Chapter 13
Integrated Modeling of Complex Production Automation Systems to Increase Dependability

Birgit Vogel-Heuser and Susanne Rösch

In current practice, analysis and development of the same mechatronic component are performed separately for both functional and nonfunctional (for definition see Sect. 3) aspects, often by different engineers and/or engineering teams, and specified in different modeling languages. This gap between the development processes of the different aspects of components leads, on the one hand, to inefficient system development processes and additional iterations between functional and nonfunctional design. On the other hand, it makes for a neglected opportunity to increase system dependability during runtime (Avizienis et al. in IEEE Trans. Dependable Sec. Comput. 1(1):11–33, 2004). By building on basic engineering information, for instance by integrating models containing selected information about a system into its control code, dynamic reconfiguration during runtime helps to increase dependability and reduce risk. Risk in this chapter is defined according to Bertsche as the "product of severity of damage and probability of occurrence" (Bertsche et al. in Zuverlässigkeit mechatronischer Systeme. Grundlagen und Bewertung in frühen Entwicklungsphasen, Springer, Berlin, 2009, p. 55) and the term dependability is used according to Avizienis et al. (IEEE Trans. Dependable Sec. Comput. 1(1):11–33, 2004): "dependability is an integrating concept that encompasses the following attributes:

- *availability* (availability in this context is considered as "the degree to which a system or component is operational and accessible when required for use, often expressed as a probability" (IEEE Std. 610.12-1990, IEEE standard glossary of software engineering terminology, The Institute of Electrical and Electronics Engineers, USA, 1990)): readiness for correct service;
- *reliability*: continuity of correct service;
- *safety*: absence of catastrophic consequences on the user(s) and the environment;

B. Vogel-Heuser (✉)
Chair of Automation and Information Systems, Department of Mechanical Engineering, Technische Universität München, Boltzmannstr. 15, 85748 Garching bei München, Germany
e-mail: vogel-heuser@ais.mw.tum.de

S. Rösch
Automation and Information Systems, Department of Mechanical Engineering, Technische Universität München, Boltzmannstr. 15, 85748 Garching bei München, Germany

C. Klüppelberg et al. (eds.), *Risk – A Multidisciplinary Introduction*,
DOI 10.1007/978-3-319-04486-6_13,

- *integrity*: absence of improper system alterations;
- *maintainability*: ability to undergo modifications and repairs" (Avizienis et al. in IEEE Trans. Dependable Sec. Comput. 1(1):11–33, 2004, p. 13).

Another important term used in this chapter is Quality of Service (QoS). This term has been used recently for different domains. In this chapter QoS is used for the quality that can be assumed when using a substitute strategy to replace another service.

This chapter contributes to the design of system availability, reliability, and safety, focusing on complex production automation systems and highlighting the results by introducing application examples from the control of a continuous thermo-hydraulic particle board press.

Keywords Automation systems · Model-based system and software engineering · Integrated modeling · Safety and functional analysis · Dynamic reconfiguration

The Facts

- Reduced time to market, and lowering of costs for product automation systems, require concurrent engineering.
- Traditional modeling methods do not support integrated development of functional and safety aspects for production automation systems.
- Additional integration of basic engineering models into the control code can be used to increase dependability of production automation systems during runtime by incorporating those models into the control code as a knowledge base for intelligent adaptive behavior.
- Integration of the different functional views of a production automation system, i.e. mechanical, electrical/electronic and software, with their constraints and restrictions will support model based dynamic reconfiguration.

1 Introduction

Today, suppliers of mechatronic products face stronger competition worldwide, resulting in a need for reduced time to market. This leads to decreasing duration times for a project and decreasing start up times, which directly influence plant manufacturers and their automation suppliers. Due to the need for reduction of project duration and time to market, concurrent and simultaneous engineering have become more important (see Fig. 1). Therefore, automation suppliers require support during the whole engineering life cycle in a more efficient way. This applies not only to the design phase as such, but to the entire life cycle. Starting about five years ago [14], forced by competition through globalization, the phases in life cycles of production automation systems, i.e. concept phase, design phase up to construction, needed to

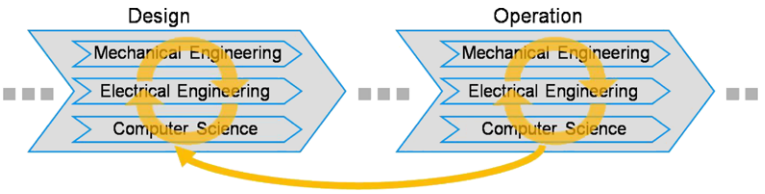

Fig. 1 Concurrent Simultaneous Engineering for design, optimization and operation (CSE) [14]

be shortened and better integrated. Up to now there has been a lack of method and tool integration. The result is a rupture in the engineering work flow between the different phases and the different disciplines.

Another challenge focused on during the last 5 to 10 years is the integration of the different views of functional aspects of a production automation system, such as mechanical, electrical/electronic, and software. Regarding these different aspects a specific challenge can be identified: the necessity to integrate the different disciplines to achieve a more appropriate solution, taking into account all aspects of a system.

During the last few years methods and technologies have been generated to allow the first promising developments of such an integration:

- From computer science, meta modeling has been introduced and is now widely spread in engineering domains, as are model coupling techniques [18, 19] and tools. *Eclipse* for example is one opportunity for the coupling of models of different engineering phases and of different engineering disciplines.
- From the discipline of automation, *AutomationML*, containing a high-level description of the topology of a system within *CAEX*, and several lower-level descriptions for specific aspects of a system such as *PLCOpen XML*, has been introduced as an XML-based description approach for engineering information [5].
- Embedded systems in production automation systems become more powerful computationally, which is a prerequisite for the use of model information and the implementation of intelligent algorithms for adaptive control systems during runtime.
- Last but not least, model based engineering is becoming more popular in the different disciplines of product and production automation, that being the prerequisite for acceptance of re-use and model coupling.

This chapter is organized as follows: first the different views from production automation are introduced and explained using a continuous thermo-hydraulic particle board press as application example. Section three highlights the modeling of functional and nonfunctional requirements, which need to be integrated into the engineering approach and the whole life cycle. Section four presents a first attempt to integrate safety and functional design by mapping the traditional safety models, for example Failure Mode and Effects Analysis (FMEA) and Fault Tree Analysis (FTA) to the functional models using an object oriented approach. The example mentioned above, a real industrial application, is given as an evaluation example. The benefit of

integrating basic model engineering information into the control code for dynamic reconfiguration during runtime is demonstrated in section five, based on the assumption that an integrated approach is to be used. This approach allows the control system to cope with malfunctions and, under given safety and operational constraints, to adapt its behavior autonomously. Section six presents some new ideas, and seven summarizes the results and gives our outlook on future work.

2 Different Views on Production Automation Systems

Thramboulidis [12] introduces a three-view model, modeled in the Systems Modeling Language (SysML).[1] The three views of Thramboulidis comprise software engineering, mechanical engineering, and electrical engineering. The central "+1" model is the mechatronic system (MTS) model which is specified using SysML. Thramboulidis et al. highlight that safety is one aspect of the MTS +1 view [13]. They claim that as a result of the synergistic modeling and the integration it is possible to make extensive dependability predictions during the development phases. Unfortunately, this is only a first concept and has not yet been implemented or proven. As a limiting prerequisite, Thramboulidis et al.'s modeling approach requires that all disciplines agree on the same component interfaces, which is rarely the case in industry. Li et al. [4] focus on the integration of mechanical models and models of information technology using SysML.

Wannagat and Vogel-Heuser [16] and Schütz and Wannagat [11] introduce a different three-view model for modeling production automation systems. It is suggested to view systems in the perspective of the technical process, the technical system, and the automation control system. The model supports the interdisciplinary work of different disciplines. Dividing the plant into different domain-specific views enables the description of the different disciplines with one modeling language such as SysML. The model allows each domain-specific engineer to have his/her own view on their components as well as a view of interfaces depicted as relations to components of other disciplines and their requirements. Schütz and Vogel-Heuser [10] use the three views and SysML as an approach to model energy aspects integrated into the functional models.

According to Wannagat and Vogel-Heuser [16], the three views can be described as follows: The technical system relates to the mechanical parts of a plant; therefore it contains information about the layout and the connections of the mechatronic components, as well as energy and material flows between them. In the automation control system controllers, networks, sensors, and actuators are included. Thramboulidis separates this view into software and electrical engineering, which is modeled as a sub-layer in our concept. The technical process itself describes the manufacturing of the product, taking account of the chronological order and all physical

[1] SysML is an extension of the Unified Modeling Language (UML), defined by the OMG, to satisfy the requirements of system engineers. In particular it offers "a semantic foundation for modeling system requirements, behavior, structure, and parametrics, ..." [26].

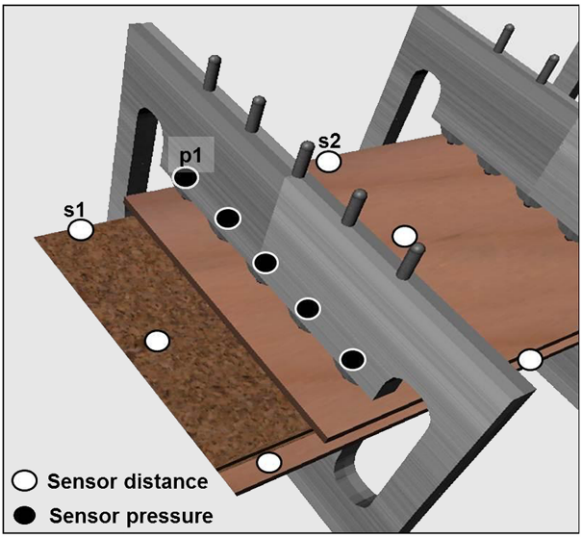

Fig. 2 Process and Instrumentation Diagram of the continuous thermo-hydraulic particle board press as application example for basic engineering information and model base for dynamic reconfiguration [16]

changes made during the process, e.g. chemical or pharmaceutical processes. In our opinion this view is essential because it represents the actual purpose of the system and is not addressed in Thamboulidis et al.'s approach. All three views allow for specialized observation of the whole system and its components. One component contains all aspects that have been assigned to it in the different views. This way the component establishes a connection concerning the content of the different views, enabling the analysis and comparison of all aspects.

Illustration 2.1 Sample application of a continuous thermo-hydraulic particle board press according to [16].

A model of a continuous production process, as basic engineering information, will be introduced in this section. With this model dynamic reconfiguration in order to increase availability will be demonstrated in Sect. 5. The continuous thermo-hydraulic particle board press is a real industrial application (Fig. 2). It is composed of up to 80 separately controlled frames (in Fig. 2 two frames are depicted). Each frame consists of 5 separately controlled cylinders with sensors for pressure (p_1) and distance (s_1, s_2).

The technical process (Fig. 3) is modeled as an internal block diagram (SysML). It shows the different sections of a continuous thermo-hydraulic particle board press from a technologist's point of view. The raw material for the particle board (wooden fibers with glue, i.e. mat) is fed into the press on the left side and will be heated and pressed. The different sections are modeled with regard to different technological functionality. From the initial description an activity diagram (Fig. 4) of this technical process is designed showing the three sections of the press. They are modeled using so-called swim lanes, which are used for structuring activity diagrams.

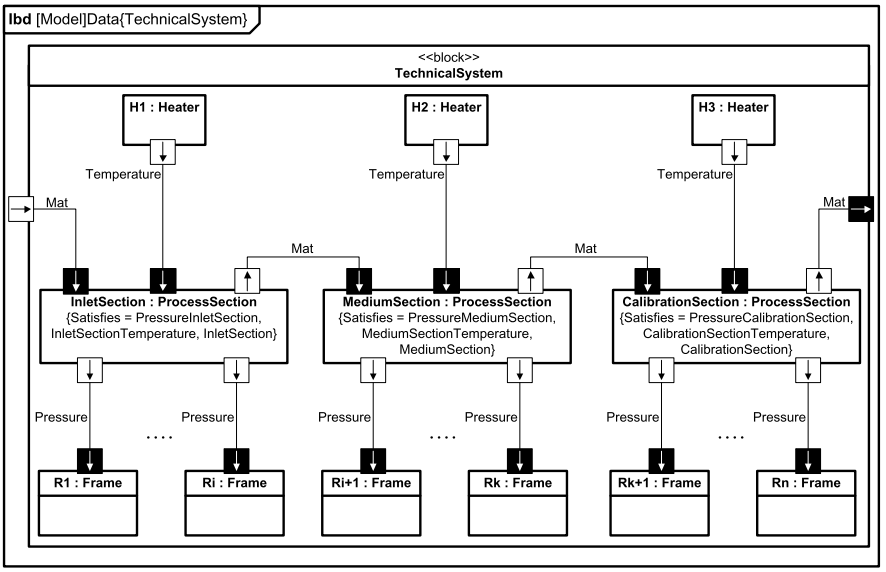

Fig. 3 Internal block diagram of the technical process view (*top level*) of the application example: continuous thermo-hydraulic particle board press [16]

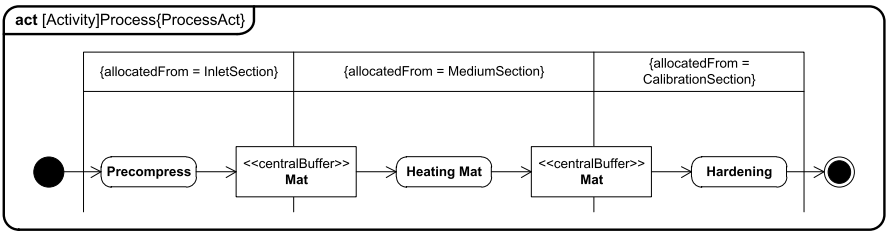

Fig. 4 Activity diagram of the technical process (*top level*) [16]

The technical system consists of a generator for electric power supply, the hydraulic system and its interface to the mat (technical process view), the hydraulic main valve and the five valves as well as pressure cylinders of each frame as can be derived from Fig. 2. The mat (bottom right Fig. 5) is pressed by these cylinders, indicated by connectors from each cylinder to the mat representing the force (F). The cylinders increase the pressure as soon as the valves are opened (connector Pressure). These structural aspects of the technical system are depicted in the internal block diagram (Fig. 5).

The automation control system (Fig. 6) represents the chosen automation concept with a classical automation device—a Programmable Logic Controller (PLC (S7))—connected via a bus coupler to the input and output connectors of the single frame of the press. Some of the components are only partially indicated behind a similar component so as to show better the most important connec-

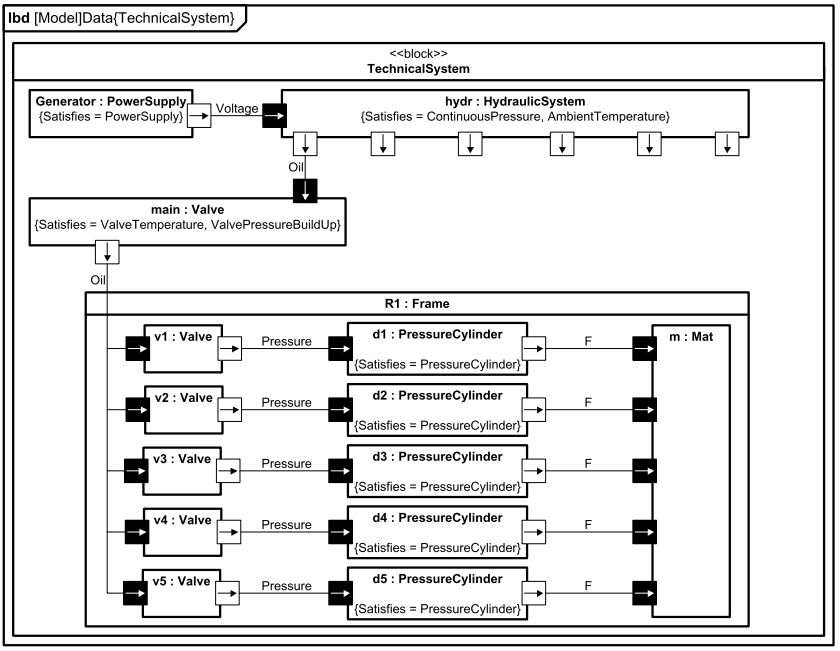

Fig. 5 Internal block diagram technical system, excerpt with five cylinders in one press frame [16]

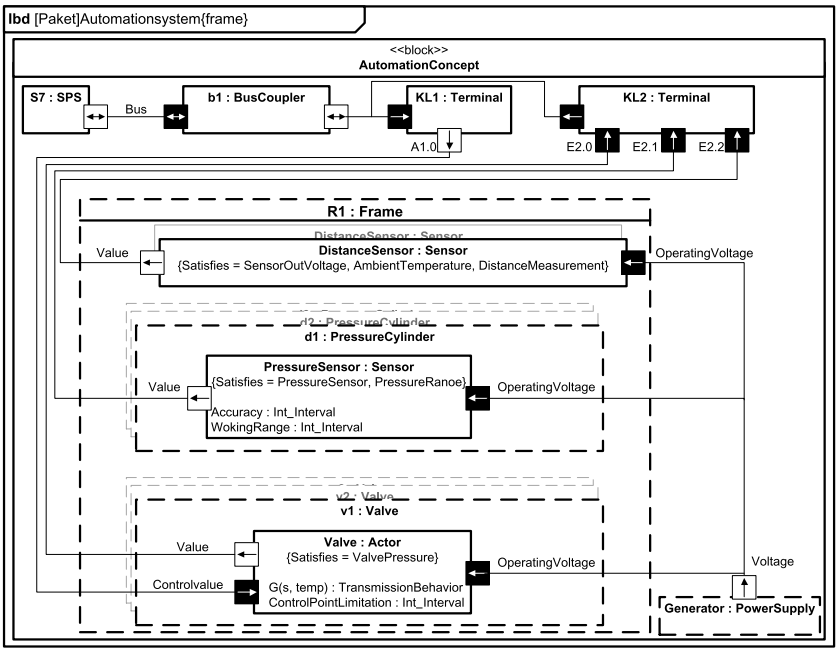

Fig. 6 Internal block diagram of the automation concept

tions within the automation concept. The generator (bottom right), the pressure cylinders, the valve and the frame show the connection to the technical system via a dashed line because they are not part of the automation control system; they belong to another view. Their sensors and actuators, however, belong to the automation system. By representing the most important links between different views using dashed lines the domain engineer has, on the one hand, an overview of his aspects, which helps to reduce complexity, and, on the other hand, the understanding of related components between the disciplines supported by the links.

Recent concepts allow us to generate code for automation applications out of more detailed SysML models [10]. After introducing the three views of the disciplines as one prerequisite, the modeling of requirements will be discussed, regarding also the three different views for one small application example.

3 Modeling Functional and Nonfunctional Requirements

Nonfunctional requirements in general are classified in [1] and from a software engineering point of view in [24]. Dependability and security (Fig. 7) as well as interoperability, maintainability, and time constraints are such nonfunctional requirements. Quality of Service (QoS) has been used recently for different domains (see Sect. 5.1.2). In the case of the continuous thermo-hydraulic particle board press, the application example introduced in this book chapter, QoS comes in where a sensor may be replaced by a calculated one in case of a malfunction (dynamic reconfiguration, Sect. 5). The replacement strategy increases the dependability of the plant under the prerequisite that a minimum required product quality can still be produced with the replacements (virtual sensors).

In our approach functional and nonfunctional requirements are included as constraints to the different views of the production automation system on different levels of detail and granularity starting from a sensor or an actuator up to the entire plant. In Sect. 5.1.2 the tolerance model is introduced as a means to trace whether the required reliability will be maintained, and whether the required probability of a given quality will be reached. If requirements apply to the behavior (dynamic) they need to be modeled in one of the behavior diagrams of the SysML such as the activity diagram. In those diagrams the requirements can be connected to the matching activities fulfilling these requirements.

Illustration 3.1 Temperature and lifespan requirements on the continuous thermo-hydraulic particle board press.

In the requirements model, which consists of different requirement diagrams (documents) according to the three views on the system (upper part of Fig. 8), the requirements of the different domains are described. The domain “technical process” requires the compliance with a defined temperature profile (up to 280 °C)

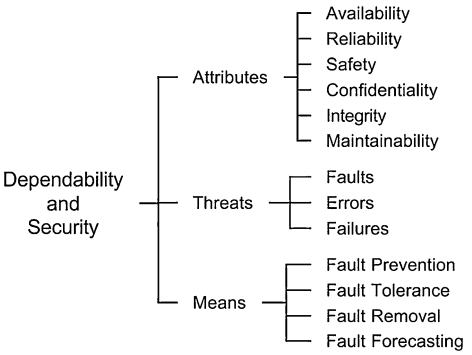

Fig. 7 The dependability and security tree [1]

inside the inlet zone of the press, whereas the domains "automation system" and "technical system" restrict the temperature to ranges in which their components cannot be damaged (automation system: up to 60 °C; technical system: up to 300 °C).

These functional requirements are traced to corresponding constraint blocks using the "satisfy" relation (middle part of Fig. 8). The constraints are instanced inside the blocks which describe the modules of the plant that has to fulfill these requirements by means of a *composite aggregation* (lower part of Fig. 8). For example the block "InletZone" instances the constraint that is linked to the requirement defining a temperature profile. To show how nonfunctional requirements are modeled, a requirement for the valve to reach its defined lifespan is modeled in Fig. 8. The nonfunctional requirement "Lifespan valve" is modeled the same way the functional requirements are modeled. It is linked to a constraint limiting the frequency of opening and closing the valve and is also categorized as part of the technical system.

A heating valve—being part of the automation system of the press—instances the constraint that restricts the temperature due to the limitations of its electronic components ("tempPAS"). Additionally, it references the constraints expressing the limitations resulting from requirements of the other two domains by means of *shared aggregations* ("TempInletZone" and "TempHeatExch"). With this information a parametric diagram can be modeled that describes the limitations regarding the temperature, which were originally stated in the requirements diagrams (documents). The SysML parametric diagram of a heating valve (Fig. 9) within the continuous thermo-hydraulic particle board press contains the different domains' limitations regarding temperature affecting this module. The constraints that are referenced by the valve only by shared aggregations are displayed with dashed lines. This indicates that the valve does not constrain the temperature to these limitations; however, the limitations of other modules of the continuous thermo-hydraulic particle board press ("InletZone" and "HeatExchanger") affect the valve. The temperature outside the valve is composed of the three modules' temperatures. To calculate the temperature inside the valve, thus the tem-

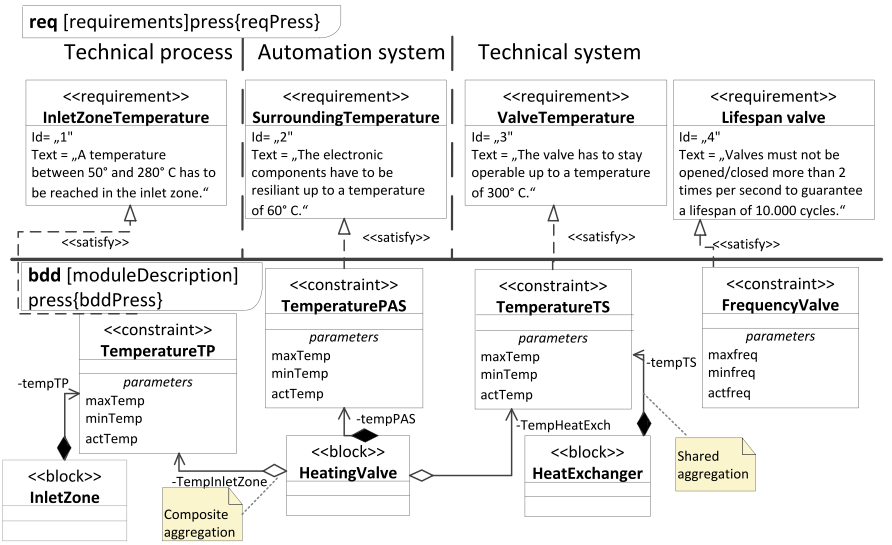

Fig. 8 Requirements and block definition diagram for the requirements concerning temperature

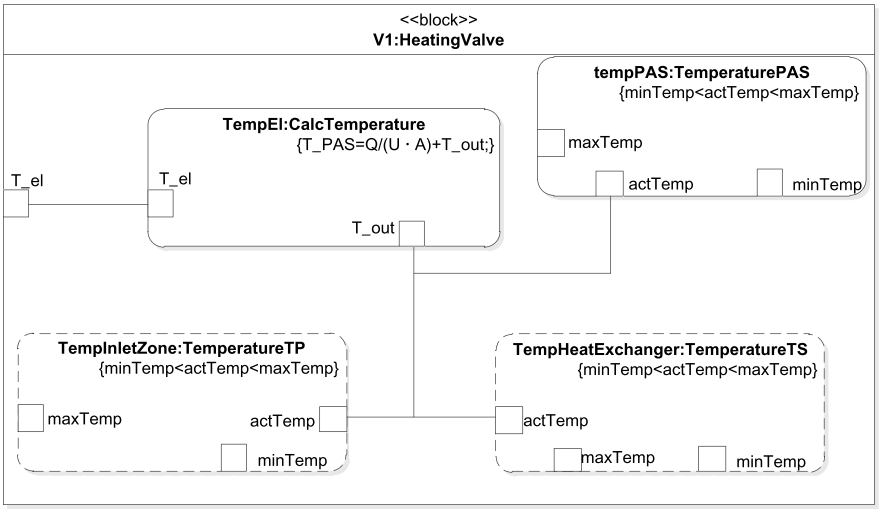

Fig. 9 Parametric diagram depicting the temperature of the heating valve

perature actually affecting the electrical components, the constraint "CalcTemperaturePAS", which expresses a formula for the heat transmission through the surface, is used.

The SysML models are the basis for coping with faults in production automation systems (see Sect. 5).

4 Integration of Potential Malfunctions and Faults as an Integrated Part of Production Automation Systems' Behavior

The task of risk analysis is to identify and evaluate risk for the entire system (see Chap. 12, [25]) at an early stage, based on the weaknesses of individual components of a system. In the first part of this section the traditional safety analysis approaches, which analyze and evaluate faults, errors, and failures, are introduced, i.e., Failure Mode and Effects Analysis (FMEA) and Fault Tree Analysis (FTA). We further discuss their application as part of production automation systems design.

Laprie et al. define faults, errors, and failures in relation to provided services or functions. An "adjudged or hypothesized cause of an error is called a fault" [1, p. 13]. The "definition of an error is the part of the total state of the system that may lead to its subsequent service failure" [1, p. 13], which means an error does not inevitably lead to a failure. A "failure is an event that occurs when the delivered service deviates from correct service" [1, p. 13].

In the automotive domain the current standard follows VDA [27] to check the safety of systems using FMEA [22] and FTA [21].

Rink[2] proposes a method that combines requirement analysis, system modeling, design of control functions, and creation of a safety concept and risk analysis of the system using an integrated object-oriented system model similar to the approach proposed in Sect. 2. The model takes both the desirable and the undesirable behavior of the system into account. Hence a universal system model is created that can be used for the design of functions as well as for the safety concept and risk analysis. Objects that are modeled as components within the system model are determined, based on the physical structure. The components' parameters and state variables appear as attributes in the object specification. The desirable and the undesirable behaviors are described as operations and graphically represented in state charts and activity diagrams. Interactions between objects are depicted by collaboration diagrams. The requirements for the nominal behavior are the basis for the design of control functions. The safety concept must recognize possible system errors and, if necessary, activate appropriate replacement functions so the system maintains or reaches a safe state again. Rink presents a use case-oriented approach for risk analysis generating the structures of the risk analysis based on mapping rules from an object-oriented system. The risk analysis starts with the construction of the FMEA system structure based on object and class diagrams of the object-oriented system model (Fig. 10).

Then the *FMEA function structure* is derived from the state charts, the activity, and the collaboration diagrams that model the desired behavior. On the basis of the undesired behavior of the system model the *FMEA malfunction structure* is created. To facilitate the generation of the FMEA-function and malfunction structure from the object-oriented system model, it is necessary to limit the variety of the means of description available in (UML/SysML) using modeling guidelines. These guidelines

[2] In the following the results of a research cooperation with an automotive company will be introduced which refers to the PhD of Anton Rink [6–9].

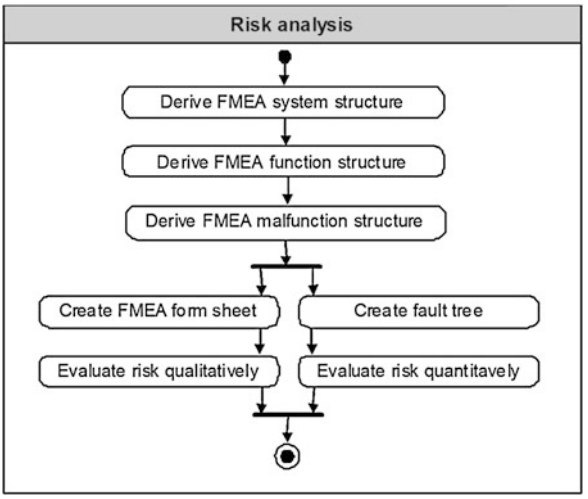

Fig. 10 Steps and procedure during risk analysis [7]

are the basis for the development and use of efficient transformation algorithms to generate the structures of the risk analysis. Both the qualitative and the quantitative risk analysis are performed by means of FMEA form sheets (see Sect. 4.1, Fig. 13) and fault trees (Sect. 4.2, Fig. 14). Depending on the results of the analysis, measures to optimize the control functions and the safety concept are taken in order to meet the requirements regarding availability and safety.

4.1 Failure Mode and Effects Analysis (FMEA)

FMEA is used to analyze whether all malfunctions of the control function have been detected in the safety concept and whether measures to avoid critical states are available. For this purpose possible errors of a system and its control function are registered and in the filled-in FMEA form sheet (see Fig. 13). With support of a fault simulation, consequences of errors and their impact on a system are determined. Due to the object-oriented structure of the system model, specific information from the FMEA forms can be considered as attributes of the objects in the object-oriented system model. Hence a system model which can be used for the functional design, the creation of the safety plan, and the risk analysis is developed.

In accordance with Bertsche et al. [2] the FMEA follows five steps:

1. Creation of a hierarchical system structure out of system elements (system structure tree)
2. Description of the functions and the function structure (function structure tree)
3. Implementation of the malfunction analysis, e.g. detection of possible errors, causes of failures and failure sequences (malfunction structure)
4. Risk evaluation in the FMEA form sheet
5. System optimization with the goal of avoiding malfunctions or reducing risks

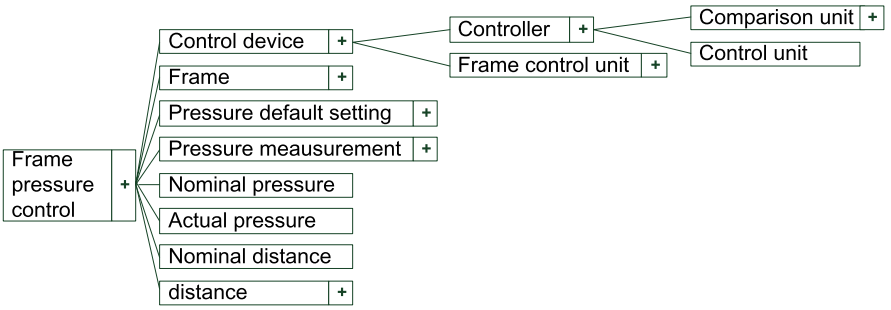

Fig. 11 FMEA—system structure tree showing the system structure of the frame pressure control; objects are modeled as system elements of the system "frame pressure control"

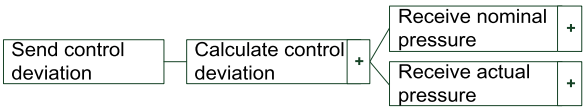

Fig. 12 FMEA—function structure tree (of step 2, Fig. 10) showing the function structure section of the control deviation calculation [8]

Illustration 4.1 Sample application: FMEA "pressure control" for the continuous thermo-hydraulic particle board press.

The FMEA system structure (Fig. 11) is derived from class and object process diagrams by depicting objects as system elements to bridge the gap between an object oriented approach and the safety analysis with FMEA (explained later in this illustration). Object and activity diagrams define the function structure.

Starting from the system function structure tree (Fig. 12) the malfunction structure can be determined. Figure 13 shows the completed FMEA form sheet for the system element "comparison unit", which can be found in Fig. 11 as a subcomponent of "controller". The function analyzed in this particular form sheet is called "Receive nominal pressure", which is only one of the functions that are fulfilled by the comparison unit. All possible malfunctions of the system are identified and entered into the FMEA form sheet (Fig. 13). By means of a fault simulation the failure sequences and their effects on the comparison unit are recognized. In this example a potential failure sequence begins with the comparison unit storing the actual pressure at too high a level. Subsequently, the pressure is assumed to be too high and a negative control difference is calculated. A potential effect of this failure is that the control receives a nominal pressure that is too low. If an error causes a critical state, a high value (maximum 10) is filled into the failure severity column (S). The next step is to check if one of the surveillance functions is available for the recognition of the failure and to evaluate the quality the detection of the failure has. In case of an absolute certainty that the failure is detected, the minimum value of 1 is filled into the column "detection probability" (D). If there is a substitute strategy, which allows avoiding the negative effects of the failure, a positive rating (low value) in

System element: Comparison unit								
Function: Receive nominal pressure								
Potential failure mode	S	Potential effect(s) of failure	Potential cause(s) of failure	Measures of prevention	O	Measures of detection	D	RPN
Store actual pressure too high	10	Received nominal pressure too high	Faulty controller specification of programming	Review and test procedures, estimate actual pressure	3	Checking and establishment of plausibility of the incoming and outgoing variables	3	90
> Store pressure too high	10							
>> Negative control difference	10		Controller hardware defect	Hardware tests	2	Checking of hardware during runtime	2	40
Store actual pressure too low	5	Received nominal pressure too low	Faulty controller specification of programming	Review and test procedures, estimate actual pressure	3	Checking and establishment of plausibility of the incoming and outgoing variables	3	45
> Store pressure too low	5							
>> Positive control difference	5		Controller hardware defect	Hardware tests	2	Checking of hardware during runtime	2	20
Don't store actual pressure	5	Nominal pressure not received	Controller hardware defect	Hardware tests, estimate actual pressure	2	Checking of hardware during runtime	2	20

Fig. 13 FMEA—form sheet for the function "Receive nominal pressure" of the system element "comparison unit"

the column of occurrence probability of the failure (O) is recorded. In this example it is possible to model the particle board press and to estimate the nominal pressure in a simulation. Therefore the plausibility of the value can be checked, the failure can be detected, and the substitute strategy of estimating the actual value can be applied. The weaknesses are ranked by risk priority numbers (RPN), which are derived for each failure by calculating the product of risk importance, detection probability and occurrence probability. Possible risk priority numbers are values between 1 (no weakness) and 1000 (extremely critical weakness).

4.2 Fault Tree Analysis (FTA)

While FMEA is used for qualitative failure analyses, the FTA enables quantitative analyses [3]. Thus it complements the FMEA and is especially effective when used in combination with it. The FTA analysis is classified as a Top-Down Analysis [2]. With the FTA [21] all possible failure combinations leading to an undesirable state

are detected. If the failure occurrence of the individual failures is known, the failure occurrence for the considered undesirable event can be calculated. In consequence if all of the individual failure rates are known, the failure rate of the different system failures may be calculated. The aim of the FTA is to define the failure combinations that lead to undesirable events and their occurrence frequency. The fault tree for an undesirable event can be derived from the malfunction structure by extending the cause-effect information with logical information. If a fault tree contains exclusively OR, AND and NOT links, the frequency of occurrence of an undesirable event can be determined with the application of Boolean algebra and probability theory. In order to determine the probabilities of failures, the parameters of the failure behavior must be established. The FTA is especially useful when identifying critical failure paths. It allows in particular the consideration of the effects of multiple failures. If an FMEA is already available, this information can be used as a basis for the FTA as a possible form of failure [2]. In what follows, the FTA will be discussed for the continuous thermo-hydraulic particle board press.

Illustration 4.2 FTA: "pressure control" for the continuous thermo-hydraulic particle board press.

One segment of the fault tree, that for the case of the pressure control not operating regularly (chosen wording "Do_not_calculate_correct_control_deviation"), is given in Fig. 14. In a fault tree the object name is given at the first position and the name for the operation responsible for the undesirable event at the second position (e.g. Fig. 14: ComparisonUnit.Do_not_calculate_correct_control_deviation). As already mentioned, individual object failures are linked by Boolean operators in such a way that the failure occurrence can be calculated using Boolean algebra and probability theory [20]. The occurrence of failures of the pressure control (p_{logic}) is calculated using the failure probabilities (p_1, p_2, p_3, p_4, p_5) of the different component failures for bus inlets, control logic, and the nominal pressure determination (see Eq. (1)).

$$p_{\text{logic}} = \big(1 - (1 - p_1) \cdot (1 - p_2) \cdot (1 - p_3)\big) \cdot \big(1 - (1 - p_4) \cdot (1 - p_5)\big). \quad (1)$$

The fault tree (Fig. 14) may also be derived from the technical system. If the nominal value of the pressure cannot be calculated, for example, the cause may be a bus problem (p_3, Message was not received), the encoder not working (p_2), or a faulty configuration or programming of the unit (p_1, reception of actual pressure too high/too low). The nominal pressure cannot be estimated if the frame model is not modeled (p_5) or the actual distance (p_4) is not received. The whole control deviation cannot be calculated if both the nominal pressure is not calculated and the estimated pressure is not available.

On the basis of qualitative (FMEA) and quantitative (FTA) risk assessment, measures to optimize system safety are initiated in a way that risks of individual failures and probabilities of system failure do not exceed specified limits. In the approach proposed by Rink, faults cannot be compensated according to the safety concept. If the unavailability of a function is detected the system is turned into a safe state until the error is corrected and the user is informed in addition. In the next section we

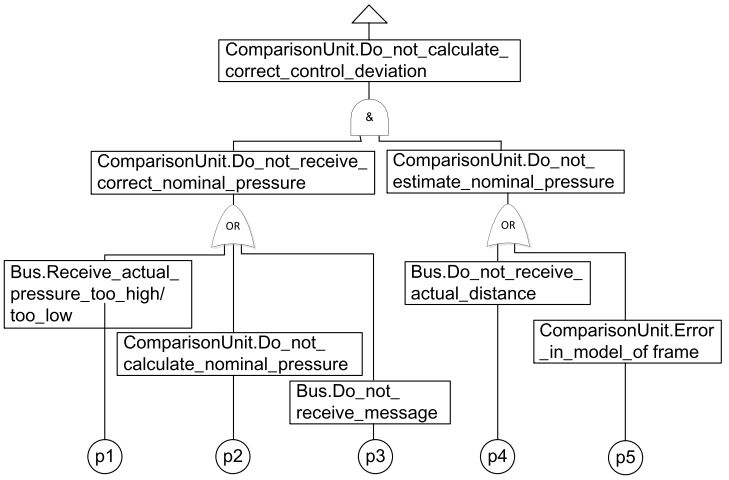

Fig. 14 Fault tree for the malfunction "Do not calculate correct control deviation"

argue why this strategy is appropriate and accepted for the automotive domain, but is not applicable to machine and production automation.

5 Coping with Faults in Production Automation Systems

After stops reached during the production process in production automation, restarts are time consuming and not always done easily. For this reason they are executed only when absolutely necessary, such as in situations of fire, hazard, or a long shut down as a result of cleaning and maintenance procedures. This is the case for the particle board press introduced in Sect. 2. Therefore, mechanisms are required to cope with the failures of one or more subsystems, devices, or sensors, and to operate the plant with a possible lower product quality until a regular shut down can be scheduled.

5.1 Adaptive Control by an Agent Based Approach

An[3] approach well suited to coping with failures of one or more subsystems is dynamic reconfiguration during runtime based on adaptive control systems using agent-oriented software. By implementing additional engineering models into the control code a basis for adaptive behavior can be reached. In the following the agent

[3]In the following the results of research of Wannagat and Vogel-Heuser will be presented which refers to [15–17].

definition of the VDI/VDE guideline 2653 is applied: “An agent is an encapsulated (hardware/software) entity with specified objectives. An agent endeavors to reach these objectives through its autonomous behavior, in interacting with its environment and with other agents. Agents represent a modeling concept for solving technical problems independently of a specific form of realization” [28].

A main part of the agents’ duties in the context of increasing availability of a control system is to detect, analyze, and handle faults. In the present state of the art, agents merely focus on instrument fault detection by using analytical redundancy between different measurement points. For every real sensor, which has functional dependencies to other sensors, we calculate additional virtual sensors using values of neighboring real sensors. The virtual sensors are used to validate the corresponding measurements and detect faults (parity space approach). If there is more than one virtual or real sensor available at one measurement-point, it is possible to detect a single fault (isolation). In principle, the virtual sensor values will never be as precise as real sensor values, but the information is beneficial for fault diagnosis purposes as well as for substitution. In the case of fault diagnosis the lack in precision may cause false alarms or a lack in sensitivity. In the case of substituting a real sensor by a virtual one, this loss of precision is relevant for closed loop controls and for the whole control strategy of the production process. Therefore it is insufficient just to calculate virtual sensors and to substitute for faulty real sensors. Additionally, the consequences of such substitutions and the constraints of the control system have to be taken into consideration. An agent knowledge base contains two main components—constraints and knowledge. Constraints define the margin of the action space that is used by the agents to take decisions. Use of basic engineering models representing knowledge about systems allows choice of alternative means to cope with failures at a certain point within the action space.

5.1.1 Requirements and Boundaries to Define an Action Space

The requirements defined for the agents, such as time and dependability as nonfunctional requirements, are the basis for specifying the action space (Fig. 15) and for defining the goals together with the parameters to achieve them.

In the first step, the requirements relating to contained modules, interfaces, and connections are collected separately for each of the system views presented in Sect. 2. Unlike the components, which can be related to more than one view, these requirements are strictly related to their own view. The same component can be viewed under different aspects. For example a valve is an actuator for the automation control system, it has a mechanical representation in the technical system, and it controls the flow rate from the technical process view. This leads to two advantages: firstly, it reduces the complexity because the requirements survey is distributed to three views for each element and secondly, it is the first integration of the requirements of different views in the same element (see Sect. 3).

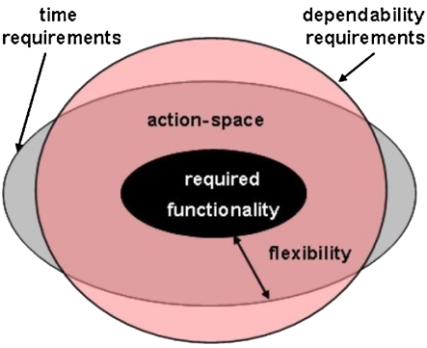

Fig. 15 Boundaries of the agent's action space [16]

In the second step, relations between functional and nonfunctional requirements defined in the first step need to be analyzed and linked to each other. The predefined functions of the modules are linked to the appropriate requirements and boundaries regardless of the three views. The result is a network of requirements and their relations, which makes it easy for the developer to get an overview of functionalities, requirements, and boundaries.

In the third step the developer evaluates if the desired flexibility may be reached with the predefined boundaries, and specifies how much flexibility the agents should have. The action space allows the agents to navigate and to achieve their goal within the appointed boundaries. Every parameter has to be checked regarding its influence on the time delay according to real time requirements and the failure probability of its related functionality in matters of dependability. Using these relations the agents are able to achieve their goal by changing the relevant parameters.

5.1.2 Diagnosis and Fault Management

To fulfill the requirements concerning the reliability of a plant, the agents have to know the effect of their actions as well as estimate the significance of a changing environment regarding their requirements within the whole system.

Similar to an FTA using the given modular structure, the developer is able to specify the relationship between the functionality of a module and its sub modules (Fig. 16). The goal is to specify the probability for correct execution of each function based on functions of the related subsystem, until the basic elements of the control system at the bottom level are reached. In this way it is possible to calculate the probability of a malfunction at the time a quality of a sensor measurement changes. The tolerance model is similar to a fault tree, but focuses not only on the top event, the function or the malfunction of the system and its probability, but also on the quality similar to a QoS with which the operation will continue under the prerequisite of a given replacement strategy (discussed in Sect. 5.1.3). This is realized by software agents, who implement the knowledge about the relation between the

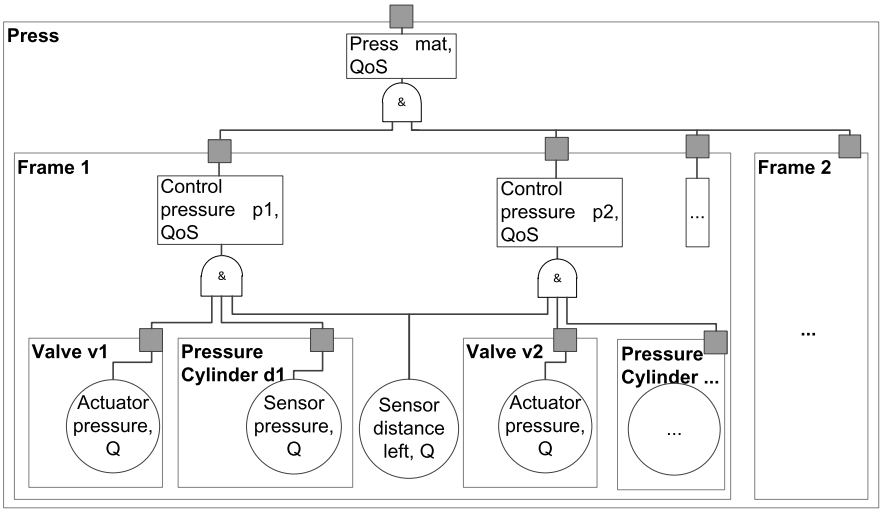

Fig. 16 Tolerance model combined with modular structure (*boxes* show module, *small grey squared boxes* depict interfaces between modules)

quality of a control system element, for example the quality of the sensors' measurements, the precision of the actuators' actions, and the functionality of a component. This relation is used to calculate both the risk of a failure regarding the observed changes and the effect of possible counteractions such as replacement of a sensor by a virtual (calculated) sensor value. Each agent is able to observe all elements of the control system and to relate the real values to the calculated values of its internal system model.

5.1.3 The Knowledge Base

Next, a knowledge base, which allows detecting sensor failures, calculating a surrogate value, and estimating the resulting precision at runtime, will be introduced. One important point for the design of such a knowledge base is that it is easy to design and implement in a Programmable Logic Controller (PLC) environment so as to be calculated during runtime. A very simple and powerful notation for this purpose, which is well known in the domain of automation, is the directed graph [20]. In this graph, each node represents a measurement point. It is equipped with a value source that can be either a real or a virtual sensor. A quality value at each node describes the accuracy of the measured or calculated value. The quality value ranges continuously from 0 to 1. The edges of the graph describe functional correlations (f, Fig. 17) between the measurement points and represent the analytical dependencies which are used to calculate virtual sensor values at runtime. The directions of the arrows indicate sensor values that are appropriate for a substitution (Fig. 17). In Fig. 17 on the right side, the sensors used within the quality model are shown.

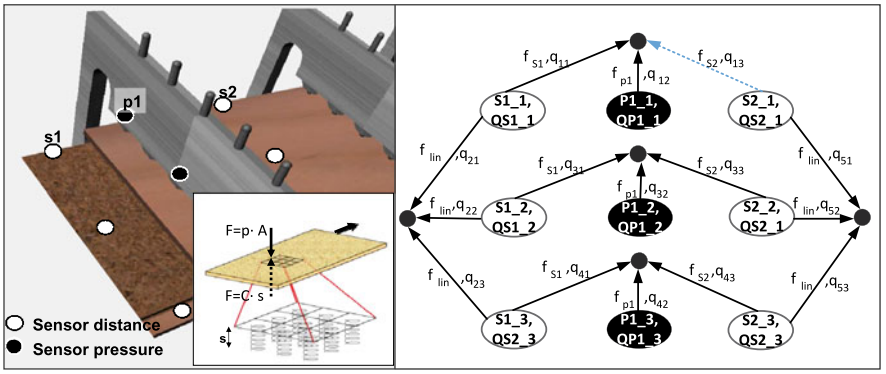

Fig. 17 Analytical dependencies of sensors in the hydraulic press in the quality model [15]

Sensor "$S2\_1$" can be calculated, using the function "f_{S2}" (edge) and the sensor values "$P1\_1$" and "$S1\_1$", for example. The function "f_{S2}" expresses the dependency between the thickness of the incoming material into the frame concerned (one of the sensors s_1), the pressure of the hydraulic cylinder (one of the sensors p_1), and the thickness of the outgoing material, using a spring model. The spring constant (C, Fig. 17, bottom left) represents the elasticity of the material and depends on the values of the actual temperature, the density, and the humidity of the wood. The black dots are used if more than one sensor is required to calculate a virtual sensor (Fig. 17, right).

It is possible to use virtual sensors as source for other virtual sensors and the probability for that rises with the number of failures and corresponding substitutions. The precision of virtual sensor values is possibly reduced by inaccurate models and time aspects, e.g. dead time or delays because of the underlying measurement, the field bus, or the calculation of virtual sensor values in the PLC. Reduced precision lowers the quality of the virtual sensors compared with original measurements. This loss of precision is given by a quality factor q similar to QoS for a measuring value (q, Fig. 17, right) which is bound to every arrow of the graph and described by values between 0 and 1. It represents the decreased quality (precision) by using this calculated virtual sensor instead of the real sensor. In addition a quality value Q (Q, Fig. 17, see QoS Sect. 3) at every node represents the precision of the real sensor itself. Therefore, the quality (precision) "q_{13}" of a virtual sensor replacing "$S2\_1$" may be calculated by the precision of the real sensor quality value "$QS1\_1$" of "$S1\_1$" and the quality value "$QP1\_1$" of the real pressure "$P1\_1$" multiplied by the loss of the quality because of the replacement represented by "q_{21}" as quality factor.

Furthermore, the agents use the quality value of a virtual sensor to determine the effect on the availability of the plant operation and to compare it with the given requirements and constraints. The reliability of sensor values is evident for processing automated production systems. Although the substitution of real sensors with calculated virtual sensors increases readiness in case of partial faults, it risks the accuracy

of the process flow. While the correctness of possible alternative strategies for static systems is determined during development time, an agent based dynamic system decides this during runtime. Both have to decide whether the production process can be continued with replaced, calculated values or if it has to be suspended. The threshold, which defines the agent's decision, is oriented by a user defined safety requirement for the specific part of the process or the technical system. The loss of a sensor triggers the reconfiguration of the control behavior automatically. The automated result may be a single parameter adjustment or an immediate shut down of an entire plant and is characterized as dynamic reconfiguration at runtime.

The introduced concept was evaluated successfully by applying it to the particle board press. The reconfiguration took a maximum of two PLC cycles to adjust. Further information about the application is given in [17].

6 Food for Thoughts

There is still a great need for engineering support for adaptive control systems in manufacturing, to increase dependability as well as their interaction with the human as the operator, and by that adaptable behavior for individuals. Our contributions in the future will focus on these tasks. Besides this, a lack in visualization of adaptive control systems is clearly apparent and needs to be covered in future research. As the acceptance of any new technology is based on transparency and trust, we need to find appropriate visualization patterns to open the black box of intelligent behavior for operators, to gain acceptance and trust. On the other hand we need to provide automatic models derived from engineering data to reduce modeling effort for industrial application. This is another challenging application-oriented research topic, because it is strongly related to different domain-specific models in different tools and their interfaces.

7 Summary

The introduced methods and measures of an integrated safety and functional analysis, including the different views in a production automation system, help to reduce time to market and cost, as mentioned in the introduction. There is still a lack of meta models for the different disciplines and domains as well as in a support of integrating different models under the constraint of changes in variants and versions. In tool support some deficiencies must be accepted as well, but the basic concepts are available, which provide the foundation for necessary improvements. We have introduced three first steps on the road to an integrated modeling of safe and dependable complex systems: the cross-disciplinary modeling with SysML, a first approach of integrating safety and functional modeling, and finally the adaptive behavior based on basic engineering models to support dynamic reconfiguration of manufacturing systems during runtime.

References

Selected Bibliography

1. A. Avizienis, J.-C. Laprie, B. Randell, C.E. Landwehr, Basic concepts and taxonomy of dependable and secure computing. IEEE Trans. Dependable Sec. Comput. **1**(1), 11–33 (2004)
2. B. Bertsche, P. Göhner, U. Jensen, W. Schinköthe, H.-J. Wunderlich, *Zuverlässigkeit mechatronischer Systeme. Grundlagen und Bewertung in frühen Entwicklungsphasen (Foundations for a Reliability Evaluation of Mechatronic Systems. Reliability of Mechatronic Systems)*. VDI-Book (Springer, Berlin, 2009), pp. 7–45
3. R. Lauber, P. Göhner, *Prozessautomatisierung (Process Automation), 1* (Springer, Berlin, 1999)
4. F. Li, G. Bayrak, K. Kernschmidt, B. Vogel-Heuser, Specification of the requirements to support information technology-cycles in the machine and plant manufacturing industry, in *14th IFAC Symposium on Information Control Problems in Manufacturing (INCOM)* (2012)
5. A. Lüder, L. Hundt, A. Keibel, Description of manufacturing processes using AutomationML, in *2010 IEEE Conference on Emerging Technologies and Factory Automation (ETFA)* (2010), pp. 1–8
6. A. Rink, Entwicklung einer Methode für die systemtechnische Auslegung verteilter und sicherheitskritischer Führungsfunktionen für Fahrzeugantriebe (Development of a method for the systemic interpretation of distributed, safety critical functions for vehicle engines). PhD Thesis, Bergische Universität Gesamthochschule Wuppertal, Wuppertal, Germany (2002)
7. A. Rink, Systemtechnische Auslegung sicherheitskritischer Führungsfunktionen für Fahrzeugantriebe (System-oriented design of safety critical control functions for vehicle drives). Autom.tech. Prax. **11**, 66–74 (2002)
8. A. Rink, B. Vogel, Objektorientierte Methode für die Auslegung eines verteilten Automatisierungssystems unter Sicherheitsaspekten (Object-oriented method with safety aspects for the design of a distributed automation system), in *Tagungsband zur Fachtagung Verteilte Automatisierung: Modelle und Methoden für Entwurf, Verifikation, Engineering und Instrumentierung*, 22.–23.3.2000, Magdeburg (2000), pp. 266–272
9. A. Rink, B. Vogel-Heuser, Auslegung von Führungsfunktionen für einen Fahrzeugantrieb anhand eines integrierten Prozessmodells unter Berücksichtigung der Sicherheitsaspekte (Design of a key function for a vehicle drive using an integrated process model taking safety aspects under consideration), in *VDI Kompetenzfeld Informationstechnik, Software-Engineering in der Praxis, Tagung*, 28.02.–01.03.2002, Düsseldorf. VDI-Berichte, vol. 1666 (VDI Verlag, Düsseldorf, 2002)
10. D. Schütz, B. Vogel-Heuser, Modellintegration von Verhaltens- und energetischen Aspekten für mechatronische Module (Integrated modeling of module behavior and energy aspects in mechatronics). Automatisierungstechnik **59**(1), 33–41 (2011)
11. D. Schütz, A. Wannagat, Domänenspezifische Modellierung für automatisierungstechnische Anlagen mit Hilfe der SysML (Domain specific modelling of technical facilities using SysML). Autom.tech. Prax. **3**, 54–62 (2009)
12. K. Thramboulidis, The 3 + 1 SysML view-model in model integrated mechatronics. J. Softw. Eng. Appl. **3**, 109–118 (2010)
13. K. Thramboulidis, D. Soliman, G. Frey, Towards an automated verification process for industrial safety applications, in *IEEE Conference on Automation Science and Engineering* (2011), pp. 482–487
14. B. Vogel-Heuser, G. Kegel, K. Bender, K. Wucherer, Global information architecture for industrial automation. Autom.tech. Prax. **1**, 108–115 (2009)
15. A. Wannagat, Entwicklung und Evaluation agentenorientierter Automatisierungssysteme zur Erhöhung der Flexibilität und Zuverlässigkeit von Produktionsanlagen (Development and evaluation of agent oriented automation systems to increase flexibility and reliability of production plants). PhD Thesis, Technische Universität München, Munich, Germany (2010)

16. A. Wannagat, B. Vogel-Heuser, Agent oriented software-development for networked embedded systems with real time and dependability requirements the domain of automation, in *Proc. of 17th IFAC World Congress* (2008), pp. 4144–4149
17. A. Wannagat, B. Vogel-Heuser, Increasing flexibility and availability of manufacturing systems—dynamic reconfiguration of automation software at runtime on sensor faults, in *Proc. of the 9th IFAC Workshop on Intelligent Manufacturing Systems* (2008)

Additional Literature

18. J. Bézivin, Model driven engineering: an emerging technical space. Lect. Notes Comput. Sci. **4143**, 36–64 (2006)
19. J. Bézivin, On the unification power of models. Softw. Syst. Model. **4**, 171–188 (2005)
20. G. Chartrand, Directed graphs as mathematical models, in *Introductory Graph Theory* (Daver, New York, 1985), pp. 16–19
21. Deutsches Institut für Normung, *DIN 25424: Fehlerbaumanalyse (Fault Tree Analysis)* (Beuth, Berlin, 1981), Part 1; (1990), Part 2
22. Deutsches Institut für Normung, *DIN EN 60812: Analysetechniken für die Funktionsfähigkeit von Systemen – Verfahren für die Fehlzustandsart- und -auswirkungsanalyse (FMEA) (Analysis for the Functionality of Systems—Method for the Failure Mode and Effects Analysis)* (Beuth, Berlin, 2006)
23. IEEE Std. 610.12-1990, IEEE standard glossary of software engineering terminology. The Institute of Electrical and Electronics Engineers, USA (1990)
24. ISO/IEC FCD 25010, Systems and software engineering—software product quality requirements and evaluation (SQuaRE)—quality models for software product quality and system quality in use
25. D. Straub, Engineering risk assessment, in *Risk – A Multidisciplinary Introduction*, ed. by C. Klüppelberg, D. Straub, I. Welpe (2014)
26. SysML specification, 06/2012: available at http://www.omg.org/spec/SysML/1.3/PDF/
27. VDA, *Qualitätsmanagement in der Automobilindustrie, Sicherung der Qualität vor Serieneinsatz, System – FMEA (VDA: Quality Management in the Automotive Industry, Securing of Quality Prior to Production*. VDA Brochure, vol. 4, 1st edn. (VDA, Frankfurt am Main, 1996), Part 2
28. VDI/VDE, *Guideline 2653: Agents in Industrial Automation, Part I* (Beuth, Berlin, 2010)