

Agents can also assist in retrofitting during production e.g. to manufacture a new slightly different product.

Example of Agent based Scheduling

In Figure 5 the concept of an agent-based scheduling for manufacturing systems is displayed. In this example the overall system can be optimized using different types of agents. See also [8].

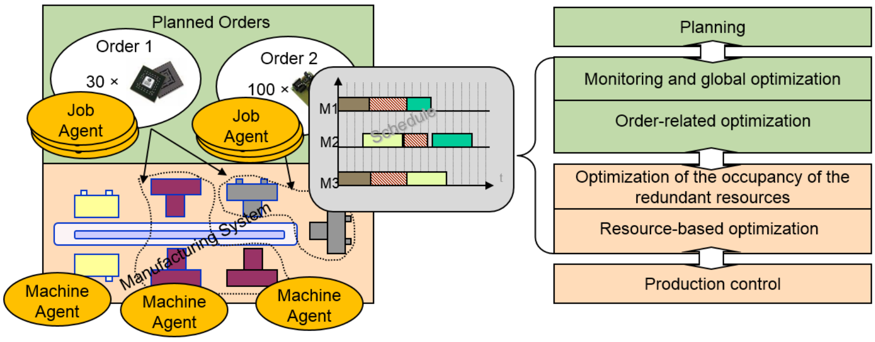

Fig. 5: Production optimization with multiple goal optimizations

For this purpose each order is represented by a “job agent”. Job agents are appointed to orders and represent them. These agents try to optimize their order. They negotiate with “machine agents”, which stand for an individual machine of the manufacturing system. The optimization target of a machine agent is to achieve the occupancy of redundant resources.

The decomposition of an order into machine jobs and the scheduling of the individual machines are done by means of the agents.

With the help of this concept, the schedule can be identified using a multilevel optimization structure taking the resource utilization and order constrains into conclusion. This approach is particularly flexible once unforeseen events occur, such as machine failures or supply shortage. The manufacturing system can automatically and dynamically adapt to a revised schedule.

Example of Agent for Self-management of Automation Systems

Manufacturing Automation systems become more flexible once they entail Self-X functionality. Self-healing, self-optimization or self-configuration provides functionality to reconfigure manufacturing during operation without modifications or retrofit. Agents implement the execution and coordination of such self-x functions.

Self-managing of automated systems consist of an operative automated system, a self-management interface and a self-management system as displayed in Fig. 6.

The self-management interface captures information out of the automated system.

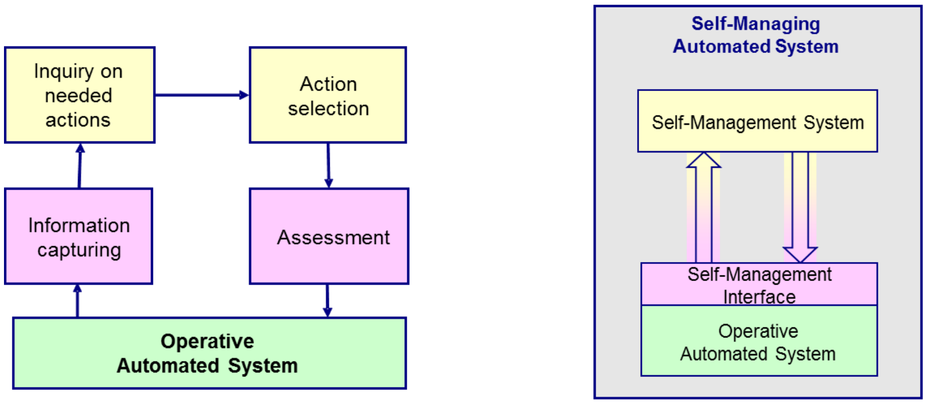

Fig. 6: Control loop in the self-management of automated systems

Often error occurs in an automated system causing a total system breakdown. Self-managing automated systems have the ability to react to these errors and try to handle them without losing the desired functionality of the system.

The self-management system uses the agent paradigm and ontologies to determine automatically if there are any actions necessary and which of them should be executed. These actions will be forwarded to the self-management Interface and then executed in the automated system.

The agent-based concept allows the autonomous execution of self-recovery and self-configuration processes in the automated system. The idea is to encapsulate the self-management functionality inside agents. Those agents are autonomous and can interact with the automation system and the operator to achieve a certain functional task.

These self-x functionalities will be provided through the agent-based self-management system where traceability and reproducibility are important requirements.

The self-management functionality depends strongly on the specific process of manufacturing automation. However, the roles of the agents can be generically defined.

A set of agents, the “self-management agents” takes charge of each individual function. Those agents are very specific to the details of the realization and focus on the individual process functionality which is to be self-managed. The agent captures the information, analyses it using a rule based approach and triggers a specific action. Both analysis as well as the decision for specific action can be supported by knowledge based processing approaches such as fuzzy logic. Additionally a “coordination agent” is deployed to synchronize the actions of the self-management agents.

Consider an elevator which has redundant sensors to detect the floor. The position at floor is detected with multiple sensors, say four sensors. If one of the sensors fails during operation the elevator would detect that issue using an internal model of the elevator. The self-management agent would identify the requirement to override the signal of the faulty sensor and take appropriate action to continue the operation as three out of four sensors are still operational. The self-management agent however would not directly interfere with the programmable logic controller (PLC), but would communicate with the coordination agent, which double-checks the self-healing case and launches the right reconfiguration program sequence in the PLC.

The advantage of the systems architecture of the multi-agent system is the structure of the control loop and the stepwise course of action. The system inquires the appropriated action after the capturing of information. Once the course of potential action is identified the system assess and decides for one course of action. Following this concept the self-management and self-healing follows a clear architecture conceived due to agent technology. See [13] for further reading.

2.3 Agents for Evaluation and Test of Automated Manufacturing Systems

One of the major parts of the creation process of an automation system is the evaluation or test phase. It is essential to prove the compliance of a manufacturing automation system with the quality requirements of the engineering and planning process.

The planning of the system evaluation, the specification, execution and test is usually a process which accompanies the whole development. System evaluation and testing are embedded in the system's lifecycle.

During its lifetime, a system is usually altered or retrofitted to adapt to new requirements and boundary conditions. Tests on the system have to be conducted regularly as any modification adds to the risk of introducing faults.

Malz et al. [4, 5] use a multi-agent system to support the evaluation and in particular the test management of software. The agents prioritize test cases by evaluating meta-information like change and fault histories of the tested software system. Those agents assign the available test resources to the test cases and trigger the execution.

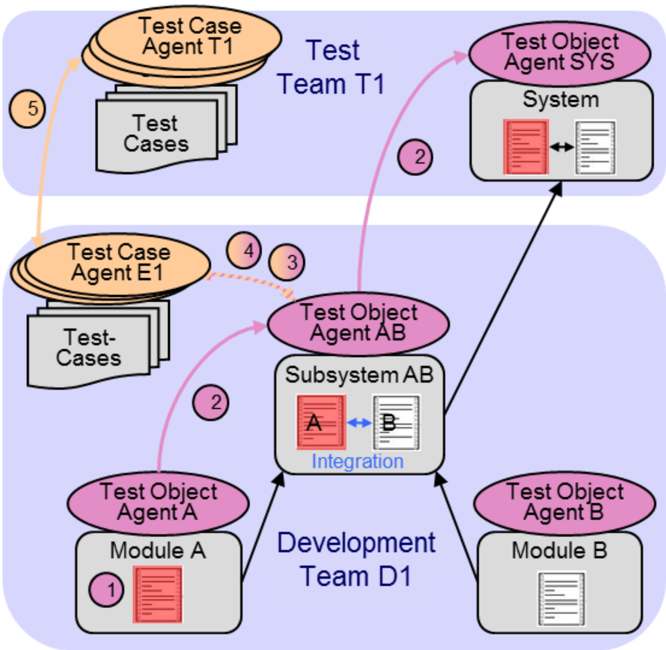

Fig. 7: Interaction of agents assigned to test objects associates to modules and sub-systems

The concept of an agent-based test management system is presented in Figure 7. System components of the automated manufacturing system especially altered software, has to be tested before it can be utilized.

All necessary test cases have to be specified before execution. However, the duration to run all those test cases taking all variants, input signal combinations, resources etc. into considering is typically short and not all cases can be applied in each test period. Hence, an optimization plan is necessary, as the test cases should be executed according to the relevance and importance keeping the timing to evaluate.

A “test object agent” represents a single software module of the manufacturing automation system. This agent considers information like how many changes have been made in this module since last testing and how critical and relevant the module is to the overall system.

A “test case agent” represents a single test case: It coordinates the resources of this test case and cooperates with other agents to prioritize all test cases and to achieve an optimal test plan and usage of test resources. The result of the test management system is a prioritized list of all test cases with the calculated priority of each test case.

Test case selection techniques reduce the test suites by identifying only relevant test cases, for example based on changes of the source code. Prioritization techniques go one step further and order the test cases by their expected benefit for the test. Unlike test case selection, a test run that is executed based on prioritized test cases may be interrupted at any time but remaining the maximal benefit. Every test case prioritization technique can also be seen as a selection technique by skipping test cases, which have a lower priority and impact than a predefined threshold value. Test case prioritization techniques are described in [4].

3 Conclusion

Currently, the importance of agent-based approaches for developing, operating and evaluation / test of industrial systems are increasing. Agent technology has multiple advantages especially in areas of software development for automated manufacturing systems. For instance decentralized systems, which are distributed over a location, modular and exposed to changes of the system structure, are particularly relevant to the application of agent technology for two reasons:

Firstly, not all information of the future requirements is available and known at the time of engineering and planning of the automated manufacturing system.

Secondly, in order to be flexible and effective it has to support dynamic changes of the system to ad hoc network sub-systems for maximum optimization.

This paper demonstrates how agent technology can support the manufacturing automation in the fields of:

- Engineering and planning of manufacturing automation systems
- Operation of manufacturing systems and
- Evaluation and testing of system.

These application areas demonstrate the importance of agent technology and concepts providing continuous support during the whole lifecycle of the systems.

In the presented examples, agents demonstrated the ability to be autonomous software units, which are goal oriented, active, cooperative, flexible and adaptive.

This makes the agent paradigm a valuable tool for the analysis, engineering and implementation of complex manufacturing automation systems.

Over the past decades many achievements have been made to consolidate the various concepts of multi-agent systems. Despite the various research approaches, terminology and definitions are wide synchronized and specified. The multi-agent system design methodology has been extensively discussed in the community and set methods and tools are available which eases the use for non-research experts in the industrial field. Additionally various agent development platforms are in existence and criterions for the selection of individual solutions have been intensively discussed in the community.

A lot of research has been done to demonstrate and understand the usage of agent Technology use cases of industrial automation of production systems, energy management modular production plant design and logistics. Some of those cases could be reviewed in this paper. More examples are available in the literature which demonstrate the ability of the approach as for instance in [12] in which ten use cases are analyzed with the goal to demonstrate the potential of agent technology. However, considering the state-of-the-art and hard facts of industry implemented there is a very limited amount of multi-agent system implementation compared to conventional approaches for industrial control.

The authors believe that this paradigm will be a central concept for many future developments of flexible and decentralized networks based on autonomous and cooperation units in manufacturing automation. Many of the future cyber-physical system developments in manufacturing automation might not utilize the agent platforms of the community nor explicitly mention the agent technology but be inspired by the design concepts and methodology. This way the complexity of systems can be coped effectively making the architecture more flexibility and easier to handle.

4 References

1. Jennings, N. R.: On agent-based software engineering, Artificial Intelligence 117, 277_296, (2000)
2. Rauscher, M., Göhner, P.: Agent-based consistency check in early mechatronic design phase. In: DS 75-9: Proceedings of the 19th Intern. Conference on Engineering Design (ICED13), Design for Harmonies, Vol. 9: Design Methods and Tools, 389_396, Seoul, Korea (2013)
3. Kratzer, M., Rauscher, M., Binz, H., Goehner, P.: An agent-based system for supporting design engineers in the embodiment design phase. In Proceedings of the International Conference on Engineering Design, ICED (2011)
4. Malz, C. and Jazdi, N.: Agent-based test management for software system test. in IEEE International Conference on Automation, Quality and Testing, Robotics (AQTR), pp. 1_6 (2010)
5. Malz, C., Jazdi, N., and Göhner, P., “Prioritization of Test Cases Using Software Agents and Fuzzy Logic,” in 2012 IEEE Fifth International Conference on Software Testing, Verification and Validation (ICST), 483_486 (2012)
6. Tompkin, J.A., White, J.A., Bozer, Y. A., Frazelle, E.H., Tanchoco, J.M.A and Trevino, J.: Facilities Planning, Wiley, New York (1996)
7. Reichl, H. and Wolf. J., Things that think. TU-Berlin- Forschung aktuell, 1887_1898 (2001)
8. Pech, S.: Software Agents in Industrial Automation Systems. IEEE Software Vol. 30, No. 3, 20_24 (2013)
9. Elbaum, S., Malishevsky, A., Rothermel, G.: Test case prioritization: a family of empirical studies. IEEE Transactions on Software Engineering 28(2), 159_182 (2002)
10. Li, Z., Harman, M., Hierons, R.M.: Search algorithms for regression test case prioritization. IEEE Transactions on Software Engineering 33(4), 225_237 (2007)
11. Rothermel, G., Untch, R., Chengyun Chu, Harrold, M.: Prioritizing test cases for regression testing. IEEE Transactions on Software Engineering 27(10), 929_948 (2001)
12. VDI/VDE 2653 Sheet 1, 2 and 3: multi-agent systems in industrial automation: Fundamentals, Development and Application. Berlin: Beuth Verlag (2010)
13. Mubarak, H., Göhner, P.: An agent-oriented Approach for Self-Management of Industrial Automation Systems, INDIN 2010 (2010)
14. Müller, J., Fischer, K.: Application Impact of Multi-Agent Systems and Technologies: A Survey. In Onn Shehory, Arnon Sturm, eds., Agent-Oriented Software Engineering: Reflections on Architectures, Methodologies, Languages, and Frameworks, 27_53 (2014)