3 Flexibility and Re-configurability in Manufacturing by Means of Distributed Automation Systems – an Overview

Daniela Wünsch[1], Arndt Lüder[2] and Michael Heinze[3]

[1] SAP Research CEC Dresden, SAP AG, Chemnitzer Straße 48, 01187 Dresden (Germany)
e-mail: daniela.wuensch@sap.com

[2] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg (Germany), e-mail: arndt.lueder@ovgu.de

[3] Otto-von-Guericke University of Magdeburg, Faculty for Mechanical Engineering, Institute for Ergonomics, Manufacturing Systems, and Automation, Universitätsplatz 2, 39106 Magdeburg (Germany), e-mail: michael.heinze@ovgu.de

Abstract The fast changing bordering conditions for industrial manufacturing systems have raised the need to increase manufacturing system flexibility regarding different types of flexibility. To enable this enhanced flexibility, manufacturing control systems must be changed resulting in new challenges which have to be tackled by management and engineers of the affected companies. In parallel, within information sciences new paradigms for structuring and implementing software systems must be developed which are also applicable to design and implementation of control architectures. This paper deals with the applicability of these new paradigms for structuring and implementing software systems to address recent challenges within the manufacturing industry. Therefore, the paradigms and challenges are described and mapped to each other.

3.1 Introduction

Fast changing economical conditions force companies based on production systems to reconsider their business models and reengineer production systems. Their basic drivers of change are

- The changing market conditions on nearly all sides including increasing customer power with increasing varieties of customer requirements regarding product quality and delivery,
- Increasing technological and technical production possibilities and useable production processes for comparable products with different economical bordering conditions,
- Fast changing raw material market conditions with sometimes highly volatile material costs, and
- Changing legal requirements such as environmental protection laws and labour laws.

All these needs drive companies to rethink their competitive advantages [1]. The main result of this rethinking process is the interest of companies in increasing their competitiveness by increasing their flexibility regarding attainable product features, useable technologies, and exploitable production resources (Fig. 3.1, [2]), and their adaptability to changing expectations regarding company embedding conditions.

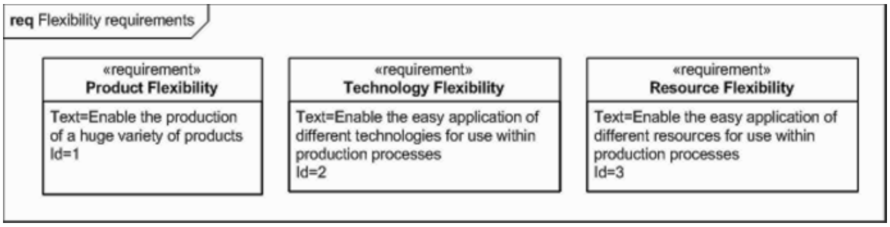
req Flexibility requirements
«requirement» Product Flexibility
Text=Enable the production of a huge variety of products
Id=1
«requirement» Technology Flexibility
Text=Enable the easy application of different technologies for use within production processes
Id=2
«requirement» Resource Flexibility
Text=Enable the easy application of different resources for use within production processes
Id=3

Fig. 3.1 Flexibility requirements

Beneath the flexibility and adaptability companies are being forced to adapt cost cutting measures within all its activities. This affects the consumption of consumables and materials in the same way as the engineering process and the consumption of noneconomic objects like nature and air.

But to reach this flexibility, adaptability, and economic resource consumption totally new technologies and architectures on all levels of control are required. Several technologies and architectures have been considered in recent years all of which aim to increase special flexibility characteristics ranging from distributed control architectures, plug-and-participate technologies, Web services, virtualization, and much more. The application of most of these technologies signifies an important step forward towards flexibility, adaptability, and economic resource consumption. But they are not necessarily compatible with each other (sometimes the application of one technology contradicts the application of another). This results in inefficient structures of the complete control pyramid (each part has its own optimal solution but the combination results in huge problems), improper system behaviour (violation of temporal conditions, wrong interpretation of data, etc.), and, in the best case, suboptimal usability (doubling of data integration, bad business processes, etc.). Nevertheless, the issue of basic architectural structures and basic technological conditions enforcing a wide-ranging step forward in the direction of flexibility, adaptability, and economic resource consumption has still not yet been considered.

In this chapter at first we first intend to briefly introduce the three basic technology paradigms of object orientation, service orientation, and agent orientation that are the current candidates for improving manufacturing system control structures and architectures. Then, we highlight a set of recent production challenges occurring in line with the intended increase of production system flexibility, adaptability, and economic resource consumption. On this basis we want to evaluate the applicability of the paradigms as a response to the production challenges.

3.2 Current Technology Paradigms

Among the most recent technology paradigms applicable to addressing the challenges to be described are object orientation, service-oriented architectures, and agent-oriented architectures.

Object orientation (OO) was developed in the area of software engineering in the late 1980s and early 1990s [3]. It is a structuring and behaviour paradigm for systems underlying characteristics, knowledge, and rights of entities within the system. It was developed to provide means for modeling, analyzing, and implementing software systems resulting in the model sets of UML [4] and SysML [5], which can be used as a description basis for OO.

Since OO is a very powerful paradigm that is not focused on software design, it has been applied very quickly in other domains as well including control system design and implementation.

The main characteristics of OO is the definition of types of objects with a specification of possible object data, admissible object behavior, and usable object interfaces (visibility of data and behavior) and the definition of different types of dependencies among object types. Based on these types objects can be instantiated.

Thus, OO provides capabilities to inherit structures and behavior between different types of objects, to encapsulate structures and behavior, and to apply different objects without direct knowledge of their internal structure and behavior in a similar way. The basic concepts are also shown in Fig. 3.2. Detailed information can be found in Booch [6].

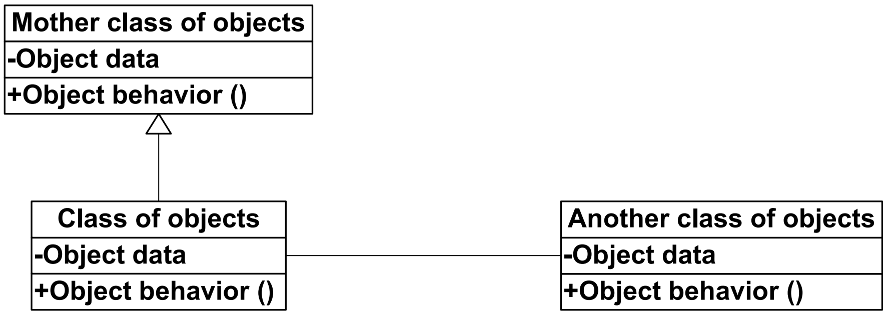

Fig. 3.2 Basic object orientation concepts

Within manufacturing system control OO can be easily exploited within the design and implementation of control applications by identifying relevant control entities and their control-relevant behavior [7]. Examples are the modeling of mechatronical units exploiting the hierarchy of objects with inheritance relations to design more detailed units or the analysis of dependencies within distributed control systems exploiting object dependencies [8, 9].

Based on the OO paradigm and riding on the wave of powerful IT devices, the Service-Oriented Architecture (SOA) paradigm has been developed. It is based on the ideas of Sculte and Natis from 1996 [10]. Nevertheless, usually SOA is defined as a structuring and behavior paradigm providing overall system functionality by exploiting the local functionalities of distributed entities in a coordinated way [11].

The main characteristics of SOA are the definitions of a service provided by a system entity and the rules by which this service can be exploited by other system entities (Fig. 3.3).

The provided service has to be accessible within a network of entities using standardized service interfaces. Thus, detailed knowledge about the internal behavior of the service is not required for the service user. In addition, the service implementation has to be independent from the service application. Usually, services are registered with some sort of yellow page service. Thus, services can be found and accessed at runtime of the system.

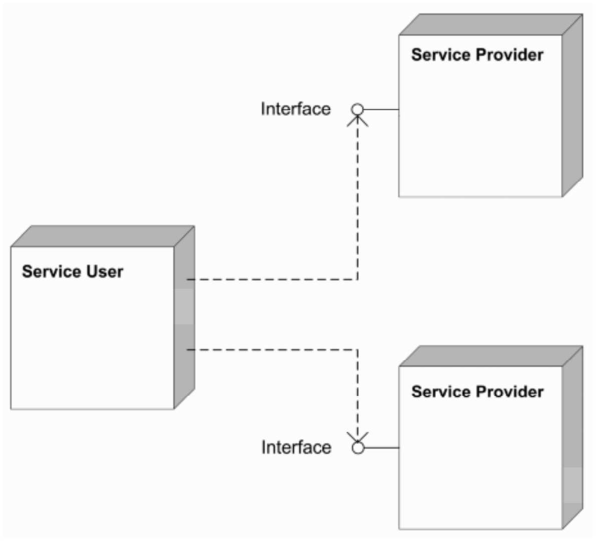

Fig. 3.3 Basic structure of SOA

The implementation of an SOA can be based on OO. Each service provider and each service user can be an object and the implementation and access to services can be based on OO mechanisms.

The use of SOA for controlling a system became relevant in recent years. Examples for its application are Web-service-based systems for device configuration [12] or Web-service-based interaction among companies [13].

Similar to SOA, Agent-Oriented Architectures (AOA) are based on OO paradigms. The main foundation of an AOA is the term of an agent. Despite various definitions of the term agent, agents are considered as independently acting entities with a dedicated environment model, agent internal aims, and the ability to act purposefully in order to reach the aim for the given behavior of the environment.

Within manufacturing system control, usually, multi agent systems (MAS) are used to enable the distribution of the control decision process among autonomous but cooperatively acting entities. There are several examples of AOA-based systems for control as given in [14, 15].

The main benefit of the application of AOA within control is the possibility to define appropriate encapsulations of control decision process parts and the explicit modeling and implementation of its interactions, as represented later within this book.

3.2.1 General Technology Application Ideas

The general idea of the application of OO, SOA, and AOA concepts in control is based on the application of mechatronical units within the design, implementation, and use of production systems. The main idea here is to compose production systems by mechatronical units in a hierarchical way [16, 17].

A mechatronical unit itself is a functionality oriented combination of mechanical, electrical, and control-related components providing functionalities to an overall system. It can be divided into a physical layer and a logical layer, as shown in Fig. 3.4, where the physical layer is responsible for the physical execution of activities necessary for the production process and the logical layer is responsible for the control of these activities.

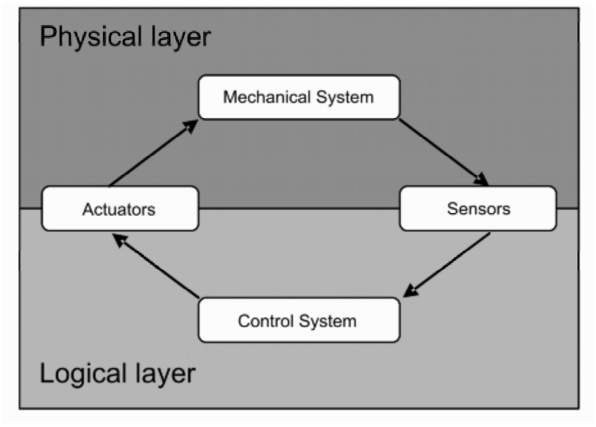

Fig. 3.4 *General structure of mechatronic systems*

The modeling of the behavior of a mechatronical unit within the design process as well as the implementation of its control part can be based on OO mechanisms. Here the object describing a mechatronical unit will provide production services or production support services to its environment. Thus, the use of OO encapsulation mechanisms enables the hiding of internal unit behavior resulting in a kind of white box behavior representation, the use of OO inheritance mechanisms enables incremental behavior enrichment of mechatronical units, and the use of OO polymorphism mechanisms enables similar usage of different objects within the control system.

In addition, SOA and AOA mechanisms can be used for the implementation of agile production systems consisting of mechatronical units. Therefore, SOA mechanisms can be used to enable plug-and-participate behavior of mechatronical units and dynamic binding of them within production, configuration, maintenance, and other processes. AOA can be used to model and implement self-aware and proactive mechatronical units acting independently but cooperatively within a system to fulfill its part of a common control system.

To enable the use of the mechatronical units within the production system additionally the order control needs to be modeled in an OO-oriented way by objects able to use the services and interfaces provided by the OO objects implementing/modeling the mechatronical units. These order objects will encapsulate order data and order execution control behavior. They will ensure application of SOA services or agent functionalities provided by mechatronical units (Fig. 3.5).

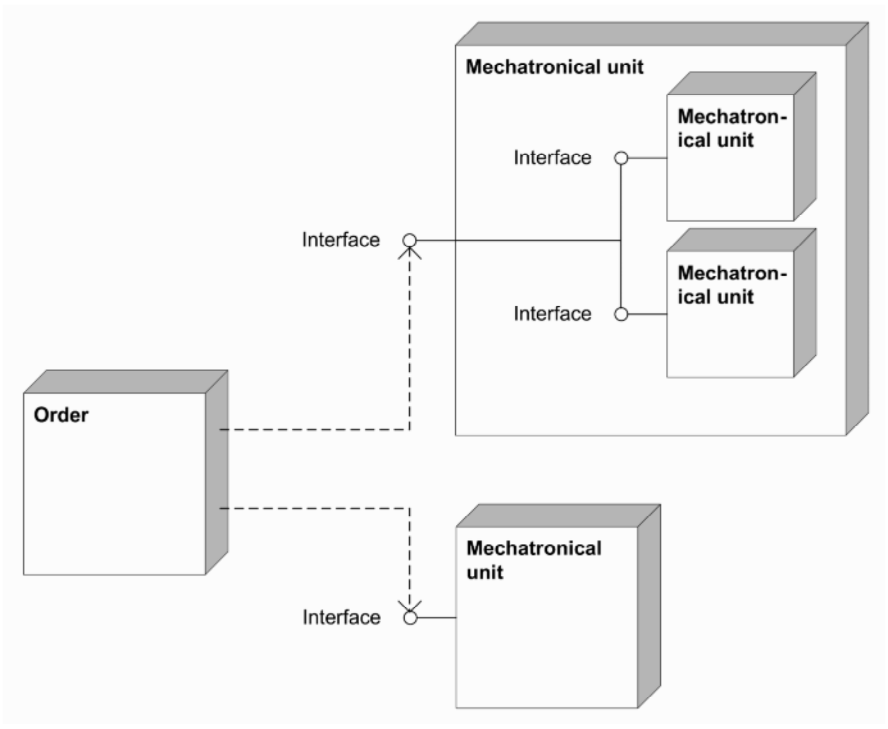

Fig. 3.5 SOA-based implementation structure of mechatronical units

3.3 Challenges in Production Control

As initially mentioned, production system control currently not provide efficient structures, proper behavior, optimal usability, and optimal integration in the overall environment. This results from different contradicting procedures. On the one hand most recent technologies are intended to be used in control systems without proper consideration of their applicability. On the other hand, the real needs of production system control are not sufficiently investigated. Finally, the technological, architectural, economic, and customer-oriented conditions of production systems, along with the conditions for production control systems, change very fast.

In the following discussion we will describe some of the recent challenges identified in Fig. 3.6 emerging from the latest developments and sketch how they can be addressed by the aforementioned basic technology paradigms. For a more detailed investigation of these challenges we refer the reader to [18].

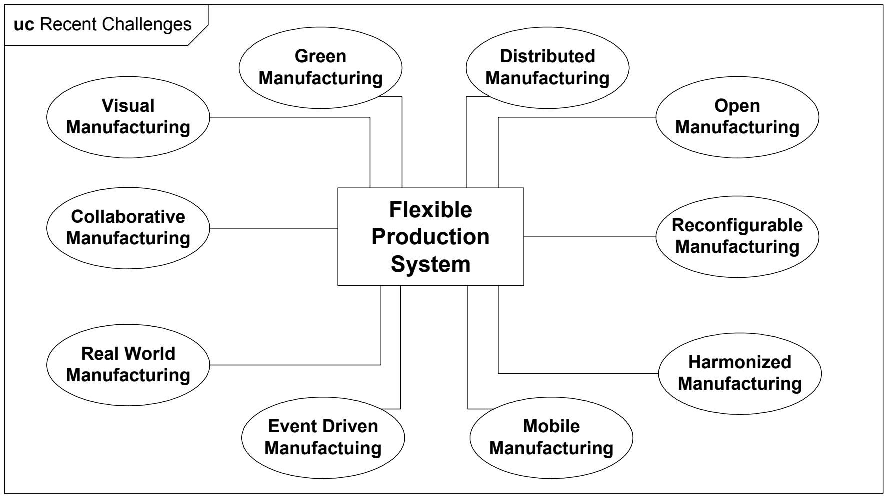

Fig. 3.6 Recent challenges

3.3.1 Visual Manufacturing

Many currently used systems within production system control (production confirmation applications, report generators, etc.) still do not consider the actual needs of the user. Often the problem is one of information overload or time-consuming manual processes to handle simple tasks.

The needs of users (in this case control engineers) requiring a proper system control interface must be met. Bad usability of the interface reflects poorly on the entire IT system. While usability problems will be overcome with time and require more iterative development work rather than groundbreaking research, other aspects of the user interface present more interesting challenges. Advanced visualization of the logistics processes within the plant is one opportunity for innovation. With today's IT support it is often difficult to get an integrated and appropriately filtered view on the current status of the shop floor as needed in recent complex and agile production systems. Low-level control systems like Human-Machine Interface (HMI), SCADA visualize technical parameters and "business" data can be retrieved from Enterprise Resource Planning (ERP) systems, but the "big picture" must be tediously constructed by looking at different sources of data. This big picture may show the plant layout and highlight the status of resources (e.g. broken machine, in repair) as well as logistical data (e.g., Work in Progress (WIP) location, missing material/personnel warnings) (Fig. 3.7). The integrated view for resource data and logistical data must have strong filtering and abstraction mechanisms so as to not overwhelm the user accompanied by a mechanism to highlight interdependencies with suppliers and subcontractors, to navigate from the past (analyze past production) through the present (monitor current operations) to the future (simulate and visualize likely scenarios), and to represent different levels of detail.

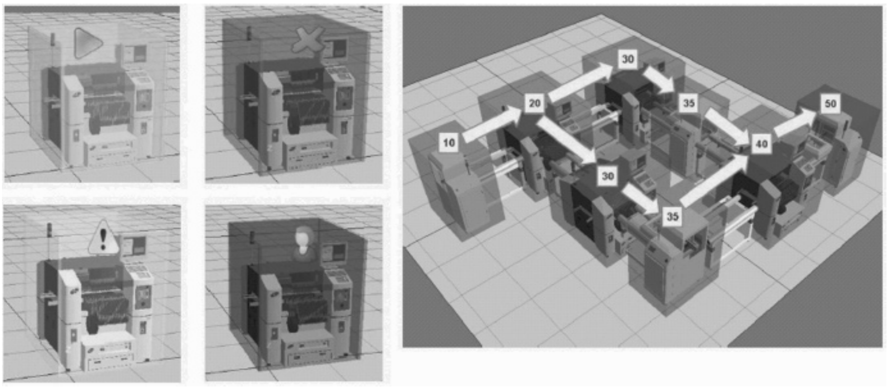

Fig. 3.7 Shop floor visualizations of resource status (left) and routing (right), realized with Visual Components 3DCreate [19]

To enable Visual Manufacturing, complete production resources will be considered as mechatronical units. Hence, production resource objects will be modeled and implemented with different object properties and behaviors encapsulated. Here, the necessary visualization properties and behavior can be integrated in the object behavior usable in larger applications. In addition, necessary data aggregation methods and behavior simulation means usable within these applications in objects can be integrated here.

3.3.2 Collaborative Manufacturing

Almost any production system consists of technical equipment and human personnel. A plant worker does not operate in isolation but is rather in a constant dialog with the machinery and coworkers [20]. He plans and executes actions, responds to exceptional events, and collaborates with coworkers to solve operational problems. Currently, only a few IT systems provide the structure and support to help plant personnel to efficiently communicate and collaborate.

To support the efficient work of plant personnel, systems are required that provide work support that is strongly knowledge based and production system and production system component functionality oriented like equipment failure reporting, diagnostic support, and maintenance assistance as well as problem solving strategy evaluation support.

In a highly networked world, collaboration is naturally not limited to the boundaries of one's own enterprise. Deeper collaboration with suppliers and subcontractors leads to better overall performance of the supply chain but requires data exchange models and security and privacy structures.

For Collaborative Manufacturing diagnosis functionalities can be integrated in the objects' modeling/implementing mechatronical units. These units can then provide diagnostic services to the overall system. In addition, object internal (and thereby mechatronical unit internal) data protection and access rules within the SOA implementation based on OO encapsulation mechanisms can enable company border crossing service applications.

Additionally, MASs can be exploited to implement collaborative manufacturing within the technical system by mapping mechatronical units to agents and implementing the control of a mechatronical unit within the agent and the collaboration of mechatronical units by agent interaction.

3.3.3 Real-World Manufacturing

State-of-the-art production control systems or MES (Manufacturing Execution System) / MOM (Manufacturing Operations Management) systems often do not support the manufacturing process according to real needs but frequently "over-engineer" solutions or, worse, address the wrong problem. Here the scheduling functionality of current MES or ERP systems are among the recent problems. As Pellerin et al. [21] have stated, most scheduling research and products are focused on highly sophisticated algorithms for building optimal production schedules but neglect the agility and required flexibility of real-world schedules. Hence, human planners or schedulers mostly do not use automated tools for schedule creation but rather need support for their highly manual and collaborative work of building and updating the schedules (usually more than one) by hand, evaluating what-if scenarios, and performing impact analyses. Planners often spend a large portion of their time on the shop floor ensuring the synchronicity of their schedules with the real process. Usually, they integrate into their schedule-generation processes engineering knowledge usually not available or not applicable in automatic scheduling procedures.

Scheduling systems should be redesigned to acknowledge these facts, become more flexible, and return the control of the scheduling process to the planner. The underlying problem of (unknowingly) ignoring the actual processes and only supporting the ideal ones needs to be addressed.

Generally, Real-World Manufacturing is an organizational and not a technical challenge. Hence, its elucidation requires organizational and management-based activities. Nevertheless, the aforementioned technologies may support this process. For example, in the case of scheduling aspects of the Real-World Manufacturing challenge, distributed scheduling mechanisms can be designed and implemented. These mechanisms may include the tracing of orders. They can be implemented by services provided by order objects of order agents and can be accessible by human operators. It gets possible to integrate resource-related scheduling functions customized to resource characteristics within the resource modeling/implementation mechatronical unit objects. Here the OO capabilities of polymorphism can be exploited. Additionally, expert systems can become integrated in these objects.

3.3.4 Open Manufacturing

Any standard software will fail to "win the hearts of the manufacturing staff" if it cannot be adapted to the particular needs of the company using it. Every shop floor is different if not unique. While standard processes and approaches do exist (like production order execution, confirmation, tracking & tracing, Kanban, etc.), usually they have very specific features that depend on the industry, the manufacturer's size, the level of automation, and many more individual characteristics forcing the user to adapt them to its special needs, making them unique. In order to address this heterogeneity, ideally every manufacturer should be able to design tailor-made applications.

Of course, growing costs prevent the implementation of all functionalities from scratch. Only a high level of reuse can guarantee a profitable solution. Therefore, reusable structures and functionalities have to exist and have to be implemented in a modular and block-oriented way. Ideally, these blocks would be based on open standards that would further facilitate ease of integration of best-of-breed solutions. The necessary reuse can be addressed by the hierarchy of mechatronical objects and its modeling/implementation by OO objects. Here each mechatronical unit in the hierarchy can be inherited from generalized mechatronical units as, for example, in [22]. Here, each mechatronical unit should provide customizable services accessible via standardized interfaces.

3.3.5 Reconfigurable Manufacturing

Flexibility of production systems is a necessary requirement to enable adaptability of the system to changing product, customer, or technology requirements. However, only reconfigurability ensures that production systems can be adapted after their initial design (see also [23]). This is what distinguishes Reconfigurable Manufacturing Systems (RMSs) from Flexible Manufacturing Systems (FMSs). FMSs provide different forms of flexibility for anticipated variations by their a-priori design. In contrast to this, RMSs provide adaptation capabilities when and where needed, i.e., “customized flexibility” delivered in a short time. They offer reconfiguration options at the hardware and software levels enabling the update of the system to face changing requirements until the complete change of the initial system.

Reconfigurability on a strategic level refers to a redesign of the overall manufacturing processes and system landscape. The Japanese “Kaizen” approach to continuous process improvement long ago revolutionized the way manufacturers develop their processes by benefiting from the creativity of their own personnel. Today, a number of tools exist to support the process of collecting and evaluating ideas. However, putting the ideas into practice is largely unsupported as current systems are not designed for change.

Hence, as above, structure and architectures are required that enable the easy integration of system changes in the running production system and, therefore, in its control system.

To attain Reconfigurable Manufacturing the classical modular architectures need to be improved. They only guarantee limited flexibility due to the complexity of single-application modules. In contrast to this, SOA facilitates the composition of customer-specific applications because the blocks of functionality are typically more fine-grained, clearly separated, and accessible as services. Ideally, the composition environment is targeted at domain experts such as production engineers and should not require much training. Reconfigurable systems based on services as described above can be adapted faster as the services are only loosely coupled. Similarly, AOA enables the composition of applications following the distribution of the control decision process resulting in structures similar to those of SOA with the main difference of a stronger focus on proactivity of control entities in contrast to the server behavior of services.

3.3.6 Harmonized Manufacturing

In order to enable smooth manufacturing operations, IT systems at the different layers of the automation pyramid (management layer, process control layer, field control layer) need to be synchronized. A series of standards exist to address this problem (e.g., ISA-95 [24] or OAGIS [25]) but these typically fall short of providing a comprehensive solution. Two major shortcomings are the synchronization of master data and of processes at the different layers. Both, the ERP system and plant-local MES/MOM system, need production master data such as resource information, routings, bills of material, and bills of operation to execute their tasks. Although these systems often need the same or similar data, they typically maintain their own master data, which tend to be difficult to consolidate and keep in sync. On the other hand, some of the data used by these systems are quite different. Therefore, simple data replication from one system to the other does not solve the problem. Rather, the different data models maintained by the ERP and MES/MOM need to be harmonized. Furthermore, the processes that operate on the data also need to be synchronized – a nontrivial problem in a distributed environment.

OO-based technologies may assist as a sound solution to this problem. Based on appropriate ontologies exploiting, for example, ISA 95 object structures can be developed handling the necessary data of the different layers. They can be adapted to the necessary data exchange among layers in an easy way reflecting the different data needs of the different layers.

The integration of these data objects within SOA-based communication architectures will enable the development of a sound and stable data exchange architecture for all levels.

3.3.7 "Green" Manufacturing

Processes and technologies should assist manufacturing enterprises to better monitor, manage, and optimize their material usage and energy consumption. For example, increased transparency and integration would enable major power consumers (e.g. steel, or paper industry) to adjust their production schedule to the changing price of the power and produce when energy capacities are in low demand (e.g., at night). Better quality control would reduce scrap or the need for rework. Optimized production schedules would reduce carbon dioxide emissions as resources that are important for environmental protections (e.g., ovens) are better utilized. The care for the environment is and is going to be one of the major challenges for the near future.

Like real Real-World Manufacturing Green Manufacturing requires management-based organizational intervention. Nevertheless, advanced technologies may support these interventions and provide a base for its technical implementation. For example, with the integration of monitoring services within mechatronical unit modeling/implementing objects, the requirements to support Green Manufacturing can be created. Here, the monitoring service search and application structures can enable the collection of data required for optimization. These services may be implemented in line with diagnostic services. The resource-related adaptation of the services and of the relevant optimization strategies can be reached by application of the OO concepts of inheritance and polymorphism.

3.3.8 Distributed Manufacturing

Virtually all current production control systems are centrally organized and tightly connected to the current state of the factory layout. Any change to the plant layout (e.g., adding new machines, changing the setup) typically requires considerable changes to the IT systems.

A decentralized approach could make production systems more flexible and adaptable. In a decentralized system, individual functionalities may be distributed to several entities that in turn may be logically or even physically distributed. Thus, each entity is able to act independently from the others and offer its services via standardized interfaces.

The implementation of Distributed Manufacturing systems can be based on the above mentioned distribution and encapsulation of control decisions within OO objects responsible for the control of mechatronical units. The control-decision-relevant data can be exchanged via interfaces using either SOA concepts or agent concepts.

3.3.9 Event-driven Manufacturing

A machine breakdown is perhaps the most prominent example of a significant event that can occur in a production environment, but it is by no means the only one. The completion of a production order, the occurrence of scrap, missing material, the rescheduling of a production plan, the press of a button on an HMI – these are more examples representing the multitude of events occurring on the shop floor.

For each event there may be none, one, or more interested parties, be it a machine, a production control system, or plant personnel. In many cases there are multiple interested parties per event. Most current production control systems do not have a clean and scalable event management architecture. Instead, a prominent approach is the data pull paradigm in which important data points are pulled in regular time intervals. This mechanism can cause considerable system load and is therefore in many cases a rather inefficient solution. Other "event mechanisms" like OPC DA subscription or OPC A&E, both currently being unified into OPC UA [26], offer event registration at the device level but do not provide any support for complex event processing. An example of a slightly more complex event that includes a temporal dimension is when one of three bottleneck resources becomes unavailable for more than 5 min. Even this basic event cannot be described by most current MES/MOM systems.

The development of event brokering services enabling event registration by event producers and event notification by event consumers as displayed in Fig. 3.8 can be used to solve the problem of Event-Driven Manufacturing. Here, the SOA can be exploited on the level of mechatronical units to design event provider services where event consumers register for events that will be transmitted only if they occur. The event provider services can be integrated with event monitoring engines within OO objects describing mechatronical units.

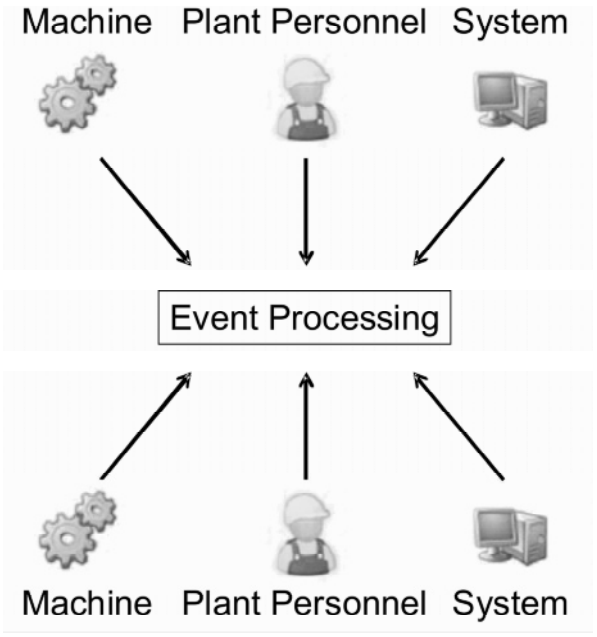

Fig. 3.8 Event processing engine for shop floor events

3.3.10 *Mobile Manufacturing*

Finding material or tools in a warehouse, performing maintenance and problem analysis on machines, configuring small devices without one's own user interface, and monitoring critical events are primary use cases for mobile and wireless technologies like PDA, wireless laptops, or cell phones.

While mobile applications have long found their place in manufacturing, the design of mobile and/or occasionally disconnected applications is still challenging.

Apart from hardware-related problems like battery life and the reliability of wireless connections, issues like the platform-independent design of a usable application or problems with data synchronization hinder further adoption. Multimodal applications (using voice, image, or even gesture recognition and other input/output modalities) increase the opportunities (e.g., enabling online support during machine maintenance) but also the costs.

Finally, the application of OO concepts enables the development of platform-independent programming and embedded networking capabilities as, for example, given in the programming technology Java[1] [27]. These platform-independent programming technologies and the agent technologies based on them like Jade [28] can be exploited to enable Mobile Manufacturing. Mobile devices can be implemented using a layered architecture including hardware, runtime environment for control applications, and hardware-independent control applications. In addition, OO concepts as developed for Distributed Manufacturing can be adapted to mobile devices enabling their on-demand integration in control applications.

3.4 Application Example

One application example of the above-described technologies for the solution of the mentioned problems is the PABADIS'PROMISE (PABADIS based Product Oriented Manufacturing Systems for Re-Configurable Enterprises) approach developed in the EC project PABADIS'PROMISE [29]. One outcome of the PABADIS'PROMISE project is a highly generic control system architecture and control system design methodology enabling a stringent decision responsibility distribution among acting control entities modeled by OO objects. It distinguishes especially between a description of plant resources providing services and products requiring services and a generic unique executor able to read these data, to find the match between supply and demand, and then to control the production that is based on SOA mechanisms and OO mechanisms (depending on the layer of control). The proposed control architecture is thus characterized by a complete separation between system data and execution environment.

[1] Sun, Sun Microsystems, the Sun Logo and Java are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries.

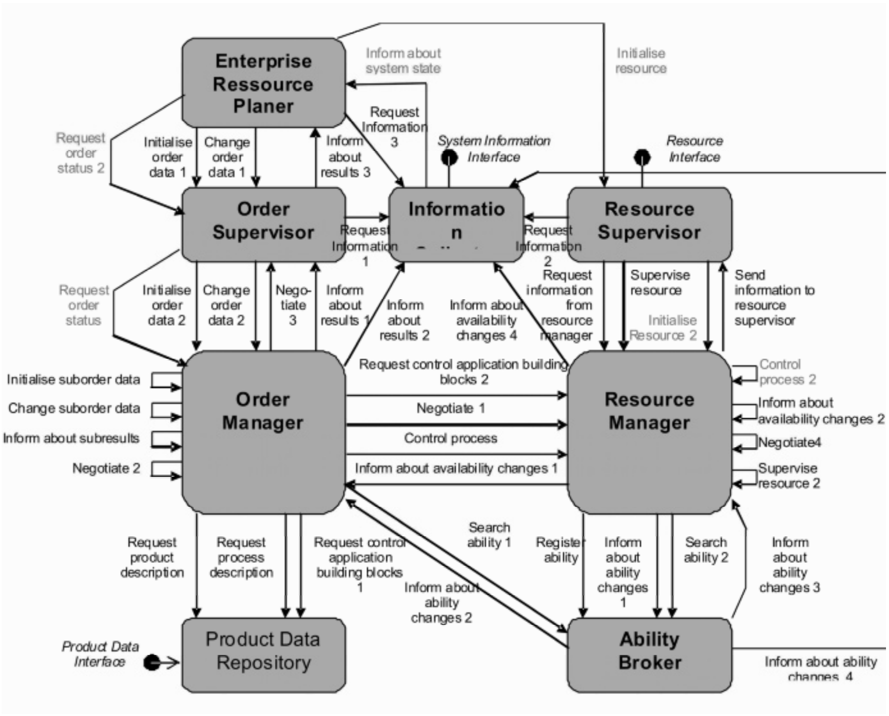

Fig. 3.9 PABADIS'PROMISE basic architecture

The PABADIS’PROMISE architecture itself (Fig. 3.9) is modeled by a set of high-level OO object entities containing the enterprise resource planner for the overall order, resource, and data management at the ERP level, the order supervisor for order execution initialization and supervision at the MES level, the order manager for order execution control at the MES level, the product data repository for product-related data provision at the MES and field control layer, the resource manager for resource control at the MES and the field control layer, the resource supervisor for resource supervision at the MES and the field control layer, the information collector for data collection at the MES and the field control layer, and the ability broker for manufacturing process execution ability announcement and search.

For the implementation of the different entities and their interaction mechanisms different OO-based technologies have been exploited. Thereby, a consistent structure has been defined applying the SOA approach at the ERP layer implementing the Enterprise Resource Planner and parts of the Resource Supervisor, Order Supervisor, Information Collector, and Product Data Repository, agent-oriented structures at the MES layer implementing Order Supervisor, Order Manager, Product Data Repository, Resource Manager, Resource Supervisor, Information Collector, and Ability Broker, and function-block-oriented structures at the field control layer implementing parts of the Resource Manager (Fig. 3.10).

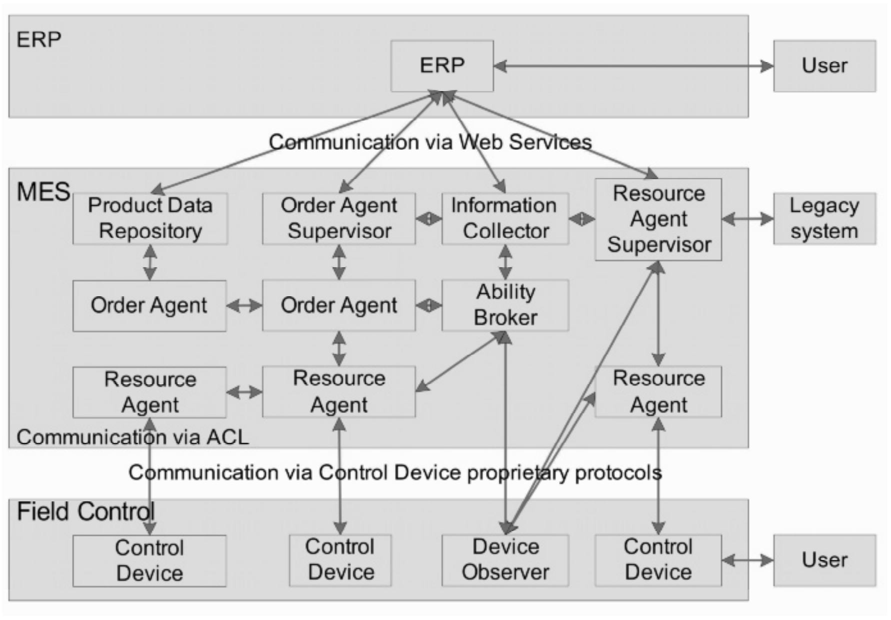

Fig. 3.10 PABADIS'PROMISE agent system architecture

The resulting structure of the PABADIS'PROMISE control system implementation integrates several of the aforementioned problem solutions successfully in the following way.

The concept of an information collector entity enables the implementation of *Visual Manufacturing, Green Manufacturing,* and *Collaborative Manufacturing.* The interentity communication between Resource Managers and Information Collectors and Resource Manager internal data collection and preprocessing enables an easy adaptable data provision to both systems while the information collector entities provide the necessary HMI for users.

The distributed scheduling mechanisms implemented in the PABADIS'PROMISE system result in *Real-World Manufacturing* structures.

The implementation of the MES layer based on the open source agent system Jade [28] has been published by the project on its Web page for further evaluation and extension. Hence, it follows the ideas of *Open Manufacturing.*

The plug-and-participate architecture for manufacturing resources controlled by Resource Managers as one example enables *Reconfigurable Manufacturing* since the Resource Mangers implement a special kind of SOA using agent interaction mechanisms to provide manufacturing services used by Order Managers.

In general the PABADIS'PROMISE architecture follows the ideas and characteristics of *Distributed Manufacturing.*

Since the plug-and-participate structure at the MES layer also involved event-based notification mechanisms for resource availability changes, *Event-Driven Manufacturing* is also covered by this architecture.

For more information on the PABADIS'PROMISE project results see [30].

3.5 Conclusions

In this chapter, three recently emerged software engineering paradigms have been presented. It has been shown how they can be applied as part of a strategy to tackle recent challenges related to factory automation.

Therefore, this chapter started with a description of the paradigms of object orientation, agent orientation, and service orientation and a consideration of their general application within industrial control exploiting the control of mechatronical units.

Then, ten major challenges were presented necessary to be considered to make industrial systems sustainable. They concern the impact of market, engineering, government, environmental protection, etc. combining technical and organizational challenges.

For each challenge it has been shown how the three paradigms can be exploited to successfully deal with the challenges.

In light of the results of this chapter it must be stated that structuring paradigms originating from information sciences and their application to mechatronical units will be an increasing trend within industrial automation and control.

References

[1] Kühnle, H. 2007. Post mass production paradigm (PMPP) trajectories. *Journal of Manufacturing Technology Management*, 18 (8): 1022-1037.

[2] ManuFuture Platform. 2005. *STRATEGIC RESEARCH AGENDA assuring the future of manufacturing in Europe.* Available from: http://www.manufuture.org/SRA_form.html [Accessed December 2005].

[3] Brügge, B. and A. Dutoit. 2004. *Objektorientierte Softwaretechnik.*, München: Pearson Studium, Prentice Hall.

[4] Booch, G., J. Rumbaugh, and I. Jacobson. 2005. *The Unified Modeling Language User Guide, The Addison-Wesley Object Technology Series*. Amsterdam: Addison-Wesley Longman.

[5] Weilkiens, T. 2008. *Systems Engineering with SysML/UML - Modeling, Analysis, Design*. OMG Press.

[6] Booch, G. 1997. *Object-Oriented Analysis and Design with Applications.* Addison-Wesley.

[7] Lüder, A., J. Peschke, and R. Sanz. 2006. Design Patterns for Distributed Control Applications, *ATP International,* 3: 32-40.

[8] Vyatkin, V., Z. Salcic, P. Roop, and J. Fitzgerald. 2007. Information Infrastructure of Intelligent Machines based on IEC61499 Architecture. *IEEE Industrial Electronics Magazine,* 1 (4).

[9] Thramboulidis, K. 2008. Challenges in the Development of Mechatronic Systems: The Mechatronic Component *(13th IEEE International Conference on Emerging Technologies and Factory Automation),* September 2008, Hamburg. Proceedings [online] Available from: http://seg.ee.upatras.gr/thrambo/dev/Papers/ETFA08paper.pdf

[10] Sculte, R. and Y. Natis. *SSA Research Note SPA-401-068: Service Oriented Architectures, Part 1and Part 2*. Gartner.

[11] Bieberstein, N., R.G. Laird, D. K. Jones, and T. Mitra. 2008. *Executing SOA – a Practical Guide for the Service-oriented Architect.* Upper Saddle River: Pearson.

[12] Neumann, P., A. Poeschmann, and R. Messerschmidt. 2008. Architectural Concept of Virtual Automation Networks *(17th IFAC World Congress Proceedings),* 6-11 July, Seoul.

[13] Papazoglou, M.P. 2007. *Web Services: Principles and Technology.* Essex: Prentice Hall.

[14] Shen, W., Q. Hao, H.J. Yoon, and D.H. Norrie. 2006. Applications of agent-based systems in intelligent manufacturing: An updated review. *Advanced Engineering Informatics,* 20 (4): 415-431.

[15] Wannagat, A., B. Vogel-Heuser, H. Mubarak, and P. Göhner. 2007. Evaluation of Agent Oriented Methodologies for the Development of Flexible Embedded Real-Time Systems in Automation. *ATP-International,* 1.

[16] Czichos, H. 2006. *Mechatronik – Grundlagen und Anwendungen technischer Systeme.* Wiesbaden: Vieweg Verlag.

[17] Habib, M.K. 2007. Mechatronics – a unifying interdisciplinary and intelligent engineering science paradigm. *IEEE Industrial Electronics Magazine,* Summer: 12-24.

[18] Rode, J. and D. Wünsch. 2007. A Research Agenda for Adaptive Manufacturing. *(12th IEEE Int. Conf. on Emerging Technologies and Factory Automation Proceedings),* September, Patras.

[19] Visual Components Oy, 3DCreate. 2008. *Bring 3D CAD Data to life*. Available from: http://www.visualcomponents.com/index.php?id=56.

[20] Bergmann, U. 2008. *Kommunikation als Optimierungskriterium: ein Beitrag zur Systematisierung der Layoutplanung von Produktionssystemen*. Thesis (PhD). Otto-v.Guericke University Magdeburg.

[21] Pellerin, R., P. Batiste, J. Robert, S. Guessous, M. Guevremont, J. Proulx, and B. Saenz De Ugarte. 2007. Lean scheduling – case studies of 10 manufacturing plants, Ecole Polytechnique Montreal. *Internal report for SAP Research.*

[22] Drath, R., A. Lüder, J. Peschke, and L. Hundt. 2008. An evolutionary approach for the industrial introduction of virtual commissioning. *(13th IEEE International Conference on Emerging Technologies and Factory Automation Proceeding-CD).* September, Hamburg.

[23] ElMaraghy, H. 2005. Flexible and reconfigurable manufacturing systems paradigms. *International Journal of Flexible Manufacturing,* 17: 261-276.

[24] The Instrumentation, Systems and Automation Society. 2000. *ANSI/ISA-95.00.01-2000: Enterprise Control System Integration.*

[25] Open Applications Group. 2007. *Open Applications Group Integration Specification (OAGIS).* Release 9.1 Available from: http://openapplications.org/oagis/9.1/index.html.

[26] OPC Foundation. 2007. OPC Unified Architecture. Available from: http://www.opcfoundation.org/Default.aspx/01_about/UA.asp?MID=AboutOPC.

[27] Savitch, W. and F.M. Carrano. 2008. *Java – An Introduction to Problem Solving & Programming.* Pearson International Edition 5th ed. Prentice Hall.

[28] Bellifemine, F., G. Caire, and D. Greenwood. 2007. *Developing multi-agent systems with JADE*, Wiley series in agent technology, Wiley publisher.

[29] Lüder, A. and J. Peschke. 2007. Order oriented manufacturing control: the Pabadis'Promise approach. *Automazione e Strumentazione,* November: 84-93.

[30] PABADIS'PROMISE consortium. 2008. PABADIS'PROMISE project homepage. Available from: www.pabadis-promise.org.