Technical Report

Industry 4.0 Asset Based Requirements Tracing in Cyber-Physical Production System Engineering

Stefan Biffl[2]
Kristof Meixner[1,2]
Arndt Lüder[3]
Jan Herzog[4]
Felix Rinker[1,2]
Anna-Kristin Behnert[3]
Dietmar Winkler[1,2]

[1] Christian Doppler Laboratory for Security and Quality Improvement in the Production System Lifecycle (CDL-SQI)
[first.last]@tuwien.ac.at

[2] Institute of Information Systems Engineering, TU Wien and CDP, Austria
[first.last]@tuwien.ac.at

[3]Otto-von-Guericke University Magdeburg, Germany
[first.last]@ovgu.de

[4]Volkswagen AG, Germany
[first.last]@volkswagen.de

Technical Report No. CDL-SQI 2021-07
Issued: May 2021

Citation: S. Biffl, K. Meixner, A. Lüder, J. Herzog, F. Rinker, A-K. Behnert, D. Winklerl: “Industry 4.0 Asset Based Requirements Tracing in Cyber-Physical Production System Engineering”, Technical Report CDL-SQI 2021-07, TU Wien, Vienna, Austria, March 2021, 2021.

Industry 4.0 Asset Based Requirements Tracing in Cyber-Physical Production System Engineering

Stefan Biffl*,§, Kristof Meixner*,† Arndt Lüder†, Jan Herzog‡, Felix Rinker*,†
Anna-Kristin Behnert†, Dietmar Winkler*,†
† Christian Doppler Laboratory for Security and Quality Improvement in the Production System Lifecycle,
* Inst. of Information Systems Eng., TU Wien and § CDP, Austria. Email: [firstname.lastname]@tuwien.ac.at
† Otto-von-Guericke University, Magdeburg, Germany. Email: [firstname.lastname]@ovgu.de
‡ Volkswagen AG, Wolfsburg, Germany. Email: [firstname.lastname]@volkswagen.de

***Abstract*—In *Cyber-Physical Production System (CPPS)* engineering, domain experts want to trace requirements from production processes to engineering artifacts on a detailed level to efficiently reuse *Industry 4.0 Assets* that automate production processes. However, the knowledge on processes and assets is often scattered on engineering artifacts and domain experts, making it hard to trace multi-disciplinary requirements. In this paper, we investigate requirements tracing for reusing robot cells, building on the domain analysis of 80 robot cell types. To integrate production process and asset knowledge, we introduce *Industry 4.0 Asset based Requirements Tracing (I4ART)*, a traceability information model and process, providing linked Industry 4.0 Assets for defining requirements trace links from production processes to software-relevant assets for reuse. We evaluate the I4ART approach with data from real-world use cases (i) by instantiating an I4ART knowledge graph, (ii) by estimating the effort for eliciting and validating trace links on different levels of detail, and (iii) by validating the findings with domain experts at four CPPS engineering companies. In the study, the effort for trace link elicitation was found reasonable and the I4ART knowledge graph efficient for accessing tracing knowledge.**

***Index Terms*—Industry 4.0, Trace Links, Tracing effort, Cber-Physical System, Knowledge Graph.**

I. Introduction

Requirements tracing [1], [2] has been successfully used to address safety-critical aspects in designing and validating software-intensive systems [3], e.g., cars and power plants, but mainly to pass certification rather than to support engineering, often due to cost-benefit questions [4], [5]. In this paper, we investigate the cost-benefit of requirements tracing for the reuse of production systems that automate processes to produce car parts [6], [7]. A key question is the viability of requirements tracing on a level of detail to validate software elements of a reusable production system, such as a robot cell to automate joining processes in car manufacturing [8].

Modern industrial production processes, e.g., for assembling cars, are automated by production systems [6], [9], [10]. The *Industry 4.0 (I40)* vision aims at providing adaptive and reusable production systems based on flexible and adaptable solution designs [6], [11]. *Cyber-Physical Production Systems (CPPSs)* represent a recent production system architecture paradigm [9] based on self-contained, interacting, and hierarchically structured system components [6], so-called *I40 components*. An I40 component, like a work cell for positioning and joining car parts, is a software-intensive system, composed of I40 Assets. These systems can describe their capabilities, called skills [12], and adapt their behavior according to their context. For instance, this is needed for adjusting production resources to changing product designs or to balance the workload between production processes in a work line [6].

An *I40 Asset* is a product, process, or production resource with a set of skills and a standardized digital representation, the so-called *I40 Asset Administration Shell (AAS)* [13]. For configuring an I40 Asset, e.g., a robot in a work cell, it is necessary to provide the asset with sufficient context on the production system and on process requirements. The coordination of several I40 Assets, e.g., two robots in a work cell (cf. Section IV), requires efficient orchestration [14]–[16].

Designing adaptive CPPSs with I40 Assets considerably increases the complexity of system design requirements in comparison to traditional production systems [17]. CPPSs produce a variety of highly-customizable products that require numerous, ideally reusable [18], types of production processes, system components, and solution design elements. Hence, a key success factor is keeping an overview of the requirements in multi-disciplinary CPPS engineering [17] on a sufficient level of detail. This requires the capability to trace system requirements that require multi-disciplinary design solutions to software requirements and solutions that determine CPPS behavior. However, the following challenges [6], [17] impede effective and efficient requirements tracing [2], [19], [20].

Challenge 1. Scattered and incomplete representation of domain knowledge for requirements tracing. CPPS design knowledge comes from (i) several departments, potentially in different companies, (ii) heterogeneous engineering artifacts of several disciplines, like mechanical and electrical engineering, and (iii) implicit domain expert knowledge. Usually, this knowledge involves high-level requirements, like product design and business processes, and low-level requirements, like the robot motion speed. In addition, the knowledge concerns requirements dependencies like the motion speed required to achieve the planned production cycle time.

The scattered engineering knowledge makes it hard to elicit and maintain *Trace Links (TLs)* [4] that represent knowledge on requirements definition and fulfillment as a foundation for automating reuse in CPPS engineering. However, the

I40 transformation from rigid production systems to adaptive CPPS requires more detailed tracing of multi-disciplinary requirements on a suitable level of abstraction.

Challenge 2. Volatile system requirements. Changes to product design, e.g., to a car part, have an impact on production process and resource requirements and design candidates [8]. However, it is difficult to represent the requirements space of changing process variants and product designs towards a CPPS to select and validate solution candidates for reuse [21].

Advances in the digitalization of CPPS engineering [6] will make TL data available in better quality and more efficiently, e.g., employing the AutomationML meta-model and exchange format, as integrated models [22], [23]. Describing a traditional asset as an I40 Assets with its digital representation [13] seems promising for integrating engineering views on the asset, but it remains unclear how to use I40 Assets and skills for requirements tracing to facilitate reuse.

In this paper, we aim at formalizing and eliciting knowledge on requirements trace links to facilitate solution design reuse and to improve shortcomings on requirement trace link definition and elicitation in CPPS engineering, using *Design Science* [24], [25]. We build on a domain analysis of 80 robot cell types with 27 robot types that assemble 6 car types in several hundred variants, which may change during CPPS design and operation. We describe the representative use case *Mounting Dashboard to Car Body* (cf. Section IV) to illustrate requirements and reusable CPPS design elements. We analyze differences between traditional and CPPS designs and needs for requirements tracing to reuse solution designs, leading to the following research questions (RQs).

We investigate **RQ1** *what model supports requirements tracing in multi-disciplinary multi-model CPPS engineering.* Building on and extending approaches from knowledge integration and requirements tracing [2], [26], we introduce a *Traceability Information Model*, the *Industry 4.0 Asset based Requirements Tracing (I4ART)* network, compatible to the I40 AAS, to formalize and integrate the knowledge needed to trace requirements to I40 Assets for reusing CPPS solution designs.

We investigate **RQ2** *what traceability process engineers can follow to efficiently elicit requirements trace links to I40 Assets in CPPS Engineering.* We describe *I4ART traceability process* steps for eliciting a technical I40 Asset network as a basis for efficiently eliciting and validating requirements traces.

To investigate the viability of our approach, we report on conducting the I4ART process with tool support, a diagram/model editor and a graph database, to efficiently design and validate the I4ART network. We estimate the *effort for trace link elicitation* (i) for samples of robot cell types in car assembly of several levels of complexity and (ii) for different levels of TL detail. We evaluate with seven senior domain experts at four production systems engineering companies their needs for requirements tracing for CPPS reuse tasks and the expected cost-benefit of the I4ART approach in comparison to their typical approaches to requirements tracing.

The remainder of this work is structured as follows: Section II summarizes related work. Section III motivates the research questions and approach. Section IV introduces the use case *Mounting Dashboard to Car Body* and identifies criteria for requirements tracing to facilitate reuse in CPPS engineering. Section V introduces the *Industry 4.0 Asset based Requirements Tracing Information Model* and the *traceability process* [2]. Section VI reports on a feasibility study with an I4ART network instance, on estimating the effort for TL elicitation, and the validation of the research results with domain experts. Section VII discusses the research results and limitations. Section VIII concludes and outlines future work.

II. Related Work

This section summarizes related work on CPPS engineering, CPPS knowledge management, and requirements tracing.

A. CPPS Engineering Background

Automotive CPPSs, an important class of software-intensive systems [10], typically consist of more than 50,000 system elements. The *reference architecture for I40* (RAMI4.0) [9], [10] categorizes these elements in six levels including work lines and work cells. A typical automotive CPPS has about 200 to 300 work cells for positioning and joining car parts [27], containing automation components with embedded systems.

CPPS engineering typically involves 10 to 30 engineering domains, including mechanics, electrics, and software engineering, from several departments and companies [17]. Each domain has different views on the CPPS, and 10 to 50 engineering tools, leading to many, often partial, local views [28], [29]. Hence, engineering tasks can benefit from *multi-disciplinary requirements tracing* on a suitable level of abstraction. In particular, product design changes, e.g., to car parts, require changes in the design and operation phases [7].

Recent research is addressing an increased reuse of engineering artifacts [30]. In the automotive industry, reuse mainly focuses on engineering artifacts related to work cells and their contained function groups [8], [31]–[33]. Examples are standardized welding cells with standardized welding robots and geo-stations or standardized function groups in assembly lines, like screwing robots. Each reuse case is based on production system knowledge on resource capabilities to execute processes based on relevant materials. These are capabilities to fulfill product and production process requirements, like position reachability for welding or screwing quality, and economic and legal requirements, like throughput and safety.

Reusing process and resource designs aims at reducing engineering and maintenance effort and cost. Yet, the division of CPPS engineering among many departments and companies with local system design decisions often leads to unnecessary design variety [34]. In this paper, we aim to model process and resource requirements in sufficient detail to facilitate process and system element identification for reuse within and across projects [35]. The foundations for this reuse are reference architectures and their use to define a system architecture enabling the integration of the artifacts to be reused [36]–[38].

B. Knowledge management in CPPS engineering

CPPS engineering knowledge is based on the core CPPS assets: products, processes, and resources, i.e., *Product-Process-Resource (PPR)* knowledge [12]. Engineering models represent objects and dependencies that reflect these PPR aspects [39], [40]. Yet, this knowledge is scattered over multi-disciplinary multi-model artifacts with various dependencies [41]. The dependency knowledge is implicit knowledge of domain experts on production processes and work cells [42].

I40 digitalization aims at more effective engineering by employing knowledge integration and reuse across engineering disciplines and projects [43]. I40 Assets are valuable objects [13], representing PPR concepts and production recipes etc. [6], [10]. The *I40 AAS* [13] is the formal digital representation of an I40 Asset utilizing object-related system modeling [44]. The I40 AAS can represent I40 Assets and integrate the knowledge coming from several discipline-specific property views. The interdisciplinary nature of *I40 Assets* that carry multi-model data and documentation combined with their AAS [13] goes far beyond the typical representation of software-intensive systems, like embedded systems. To the best of our knowledge, I40 Assets have been used to collect engineering data, but not for tracing CPPS requirements.

The emerging *skill* concept further supports multi-disciplinary CPPS knowledge integration. Pfrommer *et al.* [12] define skills as the ability of resources to perform a process, and a production skill as the demands towards resources [45]. Meixner *et al.* [21] describe how skills can abstract resource capabilities from process demands. In this paper, we build on skills as essential requirements aggregation to represent production process demands towards resources [8], [21].

Data extraction and integration requires data exchange and integration formats, e.g., AutomationML, which facilitates the elicitation of technical interfaces and dependencies between I40 Assets, and provides data logistics capabilities [46], [47].

The I40 AAS [13], skills [12], and AutomationML [22] aim at better knowledge integration by making knowledge from engineering artifacts easier to interpret for humans and computers. In this paper, we build on the I40 Assets to formalize CPPS engineering knowledge, skills as aggregation for process requirements, and AutomationML to describe technical interfaces and dependencies between I40 Assets. These contributions provide a basis for bi-directional, non-acyclic *knowledge graphs* [48] for requirements tracing in CPPS engineering.

C. Requirements tracing in Systems Engineering

There is a mature tradition on tracing requirements [1], [2], [49] for increasingly large, heterogeneous, and flexible systems of systems, in application domains such as smart energy, smart city, and CPPS engineering [3], [28], [29], [49], [50]. Fundamental challenges in requirements tracing [4], [5], [19], [51] concern methods and tools [3], [52]–[54] for the effective and efficient elicitation of *Trace Links (TLs)* [50], [55]–[57] in sufficient quality from heterogeneous artifacts in engineering practice [26]. Wohlrab [49] points out that engineers are not likely to use applications that suffer from inconsistent TLs. In the engineering of multi-disciplinary, heterogeneous CPPSs, there are many kinds of technical links and dependencies that are candidates for tracing but are scattered on multiple models [26]. In this paper, we build on Traceability Information Model and Process concepts [2], [26] to provide an integrated model for facilitating the elicitation and validation of requirements traces in multi-disciplinary CPPS engineering.

The goal for a selected task with a required level of detail for tracing is to elicit a set of TLs from engineering artifacts, with minimal, reasonable effort [19], [58]–[60]. While requirements tracing is mandatory for regulated safety-critical systems aspects, for non-regulated applications, important questions are (i) what level of tracing granularity is sufficient and achievable [19]; (ii) what is the expected tracing effort; and (iii) how to make tracing affordable for practical purposes. These issues have not been sufficiently explored in CPPS engineering. Therefore, we focus in this paper on cost estimation for tracing in a CPPS engineering use case on different levels of detail.

Multi-disciplinary CPPS engineering can benefit from *collaborative requirements tracing* [20], [61] approaches. In CPPS engineering, work groups with heterogeneous models require boundary objects [49] to bridge requirements tracing gaps, to provide context and to overcome issues from implicit domain knowledge. In this paper, we investigate the potential of I40 Assets and their AAS to provide boundary objects that enable integrating several engineering views.

III. Research Questions and Approach

To investigate requirements tracing in CPPS engineering, we followed the Design Science approach [24], [25]. To understand and explore the environment and business needs, we first (i) reviewed literature on CPPS engineering and requirements tracing (cf. Section II) and (ii) conducted a *domain analysis* from a process improvement project (cf. Section IV) at the large car manufacturing *Company A*. From this analysis, we elicited (i) the use case *Mounting Dashboard to Car Body* to illustrate reuse in CPPS engineering (see Section IV), (ii) shortcomings in typical CPPS engineering, see Challenges 1 and 2 in Section I, impeding requirements tracing for reuse, and (iii) criteria for requirements tracing capabilities as a foundation for reuse (see Section IV).

RQ1. Traceability Information Model. *What model supports requirements tracing from system to software requirements in multi-disciplinary multi-model production systems engineering?* To address RQ1, we build on the insights and knowledge acquired in the domain analysis. Leveraging the data integration capabilities (a) of the I40 AAS [13] to integrate engineering views on the asset, (b) of technical links between I40 Assets to represent trace data for a network, and (c) of skills to represent requirements of products and processes for CPPS capabilities, we design the *I40 Asset network and knowledge graph* [48] and the *I4ART Information Model* (cf. Section V) for *Industry 4.0 Asset based Requirements Tracing*. We illustrate the design and analyze trace link types

with examples from the use case *Mounting Dashboard to Car Body* (cf. Section IV, Figure 2) for reusing robot cells.

RQ2. Traceability Process. *What traceability process can engineers follow to efficiently elicit requirements trace links to I40 Assets in CPPS Engineering?* To address RQ2, we design the I4ART process for efficient elicitation of sufficiently detailed requirements trace links from heterogeneous engineering data sources to instantiate the I4ART information model coming from RQ1 (cf. Section V-B). We apply the I4ART process to design a network instance for the use case *Mounting Dashboard to Car Body* (cf. Sections IV and VI) with example data from typical CPPS engineering artifacts to explore the viability of I4ART. For samples of robot cell types on several levels of complexity, coming from the domain analysis data set, we estimate the *effort for trace link elicitation*.

We discuss the I4ART information model and traceability process results with domain experts. We do this to understand their needs for improving reuse capabilities in CPPS engineering with requirements tracing and to compare the I4ART model and method results with their typical approaches for providing requirements traceability.

This paper provides the following contributions to the RE community: (i) insights on CPPS domain concepts and tracing issues; (ii) knowledge elements for requirements tracing to bring scattered knowledge together in multi-model engineering as a basis for facilitating reuse tasks; and (iii) the I4ART network containing knowledge elements to collect engineering artifacts, requirements trace link types and instances for use as training data to facilitate automating trace link elicitation.

IV. Illustrative Use Case for Evaluation

This section introduces the illustrative use case *Mounting Dashboard to Car Body*, and derives criteria for improving requirements tracing for reuse in CPPS engineering.

The authors of this paper conducted an in-depth domain analysis in the context of a process improvement project with applied researchers at *Company A*, a large automotive CPPS engineering company. The analysis results were published on the required knowledge for requirements tracing, such as skills, and gaps in artifact exchange [8], [21], [27]. The analysis was conducted for six months in workshops with 10 domain experts from 5 domains. We investigated designs of 200 types of *positioning and joining* processes [62] with 80 robot cell types and 27 robot types. These cells screw several hundreds of parts on a car body with up to 1,800 screwing points for the assembly of 6 car types with hundreds of configurations [8], [27]. These robot cell types can be simple cells, with one screwing robot, or complex cells, with two robots for coordinated positioning and screwing (cf. Figure 1).

This use case is representative for positioning and joining processes with car parts and the required robot cells. The use case consists of two main process steps: (1) position the *dashboard* and *screw* it onto the *car body*; (2) fasten the *screw* and measure the joint. The result is the dashboard mounted to the car body. The requirements for the automation of this process are surprisingly complex due to (i) the variety of dashboards and car bodies and (ii) the integration of other processes in the work line, e.g., for reacting to timing or quality issues in previous production steps [8], [21].

Figure 1 (blue elements) shows, for a particular *work cell with two robots*, an overview of the products, production processes, skills, and production system elements and associated engineering artifacts, which hold detailed information on the system elements. The system elements range from main CPPS resources down to the software elements that control the robot cell's behavior. Further, Figure 1 (orange boxes) shows selected requirements (cf. Sections V-A and VI): high-level business requirements, production process requirements, derived multi-disciplinary skills and main CPPS resource requirements, and a chain of derived single-discipline resource requirements down to software elements. Together, these requirements allow validating the automation of the robot cell behavior and the engineering project's progress.

Reuse task. For *requirements tracing in CPPS engineering*, we focus on a *reuse task* that can benefit from tracing business and system requirements to software element solutions, represented as I40 Assets. A typical goal is to reuse the design of a *screwing process automated by a robot cell* with adaptations at stations with similar requirements, e.g., designed by different departments. The design includes the product, process, system and software elements, the engineering artifacts, and the derived requirements. We illustrate tracing requirements for reusing (i) the production process *Fasten Screw and Measure* and (ii) the system and software design for coordinating the two robots in the work cell (cf. Figure 1).

From product designs and customer requirements, the *Functional Planner* derives the requirements for the *production process* and the associated *Skill*, e.g., *Requirement FP23*, which represents the parameters to produce all car type variants on the work line that contains the work cell. These skill requirements are input (i) to identify solution design candidates for the robot cell and (ii) to define requirements for the robot cell subsystems, e.g., the *robots* and *screwdrivers*, and the *industrial PC* that orchestrates the behavior of the robots according to (i) the type of car to which the dashboard should be mounted and (ii) production information on previous production steps coming from I40 Assets in other work cells.

Figure 1 illustrates *requirements chains* starting at original business requirements and leading to derived multi-discipline and single-discipline requirements as a foundation for the validation of software requirements in the overall system context. Based on the requirements coming from the skill, the Functional Planner and the Mechanical Engineer select and *derive multi-disciplinary requirements* for the work cell's main resources, i.e., the *IPC*, *robots*, and *screwdrivers*. The Mechanical Engineer derives and addresses mechanical requirements, and hands down the device chain of a main CPPS resource, e.g., *Screwdriver-Drive-Transformer-Controller*, his design results to the *Electrical Engineer* and the *Software Engineer*, who derive and address the requirements for their disciplines and propagate their design results to construction.

Software requirements depend on multi-disciplinary system

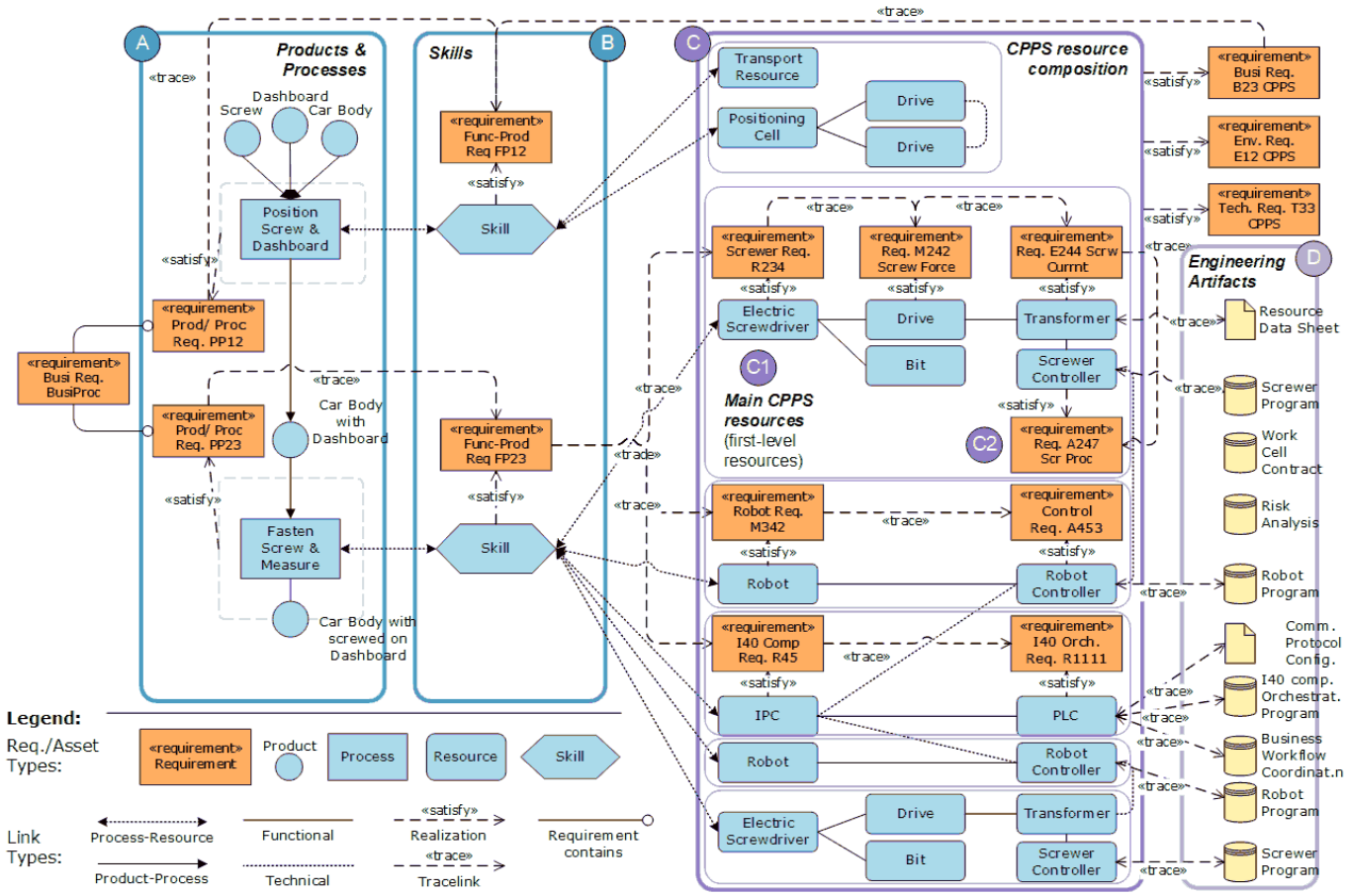

Fig. 1. Overview on selected *requirements*, *assets*, and *engineering artifacts* for the use case *Mounting Dashboard to Car Body*; for *production process steps* (tag *A*) automated by robot cell (tag *C*), a *CPPS* that is composed of *I40 Assets*; based on [63], in SysML and adapted VDI 3682 notations [21], [64].

requirements that have to be split into single-discipline requirements for the mechanical, electrical, and software engineer to work on. Further, software engineers have to adapt their requirements to the design decisions coming as engineering artifacts from mechanical and electrical engineers. For instance, the chain of requirements and associated I40 Assets starting at the *Electric Screwdriver*, leading to the *Screwer Controller*, with the software artifact *Screwer Program* in Figure 1.

The industrial PC (*IPC*) is linked via a communication network to the robots in the work cell. Derived from the multi-disciplinary system requirements to the IPC, there are software requirements for the PLC that controls the IPC and the high-level robot behavior as defined in detail in the engineering artifacts *I40 Orchestration Program* and *Business Workflow Coordination*. In robot-based placing and mounting, the robot motion paths require different designs and coordination, depending on the involved variants and status of car body, dashboard, and selected I40 components. During virtual commissioning, the *Simulation and Quality Engineer* validates the robot cell behavior against the work cell's system requirements and the derived requirements for each engineering discipline.

Shortcomings in typical CPPS engineering. Trace links are important for reuse, but are hard to collect efficiently in CPPS engineering. Unfortunately, several *shortcomings in requirements tracing* hamper reuse in CPPS engineering. Requirements tracing along the chains of derived requirements can be very useful for the early validation of the requirements for software to control the behavior of the robot cell regarding their reuse in similar contexts. Yet, in typical CPPS engineering, *derived single-discipline requirements are often not explicitly represented* but remain in the discipline-specific information silos. They only come implicitly with sharing system elements among engineers, e.g., software engineer have to rely on their expertise to infer requirements from given design elements, risking to overlook or misinterpret implicit requirements leading to unplanned cost or project delay. In general, requirements may be traced to engineering artifacts containing data on many I40 Assets, but traces to individual I40 Assets nor across engineering disciplines are not available.

Traceability strategy. Reuse of work cells in manufacturing systems, including software elements, demands capabilities for tracing multi-disciplinary requirements on the level of system elements, I40 Assets, in CPPS engineering: 1) Tracing on a suitable level of abstraction, i.e., to I40 Asset concepts, not just to heterogeneous engineering artifacts that concern many I40 Assets. 2) Tracing CPPS requirements, which concern

automating a variety of products and production processes, to skills for reusable CPPS elements. 3) Tracing from multi-disciplinary requirements to single-discipline requirements, e.g., software requirements that determine CPPS behavior.

Criteria for Requirements Tracing. From the domain analysis, we derive the following criteria for *requirements tracing* as a basis to improve reuse tasks in CPPS engineering.

R1. Trace Links in I40 Asset Network. Trace Links link product, process, skill, and resource assets to form an I40 Asset Network as foundation for tracing the technical relationships from product-transforming production processes to resources that automate the process.

R2. Trace Links in Requirements Network. Trace Links link original and derived multi-/single-discipline requirements to form a requirements network as foundation for tracing original system requirements to detailed discipline-specific component requirements.

R3. Trace Links from Requirement to I40 Asset. Trace Links link requirements to I40 Assets that define or address the requirements as a foundation for requirements' validation.

R4. Trace Links from I40 Asset/Requirement to Engineering Artifact. Trace Links link an I40 Asset or a Requirement to Engineering Artifacts that describe and/or use the I40 Asset or implement the requirement, e.g., design documents, validation reports, as foundation for tracing evidence on the rationale to I40 Asset design.

R5. Efficient elicitation of high-value trace links. Trace Link elicitation from engineering models, artifacts, and human expertise considers the cost-benefit of trace links to elicit and ensure sufficient quality of trace links that form the basis for correct and sufficiently complete trace link applications in engineering, e.g., reuse of work cells, for improving the business value coming from requirements tracing.

V. Traceability Information Model and Process

This section describes the *I4ART* information model and the traceability process [2] to facilitate the reuse of production processes and work CPPS cells.

A. I40 Asset based Requirements Tracing Information Model

To address RQ1, this sub-section introduces the *Traceability Information Model* [2] for the *I4ART network*, illustrated with example data from the use case *Mounting Dashboard to Car Body* (cf. Section IV, Figure 1). Figure 2 shows the I4ART *Information Model*, the meta model of the I4ART network (in UML notation), based on the general Traceability Information Model for heterogeneous system engineering in [26].

Trace Artifact Types. A trace artifact is a *Requirement*, an *I40 Asset*, or an *Engineering Artifact*. A trace artifact has a unique identifier, a name and properties, building on the *I40 Asset Administration Shell* design [13] to facilitate the integration of multi-disciplinary engineering data for tracing. Therefore, the properties of an I40 Asset, such as an *Electric Screwdriver* (cf. tag *C1* in Figure 1), can contain a mechanical view with the mechanical ID and the *maximal torque* and an electrical view with the electrical ID and the *maximal current*.

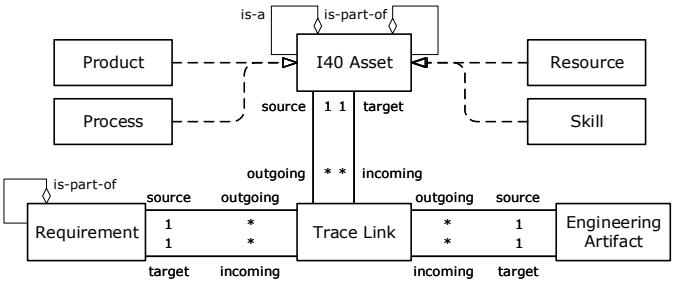

Fig. 2. *Requirement*, *I40 Asset*, *Engineering Artifact*, and *Trace Link* meta model, based on [27], in SysML notation.

A *Requirement* is an original or derived, multi- or single-discipline requirement (cf. boxes in orange color in Figure 1). An *I40 Asset* [13] is a product, process, resource [12], or skill [21] (cf. the elements in blue color in Figure 1). A *Resource* is a CPPS element (cf. tag *C* in Figure 1).

A *Skill* [21] connects a production process to resources that automate this process (cf. tag *B* in Figure 1). In this connection, the skill represents the requirements of production process variants, e.g., screwing with a torque range and maximum duration for particular car types, to enable the selection of a set of suitable main CPPS resources.

An *Engineering Artifact* can be a document, model, or database, such as a contract, a CAD plan, or a set of requirements (cf. tag *D* in Figure 1).

A *Requirement* and an *I40 Asset* can be part of a hierarchical structure, e.g., have sub-elements to represent different levels of detail, e.g., a robot cell and devices in the cell. An *I40 Asset* can be part of a type hierarchy, e.g., specific robot types as sub-types of a general robot type to allow defining different properties for sub-types.

Trace Link Types. A *Trace Link* connects exactly one trace artifact with exactly one other trace artifact (cf. the link types in Figure 1). Table I lists example trace links, shown in Figure 1, to illustrate the trace link types and sub-types to address the criteria R1 to R4 introduced in Section IV.

The trace artifacts and links form the following networks.

1. I40 Asset Network. The I40 Assets linked with discipline-specific technical links, i.e., interfaces and dependencies coming from all involved engineering disciplines (trace link types I40 Asset - I40 Asset; I40 Asset - Process - Skill, Skill - I40 Asset - Resource; sub-types functional, technical; product-process, process-resource). This network defines neighborhoods in the I40 Asset Network, e.g., all sub-components of a main CPPS resource or the assets that a software element depends on. Therefore, an I40 Asset is a boundary object between several disciplines and an entry point to the I40 Asset Network. Further, domain expert knowledge such as new dependency links between I40 Assets may be added as TLs to the I40 Asset Network.

2. Requirements Network. The chains of requirements from original business requirements to *derived multi-discipline* and *derived single-discipline requirements* (trace link types Requirement-Requirement, sub-types *Contains* and *Trace*.

ID	Source Artifact	Target Artifact	Trace link type	Sub-type
1	Req Busi 22	Prod/Proc Req FP23	Req-Req	Contains
2	Process Fasten Screw and Measure	Prod/Proc Req FP23	I40 Asset-Req	Satisfy
3	Prod/Proc Req FP23	Func/Prod Req FP23	Req-Req	Trace
4	Car Body w/Dashboard	Process Screw & Measure	I40 Asset - I40 Asset	Product - Process
5	Process Fasten Screw and Measure	Skill FP23	I40 Asset-Process-Skill	Process - Resource
6	Skill FP23	IPC	Skill-I40 Asset-Resource	Process - Resource
7	IPC	PLC	I40 Asset - I40 Asset	Functional
8	PLC	Comm. Protocol Config.	I40 Asset - Eng.Artif.	Trace
9	Drive	Drive	I40 Asset - I40 Asset	Technical

TABLE I
TRACE LINK TYPE EXAMPLES (CF. FIGURE 1), BASED ON [26].

3. Requirements-Assets Network. I40 Assets linked to requirements that define them (trace link types I40 Asset - Requirement; sub-type trace). I40 Assets, such as screwing application, skill, or *Screwer Controller* (see tags *A*, *B*, *C2* in Figure 1), are linked to or indicate associated requirements.

4. Assets-and-Requirements-to-Artifacts Network. I40 Assets and/or requirements linked to the engineering artifacts that define or implement them. (trace link types I40 Asset - Engineering Artifact, Requirement - Engineering Artifact; sub-types *satisfy*, *trace*).

Together, these networks address criteria R1 to R4 (cf. Section IV) leading to a reusable structure and content types, illustrated in Figure 1, for similar production processes and work cells. The links between the Requirements Network and the I40 Asset Network enable navigating between these networks coming from any starting requirement or I40 Asset.

B. I40 Asset based Requirements Tracing Process

To address RQ2, this section introduces the *I4ART process* for trace link definition, elicitation, and application. The role *tracing engineer* can conduct the following steps iteratively.

Step 1: Design I40 Asset Network Instance. To prepare a domain concept network as foundation for requirements tracing, the tracing engineer elicits in the scope of reuse, in the use case a process for positioning and screwing, the I40 Assets and technical links, i.e., interfaces coming from models and artifacts in multiple CPPS engineering disciplines.

In the use case context, *product* and *process* asset and link data (see tag *A* in Figure 1) was available in the *Bill of Materials* and *Bill of Operations*. Resource asset and link data came from the *Bill of Resources*; all kinds of data came in the form of spreadsheets and engineering tool export data [23].

Skills [21] are a recent type of I40 Asset (see tag *B* in Figure 1) to represent requirements of processes for capabilities of main CPPS resources (see tag *C1* in Figure 1). Skills can be instantiated from technical links between a process and main CPPS resources, coming from functional engineering results.

Domain experts add can their implicit knowledge on dependencies between I40 Assets, building on the rich data foundation of the I40 Asset Network that integrates the currently scattered knowledge on an I40 Asset and its dependencies.

Step 2: Refine Requirements and elicit Trace Links. Based on the I40 Asset Network coming from Step 1, and the list of requirements in the scope of reuse, the tracing engineer (2a) refines requirements for the I40 Assets in the network (see tags *A, B, C1, C2* in Figure 1) from original high-level business requirements to skills to main CPPS resources, then following the device chain between main CPPS resources to software elements; and (2b) elicits trace links between I40 Assets and requirements (mostly manual); and (2c) elicits TLs between I40 Assets and engineering artifacts (semi-automated, depending on artifact data quality) (see tag *D* in Figure 1).

Step 3: Validate and apply Trace Links to Reuse Task. Based on the I40 Asset Network with requirements TLs (cf. Step 2), and the list of requirements in the scope of reuse, the tracing engineer (3a) runs queries to explore production process assets in the reuse scope and their I40 Asset neighborhood; (3b) reasons on the knowledge graph to recommend new or flag TLs for inspection; (3c) visually inspects results graphs for validating TLs; and (3d) adds or improves TLs for their task-specific use.

We applied this traceability process to investigate the feasibility of the approach that addresses criterion R5 (cf. Section IV) for reuse tasks in CPPS engineering.

VI. FEASIBILITY STUDY AND TRACING EFFORT

To evaluate our research (cf. Section V), this section reports on (i) a feasibility study instantiating an I4ART network to explore querying capabilities to automate reuse tasks; (ii) the estimation of the number of trace artifacts/links for typical robot cells and the effort to elicit and validate these trace links at several levels of detail from typical data sources; and (iii) the comparison of the I4ART approach with alternative tracing approaches in practice in workshops with domain experts.

Feasibility study. Two authors of this paper conducted the I4ART process steps (cf. Section V-B) for a production process and a work cell with two robots, selected from a sample of robot cells, resulting in a I4ART knowledge graph of 126 trace artifacts and 182 trace links. The graph database provided sufficient capabilities to browse and query the I40 Asset network instance as foundation for verifying traces and for automating requirements tracing and reuse tasks.

The required information is likely to be available in typical CPPS engineering projects and can be aggregated as a sequence of data transformations from spreadsheet tables and engineering tool exports to the I4ART network, in this case a *Neo4J*[1] graph database and a custom diagram/model editor for trace link design, visualization, validation, and improvement.

Estimation of tracing effort. Tracing effort depends on the number of *Trace Links (TLs)* and the average effort for eliciting and validating a TL, which depends on data source quality and the method used for TL elicitation. The number of *TLs* depends on the number of trace artifacts, which depends on the complexity of the robot cells and the level of detail for tracing (to main CPPS resources or to detailed software elements, cf. tags *C1*, *C2* in Figure 1). In the following, we discuss data collection and analysis for tracing effort estimation.

[1]https://neo4j.com/

Data sample. Two authors of this paper analyzed the number of trace artifacts at *Company A* based on samples of screwing processes with 3 to 5 robot cell types for simple, medium, and complex types of cells (cf. Tables II and III), selected from the 80 types in the domain analysis, which represent a typical range of robot cell capabilities in automotive manufacturing. Common characteristics of the robot cells are the function to position and join car parts, their technical architecture, and their overall structure (cf. Figure 1).

Robot cell complexity levels (simple/medium/complex) mainly differ in the number of screwing applications (1/2/3), the number of robots (1/1/2), the components for synchronizing the joining process, and the robot sensors and tools for joining, varying the number of trace artifacts and links.

Number of Trace Artifacts. For the process and robot cell sample, Table II shows the minimum, average, and maximum numbers of trace artifacts by type: I40 Assets, requirements, and engineering artifacts (cf. Section V). Note that the number of requirements in Table II assumes bundling for each I40 Asset the detailed requirements for parameters, i.e., 5 to 10 parameters for a screwing application of a *Process*/*Skill* and up to 5 parameters for a CPPS *Resource*.

Asset type	#min/avg/max	**Req. type**	#min/avg/max	**Eng.Art. Type**	#min/avg/max
Product	4/6/8	Original	6/12/20	Document	15/20/26
Process	2/2.5/4	Derived	14/24/45	Spreadsheet	3/3/5
Skill	2/2.5/4	Multi-disc.	8/14/27	Model	3/5/6
Resource	12/22/38	Single-disc.	12/22/38	Program	4/6/8
Sum	**20/33/50**	**Sum**	**20/36/65**	**Sum**	**25/34/45**

TABLE II
OVERVIEW ON TRACE ARTIFACT TYPES AND NUMBER OF INSTANCES IN THE SAMPLE OF ROBOT CELLS FOR JOINING (CF. FIGURE 1).

In the study, the number of requirements grew considerably from original business requirements to derived requirements for assets within a robot cell (derived requirements, cf. tags *C1*, *C2* in Figure 1) to guide engineers in the various disciplines (by deriving multi-discipline into single-discipline requirements). Further, there were *Engineering Artifact* documents, mainly data sheets, spreadsheets specifying trace artifacts and their parameters, models relating trace artifact aspects, and programs to define the behavior of the robot cell elements.

Trace Link Type	#min/avg/max	**Factors to Trace Artifacts**
Req-Req	20/42/80	1.0 to 1.5 * # Requirements
Req-Asset	20/32/70	1.0 to 1.5 * # Assets
Req-Eng.Artif.	30/43/60	1.0 to 3.0 * # Eng.Artif.s
Asset-Asset	25/41/60	1.0 to 1.5 * # Assets
Asset-Eng.Artif.	15/38/80	1.0 to 3.0 * # Assets
Eng.Artif.-Eng.A.	2/6/10	0.1 to 0.3 * # Eng.Artif.s
Sum	110/205/360	

TABLE III
OVERVIEW ON TRACE LINK TYPES AND NUMBER OF INSTANCES IN THE SAMPLE OF ROBOT CELLS FOR JOINING (CF. FIGURE 1).

Number of Trace Links. Table III shows an overview on the minimal, average, and maximal numbers of trace links by type in the data sample, for tracing original and derived requirements to I40 Assets (incl. software elements) and to Engineering Artifacts. Further, Table III shows the main driver for the number of trace links for each link type and the range of factors to estimate the number of trace links for a type.

We investigated tracing on different levels of detail: low level of detail, i.e., original requirements to Engineering Artifacts resulted in (12/23/40) TLs; medium level of detail, i.e., original requirements to main CPPS resources to Engineering Artifacts resulted in (30/54/80) TLs; and high level of detail (cf. Table III), resulted in (110/205/360) TLs.

The number of requirements and TLs largely depended on the *complexity of the robot cell* and the *level of detail of tracing*. The number of trace links can be considerable, up to 360 TLs, when tracing from production processes (screwing applications) to software elements for a complex robot cell.

Tracing Effort. We calculate the basic effort for tracing as the number of trace links times the average effort for eliciting a trace link. To estimate the average effort per trace link elicitation, we collected data on the effort to extract a sample of trace links as input to estimate the effort for requirements tracing for a production process and robot cell. (cf. Table III). In the study, the data source quality allowed eliciting assets and original requirements with tool support, while skills and derived requirements typically came from domain experts.

In the study (cf. Table III), we measured and estimated the effort for *Trace Link (TL)* elicitation as the sum of effort for the tasks (a) preparation and import of data from spreadsheets, structured documents (1-3 work hours (wh)), (b) manual tracing from documents and models (3-5 minutes per TL; or 20-30 minutes per group of TLs that can be imported together), (c) validation by a domain expert in the I4ART graph (2-3 wh), and (d) correction of 10% to 20% TLs (2-3 wh). In the study context, *the overall tracing effort per robot cell amounted to 11 to 32 work hours*, depending on the level of tracing detail (cf. tags *C1*, *C2* in Figure 1), production process to main CPPS resources or software elements (cf. Tables II and III). Therefore, tracing at a high level of detail will benefit from advanced methods for automation/recommendation of TLs requiring good data source quality for eliciting TLs.

While the level of tracing effort per robot cell estimated in the feasibility study seems reasonable to high as extra engineering work for one work cell, the extra effort could be significantly reduced by efficiently collecting requirements and trace links as part of engineering tasks. Further, the structure of robots cells is stable over variants and versions. Therefore, domain experts could describe requirements and trace links efficiently building on a pre-filled structure of requirements and TLs (cf. the requirements and TLs in Figure 1 and Tables I and III) that they adapt to a specific robot cell during design.

Validation with domain experts. To gain insight on the benefits of the I4ART approach in CPPS engineering practice, we discussed the I4ART approach in workshops with seven senior domain experts, representing expertise on the roles functional planner, detail planner, quality manager, and reuse manager, at four CPPS engineering companies [27].

Reuse tasks. Regarding the *reuse of production processes*

and robot cells, around 42% of the domain experts reported CPPS engineering in different departments and companies, leading to high variety of technical solutions and issues during CPPS ramp-up and maintenance. Systematic reuse with requirements tracing could reduce risks of late design changes, overly long duration of CPPS validation, and maintenance cost due to high variety of solution technologies and quality levels.

Tracing requirements. Tracing is currently required for safety-critical aspects (70%) and sometimes conducted for product quality requirements to test scenarios (28%). Tracing is promising in areas where product design changes may require significant production process and CPPS design changes.

Traditional tracing approaches. The respondents reported tracing approaches in two categories. (1) *Requirements-to-Engineering Artifacts (70%).* Requirements are represented in a requirements document as elements that are traced to engineering artifacts and, sometimes, their parts. (2) *Requirements-to-CPPS-elements (42%).* Requirements are represented in a requirements database as elements that can be traced to main CPPS resources and to engineering artifacts and their parts.

However, all respondents reported major gaps in tracing regarding (a) products and production processes to CPPS resources (*skill gap*) and (b) main CPPS resources to software elements (*software gap*), making it hard to assess the impact of process changes on the CPPS design and vice versa.

Target capabilities and effort for tracing. Most respondents (85%) would consider *Industry 4.0 Asset based Requirements Tracings (I4ARTs)* based on a cost-benefit argument, if the expected benefit of tracing would significantly outweigh the cost of tracing in work hours, in line with the findings by Wohlrab *et al.* [20]. For example, reducing validation duration and effort could provide a benefit of 100 work hours for a work line with 10 robot cells. In this simple calculation, tracing would make sense, if sufficiently detailed tracing would cost less than 10 work hours per cell. This target seems achievable with good quality data sources and the automation of trace link elicitation for robot cells with similar structure and requirements, e.g., in automotive manufacturing.

The respondents found the I4ART knowledge graph efficient for accessing tracing knowledge, such as the requirements chain from a production process to the software elements that control the behavior of the CPPS that automates the process.

VII. Discussion

This section discusses results for the research questions.

RQ1. Traceability Information Model (TIM). *What model supports requirements tracing from system to software requirements in multi-disciplinary multi-model production systems engineering?* To address RQ1, Section V-A introduced the *Industry 4.0 Asset based Requirements Tracing (I4ART) Information Model* with a minimal set of trace link types to link requirements and I40 Assets for validating the requirements of software elements that control and coordinate the behavior of a production process. The *I4ART Information Model* lifts I40 Assets from engineering artifacts to a knowledge graph that facilitates describing the design chain for tracing between production process and software elements to automate reuse tasks. Based on the I4ART Information Model, we instantiated a knowledge graph in a graph database to explore tracing business requirements of production processes via skills to CPPS software elements in a´formal I40 Asset network.

These research results build on [2], [19] and go beyond the state of the art in requirements tracing [26] by modeling in the *Traceability Information Model (TIM)* (i) I40 Assets as trace artifacts that facilitate integrating engineering views based on the I40 AAS [13]; (ii) a network of technical links between I40 Assets as foundation for requirements tracing; and (iii) *skills* [21] as trace artifacts to represent the requirements of variants of a production process for several types of products to bundle and validate requirements coming from a variety of products to be produced on a work line.

RQ2. Traceability Process. *What traceability process can engineers follow to efficiently elicit requirements trace links to I40 Assets in CPPS Engineering?* To address RQ2, Section V-B introduced the I4ART process for efficient elicitation of requirements and trace links from heterogeneous CPPS engineering artifacts. The I4ART process first elicits a network of I40 Assets linked by technical links coming from multi-discipline models to design a knowledge graph that leads from a production process to main CPPS elements and further to CPPS software elements that control and coordinate process behavior. In this knowledge graph, *skills* connect production process variants to main CPPS element to bundle the requirements of the process variants. Therefore, this I4ART process step integrates knowledge scattered on multi-disciplinary engineering artifacts as a basis for requirements tracing.

The I40 Assets in the resulting I4ART knowledge graph correspond to requirements on business and technical levels, facilitate placing original requirements and deriving requirements for the involved disciplines as a foundation for reuse to validate the chain of requirements from production process to CPPS software elements. Therefore, the I4ART knowledge graph enables specific queries on the I40 Assets, their neighbor assets, and requirements to provide results for making informed decisions in CPPS design and validation for reuse.

Based on data from a production processes sample with robot cells, we estimated the number of requirements, trace artifacts, and trace links for different robot cell types and levels of detail for requirements tracing, from the process to main CPPS elements, or in detail to software elements. We also estimated the effort for TL elicitation and validation based on engineering artifacts with sufficiently complete and well accessible data on I40 Assets, their interfaces and dependencies.

These research results build on [2], [26] and go beyond the state of the art in requirements tracing [19] by a process (i) resulting in an I40 Asset network with technical links; (ii) linking a requirements network to the I40 Assets; and (iii) for detailed tracing to I40 Assets that represent CPPS software elements. Further, by estimating tracing effort and by discussing cost-benefit in the application domain multi-model CPPS engineering, confirming findings in [19], [20].

Limitations. The following limitations of the research require further investigation.

Feasibility study. The study focused on one use case in a large CPPS engineering company. This may introduce bias due to the specific selection of requirements tracing challenges and data sources considered as well as the roles or individual preferences of the domain experts. To overcome these limitations, we conducted workshops with domain experts from four CPPS engineering companies to extend the variety of environments and alternative approaches for requirements tracing. Further, we plan case studies in a wider variety of application contexts.

Assumptions on scalability. The scope of this work focuses on tracing the requirements of a small set of production processes to CPPS software elements for validating the capabilities of reuse candidates. We focused on typical production processes and work cells that occur often in automotive assembly. However, we are aware of the wide variety of CPPSs in discrete manufacturing and expect challenges for scaling up the approach to address requirements for reuse in one or several work lines. Therefore, future work should investigate the scalability of the I4ART approach.

VIII. Conclusion and Future Work

Adaptive industrial production systems, *Cyber-Physical Production Systems (CPPSs)*, require advanced capabilities for validating the fulfillment of requirements in CPPS elements, in particular software elements that coordinate and control the behavior of the CPPS and of the processes they automate [9], [28]. Tracing requirements of production processes to CPPS software elements is a promising approach to facilitate requirements validation [2], [19].

Unfortunately, in multi-disciplinary CPPS engineering, the knowledge on processes and reusable assets is often scattered on engineering artifacts and domain experts, making it hard to trace multi-disciplinary system requirements at a sufficient level of detail [19]. Further, it is hard to represent the requirements space of changing product and process design variants for requirements tracing.

Building on the results of domain analysis of 200 types of screwing processes with 80 types of robot cells and on requirements tracing approaches [2], [26], this paper introduced the *Industry 4.0 Asset based Requirements Tracing (I4ART)* information model and traceability process to integrate knowledge on production processes and assets as a basis to define sufficiently detailed requirements trace links from production processes to software-relevant assets for reuse. Following the I4ART process for a representative use case, we instantiated an I4ART knowledge graph with requirements trace links on the requested level of detail.

The *Industry 4.0 (I40) Asset* seems to be a suitable level of detail for requirements tracing as it is a concept that all CPPS engineering disciplines share, similar to the boundary object discussed in [49]. Discipline-specific properties from several disciplines come together in the *I40 Asset Administration Shell (AAS)* [13]. The *I40 Asset Network*, based on the technical links in several engineering disciplines, provides meaningful neighborhoods of I40 Assets for contextualizing reuse in CPPS engineering. The recently introduced *skills* concept represents requirements for variants of a production process to validate business requirements. Therefore, the I40 Asset Network facilitates integrating scattered and collecting implicit knowledge in CPPS engineering. We assume the concept to be useful for systems-of-systems engineering in other areas, where production/transformation processes depend on resources that automate these processes, e.g., Smart Energy.

We estimated the *effort for trace link elicitation* on different levels of granularity for robot cells in car assembly. The technical results indicate detailed requirements tracing from production process to CPPS software elements to be feasible for requirements validation with data sources that are on a suitable level of detail and accessible with automated approaches, e.g., data logistics [23].

In a validation with domain experts, they reported heterogeneous data sources with gaps regarding the level of detail and content that computers can interpret, similar to findings in [19], [59]. While the current practice does not support efficient requirements tracing on the desired level of detail, they expect digitalization to improve access to data. Hence, capabilities for efficient trace link elicitation from CPPS data sources will be a key success factor for introducing requirements tracing to CPPS engineering practice, where I40 Assets will become more widely used.

Future Work. *Trace link elicitation from CPPS data logistics.* We plan to investigate with a wider range of CPPS engineering organizations to what extent I4ART knowledge can be extracted from CPPS engineering tool data and data logistics approaches [23] to validate the findings in this paper and to facilitate automating trace link elicitation.

Elicitation of implicit knowledge. An I40 Asset can be the starting point for exploring its technical links or dependencies to neighboring concepts. Thus, I40 Assets provide the basis to elicit and represent implicit expert knowledge and local knowledge representations, e.g., issue/comment texts.

Reuse of robot cell trace information in and across projects. I4ART facilitates reuse in CPPS engineering based on trace information. As robot cells share a similar structure, it is reasonable to assume the reuse of a cell design with adaptations to cells with similar requirements in or across projects. Reusing trace information knowledge could make the elicitation and validation of trace links more efficient.

Validation Methods based on the I4ART. We plan to investigate detecting defect in early CPPS engineering, e.g., forgotten requirements that were not realized. Efficient validation after requirement or resource changes is likely to shorten CPPS engineering duration. While I4ART focuses on CPPS structure, we will consider a behavioral models, e.g., timed automata, to identify basic time-related defects before simulation.

Security. Aggregating domain knowledge in an *Industry 4.0 Asset based Requirements Tracing* network creates high-value assets. These assets require research on security concerns, e.g., intellectual property theft or using I4ART knowledge for

security attacks on I40 Assets in critical infrastructure, e.g., a large CPPS destabilizing a national power grid.

REFERENCES

[1] O. C. Gotel and C. Finkelstein, "An analysis of the requirements traceability problem," in *Proceedings of IEEE International Conference on Requirements Engineering*. IEEE, 1994, pp. 94–101.

[2] O. Gotel, J. Cleland-Huang, J. H. Hayes, A. Zisman, A. Egyed, P. Grünbacher, A. Dekhtyar, G. Antoniol, J. Maletic, and P. Mäder, "Traceability fundamentals," in *Software and systems traceability*. Springer, 2012, pp. 3–22.

[3] J.-S. Lee, V. Katta, E.-K. Jee, and C. Raspotnig, "Means-ends and whole-part traceability analysis of safety requirements," *Journal of Systems and Software*, vol. 83, no. 9, pp. 1612–1621, 2010.

[4] O. Gotel, J. Cleland-Huang, J. H. Hayes, A. Zisman, A. Egyed, P. Grünbacher, and G. Antoniol, "The quest for ubiquity: A roadmap for software and systems traceability research," in *2012 20th IEEE international requirements engineering conference (RE)*. IEEE, 2012, pp. 71–80.

[5] J. Cleland-Huang, O. C. Gotel, J. Huffman Hayes, P. Mäder, and A. Zisman, "Software traceability: trends and future directions," in *Future of Software Engineering Proceedings*, 2014, pp. 55–69.

[6] B. Vogel-Heuser, T. Bauernhansl, and M. Ten Hompel, "Handbuch Industrie 4.0 Bd. 4," *Allgemeine Grundlagen*, vol. 2, 2020.

[7] R. Mehr and A. Lüder, *Managing Complexity Within the Engineering of Product and Production Systems*. Cham: Springer International Publishing, 2019, pp. 57–79.

[8] J. Herzog, H. Röpke, and A. Lüder, "Allocation of PPRS for the plant planning in the final automotive assembly," in *ETFA*. IEEE, 2020, pp. 813–820.

[9] M. Hankel and B. Rexroth, "The reference architectural model industrie 4.0 (rami 4.0)," *ZVEI*, vol. 2, no. 2, p. 4, 2015.

[10] R. Heidel, M. Hankel, U. Döbrich, and M. Hoffmeister, *Basiswissen RAMI 4.0: Referenzarchitekturmodell und Industrie 4.0-Komponente Industrie 4.0*. Beuth Verlag, 2017.

[11] J. Siegert, T. Schlegel, L. Zarco, B. Miljanovic, A. Meyke, and T. Bauernhansl, "Ultra-flexible factories: An approach to manage complexity," *Procedia CIRP*, vol. 93, pp. 329–334, 2020.

[12] J. Pfrommer, M. Schleipen, and J. Beyerer, "PPRS: Production skills and their relation to product, process, and resource," in *18th IEEE ETFA 2013, Cagliari, Italy, September 10-13, 2013*. IEEE, 2013, pp. 1–4.

[13] Plattform Industrie 4.0 and ZVEI, " Part 1 - The exchange of information between partners in the value chain of Industrie 4.0 (Version 3.0RC01 Review)," German BMWI, Standard, Nov. 2020, https://bit.ly/37A002I.

[14] M. Sanchez, E. Exposito, and J. Aguilar, "Industry 4.0: survey from a system integration perspective," *International Journal of Computer Integrated Manufacturing*, vol. 33, no. 10-11, pp. 1017–1041, 2020. [Online]. Available: https://doi.org/10.1080/0951192X.2020.1775295

[15] A. P. Schnicke F., Kuhn T., "Enabling industry 4.0 service-oriented architecture through digital twins," in *Software Architecture. ECSA 2020, Communications in Computer and Information Science, vol 1269.*, M. H. et al. (eds), Ed. Cham: Springer International Publishing, 2020.

[16] M. Pisarić, V. Dimitrieski, M. Vještica, and G. Krajoski, "Towards a non-disruptive system for dynamic orchestration of the shop floor," in *Advances in Production Management Systems. Towards Smart and Digital Manufacturing*, B. Lalic, V. Majstorovic, U. Marjanovic, G. von Cieminski, and D. Romero, Eds. Cham: Springer International Publishing, 2020, pp. 469–476.

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

[18] B. Böhm, M. Zeller, J. Vollmar, S. Weiß, K. Höfig, V. Malik, S. Unverdorben, and S. Hildebrandt, "Challenges in the engineering of adaptable and flexible industrial factories," in *Workshops at Modellierung, Modellierung in der Entwicklung von kollaborativen eingebetteten Systemen (MEKES)*, 2018, pp. 101–111, http://ceur-ws.org/Vol-2060/mekes7.pdf.

[19] S. Maro, J.-P. Steghöfer, and M. Staron, "Software traceability in the automotive domain: Challenges and solutions," *Journal of Systems and Software*, vol. 141, pp. 85–110, 2018.

[20] R. Wohlrab, E. Knauss, J.-P. Steghöfer, S. Maro, A. Anjorin, and P. Pelliccione, "Collaborative traceability management: a multiple case study from the perspectives of organization, process, and culture," *Requirements Engineering*, vol. 25, no. 1, pp. 21–45, 2020.

[21] K. Meixner, A. Lüder, J. Herzog, H. Röpke, and S. Biffl, "Modeling expert knowledge for optimal CPPS resource selection for a product portfolio," in *ETFA*. IEEE, 2020, pp. 1687–1694.

[22] IEC, "IEC 62714:2014 Engineering data exchange format for use in industrial automation systems engineering - Automation markup language," www.iec.ch, 2014.

[23] S. Biffl, A. Lüder, F. Rinker, and L. Waltersdorfer, "Efficient engineering data exchange in multi-disciplinary systems engineering," in *CAiSE*, ser. Lecture Notes in Computer Science, vol. 11483. Springer, 2019, pp. 17–31.

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

[25] E. Engström, M.-A. Storey, P. Runeson, M. Höst, and M. T. Baldassarre, "How software engineering research aligns with design science: a review," *Empirical Software Eng.*, vol. 25, no. 4, pp. 2630–2660, 2020.

[26] N. M. F. Mustafa, "Traceability modeling for the engineering of heterogeneous systems," Ph.D. dissertation, Carleton University, 2019.

[27] D. blinded Authors, "Requirements Tracing in Cyber-Physical Robot Cells for Positioning and Joining (Case Study)," Institute, Tech. Rep. 1, Mar. 2021.

[28] B. Vogel-Heuser, M. Böhm, F. Brodeck, K. Kugler, S. Maasen, D. Pantförder, M. Zou, J. Buchholz, H. Bauer, F. Brandl *et al.*, "Interdisciplinary engineering of cyber-physical production systems: highlighting the benefits of a combined interdisciplinary modelling approach on the basis of an industrial case," *Design Science*, vol. 6, 2020.

[29] 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," in *2020 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*. IEEE, 2020, pp. 133–140.

[30] N. Jazdi, C. R. Maga, P. Göhner, T. Ehben, T. Tetzner, and U. Löwen, "Mehr systematik für den anlagenbau und das industrielle lösungsgeschäft - gesteigerte effizienz durch domain engineering (improved systematisation in plant engineering and industrial solutions business - increased efficiency through domain engineering)," *Automatisierungstechnik*, vol. 58, no. 9, pp. 524–532, 2010.

[31] A. Lüder, N. Schmidt, K. Hell, H. Röpke, and J. Zawisza, *Fundamentals of Artifact Reuse in CPPS*. Cham: Springer International Publishing, 2017, pp. 113–138.

[32] K. Hell and A. Lüder, "Wiederverwendung im engineering," *Zeitschrift für wirtschaftlichen Fabrikbetrieb*, vol. 111, no. 6, pp. 337–341, 2016. [Online]. Available: https://doi.org/10.3139/104.111531

[33] H. Röpke, "Entwicklung einer methode zur risikobeurteilung bei der wiederverwendung von entwurfselementen im anlagenengineering," Ph.D. dissertation, Otto-von-Guericke-Universität Magdeburg, Fakultät für Maschinenbau, 2019. [Online]. Available: http://dx.doi.org/10.25673/25403

[34] B. Vogel-Heuser, C. Diedrich, A. Fay, S. Jeschke, S. Kowalewski, M. Wollschlaeger *et al.*, "Challenges for software engineering in automation," *Journal of Software Engineering and Applications*, vol. 2014, 2014.

[35] *VDI Richtlinie 3695: Engineering von Anlagen – Evaluieren und optimieren des Engineerings*, Beuth Verlag Std., 2009.

[36] Q. Li, Q. Tang, I. Chan, H. Wei, Y. Pu, H. Jiang, J. Li, and J. Zhou, "Smart manufacturing standardization: Architectures, reference models and standards framework," *Computers in Industry*, vol. 101, pp. 91–106, 2018.

[37] S. Unverdorben, B. Böhm, and A. Lüder, "Reference architectures for future production systems in the field of discrete manufacturing," in *2018 IEEE 14th International Conference on Automation Science and Engineering (CASE)*, 2018, pp. 869–874.

[38] ——, "Concept for Deriving System Architectures from Reference Architectures," in *2019 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM)*, 2019, pp. 19–23.

[39] S. MirRashed, M. R. Mehr, M. Mißler-Behr, and A. Lüder, "Analyzing the causes and effects of complexity on different levels of automobile manufacturing systems," in *2016 21st IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA)*, 2016, pp. 1–4.

[40] K. Meixner, J. Decker, H. Marcher, A. Lüder, and S. Biffl, "Towards a domain-specific language for product-process-resource constraints," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, vol. 1, 2020, pp. 1405–1408.

[41] R. Drath, M. Barth, and A. Fay, "Offenheitsmetrik für engineering-werkzeuge," *atp edition*, vol. 54, no. 09, pp. 46–55, 2012.

[42] A. Strahilov and H. Hämmerle, "Engineering Workflow and Software Tool Chains of Automated Production Systems," in *Multi-Disciplinary Engineering for Cyber-Physical Production Systems*. Springer, 2017.

[43] A. Schumacher, T. Nemeth, and W. Sihn, "Roadmapping towards industrial digitalization based on an industry 4.0 maturity model for manufacturing enterprises," *Procedia Cirp*, vol. 79, pp. 409–414, 2019.

[44] R. Drath, M. Rentschler, and M. Hoffmeister, "The automationml component description in the context of the asset administration shell," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 1278–1281.

[45] J. Pfrommer, D. Stogl, K. Aleksandrov, V. Schubert, and B. Hein, "Modelling and orchestration of service-based manufacturing systems via skills," in *Proceedings of the 2014 IEEE Emerging Technology and Factory Automation (ETFA)*, 2014, pp. 1–4.

[46] P. Bihani, R. Drath, and A. Kadam, "Towards meaningful interoperability for heterogeneous engineering tools via automationml," in *2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*, 2019, pp. 1286–1290.

[47] A. Lüder, A. Behnert, F. Rinker, and S. Biffl, "Generating industry 4.0 asset administration shells with data from engineering data logistics," in *2020 25th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2020, pp. 867–874.

[48] H. Paulheim, "Knowledge graph refinement: A survey of approaches and evaluation methods," *Semantic web*, vol. 8, no. 3, pp. 489–508, 2017.

[49] R. Wohlrab, "Living boundary objects to support agile inter-team coordination at scale," Ph.D. dissertation, Chalmers University of Technology, 2020.

[50] R. Sinha, B. Dowdeswell, G. Zhabelova, and V. Vyatkin, "Torus: Scalable requirements traceability for large-scale cyber-physical systems," *ACM Transactions on Cyber-Physical Systems*, vol. 3, no. 2, pp. 1–25, 2018.

[51] N. Mustafa and Y. Labiche, "The need for traceability in heterogeneous systems: a systematic literature review," in *2017 IEEE 41st Annual Computer Software and Applications Conference (COMPSAC)*, vol. 1. IEEE, 2017, pp. 305–310.

[52] H. Dubois, M.-A. Peraldi-Frati, and F. Lakhal, "A model for requirements traceability in a heterogeneous model-based design process: Application to automotive embedded systems," in *2010 15th IEEE International Conference on Engineering of Complex Computer Systems*. IEEE, 2010, pp. 233–242.

[53] S. Nejati, M. Sabetzadeh, D. Falessi, L. Briand, and T. Coq, "A sysml-based approach to traceability management and design slicing in support of safety certification: Framework, tool support, and case studies," *Information and Software Technology*, vol. 54, no. 6, pp. 569–590, 2012.

[54] T. W. W. Aung, H. Huo, and Y. Sui, "A literature review of automatic traceability links recovery for software change impact analysis," in *Proceedings of the 28th International Conference on Program Comprehension*, 2020, pp. 14–24.

[55] D. Kolovos, R. Paige, and F. Polack, "Detecting and repairing inconsistencies across heterogeneous models," in *2008 1st International Conference on Software Testing, Verification, and Validation*. IEEE, 2008, pp. 356–364.

[56] R. Torkar, T. Gorschek, R. Feldt, M. Svahnberg, U. A. Raja, and K. Kamran, "Requirements traceability: a systematic review and industry case study," *International Journal of Software Engineering and Knowledge Engineering*, vol. 22, no. 03, pp. 385–433, 2012.

[57] E. Bouillon, P. Mäder, and I. Philippow, "A survey on usage scenarios for requirements traceability in practice," in *International Working Conference on Requirements Engineering: Foundation for Software Quality*. Springer, 2013, pp. 158–173.

[58] J. Cleland-Huang, P. Mäder, M. Mirakhorli, and S. Amornborvornwong, "Breaking the big-bang practice of traceability: Pushing timely trace recommendations to project stakeholders," in *2012 20th IEEE International Requirements Engineering Conference (RE)*. IEEE, 2012, pp. 231–240.

[59] J. Guo, N. Monaikul, and J. Cleland-Huang, "Trace links explained: An automated approach for generating rationales," in *2015 IEEE 23rd International Requirements Engineering Conference (RE)*. IEEE, 2015, pp. 202–207.

[60] N. Mustafa and Y. Labiche, "Using semantic web to establish traceability links between heterogeneous artifacts," in *International Conference on Software Technologies*. Springer, 2017, pp. 91–113.

[61] A. K. Chopra, F. Dalpiaz, F. B. Aydemir, P. Giorgini, J. Mylopoulos, and M. P. Singh, "Protos: Foundations for engineering innovative sociotechnical systems," in *2014 IEEE 22nd International Requirements Engineering Conference (RE)*. IEEE, 2014, pp. 53–62.

[62] *DIN 8593-0 Manufacturing processes joining - Part 0: General; Classification, subdivision, terms and definitions*, Beuth Verlag Std., 2003.

[63] D. blinded Authors, "Multi-Aspect Risk Exploration in Models for Positioning and Joining Simulation (Case Study) Part II," Institute, Tech. Rep. 1, Jan. 2021.

[64] *VDI Guideline 3695: Engineering of industrial plants - Evaluation and optimization*, Beuth Verlag Std., 2009.