Dear reviewers,

We kindly thank you for your feedback. Your comments greatly helped us to improve our paper. In the following, we provide a detailed list of the changes concerning each comment. All changes we made to the document are highlighted in blue font. Also, we carefully revised the paper regarding grammar, spelling, and choice of words.

Kind regards,

Quang Huan Dong, Birgit Vogel-Heuser, and Eva-Maria Neumann
### Associate Editor

<table>
<thead>
<tr>
<th>Remark</th>
<th>Response / Changes</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Please improve the manuscript according to the hints of both reviewers. Please shorten the manuscript by omitting lengthy calculations.</td>
<td>Since the applicability of the approach is evaluated on a lab-sized demonstrator, both details on rationales and calculations are provided to prepare for future work in industrial settings. Revising detailed calculations might contradict the expectation of review #1, in which no significant rework of the text is needed. We reviewed and refined the calculations according to the reviewers’ comments. We would suggest keeping the approved calculation parts. To ease the reading on the factor calculations, they are structured as follows.</td>
<td>4.4</td>
</tr>
</tbody>
</table>

1. **Calculation of Document Urgency** ($RPI_{URGENCY}$)

2. **Calculation of Document Coherence** ($RPI_{COHERENCE}$)

| Especially, please provide a critical discussion and a concise conclusion, backed by evaluations. | First, to improve the **discussion and conclusion**, we make the following adaptions in section 5 (Conclusion and outlook). The adaptions describe impact and benefit of the proposed solution. | 4.4. and 5 |

To address this issue, a Risk Prioritisation Indicator of Documentation Debt (RPI4DD) is proposed to quantify the risk of documentation debt in control software. Related work addressing the quality of control code documentation is still scarce and primarily considers source code alone. To the best of the authors’ knowledge, this is the first study to transfer the Risk Priority concept in the GAMP method to automation software and its boundary conditions. The RPI4DD approach considers not only internal software factors (e.g., complexity or functionalities) but also includes external influencing factors, such as required on-site changes or tests. Therefore, compared to existing work, the proposed approach could provide a broader view on control code documentation since aPS is developed in a multidisciplinary environment. Besides documentation, the proposed approach could be generally applied to other aspects of software quality (e.g., testing) or could be further applied to other disciplines.

Second, the applicability of the approach is **evaluated** on a lab-sized demonstrator. We make the following adaptions in section 4.4 (Proof of concept of Risk Prioritisation Indicator of Documentation Debt) and section 5 to evaluate the obtained numerical results and their benefit for company:

altogether, the RPI4DD is determined by multiplying the RPIUrgency and RPICoherence, according to equation (3). The RPI4DD results indicate that the risk of documentation debt at $F_{Restart}$ (2500) is higher than at $F_{Stamp}$ (50). Thus, document improvements or actions to prevent damage from documentation TD related to $F_{Restart}$ should be planned for corresponding module developers or application engineers. When transferring the results received for the xPPU to real-world production systems, companies may benefit by RPI4DD as a quantitative indicator for different stakeholders (e.g., from management or software development) to prioritize starting points for reducing the amount of documentation debt and, thus, avoid high long-term cost.

The applicability of the approach is evaluated on a lab-sized demonstrator. Based on the risk priorities obtained, follow-up documentation activities can be determined. No action is required in case the risk priority is low, e.g., $F_{Stamp}$ in the proof of concept presented in section 4.4. If the risk is high, e.g., $F_{Restart}$, the
factors contributing most to the risk must be analysed to plan for additional documentation tasks, e.g. to review and note changes of F_Restart at commissioning since F_Restart would mostly need to be integrated on-site by PM – as aforementioned. If the outcome is a medium level of risk, it is necessary to check if current documentation is sufficient for staff working with the control code.

Third, we make the following adaptations in section 5 since the evaluation using the selected lab-sized demonstrator represents a first proof of concept, which needs to be validated and substantiated by further analysis in future work. The adaptations discuss limitation of the work, expectation and direction for improvement of the proposed solution.

The presented factors covering an initial basis of influences need to be adapted or enlarged (e.g. quality of code comments is not yet covered). The evaluated use case is limited to the scope of a lab-sized demonstrator consisting of general functionalities. There may be unexpected company-specific software structure and documentation. More concrete factors might be required since the code size of industrial applications may vary from a hundred to several thousand lines of code per POU [12]. In future work, it is therefore planned to evaluate RPI4DD in industrial settings to enhance the factor hierarchy and calculation rules. First, the workflow with involved stakeholders and communication interfaces between departments would need a study to collect the required information for RPI4DD calculation in practice. Second, control code documentation refers not only to code comments but also to manuals or architect documents; thus, these documentation artefacts should be included in future studies to further develop the factor classification for the extensible RPI4DD. Furthermore, new tools can be integrated or designed to enable automatic assessments for the identified factors to promote the RPI4DD concept to a framework applicable to industrial development workflows.
## Review #1

<table>
<thead>
<tr>
<th>Remark</th>
<th>Response / Changes</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Overall a very-good paper that addresses current industry issues of documentation lacks and suggests an appropriate method for an identifying software for potential documentation gaps. The paper requires mostly only small improvements.</td>
<td>Thank you very much for your positive feedback.</td>
<td></td>
</tr>
<tr>
<td>At the end, I miss a somewhat more mature discussion of the results, including their critical evaluation.</td>
<td>We make the following adaptions in section 5 (Conclusion and outlook) to improve the discussion of the results and evaluation. The adaptions describe impact, benefit, expectation and direction for an improvement of the proposed solution.</td>
<td>5</td>
</tr>
</tbody>
</table>

 […] To address this issue, a Risk Prioritisation Indicator of Documentation Debt (RPI4DD) is proposed to quantify the risk of documentation debt in control software. Related work addressing the quality of control code documentation is still scarce and primarily considers source code alone. To the best of the authors’ knowledge, this is the first study to transfer the Risk Priority concept in the GAMP method to automation software and its boundary conditions. The RPI4DD approach considers not only internal software factors (e.g. complexity or functionalities) but also includes external influencing factors, such as required on-site changes or tests. Therefore, compared to existing work, the proposed approach could provide a broader view on control code documentation since aPS is developed in a multidisciplinary environment. Besides documentation, the proposed approach could be generally applied to other aspects of software quality (e.g., testing) or could be further applied to other disciplines. […] The presented factors covering an initial basis of influences need to be adapted or enlarged (e.g. quality of code comments is not yet covered). The evaluated use case is limited to the scope of a lab-sized demonstrator consisting of general functionalities. There may be unexpected company-specific software structure and documentation. More concrete factors might be required since the code size of industrial applications may vary from a hundred to several thousand lines of code per POU [12]. In future work, it is therefore planned to evaluate RPI4DD in industrial settings to enhance the factor hierarchy and calculation rules. First, the workflow with involved stakeholders and communication interfaces between departments would need a study to collect the required information for RPI4DD calculation in practice. Second, control code documentation refers not only to code comments but also to manuals or architect documents; thus, these documentation artefacts should be included in future studies to further develop the factor classification for the extensible RPI4DD. Furthermore, new tools can be integrated or designed to enable automatic assessments for the identified factors to promote the RPI4DD concept to a framework applicable to industrial development workflows. |

| Suggestions for improvement of the manuscript are included in the pdf file. These are considered as minor changes since to the opinion of the reviewer, no significant rework of the text is needed. | Thank you very much for the detailed feedback in the pdf. In the following, the comments in the pdf file are addressed. | |
What was the challenge in solving? What is new in the proposed solution?

1. To highlight the challenge in solving, we adapt the abstract as follows.

   [...] companies still face major challenges in documentation [...] insufficient documentation creates friction since it may increase the risk of malfunction and high costs, and impede system development due to lack of traceability, especially for control software being one of the main functionality carriers.

   And the proposed solution is described as follows.

   [...] to systematically capture the lack of control code documentation to avoid undesired costs due to inadequate documentation.

2. Due to the abstract length restriction (max. 600 characters), we adapt in section 1 (Introduction and motivation) to describe the challenge in solving and the work contribution as follows.

   A motivational example is introduced [...] Motivated by the example above, a question arises: How do different factors influence control code documentation? [...] The main contribution of this work is to address selected aspects of (semi-)automatically identifying a lack of control code documentation. [...]

3. Also, we adapt section 5 (Conclusion and outlook) to highlight the contribution and benefit of the approach:

   To the best of the authors’ knowledge, this is the first study to transfer the Risk Priority concept in the GAMP method to automation software and its boundary conditions. The RPI4DD approach considers not only internal software factors (e.g. complexity or functionalities) but also includes external influencing factors, such as required on-site changes or tests. Therefore, compared to existing work, the proposed approach could provide a broader view on control code documentation since aPS is developed in a multidisciplinary environment.

Source? Why the problem of low-skilled workers in lower-wage countries is more essential then the same problem in high-wage countries? Removement, reformulating or adding a relevant Source needed.

A new source [3] is added as follows.

1. Section 1 (Introduction and motivation):

   German manufacturers in this area, who used to be world-leading exporters, are confronted with growing competition worldwide and thus are struggling to stay competitive, particularly regarding work costs. Some MM and PM companies have to consider relocating engineering divisions to lower-wage countries with less experienced/qualified engineers who need more mature documentation [3].

   [...] necessary documentation artefacts (e.g., architecture documents or code comments) are required to ease the readability of complex software. The impact of insufficient documentation might be significant, especially among low-skilled engineers in lower-wage countries where the engineering departments are relocated [3].

2. References:

| Transition between paragraphs is not smooth enough. Please add short description of scientific question, which should be solved. | A short description of the scientific question is added at the beginning of the mentioned paragraph to improve the connection to previous paragraph describing the motivation example.  

[...] A motivational example is introduced in the following:

"[...] As the commissioner works as fast as possible to start up the [systems], she/he does not document the changes made in the software resulting in many software versions in the field." [6] Motivated by the example above, a question arises: How do different factors influence control code documentation? Since the development characteristics of MM and PM are typically distinct, they often follow different engineering practices. Previous studies indicated that software analysis alone is insufficient to measure software quality (e.g., maturity level [7]) or support software evolution since automation software is part of a mechatronics system involving multiple disciplines [8]. Instead, the development process characteristics must be considered. Therefore, a web questionnaire is developed to study software maturity, complexity, and documentation in MM and PM [7]. | 1 |

| Could you add at least one more source for the statement? | The source [8] is added to the statement.  

Previous studies indicated that software analysis alone is insufficient to measure software quality (e.g., maturity level [7]) or support software evolution since automation software is part of a mechatronics system involving multiple disciplines [8]. | 1 |

| If I understand it right, this work is based on [6]. If yes, could you properly reflect it in the text? | Additional text is added as suggested.  

Therefore, a web questionnaire is developed to study software maturity, complexity, and documentation in MM and PM [7]. The main contribution of this work is to address selected aspects of (semi-)automatically identifying a lack of control code documentation. First, based on the results of [7], a study is conducted to analyse the documentation aspect from expert responses to the questionnaire, especially regarding engineering practices or the automatic generation of information in MM and PM. | 1 |

| Could you shortly describe, what these guidelines are and for which purposes they will be used? | A brief description of PLCopen or MISRA is provided as follows.  

[...] IEC 61131-3 compliant languages include three graphical languages (i.e., Ladder Diagram LD, Function Block Diagram FBD, and Sequential Function Chart SFC) [...] PLCopen is a worldwide initiative of different platform suppliers to reduce engineering effort and enhance software quality by providing standards, guidelines, and education for platform users in the community of industrial automation. The PLCopen guidelines, e.g. naming conventions or structuring with SFC, support in ensuring, e.g., consistency of automation software. MISRA provides rules and directives for coding and documentation that improve software maintainability. The MISRA guidelines, e.g. using start and end comment markers, support in, e.g. portability of automation software. | 2 |

| Some illustration explaining the process would be useful here. | First, the factor hierarchy of GAMP risk priority in Fig. 2 old Fig. 4 new (factor hierarchy) is extracted as Fig. 1 new to illustrate the process in section 2 (Background). | 2 |
Second, additional text is added to section 2 (Background) to refer to the new figure.

Risk Priority in the GAMP method (i.e., criticality estimation in risk analysis) is based on three factors Severity, Probability, and Detectability (cf. Fig. 1).

More detailed description of table required e.g.: Evaluation of expert surveys on document accessibility across disciplines

Thank you very much for the hint. The table description is revised accordingly.

Table 1: Evaluation of expert surveys on document accessibility across disciplines (#72 [24])

More detailed description of table required

The table description is revised.

Table 2: Evaluation of expert surveys on automatically generating engineering artefacts across disciplines (#51 and #52 [24])

More detailed description of table required

The table description is revised.

Table 3: Evaluation of expert surveys on detail levels of in-house guidelines across disciplines (#77 [24])

Precise the column name

The column name is change from "Evaluation" to "Detail level of internal guidelines".

Could you add a table for this too?

Additional text is added to describe expert responses on usage of PLCopen and company-specific guidelines.

Question #34 examined the usage of PLCopen and company-specific guidelines. The responses of #34 show a low frequency of usage of PLCopen guidelines (10%) compared to in-house guidelines (cf. Table 4). The internal guidelines are employed by a large portion of the surveyed companies (80%); thus, the usage of guidelines from the community is less than company-specific guidelines. Furthermore, the 10% remaining responses reported that no guideline is being followed.

A new table (Table 4: Evaluation of expert surveys on usage of PLCopen and company-specific guidelines in software discipline (#34 [24])) is added as follows.

Table 4: Evaluation of expert surveys on usage of PLCopen and company-specific guidelines in software discipline (#34 [24])

<table>
<thead>
<tr>
<th>Types of guidelines in use</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>PLCopen guidelines</td>
<td>10%</td>
</tr>
<tr>
<td>in-house guidelines</td>
<td>80%</td>
</tr>
<tr>
<td>none</td>
<td>10%</td>
</tr>
</tbody>
</table>

Fig. 1: Factor hierarchy in the GAMP method [10]
This section first derives requirements of indicators for documentation debt […]

4.1 Derived requirements of indicators for documentation debt
From the state of the art, requirements of indicators for documentation debt are derived in the following.

weight names and values 2 and 3 irritate the reader: medium weight = 2 = height weight = 3

Thanks for your advice. It is revised (high weight should be 3).

– Organizational (low weight = 1)
– Messages (low weight = 1)
– Status (low weight = 1)
– Error / generic / homing (medium weight = 2)
– Motion Control (high weight = 3)
– Safety-related (high weight = 3)

source or explanation needed

An explanation is added.

In practice, 80% coverage is generally accepted as good (following the Pareto principle in testing: given that 80% of code is covered, the remaining 20% may require a significant effort but is not worth it).

Is there any reason why 20% were chosen? moderate if 20% \leq \text{TestDepth, BranchCoverage} < 80%

The 20% is based on our experience working with an industrial partner. It should be noted that expected coverage values depend on various aspects, such as software modularity or reusability. Thus, we add additional text to describe some special boundary conditions at the industrial partner.

To distinguish between a poor and a moderate test quality, additional information needs to be taken into account. More precisely, the expectation for test quality depends on various company-specific boundary conditions, such as software modularity or reusability. For instance, the boundary conditions at the industrial partner in the previous study [25] are as follows: the company has a high-modularised structure of its control code and a high degree of reuse by applying library modules and templates (about 75% of a machine's PLC project consist of reused control code). The other 25% of the machine's PLC project is newly developed machine- or customer-specific control code, e.g. data exchange between the reused library modules or implementation of the machine-specific process logic. These newly developed, machine-specific parts are the error-prone parts of the PLC project, which require thorough testing. Generally, the company's engineers know which requirements target these new code parts, and due to the company's mature code structure, they can locate the code parts in the PLC project. Consequently, the defined tests usually aim at targeting specifically the requirements linked to these critical code parts. Again, following the Pareto principle, at least 80% of the 25% new machine-specific code needs to be covered by tests. Since the 80% of the 25% new machine-specific code is about 20% of the total code, the company-specific threshold for moderate test quality is set to 20% of the entire control code in the considered PLC project. The threshold of 20% for "moderate" is set given the assumption that test cases for the machine-specific part are selected and that the remaining code is already well-tested and thus assumed to be less error-prone. Therefore, a proposal to assess the test quality with two metrics, TestDepth and BranchCoverage, is shown in equation (6).

More detailed table description needed

More details are added to the table description.
Table 5: Rating scale for Comment Compliance with guidelines or standards

<table>
<thead>
<tr>
<th>Short exact explanation of metrics would be helpful for some readers</th>
<th>An explanation of metrics is added when the metrics are firstly introduced in section 2.4 (Background - Research gap)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Neumann et al. [23] proposed some simple metrics such as HeaderCommentLinesOfCode (size of comment header), MultiLineComments (amount of comments wrapping over multiple lines), and SourceCodeCommentedRatio (density of comments) to assess the software maintainability.</td>
<td></td>
</tr>
<tr>
<td>Table 6: Rating scale for Change Type based on scope and criticality</td>
<td></td>
</tr>
<tr>
<td>More detailed table description needed</td>
<td>More details are added to the table description.</td>
</tr>
<tr>
<td>typo</td>
<td>The typo is fixed.</td>
</tr>
<tr>
<td>Regarding Comment Quality, as aforementioned, F_Restart scope is larger and relates to plant manufacturing</td>
<td></td>
</tr>
<tr>
<td>what does it mean for the company?</td>
<td>Additional text is added.</td>
</tr>
<tr>
<td>Altogether, the RPI4DD is determined by multiplying the RPI_Urgency and RPI_Coherence, according to equation (3). The RPI4DD results indicate that the risk of documentation debt at F_Restart (2500) is higher than at F_Stamp (50). Thus, document improvements or actions to prevent damage from documentation TD related to F_Restart should be planned for corresponding module developers or application engineers. When transferring the results received for the xPPU to real-world production systems, companies may benefit by RPI4DD as a quantitative indicator for different stakeholders (e.g., from management or software development) to prioritize starting points for reducing the amount of documentation debt and, thus, avoid high long-term cost.</td>
<td></td>
</tr>
<tr>
<td>More discussion about the results needed. What is New in the approach?</td>
<td>Additional text is added.</td>
</tr>
<tr>
<td>To address this issue, a Risk Prioritisation Indicator of Documentation Debt (RPI4DD) is proposed to quantify the risk of documentation debt in control software. Related work addressing the quality of control code documentation is still scarce and primarily considers source code alone. To the best of the authors' knowledge, this is the first study to transfer the Risk Priority concept in the GAMP method to automation software and its boundary conditions. The RPI4DD approach considers not only internal software factors (e.g. complexity or functionalities) but also includes external influencing factors, such as required on-site changes or tests. Therefore, compared to existing work, the proposed approach could provide a broader view on control code documentation since aPS is developed in a multidisciplinary environment. Besides documentation, the proposed approach could be generally applied to other aspects of software quality (e.g., testing) or could be further applied to other disciplines.</td>
<td></td>
</tr>
<tr>
<td>What are weaknesses of the method? What was not as expected in the Solution?</td>
<td>Additional text is added to discuss limitation of the work, scope and expectation.</td>
</tr>
<tr>
<td>The presented factors covering an initial basis of influences need to be adapted or enlarged (e.g. quality of code comments is not yet covered). The evaluated use case is limited to the scope of a lab-sized demonstrator consisting of general functionalities. There may be unexpected company-specific software structure and documentation. More concrete factors might be required since the code size of industrial applications may vary from a hundred to several thousand lines of code per POU [12]. In future work, it is...</td>
<td></td>
</tr>
</tbody>
</table>
therefore planned to evaluate RPI4DD in industrial settings to enhance the factor hierarchy and calculation rules.
The paper addresses an important issue regarding the development of automation software. Automatic identification of documentation quality is an interesting topic.

Nonetheless, the title of the paper directs the reader into the pure identification of a lack, whereas the main topic is the calculation of the numerical risk index.

Due to the title length is restricted to 99 characters, we would suggest revising "identification" to "assessment" in the title and adding a subtitle ("A risk-based approach to indicate documentation debt") describing the main topic "calculation of the numerical risk index" as follows.

Semi-automatic assessment of lack of control code documentation in automated production systems
A risk-based approach to indicate documentation debt

The structure of the paper is not really clear in following the approach. A questionnaire is analyzed, showing different practices regarding documentation. It is not clear, how the detailed results correspond to the following chapters.

To make the paper structure more clear, we add a new figure to illustrate the steps of the study and the findings corresponding to the chapters as follows.

In the text, the structure and approach are described at three main points as follows.

1. The high-level structure of the sections is described in section 1 (Introduction and motivation):

   The remainder of the paper is structured as follows. Section 2 provides the background and related work, followed by an analysis of industrial practice regarding documentation in MM and PM in Section 3. The concept and applicability of a risk-based approach to indicate insufficient documentation are presented in Section 4. Finally, Section 5 concludes the paper and provides an outlook on future work.
2. After the background is introduced in section 2 (Background and related work), the detailed results corresponding to the following sections are described.

An overview of findings from related work and expert survey corresponding to the sections is illustrated in Fig. 2.

3. The connection between section 3 (Questionnaire analysis) and section 4 (Concept of risk indicators) is described at the end of section 3:

In summary, the results show a lack of availability of required documents, low-moderate automatic generation of information in engineering, a lack of exact instructions, and high reliance on in-house guidelines in MM and PM. These findings reveal the potential triggers of documentation debt in aPS. Thus, a methodology to indicate the risk of documentation debt is necessary.

4. At the beginning of section 4 (Concept of risk indicators), the connection is explained further (please see the text below). The expert responses are included in Related sources of Figure 2 (Factor hierarchy) in section 4 and the description of individual factors.

The expert responses presented in Section 3 emphasise that insufficient documentation is still a critical challenge in industrial practice of MM and PM companies. […] The selected factors are based on current literature and expert responses identified in Section 3. The section continues with a description of the proof of concept of RPI4DD, including an evaluation using a lab-sized plant. […]

Figure 2 shows nicely the structure of the factors contributing to the risk factor. The textual description of the factors loses the reader. Thank you very much for the positive feedback on the figure structure.

Additional text is added to support the reader in section 4.3 (Factor classification).

In the following, the specification of sub-factors and corresponding rationale are described in the order presented in Fig. 4.

We addressed this comment by improving the structure of the section and enhance the mapping to the figure (Fig. 2 old - Fig. 4 new). To ease the reading of the factor descriptions, they are now separated in individual sub-sections so that the textual description follows the structure in the figure. By that, we hope to guide the reader in a more understandable way through the description of the factors.

1. **DOCUMENT URGENCY** (RPI\_URGENCY)

   […]

   1.1. **FUNCTIONALITY** (RPI\_FUNCTIONALITY)

   […]

   1.2. **REQUIRED CHANGE ON-SITE** (RPI\_ON-SITE)

   […]

   1.3. **GRADE OF TEST/QUALITY OF TEST** (RPI\_TEST)

   […]

2. **DOCUMENT COHERENCE** (RPI\_COHERENCE)

   […]

The conclusion does not evaluate the obtained numerical results. Overall, there is a lot of detailed information in the paper, which would need more room. Additional texts are added to evaluate the results obtained and their implications as follows. The applicability of the approach is evaluated on a lab-sized demonstrator. Thus, both details on rationales and calculations are provided to prepare for future work in industrial settings.
1. In section 4.4 (Proof of concept):

   Altogether, the RPI4DD is determined by multiplying the RPIUrgency and RPICoherence, according to equation (3). The RPI4DD results indicate that the risk of documentation debt at F_Restart (2500) is higher than at F_Stamp (50). Thus, document improvements or actions to prevent damage from documentation TD related to F_Restart should be planned for corresponding module developers or application engineers. When transferring the results received for the xPPU to real-world production systems, companies may benefit by RPI4DD as a quantitative indicator for different stakeholders (e.g., from management or software development) to prioritize starting points for reducing the amount of documentation debt and, thus, avoid high long-term cost.

2. Additional text is added in section 5 (Conclusion and outlook):

   The applicability of the approach is evaluated on a lab-sized demonstrator. Based on the risk priorities obtained, follow-up documentation activities can be determined. No action is required in case the risk priority is low, e.g., F_Stamp in the proof of concept presented in section 4.4. If the risk is high, e.g. F_Restart, the factors contributing most to the risk must be analysed to plan for additional documentation tasks, e.g. to review and note changes of F_Restart at commissioning since F_Restart would mostly need to be integrated on-site by PM – as aforementioned. If the outcome is a medium level of risk, it is necessary to check if current documentation is sufficient for staff working with the control code.

   The presented applicability of the approach is still limited as it was just evaluated on a lab-size demonstrator. There may be unexpected company-specific software structure and documentation. Thus, the completeness of the factors would require additional research. In future work, it is therefore planned to evaluate RPI4DD in industrial settings to enhance the factor hierarchy and calculation rules.