Software Engineering Risks from Technical Debt in the Representation of Product/ion Knowledge

Stefan Biffl[1], Lukas Kathrein[1,3]
[1]*Inst. for Information Systems Eng.*
Faculty of Informatics
TU Wien, Vienna, Austria
[first].[last]@tuwien.ac.at

Arndt Lüder[2]
[2]*Inst. of Ergonomics*
Manufacturing Sys. and Automation
OvG U. Magdeburg, Germany
arndt.lueder@ovgu.de

K. Meixner[1,3], M. Sabou[1,3],
L. Waltersdorfer[1,3] D. Winkler[1,3]
[3]*Christian Doppler Lab SQI (sqi.at)*
TU Wien, Vienna, Austria
[first].[last]@tuwien.ac.at

***Abstract*—In the multi-disciplinary *production systems engineering* (PSE) process, software engineers depend on requirements and design rationales coming from product and production process planning, summarized as *product/ion* knowledge. Unfortunately, the engineering artifacts coming from product/ion planning often represent important product/ion knowledge incompletely and not well integrated, leading to risks regarding software engineering quality. In this paper, we report on a case study at a large industrial PSE organization, investigating *Technical Debt* (TD) effects, items, and causes in PSE process documentation and configuration management according to the VDI guideline 3695 Part 2. We focus on requirements for and issues in the representation of product/ion knowledge in the engineering data provided to software engineers. Based on data elicited from PSE domain experts, we model TD concepts based on the *Quality Function Deployment* method as foundation for TD analysis and risk management. The initial validation with domain experts revealed how software engineers could benefit from improved product/ion knowledge modeling as foundation for better understanding the rationale of engineering design decisions.**

Keywords—Multi-Disciplinary Production Systems Engineering, Product/ion Knowledge, Product-Process-Resource (PPR) Model, Process Management, Technical Debt

I. INTRODUCTION

In *production systems engineering* (PSE) organizations, many different engineering disciplines work together, such as basic system and process planners, detailed automation engineers and production optimizers, for fulfilling customer requirements towards the *Industrie 4.0* vision [1] regarding, for example, production system throughput and quality. In a typical PSE process, the domain experts work in parallel in discipline-specific workgroups that exchange engineering artifacts for iterative improvement. For making informed design decisions, industrial automation and software engineers depend on the high quality of input artifacts that contain software requirements as well as results and rationale of system design decisions [4][5].

Unfortunately, the quality of *software engineering* (SE) results, such as software code governing the transportation system of a production plant, is subject to risks due to the missing or incorrect representation of product/ion knowledge, i.e., knowledge on characteristics of the product, produced by the plant or characteristics of the industrial production process and their relationships to characteristics of the production system [13].

DOI reference number: 10.18293/SEKE2019-037

An example for such a relation is a *fragile product* that imposes limitations on the maximal acceleration during the transport between production system components. If a software engineer sets the transport speed high to maximize the system throughput, the product quality may suffer, leading to the costly redesign of the overall system. Limited awareness of domain experts on the knowledge requirements of partner roles in the project may lead to insufficient descriptions of relevant engineering data and knowledge. Risks from decisions based on insufficient and often incomplete information or from unplanned effort due to unreliable communication between basic and detailed planners could be better managed with adequate knowledge representations of product/ion knowledge throughout the process. Kathrein *et al.* point in [13] out that e*ngineering organizations* (EOs), as defined in the VDI 3695 [28], tend to focus on discipline-specific outcomes rather than on the collaboration of domain experts. The domain experts suffer from low quality of collaboration artifacts, but do not, in general, have the knowledge or the budget to improve the collaboration process.

In this paper, we investigate *Technical Debt* (TD) in the representation of product/ion knowledge in engineering artifacts exchanged between PSE workgroups as foundation for analyzing and managing risks from TD effects, items, and causes in a PSE organization. An example for TD is a missing or incomplete engineering process description, which makes it hard to manage projects across several domains and work groups. In this paper we adapt the definition of TD by Li *et al.* [16] according to engineering artifacts and the collaboration process: *TD are violations in engineering artifacts compared to best practices of engineering process documentation and configuration for collaborative workgroups in the PSE domain*. Main goal is to identify TD throughout the engineering process, for better PSE process management, in particular, SE risks.

We report on results from a case study at a large industrial PSE organization on TD regarding process documentation according to the PSE domain VDI Guideline 3695 Part 2 [28] concerning the *procedure model for project activities* and *configuration management in an engineering organization*. We focus on eliciting requirements for and in the representation of product/ion knowledge supporting software engineers in their decision-making process. In the case study, we identified TD items, where one TD item is a unit bearing quality risk [16], on insufficient description of engineering process and information in the data exchange process and insufficient representation of product/ion knowledge. Based on collected data samples, we relate TD concepts to each other and investigate their relationships

based on the *Quality Function Deployment (QFD)* [19] method. The *QFD* method allows analyzing and prioritizing customer requirements together with solution options. We use the *QFD* method for analyzing TD repayment options [19], e.g., which TD items should be addressed to reduce system design cost. Better understanding TD relationships is the foundation for advanced analyses of TD risks and TD repayment options.

The remainder of this paper is structured as follows: Section II summarizes related work on PSE, on knowledge representation in PSE, and on concepts of TD in PSE knowledge. Section III introduces the research questions and approach. Section IV reports case study findings regarding TD effects, items, and causes, and relates these TD concepts in a preliminary *QFD* style model. Section V discusses results and limitations of the research. Section VI concludes and provides an outlook on future research.

II. Related Work

This section summarizes related work on *production systems engineering* (PSE), on knowledge representation in PSE, and on concepts of technical debt in PSE knowledge.

A. Production Systems Engineering (PSE)

The engineering of production systems is a multi-disciplinary task involving various disciplines, such as mechanical, electrical, and software engineering [3]. The disciplines conduct a network of engineering activities where engineering decisions are taken and engineering data are created by engineers. The engineers use appropriate input data and engineering tools optimized for the discipline. One role, the automation engineering designs and implements the hardware and software of the production system control, a main software engineering task in PSE [27].

Within PSE, the importance of digital models is increasing. New activities related to the development and use of life cycle crossing digital shadows of complete production systems and their components are envisioned to enable the *Industrie 4.0* vision [17]. These models shall contain all relevant data and knowledge on production systems aspects. This includes the description of the involved production system components, the production processes they execute, and the product resulting from the production process. Schleipen *et al.* [24] calls this *PPR knowledge* and Kathrein *et al.* [13] use the term *product/ion knowledge.* In this paper, we build on the *PPR* concept to identify shortcomings regarding knowledge representation that introduce risks to SE activities.

B. Knowledge Representation in PSE

Engineering knowledge is a specific kind of knowledge, oriented towards the production of artifacts, and, as such, requires knowledge modeling and representation approaches that differ from other types of knowledge, such as taxonomical knowledge characteristic for the life sciences domain [25]. Ontologies are information artefacts that have been used extensively to explicitly represent such engineering knowledge. This is for example investigated by Ekaputra *et al.* [7] highlighting different ontology-based data integration strategies, possible objectives an engineering organization has, as well as data source types used.

Sabou *et al.* [23] provide an overview of such ontologies and classify them in terms of the aspects of the PPR process that they cover. For example, OntoCAPE[1] [20] is an ontology for supporting *computer-aided process engineering* (CAPE) focusing on describing production process information. The *Semantic Sensor Network* (SSN)[2] ontology, developed at W3C[3], is well suited to describe process states and their observability, as well as resource states. The *Automation Ontology* (AO) captures knowledge about industrial plants and their automation systems to support engineering simulation models [21]. AO concerns mechatronic concepts to support simulation model design and integration.

The explicit modeling of PSE knowledge is characterized by the need to address recurring modeling needs specific for this domain, including: Modeling *Part-whole relations.* Legat *et al.* [14] observe that containment hierarchies are a well-accepted and frequently occurring organizational paradigm from modeling part-whole relations in (mechatronic) engineering settings. *Modeling connections between components*. Legat *et al.* [14] observe that *interface-based composition* describes the capabilities expected from an interface to enable reasoning tasks about the correctness of a system's structure.

The modeling of recurring knowledge structures can be well addressed by the reuse of *Ontology Design Patterns* (ODPs) that model best practices applicable to typical conceptualization scenarios [10]. Indeed, ODPs exist to support the conceptual modeling of (variations) of the engineering-specific modeling scenarios mentioned above. For example, modeling *Part-whole* relations can be achieved by reusing the *PartOf* ODP[4], which allows modeling part-whole relations in a transitive fashion. The *Componency* ODP[5] is a specialization of the *PartOf* ODP for modeling *part-whole* relations distinguishing between direct and indirect (i.e., transitively-assessed) parts of an object. While there are several approaches for knowledge representation in PSE they are often not used and lead thus to TD, which is addressed in this paper.

C. Technical Debt in PSE Knowledge

Avgeriou *et al.* [2] compare *Technical Debt* (TD) to friction in mechanical devices, requiring increasingly more energy to achieve the same results as parts deteriorate. This is also true for *software engineering* (SE), as short-term gains create friction over the lifetime of a software-intensive system that require extra effort and cost to address or to repay. To deal with TD, Avgeriou *et al.* [2] propose to analyze TD repayment options and to investigate TD from different viewpoints. TD in a system consists of TD items that are measurable in an SE artifact. Li *et al.* [16] identify ten different TD types, with effects ranging from

[1] OntoCAPE: https://www.avt.rwth-aachen.de/AVT/index.php?id=730
[2] SSN Ontology: www.w3.org/2005/Incubator/ssn/ssnx/ssn.owl
[3] https://www.w3.org/
[4] PartOf ODP: http://ontologydesignpatterns.org/wiki/Submissions:PartOf
[5] Componency ODP: http://ontologydesignpatterns.org/wiki/Submissions:Componency

inconveniences to crippling whole software systems, making future maintenance costly [2]. Martini *et al.* [18] point out that large SE companies invest a quarter of the development time in managing TD to continue providing their SE functions.

Dong and Vogel-Heuser [6] draw a comparison, based on results from two case studies, between TD in PSE and in SE, as similar effects, such as short-term cost savings or lack of experience, occur in both domains. Causes of TD manifest in various dimensions, for example mechanical, electrical, or software engineering [6]. As process improvement and data exchange processes are success-critical processes in *engineering organizations* (EOs), they identify crucial TD in design and architecture, knowledge distribution and documentation [6].

Martini *et al.* [18] show how architectural TD accumulates during development in a project until reaching a crisis point that makes refactoring inevitable, increasing business value as the short-term sins are repaid adequately. Case studies by Biffl *et al.* [4][5] and Kathrein *et al.* [12][13] investigated engineering processes of EOs with a focus on the structure of collaborations between workgroups [13] and how data is exchanged [4][5]. These works represent building blocks for this paper, as they define a coherent context with basic concepts needed for TD investigations. The research highlights multiple use cases with different levels of TD, and points out missing *product/ion-aware* (PPR) knowledge as a limitation.

III. Research Questions and Approach

This section introduces the research questions (RQs) following the *design science* method [29], and presents an illustrating use case to investigate TD in the representation of product/ion knowledge. Similarly, as in [26], we investigate TD as a form of software engineering risks, with effects and possible causes.

RQ1: What risks to software engineering results and activities are related to technical debt in the representation of product/ion knowledge in engineering artifacts exchanged between workgroups in production systems engineering? From the high-level RQ1, we derive the following sub-RQs.

RQ1a: What are effects of TD related to software engineering risks in PSE? We identify TD effects in the PSE process in interviews with domain experts. These TD effects can be defined as process management issues, i.e., deviations from the planned engineering process, and the process executed by individual domain experts. The identification of TD effects allows highlighting risks known to SE, but not to domain experts in PSE.

RQ1b: What are TD items regarding the VDI Guideline 3695 Part 2 in engineering artifacts exchanged between workgroups in PSE? The VDI Guideline 3695 Part 2 [28] provides valuable insights in describing engineering organizations (EOs) and potential improvement steps. The guideline provides a set of best practices that should be followed in an EO and allows analyses similar to code reviews in SE. Therefore, we define TD items in the PSE process by comparing selected target states in the VDI 3695 to the *as-is engineering process*.

RQ1c: What are causes regarding elicited TD items? As foundation for managing TD, we elicit in the case study candidate TD causes in the engineering organization. TD causes strengthen the deviation between the as-is and VDI 3695 defined process and are important to address TD items and SE risks.

RQ2: *How do TD concepts in the data exchange process relate to each other?* After identifying TD concepts, we model their relationships as foundation for analyzing the impact of TD causes on TD items and effects, with a focus on SE concepts.

RQ2a: How do TD effects and TD items relate to each other? Main outcome of this RQ is a table based on the *QFD* method [19], developed with quality managers, who are responsible for defining an ideal PSE process across all involved disciplines and for possible improvement steps. The *QFD* method facilitates prioritizing relationships between TD effects and TD items that are relevant to reduce SE risks.

RQ2b: How do TD items and causes relate to each other? There are many and diverse TD cause candidates that can have different impacts on TD items. This RQ investigates most relevant TD causes to influence the TD items and effects. Main outcome is a table depicting relationships of TD causes and items based on the QFD method [19], created with quality managers.

To answer the RQs, we followed a case study design [22] by adhering to the following case study plan. *[Objective]* Exploring an existing engineering process *[Case]* in a large PSE organization. *[Theory]* Following the design science cycle according to Wieringa [29] in a holistic case study, *[Goal]* we identify common concepts at collaboration interfaces between PSE workgroups, and identify information bottlenecks regarding TD effects. *[Method]* Through seven semi-structured interviews (in a funnel approach) [22], *[Selection]* we elicit representative data from domain experts and investigate TD effects, items, and causes.

According to the design science cycle [29], this paper focuses on workshops and interviews regarding TD effects, causes, items, and the types and strengths of the relationships between the TD concepts. We discuss likely causes for the TD items found. Based on Matook and Indulska [19], we adapt the QFD method to focus in this paper on two dimensions of the QFD *House of Quality* (see Section V). In cooperation with quality managers, responsible for improving the PSE process, we design tables based on the QFD method [19] for investigating TD cause candidates. Finally, we present a conceptual evaluation, discussing presented repayment options to address SE-relevant TD in the multi-disciplinary engineering process.

Kathrein *et al.* [12][13] elicited the illustrating use case in Fig. 1 for *data exchange in the PSE process*. In this paper, the use case frames the election of TD concepts in the case study. In the beginning, the *system planner* (SP) receives product specifications from the customer (1) and aims at providing a competitive offer and at deriving specific knowledge on the production system for later use. This process is similar to software architecture design [14]. Output of this step (lilac arrow) are resource documents regarding the plant layout, calculations, and assembly sequences, delivered to the *process planner* (PP) (2).

Upon receiving the artifacts, the *PP* investigates these artifacts with a common schema that domain experts have developed over decades. For example, the first column always is the module identifier followed by the module name and a reference to an existing CAD drawing if possible. Main goal is to derive

basic variations of the previously offered production system for detailing mechanical aspects. However, if important product aspects and design decisions are not documented, the *PP* has to call back the *SP*, e.g., via e-mail or telephone (3). Final output of this step are detailed descriptions of the production system as foundation for production optimization and PLC *software engineering* in the form of automation tasks (4).

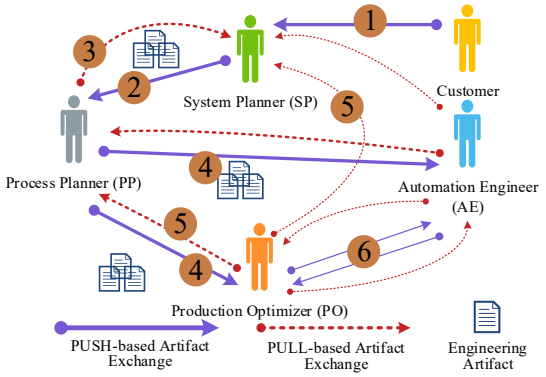

Figure 1. Use case depicting the AS-IS data exchange process.

The *production optimizer (PO)* receives all basic plans and tries to minimize the cycle times of the plant. However, this work requires different product/ion knowledge aspects and may cause many calls back to the *SP* and *PP* (5). The *PO* collaborates with the *automation engineer* (*AE*) (6), who is responsible for PLC *software engineering* tasks. From basic plans (4), the *AE* derives specific PLC software code. Goal of the *AE* is to trace design decisions as foundation for making informed design variations, such as the parameterization of the software and systems that execute production processes, e.g., the speed and acceleration of transport processes. In the next section we investigate this case study regarding TD effects, items and causes.

IV. Case Study Results and Technical Debt Model

This section reports on findings from the case study regarding TD effects, items, and causes, and relates these TD concepts in a preliminary QFD model for the case study context.

A. Case Study results on TD effects (RQ1a)

TD effects. Regarding the use case, data exchange process, the following TD effects came up frequently in workshop sessions.

TD-E1 High effort for tracing design decisions. High unplanned effort in SE activities to collect information on the rationale of design decisions to sufficiently understand what changes in the system design make sense in the production process context.

TD-E2 Data quality risks in engineering artifacts. Low quality of engineering artifacts may limit the production system capabilities and reduce reuse opportunities of system components.

TD-E3 Risk of economic project failure due to cost for unplanned effort for collecting information and due to risk of limited production system quality and capabilities.

B. Case Study results on TD items (RQ1b)

The TD item description contains the following sections: *name and acronym* of the TD item; *motivation* of the typical context and short-term benefits of the TD item; *definition* of the TD item as a violation of the VDI Guideline 3695 Part 2 [28], (see Section II.A); *measurement* definition on the presence of the TD item; r*elationships to effects* including long-term impact from the presence of the TD item; and hypothetical *relationships to causes*, including technical decisions or postponed best-practice activities. Based on the TD item description, we identified the following TD items.

Engineering process description insufficient (TD1Proc) *Motivation.* The requirements for the engineering process depend on the project and on the specific engineers conducting the engineering tasks. Therefore, engineering process models may exist on an abstract level, but do not cover engineering information exchange in sufficient detail. The domain experts focus on engineering production systems and rather than on formally defining the engineering process in detail, with the short-term benefit of starting quickly, following a method they prefer to use. The engineering process models are not maintained and often diverge from actual project practice. Similar TD concepts in SE are missing documentation of application program interfaces (APIs) and software engineering processes in general.

Definition. The VDI Guideline 3695 Part 2 [28], procedure model for project activities, defines the target state A as "*the staff knows the procedure model and can explain how they use it in the project.*" However, in the case study context, the domain experts referred to only experience-based informal processes, with limited awareness of the impact of actions in the process, in particular data exchange with process partners, beyond the immediate workgroup of the domain expert, such as extra effort and risk of engineering data consumers.

Measurement. There is no formal description of discipline-specific engineering process steps and of the collaboration processes between the disciplines involved in the engineering process. The staff does not know about their engineering process description including the impact of exchanged information and the required data maturity.

Relationships to effects. TD-E1, TD-E3 (see Table I in Section IV.C). TD symptoms include high effort for communication and rework due to shortcomings in data exchange, in particular for SE, as the SE activities depend in inputs from several disciplines that may be incomplete or even contradicting (see Fig. 1).

Relationships to causes. See Table II, in Section IV.C. In the case study, main causes came from insufficient means that hinder the description of the engineering process.

Information description insufficient (TD2Inf) (in exchanged engineering data)**.** *Motivation.* In the case study context, the domain experts focus only on the data relevant to their own discipline and do not consider dependencies to related disciplines. They often use tool-specific data exports, such as component lists or CAD drawings, and Excel as a general-purpose information exchange artifact. Short-term benefits include saving effort for the data provider and flexible choice of means for the provider when collecting the engineering data to exchange.

Similar TD concepts in SE are missing documentation of interfaces, code, and implementation details.

Definition. The VDI Guideline 3695 Part 2 [28], configuration management, defines the target state A as *"... there are discipline-specific procedures for configuration identification, configuration monitoring, [...] Within a discipline, all employees follow common guidelines.*" However, in the case study, domain experts found artifacts not to be managed, but simply to evolve over time according to engineering personnel experience, without specific consideration for dependencies between engineering artifacts in different disciplines, which poses risks for SE activities that depend on consistent and complete inputs (see Fig. 1).

Measurement. There is no formal description of engineering artifacts and data, including dependencies between engineering disciplines, such as product, process, and system design. There is no configuration history for backtracking design decisions.

Relationships to effects. See Table I, in Section IV.C. TD symptoms include high effort and risk for propagating changes to systems design across disciplines, in particular for SE, when receiving inputs from several engineering disciplines.

Relationships to causes. See Table II, in Section IV.C.

Product/ion (PPR) knowledge representation insufficient (TD3PPR) (in exchanged engineering data)**.** *Motivation.* Domain experts in production process design have product/ion (PPR) knowledge that would be, in many cases, important to engineers in later stages of PSE and optimization, in particular, for SE activities. However, the process designer tends to provide her engineering partners with hard-coded production system parameters rather than PPR knowledge as there is no dedicated tool or modeling language to allow the effective and efficient representation of PPR knowledge. Short-term benefit for the process designer is saving effort for modeling the PPR knowledge. Similar TD concept in SE would be missing information on non-functional requirements for a software system.

Definition. The VDI Guideline 3695 Part 2 [28], configuration management, defines the target state D as "*system-assisted cross-discipline*" configuration management to enable "*consistency check [...] at an early stage*". However, without sufficient PPR knowledge representation, consistency checks between production process design and production system design are difficult, error-prone, and take considerable expert effort.

Measurement. The engineering data model misses representations for expressing PPR knowledge and rationale to trace design decisions, such as production system temperature settings to the welding temperature and force of a metal joining process.

Relationships to effects. See Table I, in Section IV.C. TD symptoms include in SE activities considerable costs of errors from changes and effort for preventing defects after changes.

Relationships to causes. See Table II, in Section IV.C. In the case study context, main cause candidates include workgroup-specific optimization of the engineering organization and insufficient means to express PPR knowledge.

C. Case Study results on TD cause candidates (RQ1c)

Cause candidates linked to context in the engineering organization, often for economic and historic reasons in the EO.

A1. Workgroup-related profit centers lead to the local optimization of workgroups with limited concerns for the optimization of projects across workgroups, often at the expense of SE.

A2. Engineering habit trained by discipline-specific education leads to engineers focusing on good results in their workgroup. Engineers are, in general, not aware about work tasks, dependencies, and problems in other workgroups, unless a partner asks them for an improvement.

A3. Unclear responsibilities of domain experts in data exchange process lead to ad-hoc procedures and data definitions.

A4. Limited collaboration effort across work groups without a dedicated role for coordinating the work across workgroups.

Cause candidates from engineering process description

B1. Engineering process modelled as an artifact-based workflow, not as a data-related workflow makes it hard to describe dependencies between SE and other disciplines, such as consistency rules that relate to the data model, not to artifacts.

B2. Engineering process defined, but not useful. There is a workflow definition for a process. However, the definition may be abstract and lack important description of content dependencies, such as relationships between the product and resource design, tainting the usefulness of the definition.

B3. Engineering process defined, but not operational. There is a process description. However, missing technical foundations, such as adequate process description concepts or tool support, make it hard or impossible to conduct the process.

B4. Engineering process defined, but not known to stakeholders. There is a process description somewhere in a manual. However, the relevant actors in the project are not aware of the process description for their daily work.

Cause candidates linked to information description

C1. Description of complex dependencies required due to a large number of disciplines (often 15 or more) in a PSE project. Complex descriptions of processes and artifacts and their dependencies in an *engineering organization* (EO) lead to a very complex network (consider Fig. 1, scaled up).

C2. Industry-dependent information description. The description of information depends on the industry and has to be adapted accordingly. There is no general standard that could be applied directly. There is no general EO model as the industry domains require a variety of EO structures and behavior

C3. Tool-driven process without product/ion (PPR) information description. Often, the process is defined based on a specific tool chain. Therefore, the functional and data export capabilities of the tool determine the exchanged information. The process is not aware of PPR as the discipline-specific tools only know the PPR knowledge that is relevant within the discipline.

D. TD effect and item relationships (RQ2a)

Following an adaptation of the *QFD* method according to Matook and Indulska [19], we create a *House of Quality* (*HoQ*). Our *HoQ* provides insights into the relationships between TD effects and items horizontally (representing customer requirements in the original *HoQ*) and TD items and causes in the vertical axis (representing engineering requirements). For these two *QFD* dimensions, we design two tables expressing likely correlations and relationships. We elicited and aggregated likely relationships from a workshop with domain experts in the exploratory case study context [12][13]. As relationship types differ, we indicate the following types and strengths. *DS* indicates a *direct* and *strong* relationship (the stronger the item, the stronger the effect). *DW* indicates a *direct weak* relationship (a stronger item correlates moderately to a stronger effect), and *IW* expresses an *indirect weak* relationship (stronger cause leads to a lower TD item). *No* indicates that the TD item is not related to an effect, such as *(TD1proc) -> (II. Data Quality Risk).*

RQ2a. Table I presents the relationships between TD effects (see Section IV.A) and TD items (see Section IV.B), similar to the *HoQ* analysis [19] matrix, TD effects horizontally and TD items in the vertical axis. The relationships are of the form TD item relates to TD effect, *(TD item) -> (TD effect)*, expressing how a TD item relates to an effect.

Table I. Relationships between TD effects and TD items.

TD Effect/ TD Item	**TD-E1 High Effort**	**TD-E2 Data Quality Risk**	**TD-E3 Economic Failure**
TD1Proc	**DS**	No	**DS**
TD2Inf	**DS**	**DS**	DW
TD3PPR	**DS**	DW	DW

Legend: Relationships: DS: direct strong; DW: direct weak.

In Table I, all three TD items, relate strongly to the TD effect *TD-E1 High Effort* for SE. This is due to unclear descriptions of the process and information as well as missing product/ion knowledge, which all lead to high effort for tracing design decisions. An insufficient information description relates strongly to high risks in data quality, as artifacts are not built on common concepts or data models and thus lack any formal description. Missing product/ion knowledge is also related to the second TD effect *(TD-E2)*, however not so strongly. All three TD items have a relation to the *TD-E3 Economic Failure*, as missing information in the engineering process leads to high rework and communication overheads.

E. TD item and cause relationships (RQ2b)

Table II represents the relationship between TD items (see Section IV.B) and TD cause candidates (see Section IV.C), (TD cause) -> (TD item). Cause candidates coming from the context of the EO (A1 – A4), and from the engineering process description (B1 – B4) have a strong direct relationship to the engineering process description *(TD1Proc)*. For example, unclear relationships and descriptions which are not useful, make it very hard to describe the engineering process sufficiently to facilitate collaboration and coordination across multiple workgroups.

All three cause groups A, B, and C relate to the insufficient description of the engineering data exchange model *(TD2Inf)*. Note the *inverse relationships* of a stronger focus on engineering habits (intra process improvements) and descriptions of the engineering process as artifacts. This does not directly impact the TD item.

Table II. Relationships between TD items and TD causes.

TD Item -> TD Cause (see Sect. IV.C)	**TD1 Proc**	**TD2 Inf**	**TD3 PPR**
A1.Profit Center	DW	**DS**	**DS**
A2.Engineering Habits	DW	IW	**DS**
A3.Unclear responsibility	**DS**	DW	**DS**
A4.Limited collaboration	DW	**DS**	**DS**
B1.Eng. Proc. descr. as artifact	No	IW	**DS**
B2. Eng. Proc. descr. not useful	**DS**	DW	No
B3. Eng. Proc. not operational	**DS**	DW	No
B4. Eng. Proc. unknown	**DS**	No	No
C1.Inform. description. complex	**DS**	DW	No
C2. Inf. desc. Industry. depend.	No	**DS**	DW
C3.Tools w/o PPR	No	DW	**DS**

Legend: DS: direct strong; DW: direct weak; IW: indirect weak.

Insufficient descriptions of the engineering process make it impossible to successfully represent product/ion-aware knowledge *(causes Ax) -> (TD3PPR)*. Causes regarding the information and data exchange description do not impact product/ion knowledge representations, as a major precondition for knowledge representation is the clarification of (a) the responsibility for each part of product/ion knowledge and (b) a suitable represented approach throughout an engineering process.

F. Preliminary validation in the exploratory case study

Throughout the domain expert interviews, we collected representative data samples from engineering artifacts. We derived tables I and II from analyzing these artifacts. As the tables present vital pieces of information regarding possible correlations, we initially elicited the relationships from domain experts. We discussed the relationship candidates in detail with quality managers, who are responsible for improving the engineering process and are knowledgeable in the overall process and work group habits, including SE. We resolved divergences between the views of the domain experts and the quality managers in a common discussion. Overall, the domain experts and quality managers found the preliminary TD concepts and analysis method useful and usable for identifying and addressing high-priority TD effects, items, and causes regarding SE activities.

V. Discussion

This paper investigates risks for *software engineering* (SE) in activities related to the engineering process in a PSE organization (*RQ1*) and possible relationship between certain risks (*RQ2*). In this context, risks are TD effects for SE that occur in a PSE context with measurable probability and costs. To deal with these risks, da Luz *et al.* [26] presented a management tool for analyzing causes and effects. Similar to our work, Luz *et al. [26]* propose an approach to identify risks through selection, description and analyzation. However, the presented approach generalizes risks in a late phase, whereas we focus on organization specific TD effects and investigate these. In the exploratory case study context, SE activities depend on the requirements and design rationale from early engineering phases and often have to deal with locally optimizing workgroups, low awareness on collaboration processes, and missing understanding of requirements between work groups.

For software engineers, the *high effort* comes from risks regarding rework efforts due to frequent and late changes coming from earlier phases. As software engineers highly depend on weakly documented design decisions from early phases, a repayment option for TD is better knowledge representation of product/ion knowledge throughout the engineering process. The *low data exchange quality* impacts software engineers, who are made responsible for low quality system output, even if they write high quality code, but based on weakly communicated early design decisions. TD repayment for reducing the SE risk should focus on improving the documentation and communication of design decisions that are directly related to high-quality SE results. Finally, issues regarding unplanned efforts for reworks in the software design due to low system quality decisions may exceed the budget available to SE, leading to local *economic failure.*

The VDI Guideline 3695 Part 2 [28] was used to investigate TD items (*RQ1b*). This guideline can be seen similar to best practices for SE code development, and our analysis is equivalent to a code review. An *unclear engineering process description* makes it hard for software engineers to reliably configure production systems, as input from several disciplines may be contradicting. The missing collaboration in PSE forces software engineers to take the risky decision on which inputs to consider or ignore. Further, the *information description is not sufficient*. This makes it unclear for software engineers where to look for reliable information, as data syntax and semantics may change frequently, making it hard to validate input data and to automate the data exchange process. Software engineers thus often work based on risky assumptions. Finally, *missing product/ion aware knowledge* makes it hard in SE to take informed decisions for adapting the software system design if the preferred system design option is not feasible.

In *RQ1c*, we investigated possible causes regarding elicited TD items. Causes linked to the *context of production systems engineering* cannot be directly influenced by SE actors, but require the insight of PSE managers. We discussed the preliminary results at the case study EO with quality managers, who found the analysis useful for considering and prioritizing improvement options. Cause candidates linked to the *engineering process description* clearly motivate the need for better means of PPR knowledge representation as a foundation for process descriptions, considering conceptual, language, and usability aspects. This is similar to the need for proper software architecture descriptions identified by Guessi *et al.* [11]. The last group we identified are cause candidates linked to the *information description*. For example, it is challenging to combine methods for data integration [7] with domain-specific standards, such as *AutomationML* [7] or ontologies [23].

A *repayment option* to address TD items related to weak collaboration of workgroups is a new role, the *data curator*, similarly as presented in [9]. This role would be responsible for consolidating a common data exchange model and describe the engineering process adequately. This new role should needs to understand the requirements and limitations of the involved workgroups, in particular SE activities. As TD effects, items and causes are related to each other, we used the *Quality Function Deployment* [19] method to investigate relationships between TD effects, items and causes (*RQ2*).

In *RQ2a* we investigated how TD effects and TD items relate to each other. The *TD-E1 High Effort* is strongly and directly connected to all three TD items. This is obvious as reworks are often needed to compensate missing descriptions or information bottlenecks where especially software engineers are affected. Interesting is, that the *TD-E2 Data Quality Risk* is only weakly connected to *PPR* knowledge representation, even though this is an important information exchange concept. For SE this means that design decisions from early phases do not impact the code development so much as the overall information description, this could be for example the selection of an easily changeable component in the user interface.

RQ2b investigated how TD items and causes are related. Nearly all causes regarding the information description strongly impact the process description. This clearly motivates the need for better knowledge representation approaches, as the current engineering process is either not described, or the description is not useful or unknown to domain experts. As there are currently no tools that support the expression of PPR knowledge, software engineers could address this open issue to improve product/ion-aware knowledge representation and further allow a backflow of SE knowledge into early engineering phases as foundation for designing better reusable system parts.

Limitations. The research of this paper followed a case study in an engineering organization. However, the case study focused on only one company, which may not be representative for all EOs in general. While the domain experts in the study were very knowledgeable, their number was limited due to resource limitations of the available experts besides their daily business obligations. We found that the engineering process description may highly depend on the context, domain, and organization, thus future case studies should consider these variation points. While the domain experts found the preliminary list of TD effects, items, and causes, and their relationships useful for reflecting on TD repayment options in the case study context, these results require validation and discussion on comparable context for strengthening the external validity of the results.

VI. Conclusion and Future Work

In engineering organizations, software engineers join the PSE process in a late phase and are concerned with detailing SE aspects of the software-intensive system. However, software engineers often only receive poorly described design decisions in form of engineering artifacts making it hard for them to derive adequate new (software) engineering knowledge or tracing the earlier design decisions. These shortcomings lead to risks regarding the SE quality and impact the project effort negatively, endangering project success. In this paper, we reported on a case study at a large industrial engineering organization with the focus of investigating technical debt (TD) effects, items, and causes as risks for software engineers. Results highlight that TD can slow down engineering organizations, making it hard to manage processes where multiple domains are involved. Main insight for addressing the found challenges is the introduction of a new role, the *data curator*, to facilitate the collaboration across workgroups. The results highlight requirements for the representation of product/ion knowledge in the engineering data provided to software engineers. Engineering data is heterogeneous and there are no guidelines for basic planners, leading to a large

number of individual and local data models, and making the data exchange hard to manage, often resulting in extra effort to maintain high software quality.

The relations between TD effects, items, and causes highlighted the need for better representations for product/ion knowledge as inadequate context and artifact descriptions lead to high efforts, in particular, for software engineers, and might result in economic project failure. The research findings provide domain experts, such as project managers or software engineers with insights into the engineering process. The presented model serves as a foundation for better understanding the rationale of engineering design decisions. This leads to better SE code due to (a) better understanding of design decisions, (b) more explicit representation of system limits that relate to product characteristics, and (c) better knowledge representation for tool support.

Future work. The results of the exploratory case study should be validated with empirical data from comparable engineering companies. We focused in this paper on product/ion-aware exchange of engineering artifacts. Future research should investigate the impact of knowledge representation options on selected SE. Finally, the more comprehensive representation of integrated PSE knowledge requires improved information security. The comprehensive and well-integrated knowledge is a prime target for attackers regarding corporate espionage and regarding the intentional change of artifacts for reducing the quality of the production system or the production process. Thus, future work should investigate security auditing aspects that consider the issues and repayment options identified in this paper.

ACKNOWLEDGMENT

The financial support by the Christian Doppler Research Association, the Austrian Federal Ministry for Digital & Economic Affairs and the National Foundation for Research, Technology and Development is gratefully acknowledged.

REFERENCES

[1] L. Atzori, A. Iera, and G. Morabito, "The internet of things: A survey," Computer networks, vol. 54, no. 15, pp. 2787–2805, 2010.

[2] P. Avgeriou, P. Kruchten, R. L. Nord, I. Ozkaya, and C. Seaman, "Reducing friction in software development," IEEE Software, vol. 33, no. 1, pp. 66–73, 2016.

[3] S. Biffl, A. Lüder, and D. Gerhard, Multi-Disciplinary Engineering for Cyber-Physical Production Systems: Data Models and Software Solutions for Handling Complex Engineering Projects. Springer, 2017.

[4] S. Biffl, A. Lueder, F. Rinker, L. Waltersdorfer, and D. Winkler, "Introducing engineering data logistics for production systems engineering," Technical Report CDL-SQI-2018-10, TU Wien; http://qse.ifs.tuwien.ac.at/wp-content/uploads/CDL-SQI-2018-10.pdf.

[5] S. Biffl, A. Lueder, F. Rinker, L. Waltersdorfer, and D. Winkler, "Efficient Engineering Data Exchange in Multi-Disciplinary Systems Engineering," in Proc. Int. Conf. on Advanced Information Systems Engineering (Caise). IEEE, 2019, in press.

[6] Q. H. Dong and B. Vogel-Heuser, "Cross-disciplinary and cross-life-cycle-phase technical debt in automated production systems: two industrial case studies and a survey," IFAC-PapersOnLine, vol. 51, no. 11, pp. 1192–1199, 2018.

[7] R. Drath, A. Luder, J. Peschke, and L. Hundt, "Automationml the glue for seamless automation engineering," in 2008 IEEE International Conference on Emerging Technologies and Factory Automation. IEEE, pp. 616–623, 2008.

[8] F. J. Ekaputra, M. Sabou, E. Serral, E. Kiesling, and S. Biffl, "Ontology-based data integration in multi-disciplinary engineering environments: A review," Open Journal of Information Systems (OJIS), vol. 4, no. 1, pp. 1–26, 2017.

[9] A. Fay, U. Löwen, A. Schertl, S. Runde, M. Schleipen, and F. El Sakka, "Zusätzliche wertschöpfung mit digitalem modell," atp magazin, vol. 60, no. 06-07, pp. 58–69, 2018.

[10] A. Gangemi and V. Presutti, "Ontology design patterns," in Handbook on ontologies. Springer, pp. 221–243, 2009

[11] M. Guessi, F. Oquendo, and E. Y. Nakagawa, "An approach for capturing and documenting architectural decisions of reference architectures." in Proc. SEKE, pp. 162–167, 2014

[12] L. Kathrein, A. Lueder, K. Meixner, D. Winkler, and S. Biffl, "Process analysis for communicating systems engineering workgroups," Technical Report CDL-SQI-2018-11, TU Wien; http://qse.ifs.tuwien.ac.at/wp-content/uploads/CDL-SQI-2018-11.pdf

[13] L. Kathrein, A. Lüder, K. Meixner, D. Winkler, and S. Biffl "Product/ion-Aware Analysis of Multi-Disciplinary Systems Engineering Processes", presented at 21st Int.l Conf. on Enterprise Information Systems, Heraklion, Greece, May 2019 (in press).

[14] C. Legat, C. Seitz, S. Lamparter, and S. Feldmann, "Semantics to the shop floor: towards ontology modularization and reuse in the automation domain," IFAC Proce. Volumes, vol. 47, no. 3, pp. 3444–3449, 2014.

[15] X. F. Liu, N. Chanda, and E. C. Barnes, "Software architecture rationale capture through intelligent argumentation." in SEKE, pp. 156–161, 2014

[16] Z. Li, P. Avgeriou, and P. Liang, "A systematic mapping study on technical debt and its management," Journal of Systems and Software, vol. 101, pp. 193–220, 2015.

[17] U. Loewen, A. Schertl, S. Runde, M. Schleipen, F. El Sakka, and A. Fay, "Additional value with a digital plant model-new roles over a plant's lifecycle," ATP EDITION, no. 6-7, pp. 58–68, 2018.

[18] A. Martini, T. Besker, and J. Bosch, "Technical debt tracking: Current state of practice: A survey and multiple case study in 15 large organizations," Science of Computer Prog., vol. 163, pp. 42–61, 2018.

[19] S. Matook and M. Indulska, "Improving the quality of process reference models: A quality function deployment-based approach," Decision Support Systems, vol. 47, no. 1, pp. 60–71, 2009.

[20] J. Morbach, A. Wiesner, and W. Marquardt, "Ontocapea (re) usable ontology for computer-aided process engineering," Computers & Chemical Engineering, vol. 33, no. 10, pp. 1546–1556, 2009.

[21] P. Novák, E. Serral, R. Mordinyi, and R. Sindelár, "Integrating heterogeneous engineering knowledge and tools for efficient industrial simulationmodel support,"Advanced Engineering Informatics, vol. 29, no. 3, pp. 575–590, 2015.

[22] P. Runeson and M. Höst, "Guidelines for conducting and reporting case study research in software engineering," Empirical software engineering, vol. 14, no. 2, p. 131, 2009.

[23] M. Sabou, O. Kovalenko, and P. Novák, "Semantic modeling and acquisition of engineering knowledge," in Semantic Web Technologies for Intelligent Engineering Applications. Springer, pp. 105–136, 2016.

[24] M. Schleipen, A. Lüder, O. Sauer, H. Flatt, and J. Jasperneite, "Requirements and concept for plug-and-work," at-Automatisierungstechnik, vol. 63, no. 10, pp. 801–820, 2015.

[25] M.-a. Sicilia, E. Garćia-Barriocanal, S.Sánchez-Alonso, and D. Rodríguez-García, "Ontologies of engineering knowledge: Generalstructure and the case of software engineering," The Knowledge Engineering Review, vol. 24, no. 3, pp. 309–326, 2009.

[26] D. da Luz Siqueira, L. M. Fontoura, R. H. Bordini, and L. A. L. Silva, "A knowledge engineering process for the development of argumentation schemes for risk management in software projects." in Proc. SEKE, pp. 36–41, 2017.

[27] A. Strahilov and H. Hämmerle, "Engineering workflow and software tool chains of automated production systems," in Multi-Disciplinary Eng. for Cyber-Physical Production Systems. Springer, pp. 207–234, 2017.

[28] VDI 3695: Engineering of industrial Plants, Evaluation and Optimization, Beuth Verlag Std., 2009.

[29] R. J. Wieringa, Design science methodology for information systems and software engineering. Springer, 2014.