Title page with author details

Success factors for the design of field-level control code in machine and plant manufacturing
- an industrial survey-

Birgit Vogel-Heuser[1] (corresponding author), vogel-heuser@tum.de, ORCID: 0000-0003-2785-8819,
Juliane Fischer[1], ORCID: 0000-0002-3218-7397,
Eva-Maria Neumann[1],
Matthias Kreiner[1]

Affiliation of authors with
[1]Institute of Automation and Information Systems
Technical University of Munich
Garching near Munich,
Germany

Noname manuscript No.
(will be inserted by the editor)

Success factors for the design of field level control code in machine and plant manufacturing - an industrial survey -

Birgit Vogel-Heuser[1] (ORCID: 0000-0003-2785-8819) · Juliane Fischer[1] · Eva-Maria Neumann[1] · Matthias Kreiner[1]

Received: date / Accepted: date

Abstract The amount of software in automated production systems, including its development effort, is continuously increasing to currently up to 35-50% of the development personnel. Consequently, success factors for achieving modularity and complexity management of control software are of high economic interest. Scientific solutions are manifold but often not implemented in industry. This paper introduces the study $QoaPS_{SWE}$ (Quality of automated Production Systems' Software Engineering) providing insights into 61 machine and plant manufacturing companies to give quantitative and qualitative results to five essential research questions on success factors in the design of field-level control code. Compared to preceding surveys, $QoaPS_{SWE}$ achieves statistically significant results for software maturity (M_{MOD+}), complexity, and model-based software engineering and provides detailed insights into causes and consequences for single criteria, thus clearly identifying obstacles to be addressed in future research and with industrial countermeasures. Especially staff qualification and organizational issues are identified as obstacles to applying the object-oriented programming paradigm for control software in machine and plant manufacturing. Validity is ensured by analyzing the statistical significance of the results in addition to comparisons with earlier surveys and interviews as well as the comparison with already existing and accepted maturity levels. The provided qualitative and quantitative results will allow the benchmarking of companies' maturity and the derivation of concrete recommendations for companies depending on their M_{MOD+} value and the evaluated characteristics.

Keywords machine and plant manufacturing · automated production systems · software maturity indicator · software engineering · variant and version management · modularity · reusability · benchmark

1 Motivation and Introduction

Machine (MM) and plant manufacturing (PM) companies have to meet various challenges, including small lot sizes, high variability, and lifecycles that may last up to 50 years [1]. In Accordance to [2], [3], [4] machines or plants are referred to as automated Production Systems (aPS) in the following. aPS manufacturers in Germany face an increasingly strong worldwide competition. Used to being world-leading in the export of aPS, companies recently struggle to remain competitive, especially regarding cost and need to consider relocation of engineering departments to countries with lower wages. In addition, the large and increasing relevance of software in aPS indicates an economic need to improve modularity and complexity management strategies. More precisely, the proportion of system functionality realized by SW is growing [5], leading to a steady increase in the complexity of aPS control software (referred to as SW hereafter). Thus, concepts for supporting SW engineers in handling this complexity and maintaining the developed SW are required, e.g., by establishing modular structures and tool-based variant and version management. Therefore, the success factors for the design of field-level control software in the domain of aPS are investigated within our study $QoaPS_{SWE}$ introduced in this paper.

[1]Institute of Automation and Information System, Technical University Munich, Garching near Munich, 85748, Germany
Tel.: +49 (0)89 / 289-16400
Fax: +49 89 289 16410
E-mail: vogel-heuser@tum.de

The five targeted research questions (RQs) address only results that are accessible with a questionnaire and are introduced in the following:

RQ1. Is it feasible to divide the companies into different industrial sectors and does this provide additional insights?
Aim: compare maturity in a specific sector or in between MM and PM.

RQ2. Are correlations of maturity values statistically significant with n>60 participating companies?
Aim: provide measurable benchmark criteria.

RQ3. What are the reuse strategies in different disciplines?
Aim: identify key factors to reduce cost and increase quality.

RQ4. What are the causes and consequences of control SW complexity?
Aim: analyze complexity as a potentially disturbing influencing factor for maturity.

RQ5. Which strategies for reuse and design of control SW are successfully or unsuccessfully applied?
Aim: identify obstacles and correlations of e.g. reuse strategy and number of developers in the department.

For each RQ, detailed hypotheses are derived from the state of the art and current industrial challenges in the subsequent section (cp. summary in Table 1 depicting second-level hypotheses). To ensure validity, the findings are examined statistically and compared to existing research and survey results, e.g., results are investigated using the already established maturity levels proposed by Antkiewicz [6]. This paper also aspires to identify weaknesses in automation, including SW engineering of control SW, to derive concrete measures to improve SW maturity.

The main contribution of this paper is the introduction of a questionnaire Q4 and its results, including expert responses from a former questionnaire Q3, that provide an assessment of the maturity and success factors for the design of field-level control SW in MM and PM companies as a first step to identify weaknesses and strengths to improve a company's position at the global market. Whenever possible, quantitative and, if not possible, selected qualitative results (Q3) are delivered concerning five essential RQs introduced above based on 61 German MM and PM companies in Q4. Qualitative results are gained from Q3 especially regarding the engineering workflow for example the usage of universal modules (in electrics/ electronics (EE) with 30%, in Human Machine Interfaces (HMI) with 26% and in SW with 18%) or automatic generation from documents to engineering artefacts with highest values for EE followed by mechanics (M) and SW. $\text{QoaPS}_{\text{SWE}}$ exceeds previous surveys [7], [8], [9] in providing significant results regarding, e.g., SW maturity ($\text{M}_{\text{MOD+}}$), complexity, and Model-Based Software Engineering (MBSE) more precisely modeling tools used (cp. Table 14). To support future research, individual obstacles of the industry are identified, including insufficient staff qualification and organizational issues that hinder for example the adoption of the object-oriented programming paradigm.

The remainder of this paper is structured as follows: Based on the state of the art (Section 2), the relevance of the five RQs is substantiated and concrete hypotheses are derived. Section 3 introduces the research method used in $\text{QoaPS}_{\text{SWE}}$. The included MM and PM companies, the features of the questionnaire, the experimentation, including the selected statistical methods, and threads to validity are introduced. Sections 4 to 8 discuss the identified results for the five RQs and the corresponding hypotheses. Section 9 summarizes the results and provides an outlook on future work.

2 Related work on software modularity and reuse as a means to achieve high maturity in the design of control software for aPS

The state of the art is organized in five sub-sections. First, characteristics of aPS are introduced, including platforms, programming languages, and SW architectures, followed by a description of aPS manufacturers to understand the domain's requirements and constraints better. Since the use of models in the development process has been promoted as a solution to cope with the complexity of SW development [10], the state of the art of MBSE in aPS is presented. Subsequently, the challenges of managing reuse in a mechatronic context are discussed, as well as strategies to tackle variant and version management and software complexity. Concluding, preceding surveys that monitored prevailing challenges in the domain of aPS are presented, to be supplemented and extended by this work. In the following the detailed RQs will be derived from the state of the art and summarized in Table 1.

2.1 Platforms, programming languages, and software architectures for field-level control software in aPS

Control software in aPS is confronted with specific conditions that differ significantly from classic high-level language applications, e.g., regarding the available programming platforms, languages and the structural elements to design the software architecture.

Table 1: Research questions - overview $QoaPS_{SWE}$ (all based on Q4 besides RQ3) MM - machine manufacturer, PM - plant manufacturer

Research Question	**Related hypotheses**
Is it feasible to divide the companies into different industrial sectors and does this provide additional insights? (RQ1)	H1.1 Companies assign themselves appropriately to industrial sectors and MM vs. PM respectively H1.2 Quantitative analyses are possible due to sufficient companies per cell
Are correlations of maturity values statistically significant with $n > 60$ participating companies? (RQ2)	H2.1 There is a correlation in between M_{MOD+} and M_{TEST} H2.2 M_{MOD+} differs significantly in between MM and PM H2.3 Negative correlation in between self-assessment of control software complexity and M_{MOD+}. Positive correlation of self-assessment of companies' ability to manage control software complexity and M_{MOD+} H2.4 M_{MOD+} is correlated with company size/size of application software development department
What are the reuse strategies in different disciplines? (qualitatively) (RQ3, Q3)	H3.1 Maturity M_{MOD} is applicable to different disciplines H3.2 Universal Module is used as reuse strategy in different disciplines on the one hand and related to the company size on the other hand H3.3 Availability of documents is poor in general when engineering starts and differs due to discipline and company size H3.4 Level of detail of In-House guidelines varies between disciplines H3.5 Automatic generation of information in engineering is still rare in all disciplines
What are the causes and consequences of control software complexity? (RQ4)	H4.1 Self-assessment of control software complexity much better applicable compared to objective measures like LOC, number of POUs H4.2 Key Drivers of complexity are customer requirements and variability H4.3 Self-assessed complexity of control software in PM > MM H4.4 Self-assessed manageability of control software projects' complexity depends on type of simulation approach/tool H.4.5 Modularization and training are means to manage complexity
What strategies for reuse and design of control software are used? (RQ5)	H5.1 Analysis of reuse strategies and their relations reveal additional insights H5.2 Analysis of variant and version management and tools used reveal additional insights H5.3 Reasons for and effects of OO IEC usage can be identified H5.4 Software departments' agility relates to the number of application software engineers H5.5 Company specific guidelines/checklists are used more frequently than PLCopen

Programmable Logic Controllers (PLC) are still the typical hardware platforms to control machines and plants. They process sensor signals from the aPS in cycles consisting of read, calculate & control, and write in tasks with designated cycle-times to fulfill real-time requirements, meaning the assigned cycle times may never be exceeded. The commonly used IEC 61131-3 programming standard for PLCs defines two textual languages - Structured Text (ST) and Instruction List (IL) - and three graphical ones - Ladder Diagram (LD), Function Block Diagram (FBD) and Sequential Function Chart (SFC).

Apart from IEC 61131-3 and PLCs, also high-level/ object-oriented programming languages and more sophisticated control platforms are applied in the industry. Werner [11], Bonfé et al. [12], and Vogel-Heuser [13] highlighted the benefit of the object-oriented programming paradigm (OO). Sometimes, IEC 61131-3 Function Blocks, which need to be instantiated prior to their use, are considered as a pre-step to the object-oriented extension of IEC 61131-3 (OO IEC), which can lead to confusion regarding the terminology of OO. Tool support for OO IEC has been available for selected runtime environments for a decade [11], yet it is not frequently used in industry. Although OO IEC is applicable to all IEC 61131-3 programming languages, our experience from analyzing the small number of OO IEC applications from industry known to us showed that ST is generally used for the implementation which may be due to its similarity to high-programming languages. This assumption was confirmed by an industry expert from the IEC platform provider sector who has had access to a large number of customer projects since the introduction of OO IEC. As highlighted by the authors mentioned above, the usage of OO IEC should tremendously ease modularity, reuse, and managing variants and versions. Therefore, the relation between OO IEC and reuse strategies is analyzed as well as the reasons for and the identified effects of OO IEC usage (RQ5: H5.1 and H5.3).

In [8], the usage of OO IEC as one potentially promising paradigm to structure SW was analyzed: 42% of the participating companies did not use OO IEC at all, 48% partially, and 10% by default. However, neither dependencies between the used controller brand, the company's industrial sector and the engineering process/programming paradigm nor dependencies between usage of OO IEC and SW maturity could be proven. Therefore, Q4 aims to identify correlations of OO IEC usage with other variables (H5.3.1 considers correlations with five selected aspects, cp. Table 12). Case-sensitive additional questions (e.g., H5.3.2 - H5.3.4) ask for advantages gained and difficulties encountered by introducing OO IEC on the one hand and reasons for not using OO IEC on the other. As interfaces are one central element of OO IEC, H5.1.2.2 explicitly analyzes their benefit and impact.

As appropriate SW architectures are crucial to ensure high SW quality and to enable reusability [4], [14], they are discussed in the following. The SW architecture describes "the general structure of a system, usually expressed in components, interfaces and their interconnection" [15]. IEC 61131-3 defines so-called Program Organization Units (POUs) to enable the encapsulation of PLC programs in reusable units, which represent the components of a PLC SW project. Thereby, three types of POUs are distinguished, namely Programs, Function Blocks and Functions.

Vogel-Heuser et al. [16] confirmed the five architectural levels proposed by ISA 88 [17] by analyzing the SW architecture of companies from the MM and PM industry. Further, [18] confirms that the reuse potential of application-specific POUs correlates with the architectural hierarchy levels they are located on: the higher the level of a POU, the lower is its potential for reuse.

Vyatkin [19] introduces a SW architecture for distributed automation systems based on the IEC 61499 standard. This approach results in SW showing a composite structure and consisting of event-driven Function Blocks (FBs), which are used to describe processes. This approach's primary benefits are reduced time and effort to develop automation SW, a high degree of code modularity, and a high potential for reuse. These benefits are already proven through first industrial applications. However, this standard is not commonly used yet and "[...] has [still] a long way in order to be seriously considered by the industry" [20]. Consequently, we focus on IEC 61131-3 software.

2.2 Characteristics of aPS suppliers

Complexity and the number of variations resulting from customer-specific requirements, as well as the degree of on-site changes, are increasing from MM to PM according to prior work [4]. Furthermore, different application domains (industrial sectors) use different guidelines (e.g., SAIL in the intralogistics domain [21] or PackML in the Food and Beverage sector [17]) and need to consider distinct laws and standards, e.g., quality management in medical application (MedTech) [22]. Thus, to properly conduct a detailed analysis, the differentiation of companies into MM or PM and industrial sectors would be helpful and is addressed by RQ1. In prior questionnaires (cp. Table 14), the number of companies per cell was not enough to achieve statistically relevant results.

Further, the authors in [4] divide the lifecycle of aPS into two main phases: At first, engineering, which includes testing/quality assurance, and second, after acceptance test, operation and maintenance. Due to the nature of aPS, changes resulting from new products, unforeseen raw material, or environmental conditions often have to be made on-site [4], [16].

Additionally, prior research results [7], [8] highlight the influence of company-specific constraints on SW development. Further, [23] emphasizes the impact of the development team size on applicable reuse strategies in SW development and their potential benefits. For example, with evolution over time, universal software modules often tend to get quite complex and, consequently, changes in these modules are often not easy to detect. While a relatively small SW department is agile enough to allow direct communication of all developers with low organizational effort, e.g., in an open-plan office, the change management becomes increasingly complex and demanding for larger departments, which can even be spread across different company locations. In conclusion, the same reuse strategy (universal software modules) can have varying effects depending on company-specific constraints [23]. Thus, H2.4 considers the correlation of M_{MOD+} with organizational aspects. Furthermore, in the scope of suitable reuse strategies, the relation between a SW department's agility and the number of application SW engineers is investigated with H5.4.

2.3 Model-based software engineering in aPS

The use of models during the SW development process has been promoted as a solution to cope with the complexity of SW development [10]. Lately, research in MBSE has focused on new methods and notations to ease the development of control SW, reduce cost, and support maintainability and evolvability [4], [12], [24]. However, there exists a large gap between legacy control code (i.e., PLC code) and code gained through

newer MBSE approaches based on the Systems Modeling Language (SysML) or the Unified Modeling Language (UML) [12] in aPS companies. To bridge this gap, both code refactoring and the building of appropriate SW components are essential.

Estévez et al. propose an approach for the automatic generation of PLC programs using a component-based model of the IEC 61131-3 to define the SW architecture utilizing a markup language [25]. An approach in the process industry uses documents from the development process to identify recurring hardware combinations and the subsequent automatic generation of the required interlocks in the control SW [26]. Ladiges et al. [27] present a concept for modular process plants, which is based on the module-type package (MTP) standardized in [28]. This approach uses the PackML state machine for handling operating modes and also includes the HMI functionality. The approach is based on a standardization of all mechatronic disciplines, which can then be combined at the interfaces and thus be reused. Priego et al. [29] introduce UML profiles from three different views (control, electrics/ electronics, and SW engineering) and define mappings between them. Through model transformations, PLCopen-compliant IEC 61131-3 SW is generated from UML models.

These examples demonstrate that various MBSE approaches with different aims exist in the aPS domain. In a recent survey, Koziolek et al. [30] compared 13 automated code generation methods considering their inputs, transformations, and outputs. In conclusion they identified the merging of generated code and manually developed code as a point for future research as well as the need to establish a benchmark process to support practitioners in comparing the different approaches and choosing one.

Given the MBSE approaches from academia and the vast benefit already realized, e.g., in the automotive industry [31], the extent to which they are beneficially used to cope with complexity shall be discussed (H4.4). A further challenge for using MBSE approaches in aPS is the mostly customer-specific design of these systems. Therefore, variant and version management has to be tackled (cp. Section 2.4).

Various development processes for complex products exist that can be applied to the development of mechatronic products, including machines and plants. These range from established ones such as the V-model to more recent ones like Scrum. Scrum helps cross-functional teams accelerate development by breaking down a complex project into smaller chunks, which are realized in so-called sprints in case of deviations from the project schedule. Instead of adhering to regular intervals, meetings are held according to necessity, usually daily, during these sprints [32].

The collaboration of different disciplines, mostly mechanics (M), electrics/ electronics (EE), and SW, still poses a significant challenge in developing mechatronic products [33]. Model-based approaches and modeling tools exist not only for SW development but also for the other disciplines involved in the development of aPS. Mechanical engineers express a lot of their expertise in Computer Aided Design (CAD) data. In contrast, electrical engineers use Electrical Computer Aided Engineering (ECAE) tools ranging from the generic Excel to domain-specific tools, like EPLAN or Zuken E^3. Alvarez et al. [24] propose to combine the methods, techniques, tools, and standards from the aPS domain with methodologies from SW engineering to cover different phases of the life cycle. For this purpose, they take different modeling methods from the aPS domain to support users with their tasks while guiding them through the development phases.

This section is only a brief insight into the vast amount of modeling tools tailored to specific tasks within one discipline or aiming at supporting interdisciplinary development by combining information from different disciplines. Within this paper, we distinguish between IT-oriented modeling tools supporting the usage of modeling languages from the field of software and systems engineering (e.g. Unified Modeling Language (UML) or Systems Modeling Language (SysML)) and control-oriented modeling tools to model and simulate technical or physical systems based on numerical calculations (e.g. MATLAB / Simulink). Despite standardized exchange formats such as the XML-based AutomationML (AML) [34], many tools use proprietary formats, which complicate the continuous use of data throughout the development process, e.g., for the generation of information (H3.5).

2.4 Reuse strategies in the aPS domain: software configuration, variant and version management, and software complexity

Generally, reuse strategies in SW development can be distinguished into planned and unplanned forms [35]. Currently, during PLC SW development, the most common reuse strategy applied is still the unplanned method Copy, Paste, and Modify (C/P&M) [7], [18], [36]. However, it is known to be error-prone and time-consuming while requiring detailed knowledge of existing SW variants and versions [36]. While companies are aware of the downsides of unplanned reuse, research nonetheless shows a large quantity of such problematic cases [37].

Reasons include designers' lack of motivation to innovate and a general mistrust in solutions developed by others that are not immediately understood (the "not invented here" syndrome).

For planned reuse of IEC 61131-3-compliant SW, different strategies exist, with most of them proposing a modular SW structure. In the scope of this paper, a module is considered "a SW unit that can be reused without any modifications in different SW systems" [10]. Depending on the company's programming style, a module might be an individual POU or a group of POUs. The reuse potential and organizational effort of POUs highly depend on their granularity [38]: the higher the granularity, the higher is the reuse potential, but at the same time also the organizational effort for combining the larger number of POUs increases. Thus, Maga et al. [38] postulate that SW units should be managed according to their level of granularity.

Similarly, Gauss et al. [39] propose a modularity-based design method for mechanical engineering (M engineering), focused on the development of highly reactive and reconfigurable production systems. Thereby, shorter lead-time and lower development cost can be realized. However, the authors acknowledge the necessity to investigate whether the approach is compatible with EE and SW engineering. This risk of incompatibility is a prevailing challenge, since reuse strategies are commonly developed for individual disciplines, not considering the cross-disciplinary nature of aPS. Therefore, the commonalities and differences of strategies in different departments are investigated (RQ3), including universal modules (H3.2), availability of documentation (H3.3), use of internal guidelines (H3.4), and automatic generation of engineering documents (H3.5).

According to [35], a combination of different reuse strategies is required to reach a suitable reuse base. A previous study in the aPS domain confirms this view as it identified that market-leading companies often do not pursue a single strategy for planned reuse, but instead use mixed forms combining templates and library blocks, i.e. function blocks [9].

The development and reuse of library blocks is a common strategy. In terms of standardization, library blocks range between vendor-specific (to ease controlling a vendor's hardware) and company-specific, which can be further distinguished into company-wide or machine-type-specific libraries [18]. However, regarding the release processes of library blocks, [8] showed that even though these processes are almost standard in industry, they are not well designed in most cases. If a company develops libraries for company-wide use, designated module developers often program them in separate departments. To ensure that the integration of such modules inside the application SW is comprehensible for application engineers and supported by the chosen SW architecture, it is expected that guidelines and checklists for the application engineering exist, which is investigated in H5.1.2.1 (cp. Table 12). The effectiveness of guidelines was investigated by Reimlinger et al. [40], who used differently experienced participants from several countries in a mobile eye-tracking study, finding that novice design engineers perform well by closely interacting with the provided documents.

To support the development of library blocks, templates as predeveloped formats can be used. Thereby, templates exist for individual POUs as well as for entire control SW projects. They are used frequently in combination with MBSE approaches and code generation. A reuse strategy similar to templates is the use of design patterns, which describe abstract solutions to frequently recurring problems. The Gang of Four originally defined 23 design patterns in OO programming [41] but also approaches in the aPS domain exist. Barbieri et al. [42] derive design patterns for PackML-compliant PLC SW, including the realization of operating modes, Bonfé et al. [12] derive design patterns for mechatronics systems, and Fuchs et al. [43] identified five architectural design patterns in industrial PLC SW, which were formalized in [23].

In respect to reuse, renowned researchers including Egyed, Fay, Prähofer, Schaefer, Tichy, and Vogel-Heuser (cp. e.g., [4], [36], [44]) identified variant and version management as a prerequisite for proper SW engineering and planned reuse in the aPS domain. In aPS, variant and version management is particularly challenging, as new variants are often derived from existing legacy SW without planning or documentation [18]. Further, due to parallel operation with different machines for different customers at different sites, faults may occur in a derived variant before they arise in the initial variant, which complicates bug fixing [8]. Thus, approaches to improve and support variant and version management in the aPS domain are required.

In a planned manner, the variability of aPS SW, which usually has to be adapted to customer-specific requirements, can be addressed with so-called parameterizable, universal modules [7], [18]. They allow configuring the SW according to the requirements. In a recent survey [9], 16% of the participating companies confirmed this and 25% rated it as partially true. However, using universal modules with the correct parameters is often not a trivial task. Furthermore, universal modules may lead to high amounts of dead code.

These different planned reuse approaches, which are often used in combination, highlight the need for more detailed, quantitative results for relations of different

reuse strategies or sub-strategies, e.g., libraries, templates, and universal modules. They are addressed in H5.1.1 and H5.1.2 (cp. Table 12).

Regarding variant and version management, Software Product Line Engineering (SPLE) is a well-established approach in the informatics domain for planned reuse in variant-rich SW systems. Thereby, SW parts common to all variants are ideally implemented only once [45]. Various approaches use reverse engineering to document the variability and enable planned reuse of legacy SW, including PLC SW, e.g., Hintereiter et al. [46] and Schlie et al. with a metric-based approach [47], or ECCO aiming at the support of an enhanced application of C/P&M for development and maintenance [36]. However, SPLE is so far lacking fundamental methods and appropriate tool support to cope with the interdisciplinary nature of aPS [4]. Thus, in the field of aPS, SPLE approaches are not applicable without further adaptions to represent the mechatronic characteristics of the systems. Even though approaches for interdisciplinary product lines are already available such as [48], [49], [50], [51], they have not yet been widely established in the field of aPS. Apart from variant management, also version management and evolution are relevant factors regarding reuse. In the aPS domain, Biffl et al. [52] propose an approach for interdisciplinary version management in the aPS domain using the XML-based standard AML [34]. Wimmer et al. enlarge this approach with cardinality-based variability modeling [53]. However, the differentiation of both variants and versions is not trivial for industrial practitioners [18]. Therefore, as the authors in [8] propose, variant and version management are differentiated and analyzed in more detail in H5.2.

To distinguish different degrees of SW reuse, Antkiewicz et al. [6] introduce a virtual platform comprising seven levels transitioning from the undesired C/P&M to desired SPLE based on feature models. On the lower levels, [6] define different ways of unplanned reuse by copying code, i.e., spontaneously without any reuse strategies or processes (L0), copying but retaining information about the original code (L1) or copying code related to specific features to propagate them across projects (L2). From level L3 and higher, the approaches become more sophisticated, e.g. by adding configuration or feature models (L4), up to SPLE approaches at the highest levels (L5, L6). Further, Antkiewicz classifies reuse strategies, which have been considered in a preceding questionnaire study: The results of [7] showed that the method C/P&M is still widespread. Also, for all four case studies that were analyzed in detail in [7], the governance level according to [6] was below L4. Considering the broad range of levels and these first insights, [8] further investigated to which degree code configuration from engineering tools is realized in industry showing that from the questioned companies, 63% of MM and 45% of PM do not apply automatic configuration at all. For better applicability of Antkiewicz's approach in the aPS domain and to gain deeper insights into the maturity of reuse strategies in aPS, an adaptation of his approach to aPS is examined in H5.1.3.

Lucas and Tilbury [54] showed that 50-75% of industrial control code consists of extra-functional code, i.e. transversal tasks that are used across POUs, such as fault handling, operating data collection or switching of operation modes. These extra-functional tasks usually are not represented in the physical layout of the machine [23] and sometimes concern the interaction between aPS and human operator via HMI, e.g. in case of a fault [55]. Therefore, these parts are often more difficult to understand than standardized control code parts such as software for actuator control. Further drivers of complexity could be identified in previous work - partially including metrics to automatically assess their impact on maintainability and testability of SW and, thereby, on accumulated Technical Debt [56]. This includes adaptions of the well-established Halstead Difficulty [57] and Cyclomatic Complexity [58] for IEC 61131-3 [59], [60] as well as in-depth analyses of architectural complexity in aPS SW projects [4]. However, complexity is an inherently subjective property and, therefore, difficult to assess systematically.

2.5 Questionnaire- and interview-based surveys in aPS

The results of different preceding questionnaire studies as well as a guided interview study with six German companies focusing on reuse mechanisms, including variant and version management, are presented, revealing open questions to be addressed in this paper.

In the following, the results of the four prior studies for the different criteria are briefly introduced and the remaining gap for $\text{QoaPS}_{\text{SWE}}$ is identified (Table 14). Criteria are structured into descriptive ones, maturity indicators, and quality of engineering workflow. The descriptive criteria introduce the method used, such as the number of questions, additional interviews to validate the questionnaire results or pure questionnaires, and pure interviews. As questionnaires are the easiest way for an assessment, questionnaires without additional interviews are the preferred method to gain insights into SW maturity and its supporting workflow, which has been achieved using $\text{QoaPS}_{\text{SWE}}$ as the method introduced in this paper.

To achieve meaningful and statistically significant results regarding the maturity and engineering workflow, it is required to consider specific characteristics of MM and PM. These include classification by the industrial sector and the size as indicated by the number of employees in the software engineering department. Initially, maturity values for individual MM and PM were targeted as well as for entire industrial sectors, as different sectors show different requirements and standards [9]. The number of companies included has been too small to differentiate between MM and PM as well as between industrial sectors in all prior studies: Even though 68 companies participated in SWMAT4aPS+ [8], the companies could not be assigned unambiguously to individual industrial sectors or even distinguished regarding the classification to MM or PM. Same applies for SWMAT4aPSi/m [9]: From 70 participants, only eleven could be identified as MM or PM manufacturing companies. Therefore, the critical number of companies to be included is addressed as RQ1 in this paper. Company size and programmers in the SW engineering department were included as constraints that might influence a choice of one or the other strategy (cp. Section 2.1).

Maturity indicators and questionnaires evolved continuously from the first calculation schema for the maturity of SW modularity ($M_{MOD,Q1}$), testing/quality assurance ($M_{TEST,Q1}$), startup/operation/maintenance ($M_{OP,Q1}$), and overall maturity ($M_{\sum,Q1}$) in the study SWMAT4aPS using questionnaire Q1 [7] to the analyzed ten different criteria in $QoaPS_{SWE}$ using questionnaire Q4. The resulting questionnaire Q4 includes a concept to differentiate different reuse levels in the style of Antkiewicz et al. [6], complexity, IEC languages including OO IEC and interfaces, variant and version management, reuse strategies, and standard functions included in a SW module. Of course, comprehensive indicators would be most appreciated, like the ones proposed initially [7], but they did not show significant correlations up to now. Consequently, RQ2 addresses whether quantitative correlations of the maturity values can be found if complete questionnaire data for more than 60 companies from MM and PM are available. The second study, SWMAT4aPS+ with questionnaire Q2 [8], already focused on different disciplines, but the applied questionnaire does not include a sufficient number of questions addressing EE issues. Consequently, in the third study, SWMAT4aPSi/m with questionnaire Q3, such questions were added and are addressed by RQ3, thereby analyzing reuse strategies in different disciplines in a qualitative manner.

Antkiewicz levels have been analyzed in [7] for four companies using interviews and applying static SW analysis and compared with the maturity indicators calculated from the Q1 questionnaire, confirming a basic overlap. However, the compliance with a purely questionnaire-based study is still lacking. Consequently, this will be addressed in this paper (H5.1.3) to analyze the applicability of the approach in aPS.

As the complexity of control code might counteract SW maturity, it was analyzed right from the beginning, with [7] proposing the number of POUs and Lines of Code (LOC), number of controllers (CPUs), and number of programmers for one aPS as measures, but without success. Starting with Q2, we inserted a question addressing the most critical task and a subjective rating of complexity. In all prior studies, we could neither identify a reliable indicator for complexity nor key drivers of complexity and consequently no means to manage complexity. This issue is still open and addressed in RQ4.

To ease SW engineering in embedded systems, model driven engineering and object orientation with interfaces are well-accepted. Some of these aspects were included in our previous questionnaires, too, but revealed only qualitative results, selected moderate results, or results with low evidence [8]. Especially advantages of using OO IEC, difficulties encountered by introducing OO IEC, and obstacles when not using OO IEC could not be identified and are consequently addressed in this paper.

Prior work revealed challenges of variant and version management (cp. [7], [8], [9], [18]) and reuse strategies (cp. [9], [18]), which are well established in classical SW engineering but hardly applicable in aPS; these aspects are also included. Up to now, variant and version management were analyzed jointly, delivering only qualitative results. Diagnosis, fault handling, and mode of operation with dependencies to M and EE aggregate the usage of classical SW engineering approaches for reuse [16] and were focused in [8] and [9], too, but delivered only results with low evidence. Therefore, these aspects are addressed in RQ5 (H5.2) in detail.

Five criteria are used to compare the results gained for the quality of the engineering workflow: the procedure used for the release of library blocks and building of new variants, the efficiency of information exchange, the generation of information from different available sources for the involved disciplines, the timely availability of required documents, and the availability of guidelines in general.

The differences in release procedures of libraries for MM and PM have been identified in [7], [8], and [9]. The criteria to build a new variant were analyzed in [18], showing weaknesses in differentiation between variants and versions. Therefore, the differentiation of these as-

pects was focused in this paper and addressed with the criterion variant and version management introduced above (H5.2). The efficiency of the exchange of information was analyzed in [7] and [8]. Thereby, the automatic generation of information is considered ideal and, consequently, addressed in H3.5, including documents of different disciplines.

Efficient information exchange also requires the availability of documents in time, which has not been included in previous surveys and is, therefore, added as H3.3. To ease engineering, guidelines are often provided by the community, like IEEE Standards, VDI / VDE guidelines, or PLCopen guidelines. In [8] and [18], questions regarding guidelines were included in the studies but lacked correlation with other factors. Therefore, we investigate the level of detail of guidelines (H3.4), how it differs between disciplines, and which types of guidelines are most beneficial.

3 Research method for QoaPS$_{\mathrm{SWE}}$

The research goal addressed with QoaPS$_{\mathrm{SWE}}$ is to gain deeper insights in the state of the art in SW engineering of field level control SW and to identify weaknesses and obstacles that should be addressed in the future by method and tool suppliers as well as organizational or educational measures. Five research questions were identified that shall be addressed (cf. Table 1 for a summary).

The benchmark process used is based on SWMAT4aPS+ [8]. Some improvements were made: first, reducing the initial questions and adding new ones to analyze selected aspects in more detail and, second, rephrasing questions for a better understanding. The approach realized in this survey does not include an expert analysis of code as in [7] or additional interviews as in [8], though.

3.1 Questionnaire Q4

The applied questionnaire in QoaPS$_{\mathrm{SWE}}$, i.e. questionnaire Q4, comprises 69 questions (cp. Appendix), including all sub- and follow-up questions. Q4 is thereby structured into five sections: questions on the approach in development (15 questions), SW and reusability (40 questions), quality assurance (4 questions), handover to the customer and commissioning of the plant (5 questions), and company specifics (5 questions). Analogous to Q2 in SWMAT4aPS+, different question types were used, i.e., selection from options (single or multiple choice), free-text, and indicating percentages or scales ranging from 1-5.

Lessons learned from previous questionnaires and the analysis of the state of the art highlight the importance of design maturity, especially regarding reusability. Therefore, compared to SWMAT4aPS+, the main focus of QoaPS$_{\mathrm{SWE}}$ is shifted towards this research direction by doubling the number of questions in this section. This enables a more in-depth understanding of the concept of modularity and practices in MM and PM.

Follow-up Questions: The time needed to fill in the questionnaire should be reduced to a minimum by keeping the number of questions to be answered low. On the other hand, specific questions are relevant only if (a specific) one of the preceding selectable choices is chosen. The same holds true for free-text follow-up questions as introduced before. Follow-up questions are marked with $^{\rightarrow \mathrm{a}}$ in the Appendix, with "a" indicating the question on which they depend. For example, the multiple-choice question #5 asking for the modeling tools used for control SW is answered by all 146 respondents. The follow-up questions are selected as follows: if the selected choice is *IT-oriented* (48 companies), the follow-up question #6 is asking for the extent of its usage, otherwise the question about the extent is omitted. If participants then select to use *IT-oriented* modeling tools *only partially*, they are directed to a free-text question asking for reasons (#8 for *IT-oriented*). Further, they are asked to identify the disadvantages of not or only partially using the modeling tool (#9 for *IT-oriented*).

Free text answers and questions with additional free text: To investigate aspects of the results in [8], [9], [18] requiring a more detailed analysis such as the usage of modeling tools (#7$^{\mathrm{F}}$-9$^{\mathrm{F}}$; #11$^{\mathrm{F}}$-13$^{\mathrm{F}}$) and management of complexity (#24$^{\mathrm{F}}$, #25$^{\mathrm{F}}$), 14 free-text questions (indicated by F in the Appendix) out of 69 questions are used. For example, regarding complexity (#24$^{\mathrm{F}}$, #25$^{\mathrm{F}}$), the aim was to identify the main reasons for complexity (H4.2). Similarly, regarding modeling tools (#7$^{\mathrm{F}}$-9$^{\mathrm{F}}$; #11$^{\mathrm{F}}$-13$^{\mathrm{F}}$) QoaPS$_{\mathrm{SWE}}$ targets to identify advantages (#7$^{\mathrm{F}}$, #11$^{\mathrm{F}}$) of using them, reasons for not / partially using them (#8$^{\mathrm{F}}$, #12$^{\mathrm{F}}$), and disadvantages of only partially or not using them (#9$^{\mathrm{F}}$, #13$^{\mathrm{F}}$). Also, for OO IEC usage (#18$^{\mathrm{F}}$-21$^{\mathrm{F}}$) additionally to the predefined selectable choices, free-text answers were possible if the answer other was chosen.

3.2 Experimentation and companies included in the questionnaire

To capture a broad database and to ensure anonymity, the questionnaire is provided online via the market research group of the publisher *Verlag Heinrich Vogel*. A

German-speaking community is addressed via newsletters and web pages, which is interested in embedded systems and SW engineering as well as mechanical design in MM and PM. The participants are classified into three categories, namely MM, PM, and others.

Companies included: Based on the typical reading group of the publisher, two assumptions can be made. Regarding their role in the company, two main groups of readers are identified: 63% of readers belong to Research and Development (R&D), while 10% are technical management. Regarding the position, 15% belong to the CTO or management level, 14% are department heads, 17% are group or project managers, and 50% are on the engineering level.

While a total of 322 companies started answering the questions of Q4 in the online formula, 146 completed it. As the intended target group for this paper consists of MM and PM from the field of production automation, the 146 valid cases are firstly classified by their stated industrial sectors (#65). Thereby only companies, which assigned themselves towards at least one industrial sector of either *materials handling and intralogistics, plastics and rubber machinery, printing and paper technology, woodworking machinery*, or *process engineering machines and apparatuses*, were selected for further studies, as these are considered typical industrial sectors associated with production automation. This resulted in a first reduction of cases from 146 to 71 participants. As a second and final selection level, only companies were finally chosen, which assigned themselves either towards MM or PM in #69. This leaves a total number of 61 companies, which represent the base group for the study.

Scoring: The scoring was developed iteratively based on prior questionnaires using a five value grading schemata (from 1 to 5) jointly with feedback of industrial experts. In the following, the scoring procedure is outlined using the question of standard functions included in each module or component (#31). Thereby, modules not comprising any standard functions are rated with the lowest score of 0. In case at least standardized *HMI interfaces* are included, a score of 1 is assigned. In case either *fault detection / diagnostics* or *operation modes* are provided by the module, the participant receives a score of 3 for this question. The reason for this is that the standardization of *fault detection / diagnostics* and *operation modes* represents a demanding architectural challenge within the PLC SW and is more complex compared to providing a standardized HMI interface. Naturally, in case both *fault detection / diagnostics* and *operation modes* are implemented standardly, the highest score of 5 is achieved (cp. Table 2).

Table 2: Scores for the question "What functions does each module/component have as standard?" (#31)

Evaluation criterion	**Score**
fault detection/diagnostics **and** operations modes (with or without HMI interfaces)	5
fault detection/diagnostics **or** operations modes (with or without HMI interfaces)	3
only HMI interfaces	1
no answer	0

Based on these individual scores, the overall maturity level of each company can be calculated (cp. [7]). This is achieved by simply dividing the sum of all scores of the individual company by the sum of all achievable scores.

Scoring of free-text answers and questions with additional free-text: As introduced in section 3, 14 free-text questions are used to explore selected aspects in more detail. To analyze free-text questions, the answers given are clustered content-wise by two experts (dual coding). Thereby, similar responses are grouped into one class, with classes being chosen from the state of the art. If experts did not choose the same classification and did not agree after negotiation, the answer is not included. Surprisingly, most answers were easy to classify as usually the same words are used.

If only a few companies' answers were available or answers could not be clustered to superordinate categories, these free-text answers were not included in the analysis. For example, only eight participants were directed to the follow-up questions on control patterns (#40^{F}, #41^{F}) and only one of them provided an answer, which was consequently not used for further analysis as there were no other comparable values for this question.

Consistency checks: as an additional check of validity, selected consistency checks were conducted, e.g., *usage of version management tools* (#42) against *version tracking by the use of version management tools* (#45) and *guidelines / checklist for compliance with standard structures for comments in the module* (#37) against *manual comments recorded according to guidelines for comments* (#46) were checked for inconsistencies. Furthermore, when applying the scoring, consistency was actively assured, e.g., the usage of *IEC ST* is seen as a prerequisite for the usage of OO IEC (#17) (cf. section 2.1). In case this prerequisite was not met, the company was not awarded the score for using OO IEC (cp. Section 8.3).

3.3 Statistical Methods

In previous work, a deeper statistical analysis was hindered by an insufficient number of cases (<20). Within the $QoaPS_{SWE}$ questionnaire, a final selection of 61 cases is available, assigned to MM and PM. Thus, the following statistical tests and methods are used to provide a statistical evaluation of the observed phenomena in accordance with [61]:

Linear correlation: In order to identify the linear dependency of two items, Spearman's rank-order correlation coefficient rs is applied. Thereby, the data is firstly ranked, and then the equation for the Pearson's correlation is used on the ranked data (cp. Section 6.5.3 from [61]). Since Q4 only contains ordinal scaled items (e.g., #22), Pearson's correlation coefficient r is not feasible here. The correlation coefficient r_s indicates the correlations's effect size, which is categorized in small, medium and large for consistent interpretation (cp. Table 3). In addition to the correlation coefficient r_s, significance value p for this correlation coefficient is reported.

Differences in central tendency: To determine whether two independent groups show differences regarding a specific item (dependent variable e.g., maturity level), the non-parametric Wilcoxon-Mann-Whitney test, which can also be used for ordinal scaled items, is applied. Thereby, the groups are formed using nominal items such as the programming languages a company uses (#17). Next, the data from the dependent variable is ordered and ranked. Then, the rank-sums for both groups are calculated, and thereby the test statistic W_s, which corresponds to the rank-sum of the smaller group or the lower rank-sum in case of same size groups. After that, W_s is utilized to calculate the z-score which in turn is compared to the standard normal distribution to derive the significance value p (cp. Chapter 15.3 form [61]). The requirement of variance homogeneity of the groups is checked with the Levené's test. If not otherwise stated below, the Levené's test is not significant (level of significance = 0.05), and therefore, the requirement of variance homogenity is fulfilled. To examine the effect size, Pearson's correlation coefficient r is calculated based on the test statistics. For interpretation, the coefficient is categorized according to Table 3. The test statistic W_s, the z-score and the significance value p are reported in addition to the effect size r for the Wilcoxon-Mann-Whitney test. If more than two independent groups should be compared, the extension of the Wilcoxon-Mann-Whitney test, namely the Kruskal-Wallis test, is used to evaluate if a difference in central tendency between any two groups exists. Thereby, the rank-sums of the multiple groups are used to compute the test statistic H, which is chi-square distributed. The value for the degrees of freedom (df) for the chi-square distribution is one less than the number of groups. H is then compared to the chi-square distribution for the corresponding df to derive the significance value p (cp. Section 15.5 from [61]). The test results are reported as H(df),p.

Given a significant test result, multiple bilateral Wilcoxon-Mann-Whitney tests with an applied Bonferroni correction are used to confirm and quantify the differences.

Differences in frequency distribution: To evaluate whether there is a statistically significant difference between the frequency distribution of nominal-scaled items (contingency tables), the Pearson's chi-square test is applied. When using this test, a Pearson's chi-square χ^2 is calculated utilizing the standardized deviation of the expected frequency from the actual deviation for each cell of the contingency table. Expected frequencies are thereby calculated using row and column totals. χ^2 is then compared to the chi-square distribution for the corresponding df to derive the significance value p. For the Pearson's chi-square test df can be computed as df = (r - 1)(c - 1) in which r is the number of rows and c the number of columns (cp. Section 18.3.1 from [61]). In case the requirements for the Pearson's chi-square test regarding the number of cases in each cell is not met, the Fisher's exact test is used. When dealing with larger than 2x2 contingency tables, these tables are broken down into 2x2 tables to apply Pearson's chi-square tests or Fisher's exact tests with a Bonferroni correction nonetheless. To report the effect size, Cramér's V is calculated from the test statistics and classified according to Table 3. Additionally, the odds ratio is used, as it is an easily understood, quantitative metric.

Contrary to earlier studies (Table 14), in $QoaPS_{SWE}$, the effect size is considered as an additional quantitative measure to objectively specify the significance of an identified effect.

All results are obtained with MATLAB using the Statistics and Machine Learning Toolbox [62].

Table 3: Measures of effect size (for the different functions) according to [63], [64] and [65]

Effect size	**Pearson's r**	**Spearman's r_s**	**Cramér's V**
Small	|0.1|	|0.1|	|0.1|
Medium	|0.3|	|0.3|	|0.3|
Large	|0.5|	|0.5|	|0.5|

3.4 Threads to Validity

To avoid a systematic bias, the authors followed the guidelines of Runeson et al. [66] to construct validity and reliability of the case study as described in more detail in prior work [7], [8]. In the following, some aspects of this survey are summarized. The validity of the companies' self-assessment is ensured in different ways. To avoid a systematic bias, specifically in the questionnaire's design, the questions and the overall questionnaire structure are discussed with experts from both academia and industry in workshops. In addition, the state of the art in academia is considered, as presented in section 2, and the authors were able to build on lessons they learned from prior questionnaires, e.g., regarding the precise formulation of questions and providing appropriate answering choices. Concerning the participants, a potential bias was eliminated by publicly providing the questionnaire through a web-interface via an independent publisher. To avoid errors in the evaluation, the companies' answers are checked for inconsistencies, as introduced in section 3.3. An analysis of the comparison group's maturity relative to the average maturity further confirms the assessment. Additionally, the results are compared with the insights gained from previous surveys and interviews, as well as with the maturity levels proposed by Antkiewicz et al. [6]. Finally, the validity of the interpretation and the measure used is provided (cp. column "evaluation" and "validity" in Table 4, 5, 6, 10, and 12; validity is indicated as ++ (strong evidence), + (moderate evidence), o (low evidence), and - (no evidence/not verifiable).

4 Classification of companies into different industial sectors is feasible and provides additional insights (RQ1)

Classification includes the industrial sector synonym with application domain and the differentiation in between MM and PM. For the specification of the industrial sectors, a formulation recommended by the publisher was used for question #65, which is based on the trade associations and departments of the VDMA. Since many companies deliver to multiple industrial sectors, prior studies had difficulties in establishing a clear distinction to an industrial sector as well as in between MM and PM (cp. Table 14) because they often deliver to both or cannot assign themselves easily and it may depend on the person answering the questionnaire. To counteract this challenge, the respective question (#69) is rephrased to get the participants to make a (subjective) decision for exactly one of the options. Additionally, we aimed to include a higher number of participating companies than previously and excluded questionnaires that did not claim at least one industrial sector typically associated with production automation (#65, Figure 1). Most (59%) of the 61 companies investigated choose two to four industrial sectors, while only 18% choose just one. This confirms the results of previous studies.

Summarizing our findings (Table 4), H1.1 is true for MM and PM but false for assignment to one industrial sector. Consequently, the long-term goal to achieve Key Performance Indicators (KPIs) specific for different industrial sectors is not feasible with only 18% of 61 companies and would be ambiguous with those companies assigned to more than one sector, because it would either require that participants split their answers to the chosen two or more sectors, which is not feasible given the time for answers. To use the answers for all selected sectors would bias the results as companies choosing more sectors would count higher than those selecting only one or two. Nevertheless, we can conclude that most companies supply to more than one industrial sector and, therefore, cannot narrow down their architectures, methods, and workflow to one specific sector.

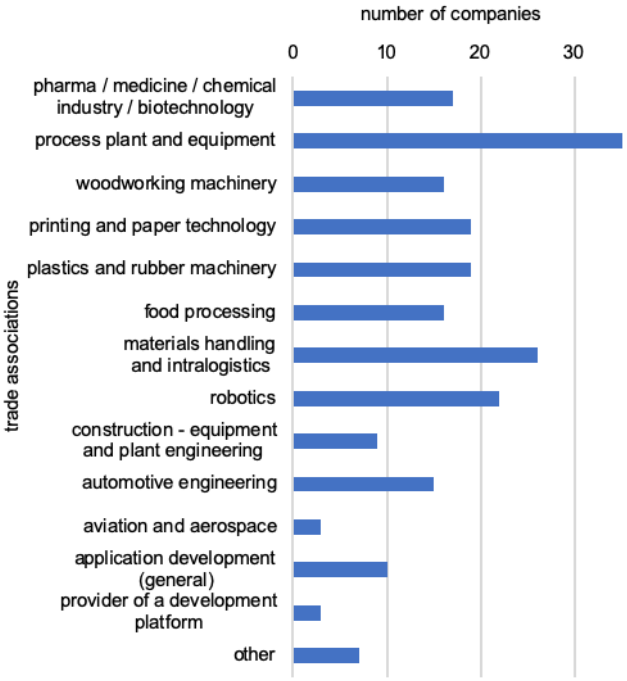

Fig. 1: Industrial sector of participating companies in Q4 #65

With initially 146 participants in Q4 and 75 in Q3, we expected, according to H1.2, that qualitative analysis is feasible due to a sufficient number of companies per category. In Q3, unfortunately, only eleven

Table 4: Classification of companies into different application domains / industrial sectors and differentiation in between MM and PM is feasible and provides additional insights (RQ1; Q3 and Q4)

Related hypotheses	Findings	Evaluation	Validity	Relev. sec.
H1.1 Companies assign themselves to a single industrial sector and either MM or PM	Numbers of industrial sectors selected (#65): 1: 18%; 2-4: 59%; > 4: 23%	False	-	Sec 4 Fig. 1
	n = 146 limited to n = 61 based on distinct assignment to either MM or PM (#69)	True	++	
H1.2 Quantitative analyses are possible due to sufficient number of companies per category	Q3: 75 participants: 11 in production automation	False	-	Sec 4
	Q4: 146 participants; 61 in production automation	True	++	

++: strong evidence, +: moderate evidence, o: low evidence, -: no evidence/not verifiable

participants could be clearly assigned to either MM or PM and answered the questions in both parts of the questionnaire, i.e. the part on EE and the part on SW engineering. Therefore, H1.2 is false for Q3 (Table 4). For many questions in Q4, reasonable and statistically significant results can be derived and will be introduced in the following. Therefore, H1.2 is true for Q4 (Table 4). Regarding the validity of our results, Q3 can only be used qualitatively and might need to be confirmed by Q4, interviews, or future questionnaires. The attempt to ask in-depth questions for both EE and SW aspects results in poorly answered parts of the questionnaire and confirms its limitation due to the limited discipline-specific expertise of the participants.

Apart from the classification to industrial sectors, an overview of used programming languages is provided. Besides the five IEC 61131-3 programming languages, high-level languages and modeling and simulation languages like MATLAB / Simulink are in use. It would be assumed that high-level programming languages will ease modularity as a prerequisite for reuse. The specification of the formulation is based on lessons learned from previous questionnaires in which only the applied programming languages were asked leading to an over-representation of high-level languages. The high percentage of high-level programming languages (cp. Figure 2) cannot be explained by the programming of HMI or Manufacturing Execution Systems (MES) because the question (#17) was clearly specified, asking only for control code excluding HMI. However, we need to assume that either validity is poor, because consistency checks with control platforms failed (74% of companies using high-level languages on control level do not match with the control platforms, as some of them do not support high-level languages) or PLC code is automatically generated from high level programming languages. The later was not confirmed by analyzing the questions regarding code generation.

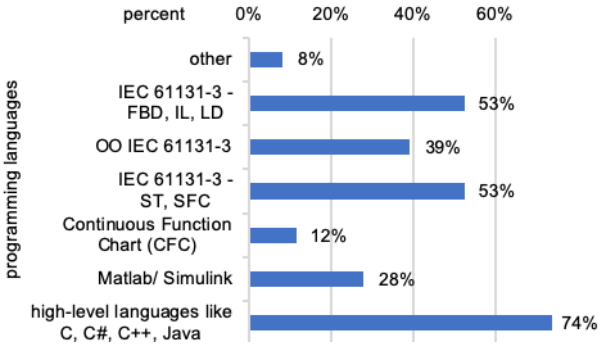

Fig. 2: Percentage of programming languages in use #17

5 Significant correlations of and influences on maturity values depending on sample size (RQ2)

In [7] using Q1 maturity values for modular design, test and operation and maintenance for automation software are introduced based on Q1 ($M_{MOD,Q1}$, $M_{TEST,Q1}$ and $M_{OP,Q1}$, respectively), but only a few correlations could be identified due to only 16 involved companies. In Q3, the SW maturity measures are transferred and adapted to EE engineering. As introduced in section 4, only eleven companies answered all respective questions, and therefore, only qualitative evaluations can be achieved (cp. Table 6, Section 6). RQ2 refers to the question which correlations with SW design modularity can be confirmed significantly with more than 60 participating companies as a minimum threshold. We propose four hypotheses (cp. Table 5) and three sub-hypotheses. Compared to [4], the calculation of M_{MOD} has been revised and enriched according to the evolution of the questionnaires to M_{MOD+} (cp. Appendix, List of questions ⧫1).

Properly designed SW shows higher M_{MOD+} values and is also easier to test (i.e., higher values for

Table 5: Correlations of maturity values significant with $n > 60$ (RQ2; Q4)

Related hypotheses	**Findings**	**Evaluation**	**Validity**	**Relev. sec.**
H2.1 A correlation between M_{MOD+} and M_{TEST} can be demonstrated	High resulting M_{MOD+} leads to high M_{TEST} • $r_s = 0.58$, $p < 0.001$, $n = 61$	True, large effect	++	Sec 5
H2.2 M_{MOD+} differs significantly between MM and PM	M_{MOD+} significantly higher in PM than in MM (#69) • Two-tailed Wilcoxon-Mann-Whitney test (WMW): $W_s = 917$, $z = 2.076$, $p = 0.0379$, $r = 0.27$, $n = 61$	True, small effect	+	Sec 5
H2.2.1 Reasons for different M_{MOD+} values in MM and PM can be identified	One-tailed WMW for each item of M_{MOD+} reveals significant higher scores for PM regarding • MBSE Tools (#5): $W_s = 918.5$, $z = 2.259$, $p = 0.012$ • Mechatronic modules (#14): $W_s = 922.5$, $z = 2.297$, $p = 0.011$ • Scope of modularization (#15): $W_s = 907$, $z = 2.042$, $p = 0.021$ • Version management tools (#42): $W_s = 870$, $z = 1.735$, $p = 0.041$ • Variant management tools (#47): $W_s = 897$, $z = 1.963$, $p = 0.025$	True, small effects	+	Sec 5
H2.3 There is a correlation between M_{MOD+} and the self-assessment of...				Sec 5
H2.3.1 Self-assessed control software complexity	Self-assessed control software complexity (#22) does not correlate with M_{MOD+} • $p = 0.381$, $n = 61 \rightarrow$ not significant	False	-	
H2.3.2 Companies' ability to manage control software complexity	Ability to manage complexity (#23) correlates with M_{MOD+} • $r_s = -0.38$, $p = 0.003$, $n = 61$	True, medium effect	++	
H2.4 M_{MOD+} is correlated with...				Sec 5 Fig. 3
Company size	M_{MOD+} significantly affected by company size (#65) • $H(2) = 9.374$, $p = 0.009$, $n = 61$ • WMW post hoc test (Bonferroni correction, significance level 0.025) • M_{MOD+} significantly higher in companies with #emp > 1500 compared to: • #emp < 250 $W_s = 389$, $z = 2.880$, $p = 0.004$, $r = 0.46$ (a) • #emp 250-1500 $W_s = 308.5$, $z = 2.326$, $p = 0.020$, r= 0.39 (b)	True a: medium - large effect b: medium effect	++	
Size of application software development department (number of employees #emp)	M_{MOD+} significantly affected by SW development department's size (#66) • $H(2) = 11.606$, $p = 0.003$, $n = 61$ • WMW post hoc tests (Bonferroni correction, significance level 0.025) • M_{MOD+} significantly higher in companies with #emp >75 compared to #emp < 25 $W_s = 406$, $z = 3.129$, $p = 0.002$, $r = 0.45$	True, medium effect	+	

M_{TEST}). Therefore, according to H2.1, a correlation between aforementioned maturity values at a significant level is expected. To evaluate this hypothesis, a Spearman's rank-order correlation coefficient r_s is calculated. Since there is a strong positive correlation between M_{MOD+} and M_{TEST}, H2.1 is considered valid (Table 5).

The second hypothesis (H2.2) examines whether there is a difference in design maturity (M_{MOD+}) in MM versus PM. As PM is even more customer-specific than MM, we assume that M_{MOD+} in MM is higher than PM. On the other hand, PM companies are familiar with such challenges and may consequently be better prepared using methods that might be rated with higher grades. Therefore, a two-tailed Wilcoxon-Mann-Whitney test is executed, as for the reasons mentioned above, no directional hypothesis can be made. H2.2 is true: companies assigning themselves towards PM have significantly higher M_{MOD+} (Median (Mdn) = 3.23) scores than companies assigning themselves towards MM (Mdn = 2.58). A small to medium effect can be demonstrated (Table 5).

A more detailed analysis of the factors of such differences is examined in H2.2.1. All scored items used in M_{MOD+} were analyzed using a one-tailed Wilcoxon-Mann-Whitney test, as H2.2 indicates a directional relationship. Statistically significant higher scores for PM than for MM can be observed for multiple items, although the effect sizes only correspondent to small effects (cp. sec 3.3). These items and effect sizes are listed in Table 5. H2.2.1 is, therefore, true and different factors were identified.

Besides the factors included in the modularity maturity calculation, also correlations to other factors are analyzed: As introduced in section 2.6 and Table 14, complexity could be a crucial disturbing factor for M_{MOD+}. To validate H2.3, this assumption is investigated in more detail. Thereby, we expect a negative correlation: the higher the self-assessed control SW complexity (#22), the lower the modularity maturity. Additionally, we assume higher self-assessments of a company's ability to manage control SW complexity (#22) to correlate with higher M_{MOD+} values. Likewise, to H2.1, two Spearman's rank-order correlation coefficients r_s are calculated. No significant correlation could be identified between M_{MOD+} and the self-assessed control SW complexity. Consequently, H2.3.1 is false. Between M_{MOD+} and the self-assessments of a company's ability to manage control SW complexity, however, a significant correlation can be verified (Table 5). Since item #22 was evaluated using a scale from 1-5 with 1 representing the highest level and 5 the lowest level of a company's ability to manage control SW complexity, r_s is, as we expected, negative (r_s = -0.38), indicating a medium effect size (cp. Table 5). Therefore, high levels of a company's ability to manage control SW complexity show a medium to high correlation with M_{MOD+} values and H2.3.2 is true.

Finally, for H2.4, we examine whether M_{MOD+} values are influenced by the size of companies and their SW development departments. Consequently, a Kruskal-Wallis test was performed for each independent variable (company size #66, size of SW development department #67). M_{MOD+} values are significantly affected by company size (cp. Table 5, Figure 3). As we expect a significant difference between large companies (>1500 employees) and small companies (<250 employees) and, furthermore, between large companies and medium companies (250-1500 employees), two Wilcoxon-Mann-Whitney tests were conducted in addition to the Kruskal-Wallis test. A Bonferroni correction was applied. Companies with more than 1500 employees show significantly higher M_{MOD+} values (Mdn = 3.50), than companies with less than 250 employees (Mdn = 2.46). The effect size is medium to large (r = 0.46, cp. Table 5). Furthermore, companies with more than 1500 employees also show significantly higher M_{MOD+} values (Mdn = 3.50) than companies with 250 to 1500 employees (Mdn = 2.62), in which the effect size resembles a medium effect (r = 0.39).

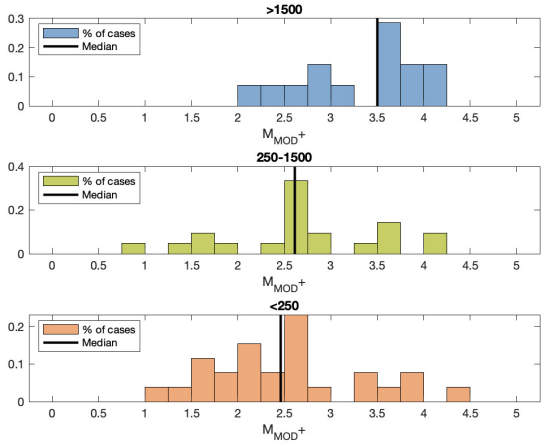

Fig. 3: M_{MOD+} values in relation to the number of employees in the company

Similar to the size of a company, the size of the SW development department significantly influences M_{MOD+} values. Again, we expect the difference in M_{MOD+} values between SW development departments with more than 75 automation / SW technicians / engineers (in the following employees) and departments with either 25 to 75 or less than 25 employees. Two Wilcoxon-Mann-Whitney tests were carried out to evaluate these differences and follow up on the Kruskal-Wallis Test. A Bonferroni correction was also applied.

While companies with more than 75 employees in their application development department show significantly higher M_{MOD+} values (Mdn = 3.5) than companies with less than 25 employees (Mdn = 2.5, r = 0.45, cp. Table 5), no significant difference in scores between companies with more than 75 employees and companies with 25-75 employees could be identified. This quantitatively confirms our qualitative findings in H3.1.1 (cp. Table 6).

6 Reuse strategies in different disciplines (qualitative results from Q3) (RQ3)

As prior questionnaires (cp. Table 14) lack results regarding EE engineering, Q3 in SWMAT4aPSi/m aimed to equally balance questions in both areas, SW and EE engineering of aPS, to elaborate specific challenges

on the one hand and to reveal on the other hand dependencies or source effect relations and similarities in design strategies like universal modules (H3.2) between SW and EE engineering. Additionally, peculiarities between the three different mechatronic disciplines should be researched (H3.1.1) as well as the sensed inequity that SW engineers always struggle harder (H3.4, H3.5) than any other discipline because they get the poorest quality of information at the latest date (H3.3) [67]. The results are summarized in Table 6.

6.1 Maturity values of different engineering disciplines (H3.1)

In SWMAT4aPSi/m, additional questions were developed for EE engineering with experts in this area to mirror the questions in SW and to balance the number of questions and level of detail in both disciplines. The key challenge was to prove the feasibility and expressiveness of maturity values also for M and EE (H3.1). Unfortunately, only eleven companies from MM and PM answered all relevant questions and thereby delivered meaningful, sound results for EE and SW (n = 11 in Q3, #76). The results for these participants will be analyzed to evaluate H3.1.2, which states that the highest average $M_{MOD,Q3}$ values are reached by different combinations of discipline-specific $M_{MOD,Q3}$ values, i.e., either the EE maturity ($M_{MOD\_EE,Q3}$) exceeds the SW Maturity ($M_{MOD\_SW,Q3}$) or vice versa. To determine these values, questions address on the one hand aspects that apply to both disciplines, such as the exchange with other disciplines, availability of functional descriptions, existing approaches for reusability, and the establishment of internal guidelines. On the other hand, the issue of variant management is covered with discipline-specific questions (concerning PLC control SW and design of circuit diagrams). The according questions can be found in the list of questions (#70-#77). To ensure comparability of the maturity in both disciplines, analogue questions were asked for SW and EE, i.e. in total seven questions each. Additionally, as our competencies and previous experience are mainly located to the SW perspective, we involved one of our industry partners with a strong background in EE in the formulation of the questions. As aPS are mechatronic systems, the question arises whether one strategy, like high $M_{MOD,Q3}$ values for both EE and SW or focusing on the optimization of one of them, might be superior. Therefore, we analyze $M_{MOD,Q3}$ values reached in EE and SW engineering. The highest values of average $M_{MOD,Q3}$ are reached by companies 194, 232, and 249 (cp. Figure 4); all of these companies reached higher modularity maturity in SW ($M_{MOD\_SW,Q3}$) than in EE ($M_{MOD\_EE,Q3}$). Only companies 86 and 131 show higher $M_{MOD\_EE,Q3}$ than $M_{MOD\_SW,Q3}$. We can conclude that for the best overall rating, high values in both categories can be observed with $M_{MOD\_SW,Q3}$ outperforming $M_{MOD\_EE,Q3}$. Therefore, H3.1.2 is true but due to the small number of companies only qualitatively.

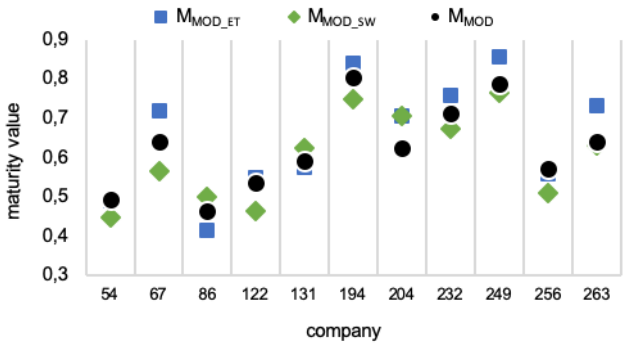

Fig. 4: Maturity values for modularity in electrical ($M_{MOD\_EE,Q3}$), software engineering ($M_{MOD\_SW,Q3}$) and average maturity ($M_{MOD,Q3}$) for aPS H3.1 questions #70-#77

As we derived from several case studies and interviews [18], smaller and mid-size companies may not be able to invest in modularity. Therefore, we assume that the modularity of the different disciplines depends on the company size (H3.1.1). Modularity ($M_{MOD\_SW,Q3}$, $M_{MOD\_EE,Q3}$ M_{MOD_M}) shows the highest values for companies with more than 1,000 employees, decreasing with company size. This is true for SW and EE engineering but false for M engineering as companies with 251 to 1,000 employees show slightly higher values than the smaller and larger ones. This observation is confirmed by the quantitative results of Q4 (cp. Section 5, H2.4). Summarizing the evaluation results for H3.1.1 and H3.1.2, we can conclude that maturity of different disciplines is in theory applicable but would need at least two participants (from SW and from EE) sharing their experience in answering one questionnaire. This is not feasible in practice. Overall, H3.1 is considered as partially true.

6.2 Availability of required documents (H3.3) and level of in-House guidelines (H3.4)

To examine the availability of functional descriptions, a single choice question was asked for each discipline about when the functional descriptions are available (#72). The availability of documents is poor in general when engineering starts and differs due to discipline and company size. SW engineering, compared to

Table 6: Reuse Strategies in different engineering disciplines (qualitatively) (RQ3; Q3)

Related hypotheses	**Findings**	**Evaluation**	**Validity**	**Relev. sec.**
H3.1 The maturity metric M_{MOD} is applicable to different disciplines	Only a few participants could answer detailed questions for both SW and EE (#76, n = 11)	Partially true	o	Sec 6.1
H3.1.1 $M_{MOD\_EE,Q3}$/$M_{MOD\_SW,Q3}$/ $M_{MOD\_M,Q3}$ are dependent on the company's number of employees	$M_{MOD\_EE,Q3}$/$M_{MOD\_SW,Q3}$/ $M_{MOD\_M,Q3}$ show highest values for companies with #emp > 1000, decreasing with company size (#66): true for SW and EE, false for M. Comp. where 251 < #emp < 1000 (#66) have slightly higher values than smaller and larger ones (n = 11)	Partially true	o	Sec 6.1
H3.1.2 Highest overall $M_{MOD,Q3}$ values are reached by different combinations of $M_{MOD\_SW,Q3}$ and $M_{MOD\_EE,Q3}$	Best overall rating if high values in both categories $M_{MOD\_EE,Q3} > M_{MOD\_SW,Q3}$ for 8 out of n = 11	True	o	Fig. 4
H3.2 Universal Module is used as a reuse strategy in different disciplines. It is, further, related to the company size	Percentage of application (yes / partially) in EE (30 / 25) > HMI (26 / 23) > SW (18 / 23)	True	+	Sec 6.3
	Highest values where #emp > 1000, smallest where #emp < 50 (n = 57)	True		
H3.3 Availability of documents is poor when engineering starts. It later differs due to discipline and company size	Percentage of availability in early phases according to disciplines (#72): SW < EE = M	True		Sec 6.2 Tab. 7
	According to company size (#66): Highest availability of functionality specification (#72) in companies with #emp > 1000, lowest with #emp = 50..250; valid for all disciplines	True	+	
H3.4 Level of detail of In-House guidelines varies between disciplines	Most stringent guidelines (#77) in SW (22%), followed closely by EE (19%), and M (10%) (n = 75)	True	+	Sec 6.2 Tab. 8
H3.5 Automatic generation of information in engineering still rare in all disciplines	Automatic generation is applied in between 34% (HMI) and 57% (EE) with requirements documents being the most frequently used source	True	+	Sec 6.3 Tab. 9

EE and M, shows the lowest availability (cp. Table 7), justifying the aforementioned gut feeling of practitioners. The availability of functional specifications shows the highest values for companies with more than 1,000 employees and the lowest ones for companies with 50 to 250 employees across all disciplines. Therefore both aspects of H3.3 are true. The percentage of available documents when design starts or sequentially delivered at milestones shows the lowest values for SW. The lowest values for the specification of functionality are found for companies with 50 to 250 employees.

To enhance engineering strategies in the future and to reduce duration of engineering and shorten time to market, these findings should be considered to enlarge the availability of required documents as a prerequisite to establish parallel, interdisciplinary development workflows.

Table 7: Availability of documents in different disciplines H3.3 #72

Discipline/Availability of documents	**SW**	**EE**	**M**
when design starts	9%	13%	13%
often too late	28%	29%	25%
delivered sequentially at milestones	25%	31%	25%
need to be requested by engineer	20%	8%	13%
engineer must clarify details with the customer	3%	4%	3%
not applicable	15%	15%	20%

Companies frequently rely on in-house guidelines (for details cf. section 8.4). H3.4 considers potential differences between disciplines regarding the level of de-

tail of these guidelines (cp. Table 8): Most exact ones are provided in SW engineering (21%), closely followed by EE (20%), and M (11%). Despite limited access to documentation, SW engineers lead in precise formulation of guidelines. H3.4 is true. This may confirm the perceived inequity between disciplines.

Table 8: Stringency of in-house guidelines in different disciplines H3.4 Q3 #77

Discipline/Evaluation	**SW**	**EE**	**M**
coarse-grained guidelines	33%	36%	44%
detailed guidelines	31%	29%	24%
exact instructions	21%	20%	11%
not applicable	15%	15%	21%

Table 9: Usage of code generation in different disciplines in % H3.5 #76

Generate from (column) for (lines)	**PP**	**M**	**EE**	**SW**	**HMI**	**BOM**
contracts	14%	-	-	-	-	-
requirements documents	30%	26%	29%	38%	26%	12%
component lists	-	11%	11%	-	-	24%
E-Plan	-	7%	17%	-	-	10%
models	-	-	-	12%	8%	-
$\sum$ from above	44%	44%	57%	50%	34%	46%
nothing	20%	18%	17%	16%	20%	17%

Project planning (PP), Mechanical (M), Electrics/ electronics (EE), Software (SW), Human Machine Interface (HMI), Bill of Material (BOM)

6.3 Universal module (H3.2) and code generation (H3.5) - reuse strategies in different disciplines

As introduced in [7] and section 2.6, different design approaches exist, such as universal modules (H3.2) and automatic generation from different design documents (H3.5).

Universal modules are a classical reuse strategy in control SW engineering but even more frequently used in EE and HMI (cp. Table 6). Interviews and code analysis of different companies revealed that the company size seems to be related to the usage of this strategy. Therefore, the company size was also analyzed. The highest percentages for universal module usage were qualitatively identified for companies with more than 1000 employees and the lowest usage for companies with less than 50 employees. Quantitative results are available from Q4 (cp. RQ5, H5.1.1).

Analyzing automatic code generation and configuration (cp. Table 9), sources needed to be identified, revealing that across disciplines and particularly in SW engineering, requirements documents are most frequently used. Overall, however, the observed values are still low. Consequently, H3.5 is true.

7 Causes and consequences of control software complexity (RQ4)

As control SW complexity is assumed to complicate SW engineering and, thereby, impact SW maturity, complexity should be analyzed in more detail (cp. Section 2.4). This analysis enables the development of methods, workflows, and tools as countermeasures. SWMAT4aPS [7] intended to measure SW complexity using the number of CPUs per machine or plant, LOC, or number of POUs as single or combined indicators. Unfortunately, most companies were neither willing nor able to share these values. Consequently, [4] proposed to consider additional measures for complexity. In SWMAT4aPS+ (Q2), not only the program size was asked as a complexity indicator, but the participants were also asked for a subjective rating of SW complexity, leading to the result that subjective ratings might work, but also showing deviations between program size and self-assessment (cp. [9] Table 1, complexity and size).

Recent results (cp. [8]) proved that companies rate complexity as a highly relevant characteristic, as complexity correlates to the necessary effort to realize a desired function. Unfortunately, neither the subjective measure nor the numbers of CPUs, POUs, or LOC delivered significant results. In the following, we introduce the causes and consequences of control SW complexity, as revealed by Q4. We can prove the applicability of the self-assessment of control SW complexity (H4.1) and its manageability (H4.4). We can show dependencies to PM and MM (H4.3) as well as key drivers of complexity (H4.2), means to cope with complexity (H4.5), and obstacles of using them. The results derived in this section are summarized in Table 10 and introduced in the following.

7.1 Applicability of self-assessed control software complexity (H4.1, H4.3) and Key Drivers of complexity (H4.2)

The self-assessment of control SW complexity using two questions (#22, #23) delivered much better results

Table 10: What are the causes and consequences of control software complexity? (RQ4)

Related hypotheses	**Findings**	**Evaluation**	**Validity**	**Relev. sec.**
H4.1 Self-assessment of control software complexity is better applicable than objective measures like LOC or number of POUs	Questions (#22, #23) asked for complexity and its manageability, respectively. Informative dependencies are revealed	True	+	Sec 7.1
H4.2 The key drivers of complexity are customer requirements and variability	Main reasons for complexity (#24^{F}) are customer requirements (30%), variability (about 11%), and technical and process related details (about 11%) (n = 61)	True	++	Sec 7.1
H4.3 The self-assessed complexity of control software is larger in PM than in MM	Correlation of complexity (#22) and self-classification (#69) • WMW: W_s = 916.5, z = 3.055, p = 0.001, r = 0.39, n = 61	True, Medium effect	++	Sec 7.1
H4.4. The self-assessed manageability of control software projects' complexity depends on the type of simulation approach/tool used	Correlation of approach/tool (#5) and complexity (#22) • H(2) = 2.819, p > 0.1, n = 61 Correlation of approach/tool (#5) with manageability (#23) • H(2) = 7.153, p = 0.067, n = 61	False True	- o	Sec 7.2
H4.4.1 Self-assessed manageability of control software projects' complexity is influenced by the usage of MATLAB/Simulink	Correlation of MATLAB/Simulink usage (#17) and manageability of complexity (#23) • WMW: W_s = 357, z = -2.861, p = 0.004, r = -0.37, n = 61	True, Medium effect	+	
H4.4.2 Self-assessed manageability of control software projects' complexity is influenced by the extent of IT-oriented modeling tools used	Correlation of IT-oriented modeling (#6) and complexity management (#23) • WMW: W_s = 34.5, z = -3.014, p = 0.003, r = -0.71, n = 18	True, Large effect	+	
H4.4.3 There exist obstacles for usage of IT or control-oriented modeling	Hindrances for IT-oriented modeling (#8^{F}, #9^{F}): 29 entries For simulation/control-oriented modeling (#12^{F}, #13^{F}): 23 entries	True	++	Sec 7.2
H4.5 Modularization and training are means to manage complexity	Implemented measures (#25^{F}): modularization (19%), training (14%), documentation (7%), communication and simulation (5%) (n = 42)	True	++	Sec 7.3

compared to the objective measures mentioned above. It reveals more informative dependencies than identified in Q2, i.e., between the overall complexity of the industrial sector and the complexity of individual companies operating in this sector.Therefore, H4.1 is true.

As PM is, in general, more customer-specific than MM, e.g., regarding size and location of the plant premises, one would expect control SW in PM to be more complex, or at least to be perceived more complex by the developers. To validate this hypothesis (H4.3), a one-tailed Wilcoxon-Mann-Whitney test was used. Thereby, H4.3 can be confirmed, as companies assigning themselves towards PM assess the complexity of their control SW significantly higher (Mdn = 4, scale from 1 for low to 5 for high) than companies assigned towards MM (Mdn = 3). The effect size can be specified as medium (Table 10). It should be noted that in individual cases the assignment between MM and PM is not trivial and in these cases the limits of the questionnaire study are reached. Thus, additional interviews would be necessary for a further in-depth analysis.

Key drivers for complexity (H4.2, #24^{F}) are *customer requirements* (30%), *variability* (about 11%), and *technical and process-related details* (about 11%). *Customer requirements* as well as *technical and process-related details* are not negotiable and need to be coped

with, but *variability management* can be supported as introduced in Section 8. Consequently, H4.2 is true.

7.2 Modeling and Simulation influences self-assessed manageability of control software projects complexity (H4.4)

The few selected companies in ([9], Table 1) showed that whenever subjective complexity values are higher than or equal to 5 (highest rating is 6), MBSE tools such as IT-oriented and control-oriented ones are also applied, with only one outlying company not using any of these tools. Therefore, a more detailed analysis of complexity regarding MBSE tools seems promising.

To better differentiate between different approaches in this questionnaire, we followed the advice of industrial experts and asked explicitly for IT-oriented modeling tools and simulation / control-oriented ones (#5). Using this question, the companies that answered can be divided into four groups (usage of both approaches (n=11), usage of only IT-oriented approach (n=12), usage of only simulation / control-oriented approach (n=12), no usage of modeling tools (n=17)). We assume the self-assessments of control SW complexity and manageability of complexity (#22, #23) to be influenced by the usage of modeling tools [4] and, therefore, expect verifiable significant differences in these complexity values between the aforementioned groups using a Kruskal-Wallis Test. While the self-assessment of control SW complexity is not significantly influenced, the self-assessment of the manageability of complexity is affected at a weakly significant level ($p = 0.067$) by the modeling approach used (Table 10). While post hoc tests are deemed not feasible due to the already only weakly significant Kruskal-Wallis test, a simple comparison of medians shows companies using both approaches to assess themselves best (usage of both approaches Mdn = 1.5, usage of only IT-oriented approach Mdn = 2, usage of only simulation / control-oriented approach Mdn = 2, no usage of modeling tools Mdn = 3, lower values correspond to better manageability) especially compared to companies not using modeling tools. Of course, this result is not of representative significance and will need future research.

Analyzing both approaches in more detail reveals the following: As MATLAB / Simulink can be used for both modeling approaches, the question arises, whether the usage of MATLAB / Simulink influences the self-assessment of manageability of complexity, too. Therefore, for H4.4.1, a Wilcoxon-Mann-Whitney test was calculated to identify differences in the self-assessed manageability of their control SW projects' complexity depending on the usage of MATLAB / Simulink. A statistically significant difference between both groups was identified: Companies using MATLAB / Simulink assess the manageability of the complexity of their control SW projects better (Mdn = 1, lower values correspond to better manageability) than those not using MATLAB / Simulink (Mdn = 2) with a medium effect (Table 10). Most companies selected the usage of MATLAB / Simulink among other programming languages (#17) instead of as a modeling tool (#5) as we expected.

Free text questions asked for the advantages and disadvantages of using simulation / control-oriented tools exclusively, partially, or not at all. In total, nine participants named advantages that can be categorized as *close connection to the finished product, time and cost savings due to optimized design, and early detection of errors / weaknesses*. Regarding disadvantages in case of not using these tools, ten answers were given, which can be categorized as *difficulty to predict consequences and effort, late recognition of design*, and finally, *possibly longer commissioning times*.

Interestingly, companies using IT-oriented modeling tools for entire projects compared to those using it only partially show significantly better values in the self-assessed manageability of their control SW projects' complexity (Mdn = 1 for entire projects, Mdn = 2 for partial use, lower values are better). This corresponds to a large effect (Table 10). In contrast to this result, the self-assessed manageability of control SW projects' complexity in companies using simulation / control-oriented tools does not differ significantly between those who used them only partially or for the entire project. This confirms H4.4.2.

Additionally, seven companies mentioned advantages of using IT-oriented modeling tools in free-text questions, including *comprehensibility, reusability*, and *optimized structure*. Fourteen companies that do not or only partially use IT-oriented tools named disadvantages that can be categorized as *lack of knowledge transfer, lack of SW documentation, long development time*, and *poor SW quality*.

Obstacles for the usage of modeling (cf. H4.4.3) are of particular interest, as they might be resolved in the future to ease application of such techniques. Twenty-nine entries for the IT-oriented and 23 for the simulation / control-oriented approach were named. For both, the most frequently named entries are *time expenditure, effort*, and *complexity* (IT 14%; control 22%), followed by *lack of knowledge / experience / training* (IT 10%; control 13%) and *usage of own systems / concepts / tools* (IT 10%, control 9%).

As further reasons for not using modeling tools, companies mentioned *lack of functionality, no seamless*

integration with IEC programming, and *low profitability for components with low reuse*. Furthermore, some participants explained that modeling tools are only used partially, since aspects of EE and M cannot be fully described in models.

7.3 Means to manage complexity (H4.5)

In total, 42 companies answered a free-text question regarding means for managing complexity and mentioned modularization (19%), training (14%), documentation (7%), communication (5%), and simulation (5%). Especially modularization has been confirmed and discussed using M_{MOD+} in Section 5. Training and qualification aspects have also been mentioned to hinder the application of OO IEC (cf. Section 8.3). As a result, the importance of modularity is further emphasized.

Conclusions of detailed analysis of complexity and its manageability: We could prove that the self-assessments revealed meaningful results. Furthermore, we were able to show that the complexity of control-level SW in PM is more complex than in MM, as well as its dependency on the use of modeling notations. Modeling approaches are still not used by 28% of the companies, but its importance to cope with complexity was underpinned. Finally, we identified weaknesses in current approaches, like including M and EE aspects beyond SW, as well as time effort and training aspects indicating future research and development challenges.

8 Strategies for reuse and design of control software (RQ5)

Following the research gap from Section 2.6, we analyzed reuse strategies in industry (H5.1), focusing on variant and version management and the corresponding tools (H5.2), as aPS are variant-rich and evolve over decades [16]. OO as one paradigm to ease reuse is analyzed in more depth, especially with free-text questions on advantages and difficulties of implementing OO IEC, or reasons why it is not used (H5.3). Also, guidelines and checklists to foster higher reuse were explicitly examined. Additionally, we analyzed organizational factors, such as department size and its relation to department agility.

8.1 Analysis of reuse strategies (H5.1)

Reuse strategies have been of interest for years in embedded systems and for more than 20 years also in aPS. Classical and more sophisticated strategies have been introduced. Prior analysis (cp. Table 14) revealed some results, but mostly qualitatively. We analyzed, according to Table 12, four hypotheses detailed by two sub-hypotheses that will be introduced in the following.

8.1.1 Most frequently used reuse strategies (H5.1.1) and dependencies in between them (H5.1.2)

This subsection analyses frequently applied reuse strategies and their dependencies including the relevance of guidelines and checklists in regard to separate module development departments. Question #51, which asks for the applied reuse strategies, is formulated as a multiple-choice question and allowed to insert further ways of reuse in an additional text box. However, the evaluation showed that the most common reuse strategies are covered by the choices offered. In detail, companies specified to use the following reuse strategies: *libraries* (used by 71% of companies), *templates* (53%), *universal modules* (44%), and *code generation / code configuration* (31%), which confirms that H5.1.1 is true. On the other hand, the comparison of the mean values for the different reuse strategies revealed that dependencies between *templates* and *universal module, universal module* and *code generation / code configuration, code generation / code configuration* and *libraries* could not be identified, and, therefore, H5.1.2 is considered as true.

The relationship between companies with a dedicated module development department and companies using guidelines and checklists was assessed in H5.1.2.1. (cf. Table 12). In [68], companies having a separate department or person in charge of development and maintenance of company-specific standard functions were addressed, dividing control SW engineers into module developers responsible for the management of module libraries with sound knowledge about reuse and architectures, and application engineers using these modules to compose an application based on their knowledge about the technical process or application. From the 46 companies that answered the question on the availability of a module developer or module developer department (25 from MM and 21 from PM) more than 50%, more precisely 26 companies, do have a module developer or the respective department. Only 38 companies from these 46 answered the questions on usage of guidelines and checklists. A significant relation could be identified between the companies with a module developer or module developer department and the usage of guidelines and checklists for the creation of application- and / or customer-specific control modules (two-tailed Fisher's exact test, $p = 0.016$, cp. Table 11). Such companies seem to be more likely to use guidelines / check-

Table 11: Relations between the availability of a module developer or module developer department (#27) and the usage of guidelines and checklists (#37, different selectable answers) (↑ - positive impact/higher usage, ↓ - negative impact/less usage)

Evaluation criterion	Avail. y/n	Impact	p*	eff.s. V**
creation of library blocks/blocks with standard functionality	21/8	-	> 0.1	0.05
creation of application- and/or customer-specific control modules	22/3	↑	0.016	0.45
naming conventions for variables, POUs etc.	22/5	-	> 0.1	0.28
software architecture (call graph, hierarchy levels, fault handling and operation modes)	19/5	-	> 0.1	0.16
standard structure for comments in modules (e.g. date and author)	17/3	-	> 0.1	0.27

* probability value
** Cramér's V (Validity of results: low ($p > 0.1$, orange), medium ($p \leq 0.1$, $V < 0.45$, yellow), medium to large ($p \leq 0.1$, $V \geq 0.45$, green); column Avail., y-yes for in companies with module developers and n-no for companies without)

lists for the creation of application- and / or customer-specific control modules. The effect size is medium to large ($V = 0.45$). Independent from the availability of a module developer or the respective department 13 of the 38 companies do not use guidelines or checklists at all. While all the other selectable choices in #37 show a high availability of guidelines for companies with a module developer (department), no further significant relations could be identified (cp. Table 11, $p > 0.1$), which is unexpected, especially for the easy to develop and implement naming conventions and comments. Interestingly, the standard structure for components in modules shows the lowest numbers.

To summarize, H5.1.2.1 is true for only one type of guideline / checklist (i.e., guidelines / checklists for the creation of application- and / or customer-specific control modules). Nevertheless, all companies with module developers show higher numbers for the availability of all guidelines.

Additionally, a significant relation between the implementation of standard functionalities and the usage of guidelines could be identified: The implementation of *fault detection / diagnostics* and *operation modes* as standard functionality in each module / function block is significantly associated with the usage of guidelines / checklists for the *creation of library blocks / blocks with standard functionality* (Pearson chi-square test $\chi^2(1) = 9.462$, $p = 0.002$, $V = 0.44$). Furthermore, if *HMI interfaces* are implemented in addition to *fault detection / diagnostics* and *operation modes*, the relation to the usage of guidelines / checklists for the *creation of library blocks / blocks with standard functionality* is significant as well ($\chi^2(1) = 11.561$, $p < 0.001$, $V = 0.49$). In both cases, it seems that companies using guidelines / checklists for the *creation of library blocks / blocks with standard functionality* are more likely to use standard functionality in each module / function block and vice versa (20 times more likely if *fault detection / diagnostics, operation modes*, and *HMI interfaces* are used, 7.6 times more likely if *fault detection / diagnostics and operation modes* are used).

Relations between usage of *interfaces / properties (OO)* and *selected criteria* (H5.1.2.2): In [23], [18], interfaces for data exchange were identified as enablers to ease reuse. More mature code also included the usage of mode of operation, diagnostics and fault handling. Using this background knowledge, we manually selected potentially interesting combinations for the categories PLC type (#16), interfaces (#30), functions included in modules (#31), checklist (#34), version management tools (#42), and variant management tools (#47) to analyze whether there are relations (shown in Table 13).

Deeper analysis showed that participants answered unexpectedly: we assumed that companies utilizing *interfaces / properties (OO)* would use *OO IEC* as a prerequisite, but over 57% of companies using *interfaces / properties* do not (53% in MM, 61% in PM). Consequently, participants answering the questionnaire referred to other mechanisms than the *interfaces / properties of OO IEC*. For example, in classical PLC programming, FBs are used in analogy to OO-classes by defining different variable ranges to distinguish public and private variables for data encapsulation. Also, other workarounds exist to achieve higher quality and use classical quality management tools like SonarQube: the PLC code (e.g., STEP7 SCL files, cp. Fig. 5) is converted to C and subsequently analyzed based on the rules defined in the underlying quality model of the tool. Using the tool results, first, the C file is analyzed and, subsequently, the PLC code is optimized based on the issues identified by the tool.

Nevertheless, there is a significant relation between the usage of classical PLCs (e.g., Siemens) and the usage of *interfaces / properties (OO)* (cf. Table 13, group (')). Thereby, companies that specify to use classical PLCs are less likely to use *interfaces / properties* (0.18 times). Furthermore, we assumed a relation between *interfaces / properties (OO)* and other typically used ways to implement interfaces in an average control SW project (input / output variables of the function blocks

Table 12: What strategies for reuse and design of control software are used? (RQ5)

Related hypotheses	**Findings**	**Evaluation**	**Validity**	**Relev. sec.**
H5.1. Analysis of reuse strategies and their relations reveal additional insights				
H5.1.1 Libraries, templates and universal modules are the most frequently used planned reuse strategies	Companies apply multiple planned reuse strategies (#51): libraries (71%), templates (53%), universal modules (44%), code generation / code configuration (31%)	True	+	Sec 8.1.1
H5.1.2 No relations between the different reuse strategies can be identified	Types of planned reuse (#51) lack relations between each other	True	+	Sec 8.1.1
H5.1.2.1 If a module developer department exists, guidelines and checklists are prescribed	Companies with a module developer department (#27) prescribe design guidelines regarding application-specific and/or customer-specific FBs (#34)	True (for one of five criteria)	cp. Tab. 11	Sec 8.1.1
H5.1.2.2 Dependencies exist between the usage of interfaces/properties (OO, including OO IEC) and selected variables	Multiple dependencies could be identified. Correlations of: • Usage of interfaces/properties and classical PLCs: $\chi^2(1) = 5.012$, $p = 0.025$, $V = 0.29$ • Usage of interfaces/properties and other means of implementing interfaces: $\chi^2(1) = 5.966$, $p = 0.015$, $V = 0.31$ • Usage of interfaces/properties and exchange via global variables: $\chi^2(1) = 12.876$, $p < 0.001$, $V = 0.46$ • Usage of interfaces/properties in combination with OO IEC and usage of standardized functionality: two-tailed Fisher's exact test, $p = 0.037$, $V = 0.28$ • Usage of OO IEC and classical PLCs: $\chi^2(1) = 3.974$, $p = 0.046$, $V = 0.26$ • Usage of OO IEC and PC-based PLCs: $\chi^2(1) = 4.481$, $p = 0.028$, $V = 0.28$	True	cp. Tab. 13	Sec 8.1.1
H5.1.3 Simplified Antkiewicz levels C0-C3 show the same tendencies as M_{MOD+}	Simplified 4-level classification considering information of changes and selected reuse strategies significantly affects M_{MOD+} • $H(3) = 14.107$, $p = 0.00$	True	++	Sec 8.1.2
H5.2 Detailed analysis of variant and version management and tools used reveal additional insights				Sec 8.2
H.5.2.1 State of the art variant management tools are not used	69% of MM or PM use a variant management tool (#45, multiple selections allowed): Excel (Macros) (50%), TIA portal and SAP/ERP (both 38%), others (17%)	True	+	Sec 8.2.1
H5.2.2 Version management tools are used for software change tracking	77% of MM or PM use a version management tool (#43, #42, multiple selections allowed): Tortoise SVN (63%), Subversion (37%), SAP (22%), Team Viewer (19%) (n = 27)	True	+	Sec 8.2.2
H5.2.3 PM are more likely to use variant management tools than MM	Correlation of Version tracking method (#45) with self-classification (#69) • Fisher's exact test: $p = 0.035$, 2-tailed, $n = 61$ • Cramér's $V = 0.32$	True, Medium effect	+	Sec 8.2.3
H5.3 Reasons for and effects of OO IEC usage can be identified				Sec 8.3
H5.3.1 OO IEC usage correlates with selected aspects.				Sec 8.3
H5.3.1.1 OO IEC is used more in MM than in PM	Usage of OO IEC (#17): 44% (or 42%; data inconsistency) in self-classified PM (#69), 36% in MM (n = 61)	False	-	Sec 8.3
H5.3.1.2 Some reuse strategies are applied more often if OO IEC is used	Correlation of OO IEC usage (#17) with usage of libraries (#51) • Pearson's chi squared test: $\chi^2(1) = 5.503$, $p = 0.019$, $n = 61$ • Cramér's $V = 0.30$	True	-	Sec 8.3

Related hypotheses	Findings	Evaluation	Validity	Relev. sec.
H5.3.1.3 OO IEC influences the self-assessed control software complexity and its manageability	Correlation of OO IEC usage (#17) with complexity (#22) and its manageability (#23) • WMW for #22: W_s = 789.5, z = 0.695, p > 0.1, n = 61 • WMW for #23: W_s = 693.5, z = 0.775, p > 0.1, n = 61	False	-	Sec 8.3
H5.3.1.4 With an increasing amount of customer specific code, usage of OO IEC decreases	Correlation of OO IEC usage (#17) with proportion of customer-specific code (#32) • Pearson's chi squared test: $\chi^2(3) = 1.697$, p > 0.1, V = 0.17 → not significant	False	-	Sec 8.3
H5.3.1.5 Companies using OO IEC reach higher M_{MOD+} values	Correlation of OO IEC usage (#17) with resulting M_{MOD+} • WMW: W_s = 945, z = 2.976, p = 0.003, r = 0.38, n = 61	True, Medium effect	++	Sec 8.3
H5.3.2 OO IEC usage offers various advantages	Advantages listed by companies (#19): increased code quality, use of standard tools, comprehensive code, design cost / design time saving (n = 24)	True	++	Sec 8.3 Fig. 7
H5.3.3 There are reasons / obstacles for not using OO IEC	Obstacles listed by companies (#18): lack of qualification of service and / or application engineers, lack of platform support (n = 25)	True	++	Sec 8.3 Fig. 9
H5.3.4 Companies encounter difficulties while implementing OO IEC	Difficulties listed by companies (#20): employees' resistance, effort, compatibility concerns (n = 24)	True	+	Sec 8.3 Fig. 8
H5.4 The software department's agility is influenced by the number of application software engineers / software development department's size	Correlation of SW department's agility (#68) with its size (#67) • H(2) = 8.274, p = 0.016, n = 58 Post hoc test: • Significantly higher agility values in departments with #emp < 25 compared to departments with #emp > 75: W_s = 366, z = -2.647, p = 0.004, r = -0.38 • No significant difference between departments with #emp < 25 and 25 ≤ #emp ≤ 75: W_s = 324, z = -1.559, p > 0.025	Partially true	+	Sec 8.4 Fig. 10
H5.5 Company-specific guidelines/checklists are more frequently used than the PLCopen Guidelines	Guidelines/checklists listed by companies (#34): 80% internal guidelines and checklists; 10% PLCopen; 10% none (n = 61)	True	+	Sec 8.4

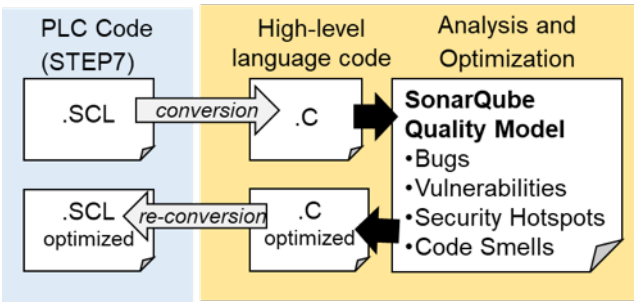

Fig. 5: Workaround to apply quality management tools from high level programming languages to PLC software

/ functions, data exchange via global variables). Companies utilizing interfaces / properties seem to be 0.16 times less likely to use one or both other ways of implementing interfaces. This finding is further supported by the fact that usage of *interfaces / properties (OO)* is associated with the usage of data exchange via global variables, whereby, again, companies using *interfaces / properties (OO)* seem to be 0.13 times less likely to implement their data exchange via global variables (Table 12). However, it has to be noted that the companies answering the questionnaire might have misunderstood the term *interface*, since a concept of interfaces has been introduced about three years ago in classical PLCs. Thus, further analysis by means of interviews are required to verify this result.

Extra-functional tasks included in any function block are of great interest as they increase code complexity (#31). Therefore, we analyzed on the basis of the 24 companies using *interface / properties (OO)* and *OO IEC*, whether they, by default, include *fault detection / diagnosis, operation modes*, or *HMI interfaces* as well

Table 13: Relations between usage of interfaces/properties (OO)(')/the usage of (interfaces/properties (OO) & OO IEC)(") and PLC type (#16), interfaces (#30), functions included in modules (#31), checklist (#34), version management tools (#42), version management tool (#47) (↑ - positive impact/higher usage, ↓ - negative impact/less usage)

Evaluation criterion usage of		**Impact**	**p***	**eff.s. V****
classical hardware PLC' (like Siemens)		↓	0.025	0.29
data exchange via global variables'		↓	< 0.001	0.46
interfaces other than interfaces / properties (OO)'		↓	0.015	0.31
operation modes & fault detection/diagnosis	'	-	> 0.1	0.16
	"	↑	0.037	0.28
intern checklists'		-	> 0.1	0.07
version management tools'		-	> 0.1	0.15
variant management tools'		-	> 0.1	0.21

* probability value
** Cramér's V (Validity of results: low (p > 0.1, orange), medium (p ≤ 0.1, V < 0.45, yellow), medium to large (p ≤ 0.1, V ≥ 0.45, green)

as their combinations. Of the 24 companies, 14 chose *others, not applicable*, or *none*. All of the remaining ten companies include *fault detection / diagnosis* as standard, nine *operation modes*, seven *HMI interfaces*, and six include all three. While there is no significant relation between *interfaces / properties (OO)* and the implementation of *operation modes* and *fault detection / diagnosis*, the relation turns out to be significant when *interfaces / properties (OO)* are considered in combination with OO IEC. Companies using *interfaces / properties (OO)* in combination with *OO IEC* are 8.0 times more likely to use operation modes and fault detection / diagnosis as standard functionality in their modules / function blocks. This seems to validate the fact that companies use *interfaces / properties (OO)* within the extension of *OO IEC* to implement standard functionalities in a more modular way (Table 12). Thereby, companies could potentially produce more comprehensible code, ensure a higher quality of control SW, and save cost and time. These are all deemed typical advantages of *OO IEC* (cp. Section 8.3).

It is expected that the use of interfaces results in a higher degree of modularity - e.g., by enabling information hiding - and thus, there would be additional benefits when using version or variant management. Therefore, we expected the implementation of interfaces by utilizing *interfaces / properties (OO)* to be associated with the application of version or variant management, but in both cases, no significant relation could be verified (both two-tailed Fisher's exact test, version management: p > 0.1, V = 0.15; variant management: p > 0.1, V = 0.21, cp. Table 13). Future research including, apart from interfaces and properties, also the *OO IEC* concepts methods and inheritance will need to analyze these effects in more detail. However, to investigate this relation, in-depth questions about the SW architecture are required, which is not feasible in a purely questionnaire-based study.

8.1.2 Simplified Antkiewicz levels for maturity assessment can be derived from Q4 (H.5.1.3)

Results of prior questionnaires (cp. Table 14), as well as the state of the art (cp. Section 2.3), show that there are still significant obstacles in the modularization of control SW in aPS due to the high complexity of the systems, the closely interwoven interaction of different disciplines [8], and the lack of systematic reuse approaches. The latter is also reflected by the lower levels defined by Antkiewicz et al. [6]. In the following, we discuss to what extent the questions of Q4 can be used to identify different maturity levels in SW, e.g. the lower Antkiewicz levels L0 to L3. Companies that mainly apply ways of reuse described in Level L0 are characterized by using universal modules to reuse control SW (#51), not using tool support for variant (#47) or version management (#42), and usually not merging variants that have been developed in parallel (#55). Ways of reuse from Levels L1 and L2 show first attempts to manually merge variants (#55), to track changes of versions manually (#45), and to exchange information about changes within development teams (#1). Reuse levels above L3 are characterized by using code configuration (#51) but cannot be precisely distinguished using the questionnaire results: These higher levels are described by systemized reuse using SPLE based on feature models. Although such approaches are also available in automation, they are so far rarely used, which is why they were not part of the questionnaire (cf. Section 2). Applying the derived criteria to the data set revealed that there are no companies to be assigned to any of the levels higher than L1, because the criteria cannot be directly transferred to the participating companies dealing with domain-specific challenges and boundary conditions that strongly differ from the ones in classical software engineering, which is focused by Antkiewicz et al. [6].

Since the original Antkiewicz levels are not applicable, in the following, the idea is adapted as a four-level concept of clusters C ranging from C0 to C3, which provides a feasible classification. We selected, according to Antkiewicz [6], the type of information of changes as one important criterion and the reuse strategies code

generation and templates as a second criterion resulting in four different groups (cp. Figure 6):

(1) C1 - poor information of changes (meetings, phone, email) and lack of reuse strategies (*code generation, templates*): 11 companies
(2) C2 - information of changes (Product Lifecycle Management (PLM) systems, Enterprise Resource Planning (ERP), change information, or AML) and lack of reuse strategies (*code generation, templates*): 10 companies
(3) C3 - information of changes (PLM systems, ERP, change information, or AML) and advanced reuse strategies (*code generation, templates*): 31 companies
(4) C0 - deviating cluster with poor information of changes (meetings, phone, email) but advanced reuse strategies (*code generation, templates*): 9 companies

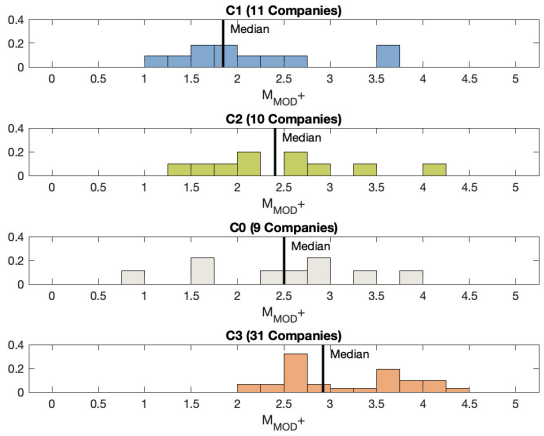

Fig. 6: M_{MOD+}-values of Antkiewicz levels (identified clusters C0 - C3 sorted by M_{MOD+} median values)

The maturity values M_{MOD+} and the C levels show the same tendency besides group C0 with a huge scatter. H5.1.3 is true, showing the necessity for further research to, at first, propose a concept similar to Antkiewicz et al.'s levels L_3 and higher [6] for aPS SW and, secondly include additional questions into the questionnaire on the basis of this concept.

Furthermore, M_{MOD+} values are significantly affected by the adapted Antkiewicz levels C0 - C3 (Table 12). As we expected companies from C3 to score the highest M_{MOD+} values, three right-tailed Wilcoxon-Mann-Whitney tests were used with a Bonferroni correction (cp. section 3.4) for further analysis. Thereby, all effects are reported at a significance level of 0.0167. While M_{MOD+} values of companies assigned to C3 differ significantly from the M_{MOD+} values of companies from C1 ($W_s = 120.5$, $z = 3.307$, $p < 0.001$, $r = 0.510$) and C2 ($Ws = 136.5$, $z = 2.218$, $p = 0.013$, $r = 0.346$), they do not appear to be different to M_{MOD+} values of companies in C0 ($Ws = 123.5$, $z = 1.929$, $p = 0.027$, *not significant*, $r = 0.305$).

Concluding, there is a group of companies C0 that show the lowest values according to Antkiewicz (L0) but higher values for M_{MOD+} and may be interpreted as specific approach.

8.2 Detailed analysis of variant and version management and tools used reveal additional insights (RQ5 H5.2)

Vogel-Heuser et al. [4] identified variability management as "a key issue to cope with the complexity of evolution in aPS [..], because aPS are variant-rich and require cross-discipline variability models coping also with parallel evolution and different versions." Consequently, we conducted further analysis in detail.

8.2.1 Usage of variant management tools (H5.2.1)

Earlier analysis revealed that more sophisticated variant management approaches like configuration of code out of product lines or feature models supported by tools such as pure::variants are rarely used in aPS software engineering. This survey confirms these findings as the highest ratings are gained by *Excel (Macros)* (50%), *TIA portal* and *SAP / ERP* (both 38%), and *others* (17%); (multiple selections allowed). Overall, only 69% of MM or PM use a variant management tool.

8.2.2 Reference to version management tool used for software change tracking (H5.2.2)

To ensure consistency for evaluating this hypothesis, the intersection of two questions is analyzed, i.e. companies specifying to use a specific version management tool (#42) and at the same time indicating to use a version management tool for change tracking (#45).

A *version management tool* is used by 77% of MM or PM, i.e. by 47 of 61 companies (#42). In case SW change tracking (#45) is realized via a version management tool, we assume a higher maturity compared to manual tracking in the function block itself or separate files such as Excel sheets. Consequently, we analyzed the applied tools. SW changes are tracked by 57% of the companies (n = 27 out of 47, 14 MM, 13 PM) via a *version management tool* (only companies

that use version management are included); most frequently, TortoiseSVN (63% of 27), followed by Subversion, SAP, and Team Viewer were named (Multiple choices allowed). The distribution seems inconsistent at first glance: Since TortoiseSVN is a graphical client of Subversion, the participants who chose TortoiseSVN should be a subset of the SVN users, but twelve of the 61 companies stated that they use TortoiseSVN and *not* SVN. However, this uncertainty is plausible considering the background of the participants - many of the study participants work in management positions, who are less acquainted with the classification of the different systems. Analyzing change tracking of versions in more detail, only 21% of the companies document the variants and versions implemented in the delivered aPS in SAP; equally distributed in MM and PM, and 17% integrated the workflow in a PLM system. Regarding maturity in managing versions, these findings show important weaknesses to be addressed. Consequently, H5.2.2 is true.

8.2.3 PM are more likely to use variant management tools than MM (H5.2.3)

Since aPS from PM are usually more complex and customer-specific than aPS from MM, we expect that the need of PM to use variant management tools is higher than the need of MM. In the questionnaire answers a statistically significant relation could be identified between the classification into MM and PM and the usage of variant management tools (two-tailed Fisher's exact test, $p = 0.035$). Thereby, PM are 9.1 times more likely to use variant management tools compared to MM. The effect size can be described as medium ($V = 0.32$). Therefore, H5.2.3 is true.

8.3 Reasons for and effects of OO IEC usage (H5.3)

The benefits of OO IEC are controversial and, therefore, its usage is examined, identifying correlations between the usage of OO IEC and other aspects, as well as advantages of using it (cp. Figure 7), difficulties identified if used (cp. Figure 8), and obstacles hindering its usage (cp. Figure 9).

At first, the percentage of usage in MM and PM should be introduced (H5.3.1.1), expecting that MM (36%) would have a higher amount compared to PM (42%, corrected for inconsistencies, cp. Section 3.2) as they should face less customer-specific requirements, and therefore, be able to apply more systematic approaches. H5.3.1.1 is false, however, as more PM use OO IEC than MM, but this confirms the maturity values already discussed in section 5 (H2.2).

In H5.3.1.4 we investigated, if an increasing amount of customer-specific SW leads to a decreasing usage of OO IEC. However, the analysis showed that there is no statistical significant relation between the percentage of customer specific code and usage of OO IEC (Table 12). Thus, H5.3.1.4 is false The relation of modularity maturity M_{MOD+} and OO IEC adoption can be compared to a hen and egg problem. On the one hand, usage of OO IEC should enable improved modularity, on the other, modular code is a prerequisite for the adoption of OO IEC. The analysis of H5.3.1.5 confirmed that companies using OO IEC have significantly higher M_{MOD+}' values (Mdn = 3.29) than companies not using OO IEC (Mdn = 2.50), Table 12 (to avoid influences on the calculation of M_{MOD+} and therefore influences on the outcome of the statistical test, the scoring for OO IEC was excluded in M_{MOD+}').

Analyzing dependencies between OO IEC and reuse strategies, a significant correlation with the usage of libraries could be identified (Table 12, H5.3.1.2). OO IEC claims to ease the management of complex SW, but the usage of OO IEC seems to neither influence the self-assessed control SW complexity nor the self-assessed manageability of control SW projects' complexity (Table 12). Therefore, H5.3.1.3 is deemed invalid.

Analysis of the usage of OO IEC in relation to PLC platforms does not provide unambiguous results because companies often need to deliver/support more than one PLC platform and, consequently, more than one answer for question #16 (type of PLC used) is given. Therefore, companies cannot be separated precisely into groups which only use one type of PLC.

For companies, which use only one platform type, the distribution of OO IEC is as follows: 68% classical, 12% PC-based, and 20% others. This seems contradictory, as most classical PLCs do not explicitly support OO IEC, but it could be revealed from interviews that either companies wrongly claim to use OO IEC with classical PLCs without using the OO IEC commands in code according to the standard [11] or they use an OO IEC preprocessing step and convert or import to classical PLCs via an import functionality (cp. Figure 5). Companies using OO IEC named the following advantages of implementing OO IEC (H5.3.2): *increased code quality* (63%), *more comprehensible code* (58%), and *savings in both design cost and design time* (42%) (Figure 7). The *use of standard tools from SW development* (67%) is unexpected but in line with the unexpected combination of classical PLCs. We assume that companies developed specific solutions to ease design using standard tools. This finding will need further analysis using interviews.

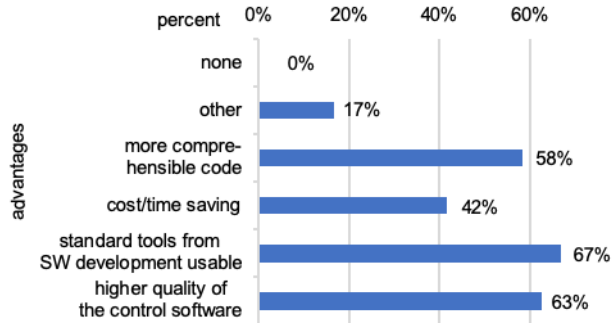

Fig. 7: Advantages resulting from the use of the object-oriented extension of the IEC 61131-3 programming languages H5.3.2, #19

Difficulties when implementing OO IEC (Figure 8) were analyzed to understand and eliminate them (H5.3.4): Most frequently, *resistance to change among employees* (63%) was named by companies using classical PLCs, followed by *compatibility concerns* (50%) by companies using PC-based platforms. Additionally, *training effort* (29%) and *high cost* (21%) as well as *necessary technical conversion* (46%) are named. As reasons for not using OO IEC (H5.3.3), the *lack of qualification of application engineers* (38%) or *service* (35%) and *lack of platform support* (32%) are rated the highest (Figure 9). Further analysis between usage of OO IEC and platforms reveal problems similar to [8], because most companies use more than one platform. Concluding the more detailed findings regarding OO IEC, many well-known benefits have been confirmed. The creativity of industrial companies with high variability seems to create unexpected proprietary beneficial solutions, combining standard tools with all types of PLCs. Nevertheless, the resistance of employees and the qualification of both application engineers and service technicians needs to be addressed if OO IEC shall be used successfully by a higher percentage of companies.

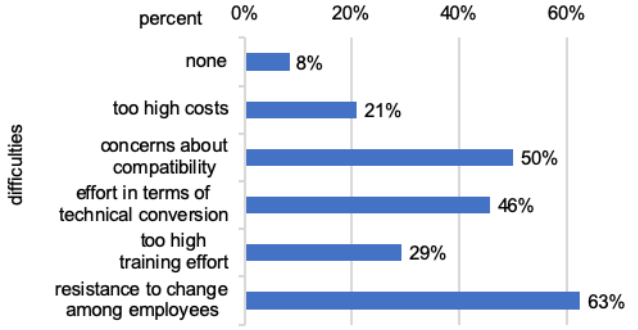

Fig. 8: Difficulties that have arisen during implementation of OO IEC H5.3.4, #20

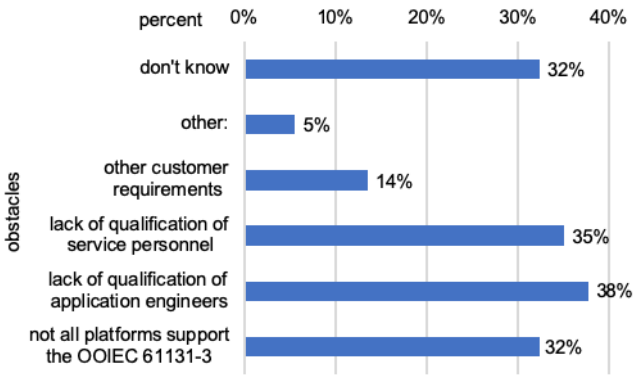

Fig. 9: Obstacles of OO IEC usage H5.3.3, #18

8.4 Software departments' agility (H5.4) and usage of guidelines (H5.5)

As emphasized in [23], a company's size and agility are highly relevant factors to be considered when assessing its reuse strategy. More precisely, the same reuse strategy can have a positive or negative impact on changed boundary conditions, such as the SW department's size.

Thus, H5.4 assumes a significant influence of the software department's size on a company's self-assessed agility, characterized by the number of automation technicians and SW engineers. To validate this hypothesis, firstly, a Kruskal-Wallis test was carried out and confirmed that assumption (Table 12).

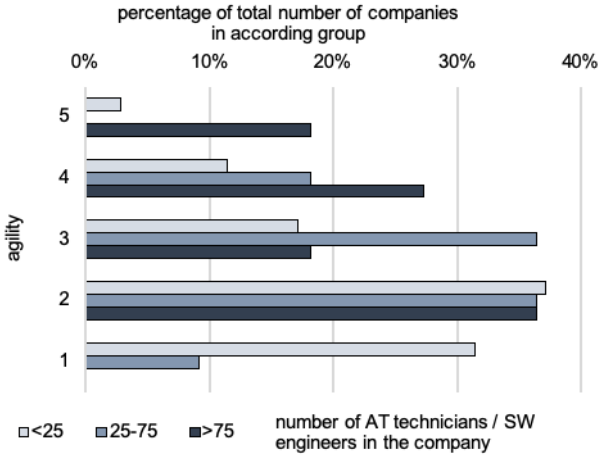

Fig. 10: Assessment of the agility of the company's software development department from 1 - "can work on call" to 5 - "formal processes necessary for exchange" H5.4 #68

We expected small departments (<25 employees) to be more agile compared to medium-sized (25-75 employees) and large departments (>75 employees). Two left-tailed Wilcoxon-Mann-Whitney tests with Bonfer-

roni correction (all effects reported at a significance level of 0.025) were conducted as post hoc tests. While there is actually a statistically significant difference in the self-assessed agility between departments with less than 25 employees (Mdn = 2) and more than 75 employees (Mdn = 3), no significant difference could be identified between departments with less than 25 and departments with 25 to 75 employees (Table 12). Consequently, H5.4 is partially true.

Usage of guidelines was addressed in [8] for testing. Available guidelines / checklists represent joined knowledge and should support and ease gaining higher maturity across the community. Especially PLCopen guidelines aim to ease the effort and increase the quality developed by all members of PLCopen. Question #34 examined the usage of guidelines and checklists to ensure high quality and traceability of SW, showing that internal guidelines and checklists are frequently used compared to only 10% PLCopen usage. In conclusion, PLCopen - representing one type of community guideline - is used less than internal guidelines. The latter are used by 80% of the companies and, consequently, H5.5 is true.

9 Conclusion and Outlook

By introducing the Q4 questionnaire, its results and expert responses of Q3, the assessment of the maturity and success factors for the design of field-level control SW in MM and PM companies was demonstrated to enable the identification of weaknesses and strengths. This assessment is a first step to improve a company's position at the global market. Whenever possible, quantitative results were introduced concerning the five RQs. Additional qualitative results are gained from Q3 especially regarding the engineering workflow. $\text{QoaPS}_{\text{SWE}}$ exceeds previous surveys (cp. Table 14, right column) in providing significant results regarding, e.g., SW maturity ($\text{M}_{\text{MOD+}}$), complexity, and MBSE, more precisely, modeling tools used. To support future research, individual obstacles of the industry are identified and some related aspects, which need to be addressed, are introduced in the following: at first aspects regarding the obstacles for industrial companies and need for improved tool support, next regarding the improvement of questionnaire-based approaches and last but not least the combination with interview or code analysis based approaches as further step. Based on assessed results using $\text{QoaPS}_{\text{SWE}}$, detailed pain points can be identified or confirmed by further interviews, workflow analyses and code audits to find means to improve aPS SW.

Obstacles for industrial companies and need for improved tool support

The usage of OO IEC was found to correlate with the usage of standard functionality, which increases quality and reduces complexity of aPS SW to save costs and time. However, OO IEC is in most companies not or only partially used. $\text{QoaPS}_{\text{SWE}}$ further showed that the usage of IT-oriented and control / simulation-oriented modeling supports the handling of complexity. Variant and version management still require improvements: appropriate tool support is necessary, but due to the heterogenous engineering tools used in the different companies, this is not a simple task.

Despite the benefits of IT-oriented modeling, it lacks functionality as well as seamless integration with IEC programming. The effort of its use is only worthwhile for basic programs that are frequently re-used contrary to software, which is frequently affected by customer-specific adaptions, e.g., in the field of special purpose machinery (the same applies for simulation / control-oriented modeling). Furthermore, M and EE aspects cannot be modeled but only SW, which hampers the standard use of these tools for the development of mechatronic systems. Generally, $\text{QoaPS}_{\text{SWE}}$ showed the challenges but also the remaining open questions in the design of EE engineering parts of an aPS.

Version tracking is still conducted with largely manual methods such as naming conventions. Interfaces are used, but not to the possible extent provided by OO IEC [11]. Lack of platform support is an essential technical obstacle for OO approaches, whereas Figure 9 reveals that the qualification of application engineering and service staff constitutes further organizational obstacles. Besides qualification, 63% of the companies also criticized the staff's resistance to change (Figure 8). From interviews, we learned that some companies do not know how to achieve an OO or even service-oriented SW architecture and are aware of the risk of failing with such a mid-term development. For example, the entire application might not fulfill the real-time requirements anymore or require a more costly CPU, or the transformation may simply turn out too difficult or time consuming. In principle, many technological parts are available, but the usage is limited, as many barriers exist. Concluding, the purely technical approach currently favored will not be enough and needs to be supported by qualification, training, and organizational measures (cp. [69]). Contrary to expectations, community guidelines such as PLCopen are used less frequently than company-specific guidelines.

Table 14: Comprehensive overview of the State of the Art and comparison with results of previous studies

Criteria	**SWMAT 4aPS [7]**	**SWMAT4aPS+ [8]**	**SWMAT4a PSi/m [9]**	**Interview Study [18]**	**QoaPS$_{SWE}$**
General Focus	Maturity for aPS (focus on SW) in all engineering and operation phases	Maintainability of aPS for SW update in operation phase	Typical maturity values and variations for metrics within the same company (sensitivity) and for selected research approaches (OO, MBSE)	Reuse mechanisms, variant and version management, transfer to planned reuse with SPL	Design maturity M_{MOD+} (Q4) sig. correlation with MM vs PM; complexity; OO; MBSE, disc. specific design maturity (Q3- qual.) and overall optimum, RQ3
Method	Q1, Code analysis (call graphs), Interviews (companies known)	Q2, Selected interviews	(Q1 & Q2), Selected aspects (companies known) and Q3	Guided Interviews	Q3 & Q4; only questionnaire byQ
#Companies (Other, MM, PM) and industrial sector	16 PS (2); MM (12): food (9), pharma/logistics (3), health (2), automotive (2), engineering (2); PM (2): plastic, logistics (2)	68 **x**: assignment to industrial sector, cp. [8] Fig. 2 ; others (40), automotive (20), operate in more than 2 sectors (51)	7 with 14 local engineering dept. SPM and PM Selected qual.: steel (1 with 4), food&beverage (7), pharma (5), logistics (2)	6 Assignment by authors: (Serial) MM (4): food& beverage (2), medicine / pharma (2), PM (2): wood- working (1), automotive (1)	293 Q4: 207; MM(146), PM(61) Q3: 86; MM (75), PM(11) **x**: industrial sector ambiguous despite VDMA classification ≥2 sectors chosen
Differentiation in betw. MM and PM	**-**: assigned by authors, lack of effects	**x**: only 28% assignable (MM: special purpose/standard)	**x**: companies known	**x**: companies known	++: MM ⇔ PM by criteria and signif. correlation, RQ1
a) Companies size (# employees);	**-**: none	Comparably large companies (26<250, 12 = 250-1000, 30>1000) (cp. [8] Fig. 3);	**x**	**x**	+: agility correlates with number of employees
b) Programmer in department (dep.)	Without automation comp.: 2<20; 3<30; 2<50; 2<100; 2>100	**-**: none	**x**	**x**	+: agility correlates with programmers in dep. (RQ5)
Maturity: Maturity indicators (M)	**o**: $M_{MOD,Q1}$, $M_{TEST,Q1}$, $M_{OP,Q1}$, $M_{\sum,Q1}$: either $M_{MOD,Q1}$ or $M_{OP,Q1}$ high for highest $M_{\sum,Q1}$ Focus Control software	**x** (not applicable for different disc.): highest influence on $M_{\sum,Q1}$: quality assurance, release proc. for library blocks, interdisc. usage of version mgm. tool, testing simulation, automated configuration	**o**: identification of typical M values and variations within the same company or within a specific industrial sector ≥ 3 different approaches reach high values	**o**: mature answers as potential for a transfer to Software Product Line (SPL) highlighted by authors; Identification of potential and challenges supporting / hindering transfer to SPL	Q4: ++: M_{MOD+} significant and details for MM/PM, RQ2 Q3: +: either M_{MOD_SW} high or M_{MOD_ET} for highest overall M_{MOD}, RQ3
Measurement of complexity as disturbing variable for maturity M_{MOD}:					
Code size metrics (POU, LOC)	**-**: inconsistently answered for POU and LOC	**-**: code size metrics fall short of prop. illustrating complexity (RQ 3.4)	**-**: self-assessment (1 question)	**x**	+: self-assessment (2 questions) proves to be a more appropr. approach to measure complexity, RQ4
CPU, programmers/ applic. or machine	**-**	**-**	**x**	**x**	
Other artefacts like most critical task	**x**	**-**: Artefacts more construc- tive but less objective	**x**	**x**	
Key driv. of compl./ means to mng.	**x** / **x**	**x** / **x**	**x** / **x**	**x** / **x**	+/+: identified: H4.2. key drivers; H4.5 means to mng.

++: strong evidence, +: moderate evidence, o: low evidence, -: no evidence/not verifiable, x: not applicable/not examined

	Criteria	SWMAT 4aPS [7]	SWMAT4aPS+ [8]	SWMAT4aPSi/m [9]	Interview Study [18]	$QoaPS_{SWE}$
Maturity	Antkiewicz levels	Interview-based for 4 companies: L0-L3	**x**	**x**	**x**	**x** (adapted); simplif. applic., correlates sig. with M_{MOD+}
	IEC languages	**-**	**-**: 82% high level languages, usage unknown	**-**	**x**	**o**: (HMI/PLC/modeling tools still inconsistent)
	Modeling	**-**	**-**: usage of different tools cp. [8] Fig. 6, no classification into IT vs. simulation/ control-oriented	**-**	**x**	**o**: IT vs. simulation/ control-oriented: sig. correlations with complexity management; advantages, disadvantages, obstacles (free text); RQ4
	OO IEC	**-**	+: qual and platform dependencies need more research	**x**	**x**	**o**: sig. detail, advantages, disadvantages, obstacles (free text); H5.3
	Interfaces	**-**	**-**	Included in approach A (large company, large plant)	**x**	**+**: correlation sig.; RQ5, but inconsistent; further analysis required
	Variant/ Version management (mgm.) tools	**o**: version mgm. tool and generation of new variants addressed	**o**: SW status of operating plant often unknown; 44% do not use variant mgm.; Some analysed, but variant and version mixed	**o**: reasons for not using variant construction Q3 cp. [9] Fig. 6; some named	**o**: 50% (3) not using variant mgm. tool, 33,3% use no support for variant mgm.; challenges: no global knowl. base	**++**: PM more likely to use variant mgm. than MM; H5.3 Variant and version differentiated- confirming less sophisticated tools for variant and classical software version mgm. Tools => lack of appropriate tools and engineering workflow
	Reuse strategies: universal module dependencies between strategies & organis. aspects	**x**	**x**	3 clusters	Library POUs (5) Code-configuration / generation / templates (4) Copy, Paste & Modify (4) Reuse Levels (archit. asp.)	Q4: usage differs, H5.1.1 **-**: module developer (exist) =>guidelines for creation of application- and or customer-specific control modules. Q3: **o**: dependency between usage of Universal Module and company size (H3.2)
	Standard functions in module	**x**	**o**: H3.5 65% Fault detection, and 50% diagnosis, 46% mode of operation	Included in approach A (large company, large plant)	**x**	**o**: qualit. for 24 comp. using interface / properties:
Engineering Workflow	Engineering Workflow	Release proc. of libraries	Release procedure of libraries	Release procedure of libraries	**-**: criteria to build new variant	**x**
	Effic. in- form. exch.	+	+	**-**	**x**	**x**
	a)Generation from x to y	From MBSE or ECAD to PLC	From MBSE to PLC Code	From MBSE or ECAD to PLC and to HMI (Q3)	Code configuration / generation / templates (4)	**+**: generation from x for y in Q3 (qual); RQ3
	b)Docum. in time	**x**	**x**	**x**	**x**	**++**: Q3 (qualitative for different disciplines); H3.3
	Availability of guidelines	**x**	+	**-**	Guidelines in 3 comp., under dev. in 1	**+**: details and PLCOpen; Table 7

Improvement of questionnaire or self-assessment based approaches
We identified weaknesses in current approaches, e.g. regarding the interdisciplinary exchange of development documents as well as time effort and training aspects, indicating potential for future research and developments. The attempt to ask in-depth questions for both EE and SW aspects in one questionnaire results in poorly answered sections and confirms the limitations of a questionnaire due to the limited discipline-specific expertise of the participants. Consequently, future iterations need to address at least two people representing the two different disciplines to gain meaningful answers or an interview based approach.

Additionally, also the assessment of a company's maturity level similar to Antkiewicz et al. [6] bears the potential to derive recommendations for action. While the higher levels of reuse according to Antkiewicz et al. [6] are not subject of the survey, since SPLE has not been widely established in the field of aPS yet, the questionnaire results instead reveal degrees of reuse of control SW, which are not covered by these levels. Based on the results, different ways of reuse can be derived, depending on, e.g., the usage of OO IEC, the modularization of different disciplines, or the development of reusable library blocks. In future work, it is, therefore, planned to enlarge these levels based on current and future questionnaire results but also based on lessons learned from industrial research projects to develop a classification for modularity of control SW. By means of such a classification, after their assessment companies could be informed about their improvement potential in order to raise their SW to the next higher maturity level.

Combination with interview or code analysis based approaches as second step
Static software analysis as second step: Currently, a holistic maturity assessment approach considering the strength of the individual company, its unique requirements, e.g. requirements of the company's industrial sector, and its abilities is lacking. Regarding the maturity assessment, static code analysis and the use of SW metrics have proven beneficial in identifying weaknesses in PLC SW. Further, recent research promotes the derivation of recommendations for actions based on the insights gained through target-oriented SW analysis [70]. Enlarging the static analysis with a structured analysis procedure including maturity estimation based on a combination of questionnaire and guided interviews, could be a promising approach to bridge the aforementioned gap. Furthermore, it has the potential to enable domain experts to analyze their SW by themselves and derive recommendations for actions targeting the improvement of identified weaknesses while taking their company's boundary conditions into account.

In summary, there is still a lot to be done to improve the design of field-level control code in machine and plant manufacturing especially in high wage countries like Germany.

Acknowledgements We thank all companies who answered the questionnaire.

Funding
This study was partially funded by the DFG (German Research Foundation) (VO 937 / 31-1). We also thank experts from EFIMA consortium (AZ-1208-16) for their support in the design and evaluation of Q3.

Conflicts of interest/Competing interests
The authors declare that they have no conflict of interest.

Availability of data and material
Will be provided later.

Code availability
Not applicable.

Authors Contributions

Birgit Vogel-Heuser: Conceptualization; Methodology; Investigation; Resources; Supervision; Data curation; Formal analysis; Validation; Writing - original draft; Writing - review & editing.
Juliane Fischer: Conceptualization; Methodology; Investigation; Data curation; Writing - original draft; Writing - review & editing, Visualization.
Eva-Maria Neumann: Conceptualization; Methodology; Investigation; Data curation; Visualization; Writing - original draft; Writing - review & editing.
Matthias Kreiner: Data curation; Formal analysis; Visualization; Writing - original draft; Writing - review & editing.

Ethical Approval
Participation in the questionnaire was voluntary and data was collected in anonymous form. It was possible for the participants to leave questions unanswered by indicating "no answer". Additionally, the paper's main data is original, has not been published and is not under consideration for publication elsewhere.

Consent to Participate
All authors declare that they have agreed for authorship, have read and approved the manuscript, and have given the consent for submission and subsequent publication of the manuscript.

Consent to Publish
All authors are consent to publish this article with its included data in The International Journal of Advanced Manufacturing Technology and approve this submitted version.

Appendix

In the following, the questions from the questionnaire Q4 are provided. Sub items included in M_{MOD+} (⧫1), M_{TEST} (⧫2) and M_{OP} (⧫3) are marked accordingly. ^{F} Free-text question. $^{\rightarrow a}$ Follow-Up question from question a.

Sub items regarding development approach

1. ⧫1 How do the different disciplines exchange information during the development phase?
2. Who is involved in the meetings for exchange between the different disciplines?
3. How often do the various disciplines exchange information on average?
4. ⧫1 In what form is the requirements specification usually available?
5. ⧫1 What kind of modeling tools (m.t.) do you use for PLC-based control development and test?
6. $^{\rightarrow 5}$ To what extent do you use IT-oriented m.t.?
7^{F}. $^{\rightarrow 6}$ What are the advantages of IT-oriented m.t.?
8^{F}. $^{\rightarrow 5,6}$ Why do you not use/ony partially IT-oriented m.t.?
9^{F}. $^{\rightarrow 6}$ What are the disadvantages of not using/partial use of IT-oriented m.t.?
10. $^{\rightarrow 5}$To what extent do you use simulation/control m.t.?
11^{F}. $^{\rightarrow 10}$ What are the advantages of simulation/control m.t.?
12^{F}. $^{\rightarrow 5,10}$ Why do you not/ony partially use simulation/control m.t.?
13^{F}. $^{\rightarrow 12}$ What are the disadvantages of not using/partial use of simulation/control m.t.?
14. ⧫1 Does your company have interdisciplinary mechatronic modules?
15. ⧫1 What is modularised in your company?

Sub items regarding software and reusability

16. What types of control systems do you use?
17. ⧫1 Which programming languages are used in your company for the control software (not HMI)?
18^{F}. $^{\rightarrow 17}$ Why is the object-oriented extension of IEC 61131-3 not used?
19^{F}. $^{\rightarrow 17}$ What are the advantages of using the object-oriented extension of the IEC 61131-3 programming languages?
20^{F}. $^{\rightarrow 17}$ What difficulties have arisen during the introduction?
21^{F}. $^{\rightarrow 17}$ Would Control Patterns in your company help to use the object-oriented extension of IEC 61131-3?
22. How complex do you consider the control software projects at the control/field level in your company to be?
23. How well does your company manage this complexity?
24^{F}. What are the main drivers of this complexity?
25^{F}. What measures do you implement to manage this complexity?
26. What types of standard software functions are used for control software projects in your company?
27. $^{\rightarrow 26}$ Is there a responsible person/department for the development of company-specific standard functions?
28^{F}. $^{\rightarrow 26}$ Which libraries are most frequently used?
29^{F}. $^{\rightarrow 26}$ Which additional libraries should be developed in the near future or which would be most helpful for you?
30. How are interfaces predominantly implemented in an average control software project?
31. ⧫1 What functions does each module/component have as standard?
32. What is the proportion of project/customer-specific codes in an average control software project?
33. ⧫1 What is the usual process from creation to release of a library block?
34. What guidelines/checklists does your company use to ensure high quality and traceability software?
35. $^{\rightarrow 34}$ What are the PLCopen guidelines/checklists?
36^{F}. Which additional PLCopen guidelines/checklists would you consider useful?
37. $^{\rightarrow 34}$ What are these policies/checklists aimed at?
38^{F}.$^{\rightarrow 34}$ What problems arise from not using guidelines/checklists?
39. What control patterns are used by your company?
40^{F}. $^{\rightarrow 39}$ What are they (e.g. motion control)?
41^{F}. $^{\rightarrow 39}$ What are advantages of additional Control Patterns?
42. ⧫1 Does your company use a version management tool, and if so, which one?
43. $^{\rightarrow 42}$ Why do you not use a version management tool?
44. Which disciplines use a version management tool?
45. ⧫1 How does version tracking work?
46. $^{\rightarrow 45}$ In what form are the manual comments recorded?
47. ⧫1 Which variant management tool do you use?
48. ⧫1 Are there variants of mechatronic modules or are the variants discipline-specific?
49. $^{\rightarrow 47}$ Why is variant construction not used?
50. $^{\rightarrow 47}$ What problems arise from not using variant construction?
51. What types of planned reuse do you use for the development of open-/closed-loop or drive software?
52. ⧫1 $^{\rightarrow 51}$ From which tools/models is code generated?
53. $^{\rightarrow 51}$ Why is the largest part of the so-called glue code, i.e. code for linking, necessary?
54. $^{\rightarrow 51}$ Are there library blocks that can be used in dif-

ferent machine types (e.g. control of a standard drive)?
55. How are parallel developed variants combined?

Sub items regarding quality assurance

56. ♦2 Which quality assurance measures are used in your company?
57. ♦2 What proportion of the open-/closed-loop and drive software (individually designed according to customer requirements) is tested in your company?
58. How are model-based development and simulation used in your company?
59. ♦2 Which part of the control software (application software) is tested by means of simulation?

Sub items regarding handover to the customer and commissioning of the machine or plant

60. ♦3 Does the application engineer/programmer on site commission the machines or systems?
61. ♦3 How is the software handed over to the customer?
62. ♦3 How do you install updates/adjust the software?
63. How often is the software of a machine/system updated during its lifetime after acceptance? (on average)
64. ♦3 When does the service/software department in the company usually exchange information with the customer about current software statuses on site?

Sub items regarding company specific questions

65. Which sales markets is your company active in?
66. How many employees work in your company?
67. How many automation/software engineers does your company have in application development?
68. How agile do you consider your company's software development department to be?
69. Would you classify your company more as a machine or plant engineering company?

Questions from Q3 used for the discipline specific assessment of $M_{MOD\_EE,Q3}$ / $M_{MOD\_SW,Q3}$ (cp. Fig. 4)

70. How well does the exchange with the respective discipline work?
71. How often does the exchange take place?
72. When are the functional descriptions available in the respective department?
73. (cp. ♦1 #15)
74. Does the control software (PLC) contain all possible variants (universal module)? (specific question for $M_{MOD\_SW,Q3}$)
75. Does the circuit diagram contain variants/options that are not installed in the specific machine/product? (specific question for $M_{MOD\_EE,Q3}$)
76. (cp. ♦1 #52 $^{\rightarrow 51}$)
77. Does your company use internal guidelines for the respective discipline that serve to achieve results of the highest possible quality and traceability?

Questions from Q1 used for $M_{MOD,Q1}$

78. How is the in-house cooperation arranged?
79. Which documents are exchanged during a development project?
80. How is the development project documented?
81. Who started the initiative to use modularization?
82. (cp. ♦1 #15)
83.Is continuous integration used?
84. $^{\rightarrow 83}$If yes, what is the tool chain you use?
85. (cp. ♦1 #17)
86. How often are library components used?
87. (cp. ♦1 #33)
89. How is the decision to form new variants made?
90. (cp. ♦1 #42)
91. (cp. ♦1 #45)
92. How often is code generation from EPLAN or other engineering tools applied?
93. (cp. ♦1 #52)
94. Are projects configured automatically from libraries based on templates?

References

1. B. Vogel-Heuser, S. Feldmann, J. Folmer, J. Ladiges, A. Fay, S. Lity, M. Tichy, M. Kowal, I. Schaefer, C. Haubeck, W. Lamersdorf, T. Kehrer, S. Getir, M. Ulbrich, V. Klebanov, and B. Beckert, "Selected challenges of software evolution for automated production systems," *Proceeding - IEEE Int. Conf. on Industrial Informatics (INDIN)*, pp. 314–321, 2015.
2. A. Lüder, A. Klostermeyer, J. Peschke, B. A., and T. Sauter, "Distributed automation: Pabadis versus hms," *IEEE Trans. Ind. Inform*, vol. 1, no. 3, pp. 31–38, 2005.
3. G. Rzevski, "On conceptual design of intelligent mechatronic systems," *Mechatronics*, vol. 13, no. 10, pp. 1029–1044, 2003.
4. B. Vogel-Heuser, A. Fay, I. Schaefer, and M. Tichy, "Evolution of software in automated production systems: Challenges and research directions," *Journal of Systems and Software*, vol. 110, pp. 54–84, 2015.
5. K. Thramboulidis, "The 3+1 SysML View-Model in Model Integrated Mechatronics," *Journal of Software Engineering and Applications*, vol. 03, no. 02, pp. 109–118, 2010.
6. M. Antkiewicz, W. Ji, T. Berger, K. Czarnecki, T. Schmorleiz, R. Lämmel, Ş. Stănciulescu, A. Wąsowski, and I. Schaefer, "Flexible product line engineering with a virtual platform," *Companion Proc. of the 36th Int. Conf. Softw. Eng.*, pp. 532–535, 2014.
7. B. Vogel-Heuser, J. Fischer, S. Feldmann, S. Ulewicz, and S. Rösch, "Modularity and Architecture of PLC-based Software for Automated Production Systems: An analysis in industrial companies," *Journal of Systems and Software*, vol. 131, pp. 35–62, 2017.

8. B. Vogel-Heuser and F. Ocker, "Maintainability and evolvability of control software in machine and plant manufacturing - An industrial survey," *Control Engineering Practice*, vol. 80, pp. 157–173, 2018.
9. B. Vogel-Heuser, F. Ocker, and E.-M. Neumann, "Maturity variations of PLC-based control software within a company and among companies from the same industrial sector," *IEEE Industrial Cyber-Physical Systems (ICPS)*, pp. 283–290, 2018.
10. V. Vyatkin, "Software Engineering in Industrial Automation: State-of-the-Art Review," *IEEE Trans. Ind. Inf.*, vol. 9, no. 3, pp. 1234–1249, 2013.
11. B. Werner, "Object-oriented Extensions for IEC 61131-3," *IEEE Industrial Electronics Magazine*, vol. 3, no. 4, pp. 36–39, 2009.
12. M. Bonfé, C. Fantuzzi, and C. Secchi, "Design patterns for model-based automation software design and implementation," *Control Engineering Practice*, vol. 21, no. 11, pp. 1608–1619, 2013.
13. B. Vogel-Heuser, "Usability experiments to evaluate UML / SysML-based model driven software engineering notations for logic control in manufacturing automation," *Journal of Software Engineering and Applications*, vol. 7, no. 11, 2014.
14. M. Konersmann, Z. Durdik, M. Goedicke, and R. Reussner, "Towards Architecture-Centric Evolution of Long-Living Systems (The ADVERT Approach)," *ACM SIGSOFT Int. Conf. on the Quality of Software Architectures*, 2013.
15. R. Reussner, M. Goedicke, W. Hasselbring, B. Vogel-Heuser, J. Keim, and L. Märtin, *Managed Software Evolution*. Cham: Springer International Publishing, 2019.
16. B. Vogel-Heuser, J. Fischer, S. Rösch, S. Feldmann, and S. Ulewicz, "Challenges for Maintenance of PLC-Software and Its Related Hardware for Automated Production Systems: Selected Industrial Case Studies," *IEEE Int. Conf. on Software Maintenance and Evolution*, pp. 362–371, 2015.
17. "IEC 61512-1:1997 Batch control - Part 1: Models and terminology, vol. 1.0," 1997.
18. J. Fischer, S. Bougouffa, A. Schlie, I. Schaefer, and B. Vogel-Heuser, "A Qualitative Study of Variability Management of Control Software for Industrial Automation Systems," *34th Int. Conf. on Sw. Maintenance and Evolution (ICSME)*, pp. 615–624, 2018.
19. V. Vyatkin, "IEC 61499 as Enabler of Distributed and Intelligent Automation: State-of-the-Art Review," *IEEE Trans. Ind. Inform.*, vol. 7, no. 4, pp. 768–781, 2011.
20. K. Thramboulidis, "IEC 61499 as an Enabler of Distributed and Intelligent Automation: A State-of-the-Art Review - A Different View," *Journal of Engineering*, 2013.
21. "System architecture for intralogistics (SAIL), VDI/VDMA 5100," 2016.
22. "ISO 13485:2016 - Medical devices - Quality management systems - Requirements for regulatory purposes," 2016.
23. E.-M. Neumann, B. Vogel-Heuser, J. Fischer, F. Ocker, S. Diehm, and M. Schwarz, "Formalization of Design Patterns and Their Automatic Identification in PLC Software for Architecture Assessment," *21st IFAC World Congress*, 2020, in press.
24. M. L. Alvarez, E. Estévez, I. Sarachaga, A. Burgos, and M. Marcos, "A novel approach for supporting the development cycle of automation systems," *Int. Journal of Advanced Manufacturing Technology*, vol. 68, pp. 711–725, 2013.
25. E. Estévez, M. Marcos, and D. Orive, "Automatic generation of PLC automation projects from component-based models," *Int. Journal of Advanced Manufacturing Technology*, vol. 35, pp. 527–540, 2007.
26. R. Drath, A. Fay, and T. Schmidberger, "Computer-aided design and implementation of interlock control code," *IEEE Conf. on Comp. Aided Control System Design*, pp. 2653–2658, 2006.
27. J. Ladiges, A. Fay, T. Holm, U. Hempen, L. Urbas, M. Obst, and T. Albers, "Integration of Modular Process Units Into Process Control Systems," *IEEE Transactions on Industry Application*, vol. 54, no. 2, pp. 1870–1880, 2018.
28. "NAMUR recommendation NE 148, "Automation requirements relating to modularization of process plants", NAMUR, Leverkusen, Germany," 2013.
29. R. Priego, A. Armentia, E. Estévez, and M. Marcos, "Modeling techniques as applied to generating tool-independent automation projects," *at - Automatisierungstechnik*, vol. 64, no. 4, pp. 325–340, 2016.
30. H. Koziolek, A. Burger, M. Platenius-Mohr, and R. Jetley, "A classification framework for automated control code generation in industrial automation," *Journal of Systems and Software*, vol. 166, 2020.
31. M. Broy, "Challenges in automotive software engineering," *Int. Conf. on Software Engineering (ACM)*, 2006.
32. "Scrum, online," *www.scrumalliance.org / why-scrum (retrieved 2017-05-26)*.
33. M. Follmer, P. Hehenberger, S. Punz, R. Rosen, and K. Zeman, "Approach for the creation of mechatronic system models," *Int. Conf. on Engineering Design*, pp. 258–267, 2011.
34. "Engineering data exchange format for use in industrial automation systems engineering - AutomationML, IEC 62714," 2014.
35. C. Mahler, "Automatisierungsmodule für ein funktionsorientiertes Automatisierungsengineering," 2014.
36. S. Fischer, L. Linsbauer, R. E. Lopez-Herrejon, and A. Egyed, "Enhancing clone-and-own with systematic reuse for developing software variants," *Int. Conf. on Software Maintenance and Evolution*, pp. 391–400, 2014.
37. J. Busby, "The problem with design reuse: an investigation into outcomes and antecedents," *Journal of Engineering Design*, vol. 10, pp. 277–296, 1999.
38. C. Maga, N. Jazdi, and P. Göhner, "Requirements on engineering tools for increasing reuse in industrial automation," *IEEE Int. Conf. on Emerging Technologies & Factory Automation (ETFA)*, 2011.
39. L. Gauss, D. Lacerda, and M. Sellitto, "Module-based machinery design: a method to support the design of modular machine families for reconfigurable manufacturing systems," *Int. Journal of Advanced Manufacturing Technology*, vol. 102, pp. 3911–3936, 2019.
40. B. Reimlinger, Q. Lohmeyer, R. Moryson, and M. Mebold, "Exploring how design guidelines benefit design engineers: An international and global perspective," *Design Science*, vol. 6, 2020.
41. E. Gamma, R. Helm, R. Johnson, and J. Vlissides, "Elements of reusable object-oriented software," *Journal Design patterns*, vol. 39, 2011.
42. G. Barbieri, N. Battilani, and C. Fantuzzi, "A PackML-based Design Pattern for Modular PLC Code," *IFAC-PapersOnLine*, vol. 48, pp. 178–183, 2015.
43. J. Fuchs, S. Feldmann, C. Legat, and B. Vogel-Heuser, "Identification of Design Patterns for IEC 61131-3 in Machine and Plant Manufacturing," *IFAC Proceedings Volumes*, vol. 47, no. 3, pp. 6092–6097, 2014.

44. D. Rabiser, H. Prähofer, P. Grünbacher, M. Petruzelka, K. Eder, F. Angerer, M. Kromoser, and A. Grimmer, "Multi-purpose, multi-level feature modeling of large-scale industrial software systems," *Software & Systems Modeling*, 2016.
45. K. Pohl, G. Böckle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles and Techniques.* Springer-Verlag New York, USA, 2005.
46. D. Hinterreiter, H. Prähofer, L. Linsbauer, P. Grunbacher, F. Reisinger, and A. Egyed, "Feature-Oriented Evolution of Automation Software Systems in Industrial Software Ecosystems," *IEEE 23rd Int. Conf. on Emerging Technologies & Factory Automation (ETFA)*, pp. 107–114, 2018.
47. A. Schlie, K. Rosiak, O. Urbaniak, I. Schaefer, and B. Vogel-Heuser, "Analysing variability in automation software with the variability analysis toolkit," *Int. Systems and Software Product Line Conference (SPLC)*, pp. 1–8, 2019.
48. T. Simon, B. Vogel-Heuser, and J. Fischer, "Variability Management for Automated Production Systems Using Product Lines and Feature Models," *14th IEEE Int. Conf. on Industrial Informatics (INDIN)*, pp. 1231–1237, 2016.
49. S. Feldmann and B. Vogel-Heuser, "Interdisciplinary product lines to support the engineering in the machine manufacturing domain," *Int. Journal of Production Research*, vol. 55, no. 13, pp. 3701–3714, 2017.
50. M. Fang, G. Leyh, J. Doerr, C. Elsner, and J. Zhao, "Towards model-based derivation of systems in the industrial automation domain," *19th Int. Conf. on Software Product Line*, pp. 283–292, 2015.
51. S. Schröck, A. Fay, and T. Jäger, "Systematic interdisciplinary reuse within the engineering of automated plants," *Annual IEEE Int. Systems Conf. (SysCon)*, pp. 508–515, 2015.
52. S. Biffl, E. Maetzler, M. Wimmer, A. Lueder, and N. Schmidt, "Linking and versioning support for AutomationML: A model-driven engineering perspective," *IEEE 13th Int. Conf. on Industrial Informatics (INDIN)*, pp. 499–506, 2015.
53. M. Wimmer, P. Novak, R. Sindelar, L. Berardinelli, T. Mayerhofer, and A. Mazak, "Cardinality-based variability modeling with AutomationML," *22nd IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, pp. 1–4, 2017.
54. M. Lucas and D. Tilbury, "Methods of measuring the size and complexity of PLC programs in different logic control design methodologies," *Int. Journal of Advanced Manufacturing Technology*, vol. 26, pp. 436–447, 2005.
55. B. Vogel-Heuser, J. Fischer, D. Hess, E.-M. Neumann, and M. Würr, "Managing Variability and Reuse of Extra-functional Control Software in CPPS," *Design, Automation and Test in Europe Conference (DATE)*, 2021.
56. B. Vogel-Heuser and S. Rösch, "Applicability of Technical Debt as a Concept to Understand Obstacles for Evolution of Automated Production Systems," *IEEE Int. Conf. on Systems, Man, and Cybernetics (SMC)*, pp. 127–132, 2015.
57. M. H. Halstead, "Elements of software science," 1977.
58. T. J. McCabe, "A Complexity Measure," *IEEE Trans. Software Eng.*, vol. 4, pp. 308–320, 1976.
59. L. Capitán and B. Vogel-Heuser, "Metrics for software quality in automated production systems as an indicator for technical debt," *13th IEEE Conf. on Automation Science and Engineering (CASE)*, pp. 709–716, 2017.
60. J. Wilch et al., "Introduction and Evaluation of Complexity Metrics for Network-based, Graphical IEC 61131-3 Programming Languages," *45th Annual Conf. of the IEEE Industrial Electronics Society (IECON)*, pp. 417–423, 2019.
61. A. Fields, *Discovering Statistics using IBM SPSS.* Sage, London, 2009.
62. "MATLAB: Statistics and Machine Learning Toolbox," *https://www.mathworks.com/products/statistics.html.*
63. J. Cohen, *Statistical power analysis for the behavioral sciences.* Academic Press, New York, 1988.
64. P. D. Ellis, *The essential guide to effect sizes: Statistical power, meta-analysis, and the interpretation of research results.* Cambridge University Press, Cambridge, New York, 2010.
65. H.-Y. Kim, "Statistical notes for clinical researchers: Chi-squared test and fisher's exact test," *Restorative Dentistry & Endodontics*, vol. 42, pp. 152–155, 2017.
66. P. Runeson, M. Höst, A. Rainer, and B. Regnell, "Case Study Research in Software Engineering: Guidelines and Examples," 2012.
67. B. Vogel-Heuser and S. Rösch, "Applicability of Technical Debt as a Concept to Understand Obstacles for Evolution of Automated Production Systems," *IEEE Int. Conf. Systems, Man, and Cybernetics*, pp. 127–132, 2015.
68. B. Vogel-Heuser, J. Fischer, E.-M. Neumann, and S. Diehm, "Key maturity indicators for module libraries for PLC-based control software in the domain of automated Production Systems," *16th IFAC Symposium on Inf. Contr. Probl. in Manuf. (INCOM)*, pp. 1610–1617, 2018.
69. M. Zou, B. Vogel-Heuser, M. Sollfrank, and J. Fischer, "A Cross-disciplinary Model-Based Systems Engineering Workflow of Automated Production Systems Leveraging Socio-technical Aspects," *IEEE Int. Conf. on Indust. Eng. and Eng. Management (IEEM)*, pp. 133–140, 2020.
70. B. Vogel-Heuser, J. Fischer, and E.-M. Neumann, "Goal-Lever-Indicator-Principle to Derive Recommendations for Improving IEC 61131-3 Control Software," *IEEE Int. Conf. on Indust. Eng. and Eng. Management (IEEM)*, pp. 1131–1136, 2020.

Success factors for the design of field level control code in machine and plant manufacturing - an industrial survey -

Birgit Vogel-Heuser[1] (ORCID: 0000-0003-2785-8819)
Juliane Fischer[1] Eva-Maria Neumann[1] Matthias Kreiner[1]