Requirements Management for Flow Production of Precast Concrete Modules

Simon Kosse, Oliver Vogt, Mario Wolf, Markus König, and Detlef Gerhard

Abstract For efficient and sustainable use of precast concrete modules, all relevant information must be collected digitally and in real-time and made available in digital twins. Digitization should be carried out in such a way that continuous quality management is possible at all times. This also includes whether the produced concrete modules also meet all the requirements from the initial design. For example, the precast concrete parts must be able to absorb certain forces or have precise connections and joining options. The Requirements Interchange Format (ReqIF) can be used to describe requirements digitally and exchange them between different IT systems and stakeholders. The creation of automated quality control (QC) protocol for the flow production process can be implemented based on this already structured and formalized requirements format. In this paper, the Asset Administration Shell (AAS) from the context of Industry 4.0 is enhanced to enable the formal description and automated verification of requirements for precast concrete based on the ReqIF interchange format. For this purpose, a smart service integrates the ReqIF-compliant requirements into an AAS submodel. Via this smart service, a mapping assistance tool lets stakeholders assign measurable properties of the precast concrete modules to the requirements, thus enabling an automated quality check. The presented approach is validated based on a virtual precast concrete wall for which a chain of linked requirements is described and automatically checked within the scope of a case study.

Keywords Industry 4.0 · Digital Twin · Precast Concrete · Requirements · Rule-based Checking

S. Kosse (✉) · M. König
Chair of Computing in Engineering, Ruhr University, Bochum, Germany
e-mail: simon.kosse@ruhr-uni-bochum.de

O. Vogt · M. Wolf · D. Gerhard
Digital Engineering Chair, Ruhr University, Bochum, Germany

S. Skatulla and H. Beushausen (eds.), *Advances in Information Technology in Civil and Building Engineering*, Lecture Notes in Civil Engineering 357,
https://doi.org/10.1007/978-3-031-35399-4_48

1 Introduction

Industry 4.0 refers to using Internet of Things (IoT) technologies to automate the manufacturing industry. Products, machines, and processes are interconnected to form decentralized cyber-physical systems within which data and information are exchanged for self-controlled and self-managed production. These systems are made possible by the use of modern information and communication technologies. Initial applications to the production of precast concrete modules show the potential for significantly increasing efficiency through end-to-end digitized and automated production of precast concrete modules in modern flow production processes. In terms of precast construction, building structures are segmented into modules. As part of this segmentation, modules must fulfill certain functions and requirements, such as the transfer of specific forces, compliance with tolerances, or fulfillment of building physical properties such as fire resistance, thermal insulation, and sound insulation. Requirements also exist regarding the components' connections, dimensions, and weight. Concepts for automated testing must be developed to verify the requirements imposed on the components.

A prerequisite for implementing cyber-physical systems according to the I4.0 model is the digital twin, which manages and provides data and information in a centralized manner for each asset. It contains all data and semantic information relevant to an asset throughout its life cycle. Intelligent and consistent requirements management is essential for smart products as well as smart production systems. Faults or deviations from requirements must be detected and corrected early to guarantee the quality of the end product. The digital twin should not only have information about its state but also be able to represent and check requirements. A prerequisite for automated testing is the formal description of requirements, which can be functional or quality requirements. The requirements brought into the digital twin of a concrete module must be complete, unambiguous, consistent, and verifiable.

This paper presents an approach for formally describing requirements for precast concrete modules in the context of industrialized manufacturing based on the Requirements Interchange Format (ReqIF). The ReqIF format is an exchange format based on the XML schema that enables the formal description of requirements. The requirements described in ReqIF are integrated into the AAS, which is the technical implementation of the digital twin. A submodel extends the AAS for capturing requirements described in ReqIF format and mapping of the requirements to properties and characteristics in the AAS. Verification of the requirements is performed using an external service. The following research questions are answered:

- How can requirements be formally described and integrated into the digital twin?
- How can the AAS be extended to facilitate the automated verification of requirements?
- How can the automated verification of requirements be integrated into the industrialized production of precast concrete modules?

2 Related Work

2.1 Industry 4.0, Digital Twin, and the Asset Administration Shell

Industry 4.0 refers to the digitalization and automation of the manufacturing industry by networking products, machines, and processes into cyber-physical systems. Real and virtual space are linked through innovative communication and interaction technologies, enabling industrialized production in which intelligent products independently seek an optimal path through the production system, interacting and communicating with machines and processes. For the implementation of Industry 4.0, the digital twin plays an essential role as a virtual representation for collecting, evaluating, and providing data and information.

Since introducing the concept of the digital twin as the conceptual ideal of Product Lifecycle Management (PLM) [1], the concept has been taken up by many different industries, each developing its definition. As a result, no universal conceptualization of the digital twin exists [2]. However, in most definitions, the digital twin is described as consisting of a physical and virtual representation and a data link that connects the two. A key issue in construction deals with the role of Building Information Modeling (BIM) in the context of the digital twin. In their paper, Davila Delgado and Oyedele [3] state that both concepts are unique approaches for different industry requirements in each case. While BIM has been designed to increase efficiency in design and construction, the digital twin has been designed to represent the current state during manufacturing operations through real-time data that enables data-driven decision-making and simulation of what-if scenarios [3–5].

The AAS corresponds to the technical implementation of the digital twin in I4.0. It is composed of many individual submodels to form an overall digital representation. Each submodel represents an aspect or use case. The submodels have all the relevant data and information required to represent the corresponding use case. The actual contents of the submodels can include attributes and properties in the form of key values and references to external files. All data have concept descriptions that uniquely specify the data ensuring machine readability and thus enabling automated data exchange between AAS and machines. File formats include serializations in.xml and. json. In addition, a package format (.aasx) is available that has all other contents and references in addition to serializations in.xml or. json. The AAS can be used as a passive AAS for interoperable data exchange or as a passive or active server component. The AAS data can be accessed via standard REST query, OPC UA or MQTT [6]. A special I4.0 language is being developed for autonomous communication and interaction between AASs [7].

2.2 Requirements Management in Construction

Requirements management has its origins in systems and software engineering. Here, it is a branch of software technology that deals with the objectives, functions, and constraints of software systems, as well as the specifications of software behavior and their evolvement over time [8]. The main activities of requirements management include eliciting, modeling and analyzing, communicating, agreeing, and developing requirements [9]. According to Fernie et al. [10], transferring requirements management to the construction industry helps comply with costing and time schedules, view project decisions from a life-cycle perspective, and improve overall customer satisfaction. Several types of requirements exist in the construction industry. Customer requirements combine with site, environmental, and regulatory requirements to form design and construction requirements [11]. The goal of requirements management in construction is to fulfill all needs and requirements of all stakeholders involved in the construction of a building [12]. In the construction industry, requirements management is often not considered over the entire life cycle of a building but is increasingly applied in the early planning phases. Requirements are assessed once, not updated, or made available in a central repository for later lifecycle phases and use [13]. In their work, Jallow et al. [14] present a platform for managing requirements across all life phases of a building project. For the platform, they follow an information-centric, process- and service-oriented approach. Compliance with tolerance requirements is integral to requirements management, especially for precast concrete structures. Talebi et al. [15] report that the construction industry lacks a systematic practical process to avoid tolerance problems that often occur during assembly on construction sites. Their study reviews existing guidance in the literature and divide tolerance management into the stages of identification, planning, communication, and control. They present a holistic conceptual framework that integrates all stages of tolerance management and aims to improve conventional tolerance management. Rausch et al. [16] developed a domain model for tolerance management that consolidates scattered knowledge on tolerance management into a standardized uniform framework. The tolerance knowledge is formalized to make it unambiguously interpretable by software systems enabling automated tolerance management. Their approach is based on an initial BIM and an analysis of key tolerance relationships through a sequence of inference rules to create a semantically rich BIM with tolerance management concepts. The enriched BIM can support effective tolerance management through analysis and simulation.

2.3 Requirements Interchange Format - ReqIF

The Requirements Interchange Format (ReqIF) is an open, non-proprietary exchange format with which requirements can be exchanged in a formalized and interoperable manner [17]. The metamodel of ReqIF is based on XML schemata and contain the requirement meta data and content in the form of attributed data elements and other associated data such as documents and images. The XML schema consists of a root element consisting of a header and the requirement data. The requirement data consists of the data elements SpecObjects, by which the requirements are represented, Specification, that a hierarchical structure of SpecObjects represents, SpecRelations, that represents links between SpecObjects as well as SpecType, which specifies which data types an element can have. Further information can be found in the documentation of the exchange format. For further information please refer to the official documentation [18].

3 Concept

Requirements of various categories are relevant to concrete modules during and after modularization and for prefabricated construction. This is true especially for structural requirements in the form of minimum load-bearing capacities and dimensional accuracy, as well as functional requirements regarding heat, moisture, and fire protection. In addition, industrialized production imposes requirements for each process step that must be checked either during or after a process step. These can include, for example, minimum concrete strengths for stripping the modules of their formwork or temperature ranges to be maintained during heat treatment of the modules. The technical requirements for construction are distributed over several standards and guidelines, such as DIN EN 13,369 [19], which specifies general rules for precast concrete components, or product standards that specify component-specific requirements, such as DIN EN 14,992 for wall elements [20]. For quality-assured production in the sense of I4.0, requirements must be integrated into the digital twin of the modules to enable automatic verification and ultimately informed decision-making for the humans in the production process.

Figure 1 shows the schematic representation of the concept for integrating requirements into the AAS. Initially, the requirements are created, maintained and distributed across the various planning phases, from initial planning, through the preliminary, draft, and approval planning, to detailed design. Through these planning processes, the requirements for individual building components evolve based on customer requirements in conjunction with requirements for the construction site, environment, and regulatory requirements into design and, finally, construction engineering requirements for the individual building components. During the planning phases, the digital representation of a concrete module exists only as a type AAS, which already has the required data structure but does not yet contain any instance-specific

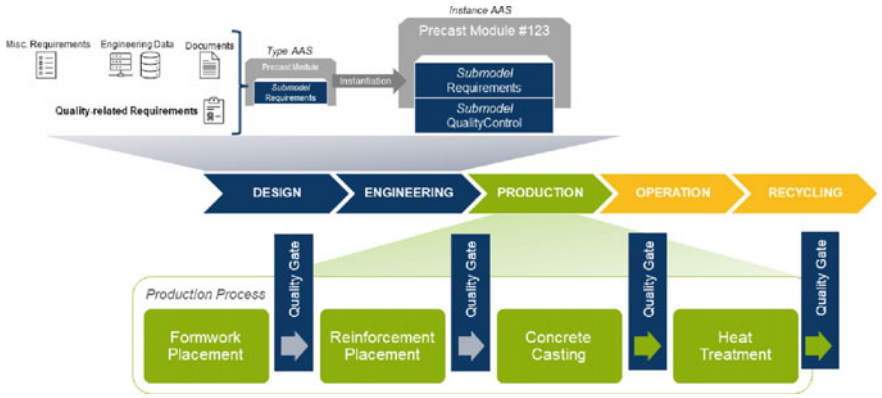

Fig. 1 Concept for the integration of in ReqIF formalized requirements into the AAS

values, as there is no physical counterpart yet. The instance AAS of the respective concrete module is created only by transferring the planning data into the AAS at the end of the detailed design phase. It is assumed that the requirements determined during the planning phases are available in a digital building model.

These requirements can be defined in different software tools, which can result in them having different structures and formats. This means that simple integration into the digital twin is not possible. Therefore, an interoperable data format is necessary that stores requirements in a structured manner independent of the software platform. For this purpose, the ReqIF data exchange format, known from mechanical engineering and based on an XML data structure, is ideally suited.

Since ReqIF is only an exchange format, certain defined attributes must still be specified for the requirements description so that the requirements have a uniform structure. A template is developed for this purpose in this paper. Based on this template, the requirements can be exported as an ReqIF-File and then be integrated into the digital twin. To transfer the requirements data to the AAS, the requirements are exported to a ReqIF file and send to the'AAS' import service.

For this purpose, an AAS submodel definition for managing the requirements is developed. To make the requirements automatically testable, it is also necessary that a mapping to measurable properties is stored in this submodel. Likewise, rules or conditions must be stored for the mapped properties, which must be met for the condition to be fulfilled. The submodel contains a data structure for storing the requirements and their metadata, as well as information for mapping the requirements to other properties and characteristics in the AAS, e.g., data stored in the QualityControl submodel as part of quality control.

Requirements for concrete modules differ with respect to the module type. This enables the predefined requirements to be mapped to the data contained in the AAS enabling fast integration and automated checking of these requirements. At the end of the detailed design, the digital twin of a precast concrete module is created that has all relevant data for production, including production and quality requirements.

During production, services can continuously check the requirements, e.g., against collected quality data.

The work is based on the template of an AAS for general precast concrete modules, which was developed as part of the authors' extensive preliminary work [21]. The template is extended by a submodel for capturing and mapping requirements to properties and features in the AAS.

In the following, an attribute schema is proposed for a ReqIF template. Attribute schemas are not standardized and must be specified separately for each use case, similar to submodel definitions in AASs. In addition, the requirements submodel developed within the scope of this work is described concerning contents and intended use cases.

3.1 ReqIF Template for Requirements of Precast Concrete Modules

Essential for the development of ReqIF templates for the formalization of precast concrete requirements is the definition of an attribute schema. Attributes add important semantic information to requirements that, among other things, allow requirements and their intentions to be defined, maintained, and verified. Attribute schemas are not standardized and must be defined for the particular application. Different sources in literature recommend structures for such attribute schemas, e.g., [22]. In general, adding attributes to the schema is only recommended if a clear added value or use case exists through the attribute so that the schema is not cluttered with unnecessary information and eventually becomes confusing. Besides standard attributes like GUID, name and description, some use case specific attributes were selected (cf. [22]):

VerificationMethod: Specifies the recommended method for checking the requirements. Verification of tolerance requirements can be performed, for example, by a 3D point cloud acquired by laser scanning and a nominal/actual comparison.

VerificationPhase: Time or life phase in which the requirement is checked. For example, the time for checking a minimum compressive strength for stripping could be specified here. The attribute is particularly essential for coordination with regard to automated checking.

VerificationResults: This attribute specifies the result of the verification of the requirement. Document verification results are summarized by a short string, e.g., successful, or unsuccessful. Some requirements required verification of multiple properties and characteristics.

VerificationStatus: Status of the review of the requirement. The attribute summarizes the result of all performed checks.

ComparisonOperator: This attribute specifies a comparison operator that is used for automated requirements testing by an external service as part of quality assurance.

Mapping: Assignment of the requirement to the corresponding properties in the AAS.

LimitValue: Defines the outermost value that must not be exceeded or fallen short of in order to comply with the requirement.

3.2 Submodel Requirements

Submodels are elementary components of administration shells that contain all essential data in a context-specific manner and provide the administration shell with functionality. Standardization of these submodels is sensible in order to maintain interoperability and adaptability. Some basic submodels, such as those for identifying, documenting, and representing technical data of assets, have already been standardized by the Standardization Council Industrie 4.0. For specific use cases, submodels and their data contents must be developed accordingly. For integrating requirements of precast concrete parts into the AAS, a submodel is developed that stores the requirements along with their metadata. It specifies a mapping of the requirements to the data contents of the administration shell as a basis for automated testing. The focus is in particular on:

- integration of requirements for precast concrete modules into the AAS using the ReqIF exchange format,
- mapping of requirements to properties and attributes in the AAS for requirements verification,
- machine-readable provision of requirements for automated compliance checking through an external service.

The requirements are first extracted from a ReqIF file in the intended use case and imported into the submodel. The enhanced semantic description in the AAS by concept descriptions enables the machine readability of requirements data. In conjunction with a reference link to the corresponding data in the AAS, it enables the requirements to be checked in a nominal/actual comparison.

The submodel represents every requirement by a Submodel Element Collection (SMC) containing the attributes specified in the ReqIF file (cf. Figure 2). Some requirements require the verification of multiple properties and attributes. Dimensional accuracy, for example, is a requirement for a precast concrete component that can only be verified by checking various tolerance values. The resulting partial requirements are each stored in a further SMC together with their metadata.

Fig. 2 Template of Submodel Requirements

4 Case Study

For validation, the concept is applied to the single-shell lining segment with conventional steel reinforcement, illustrated in Fig. 3. Linings are manufactured in precast plants with extensive quality assurance measures. The functions to be fulfilled by lining segments comprise essentially the absorption of actions from dead weight, superimposed loads, and sealing against groundwater. In single-shell structures, all structural and design requirements are taken over by the lining segments, which is

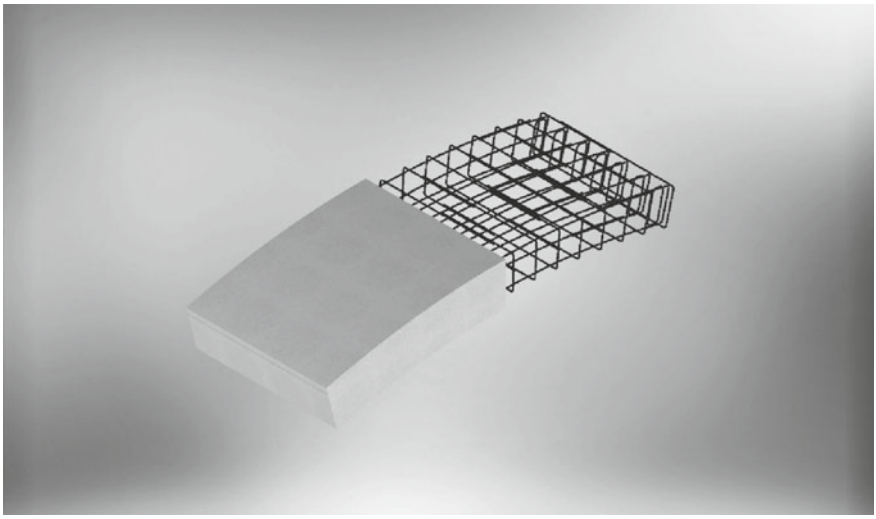

Fig. 3 Tunnel lining segment

why the requirements for single-shell structures are particularly high. The requirements placed on the lining segment in this case study are taken from the DAUB Recommendation for the Design, Manufacture, and Installation of Segments and from ZTV-ING Part 5, Sect. 3 [23, 24]. An type AAS is first created based on the template presented in [21] for the illustrated lining segment. Subsequently, the AAS is extended according to the presented approach by submodels for capturing the requirements formalized in ReqIF and mapping the requirements to the attributes and properties in the submodels of the AAS. The functionality is demonstrated by an external smart service that validates the requirements.

4.1 Integration of Requirements into AAS

Figure 4 shows the the ReqIF studio editor with the content of the ReqIF-file, which has been created for the tunnel lining module based on the attribute scheme presented in Sect. 3.1. The file contains exemplary requirements for the minimum compressive strength for stripping the module of its formwork as well as for the minimum reinforcement and dimensional accuracy based on [24]. Both the minimum reinforcement and dimensional accuracy requirements require the verification of several sub-requirements. Accordingly, a hierarchical arrangement of these requirements has been made. The result of the verification of all sub-requirements is indicated by the VerificationStatus attribute of the superordinate requirement. Figures 5 and 6 show the requirements and quality assurance submodel after successfully integrating the requirements and performing measurements for quality assurance.

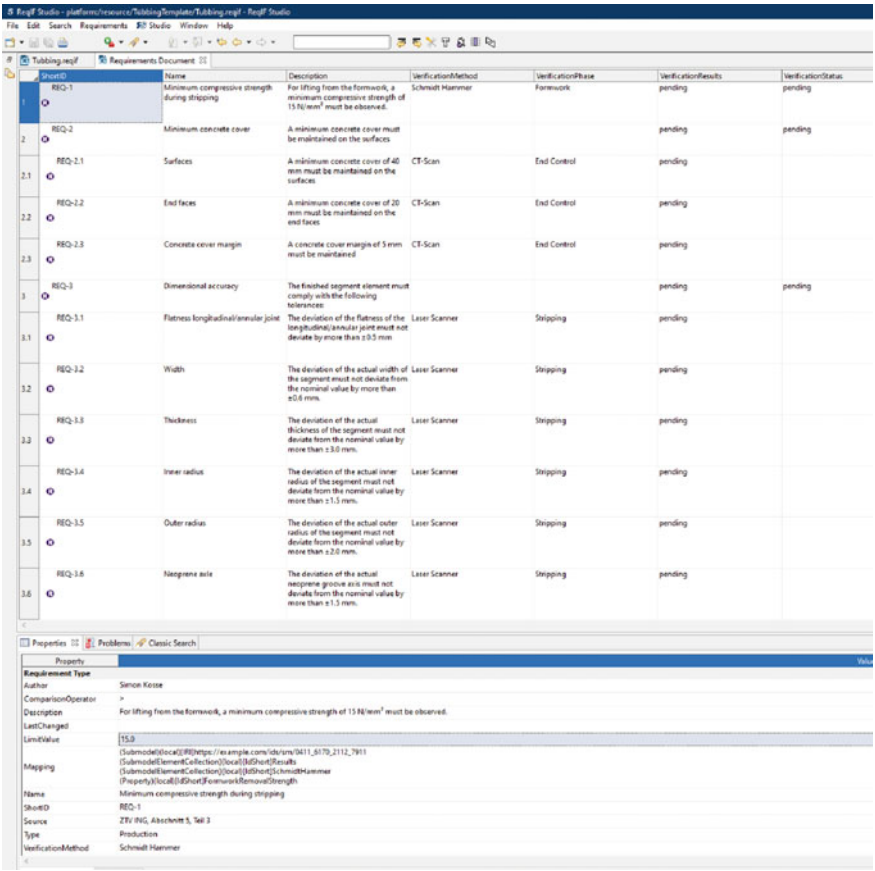

Fig. 4 ReqIF file of tunnel lining module

4.2 Compliance Checking of Requirements

The automated execution of the requirements compliance check is based on the submodel structure presented earlier in this paper. The AAS is made available on a server so that the contents of the AAS can be queried, updated or modified via the Rest API. The server used in this case study is an instance of the open-source AASX server [25]. Initially, the AASX file is loaded by the AASX server's services, making its contents retrievable via a standardized REST API.

Since it is possible to communicate with the server via the Rest API, different implementation options for the requirements checking service are possible. For this case study, a web-service based on Python (respectively PyWebIO) is developed, which communicates directly with the AASX server. The accordingly created user interface can be used to select which asset the automated check should verify. To do this, the service queries all type-matching AAS hosted on the server and provides

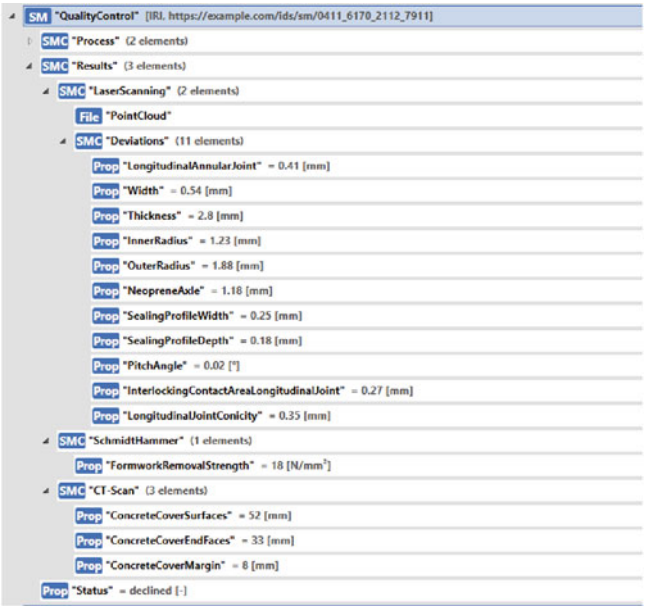

Fig. 5 Submodel Quality Control of tunnel lining module

the user with the results as a drop-down selection. Once the desired asset has been selected, the service queries the contents of the requirements submodel of the selected AAS. The web service presents this content graphically to the user. For this purpose, the requirement as such is displayed with its status as well as each individual condition that is required for the fulfillment of the requirement (see Fig. 7).

At this point, no check has been performed yet. The check is started by clicking the "run requirements check" button. The service now processes the requirements to be checked one after the other. For each requirement, the individual conditions are checked, provided the status of the requirement is not already set to "fulfilled". To do this, the service queries the actual value stored in the AAS for the property to be checked and compares this with the NominalValue specified in the requirements file on the basis of the comparison operator selected. If the condition was met, the status of the condition is set to "fullfilled" and the next condition is checked. If all conditions of a requirement were checked, the status is set to "fulfilled" or "not fulfilled" depending on the status of each condition. The individual statuses and the current value of the property are also displayed to the user in the web interface.

Since these values are stored temporarily in the web service, they must finally be transmitted to the AAS via the Rest API. Thus, the results of the check are stored and can be used for further use cases.

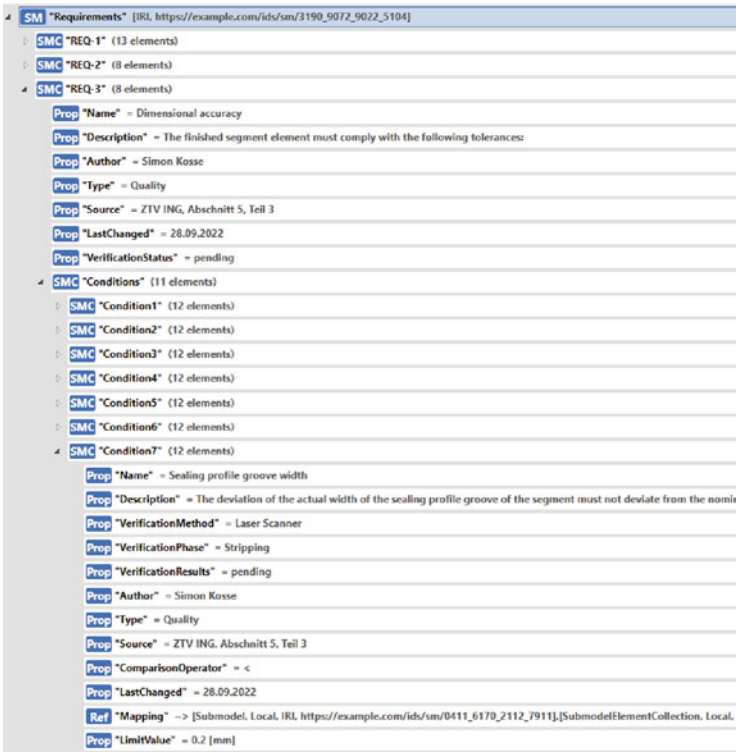

Fig. 6 Requirements submodel of tunnel lining module

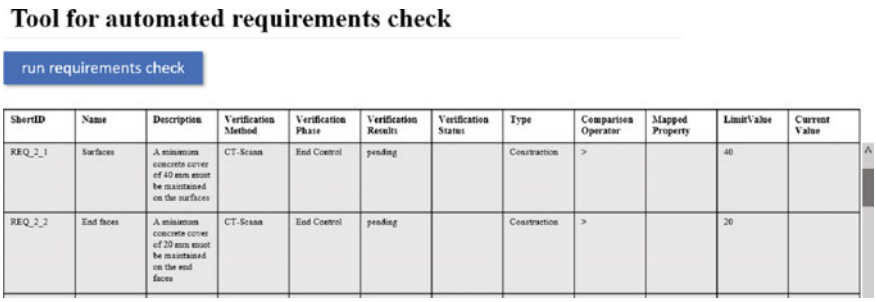

Fig. 7 Tool for automated requirements check

5 Conclusion and Outlook

Precast construction is characterized by the segmentation of building structures into individual modules, which have to fulfill specific requirements and functions. In the context of industrialized production based on the I4.0 model, compliance with

these requirements must be continuously checked in an automated manner in order to ensure the quality of the precast concrete part. A prerequisite for the automated verification of the requirements is their formal description. This paper presents an approach to extending the AAS known from I4.0 by requirements specified in ReqIF format and automated verification. A submodel was developed to capture the formalized requirements, mapping the requirements to the features and properties available in the AAS. An external smart service was developed for automated verification. The results of the checks are the basis for information-based decision-making and are an essential prerequisite for implementing a decentralized and autonomous production system.

This paper shows the first prototypical implementation of automated requirements testing. It has not yet been applied in the context of a fully automated and digitized production system. The approach is limited to the requirements of single modules. The verification of requirements of multiple components is essential in the context of a modularized design.

Future work will address the handling of the results of the verification. The question of how to deal with concrete modules that do not meet the requirements is to be clarified. The handling of rejected modules has a direct impact on production planning. Integration of a real-time simulation of the production system for continuous optimization and control in order to be able to react dynamically to these changes is planned.

For the testing of geometric and material requirements, measurement data are usually used that have a certain degree of uncertainty. Consequently, a test of these requirements does not always provide a discrete result. Concepts of probability logic can be integrated into the presented approach. Finally, all resulting changes regarding human, technology and organization will be investigated further in the coming implementation phases.

References

1. Grieves M (2016). Origins of the Digital Twin Concept
2. Kritzinger W, Karner M, Traar G, Henjes J, Sihn W (2018) Digital twin in manufacturing: a categorical literature review and classification. IFAC-PapersOnLine 51:1016–1022
3. Davila Delgado JM, Oyedele L (2021) Digital twins for the built environment: learning from conceptual and process models in manufacturing. Adv Eng Inform 49:101332
4. Shahzad M, Shafiq MT, Douglas D, Kassem M (2022) Digital twins in built environments: an investigation of the characteristics, applications, and challenges. Buildings 12:120
5. Khajavi SH, Motlagh NH, Jaribion A, Werner LC, Holmstrom J (2019) Digital twin: vision, benefits, boundaries, and creation for buildings. IEEE Access 7:147406–147419
6. Plattform Industrie 4.0. Diskusspapier - Verwaltungsschale in der Praxis (2019)
7. Plattform Industrie 4.0. Diskussionspapier I4.0-Sprache - Vokabular, Nachrichtenstruktur und semantische Interaktionsprotokolle der I4.0-Sprache (2018)
8. Zave P (1995) In: Proceedings of 1995 IEEE international symposium on requirements engineering (RE 1995) (IEEE Comput. Soc. Press), pp 214–216
9. Nuseibeh B, Easterbrook S (2000) In: Proceedings of the conference of the future of software engineering, pp 35–46

10. Fernie S, Green SD, Weller SJ (2003) Dilettantes, discipline and discourse: requirements management for construction. Eng Constr Archit Manag 10:354–367
11. Kamara JM, Anumba CJ, Nosa FO (2002) Evbuomwan. Capturing client requirements in construction projects (Thomas Telford Ltd)
12. Yu ATW, Chan EH (2010) Requirements management in the architecture, engineering and construction (AEC) industry: the way forward in construction. In: Proceedings: W096 - special track 18th cib world building congress, pp 1–11
13. Karim Jallow A, Demian PN, Baldwin A, Anumba C (2014) An empirical study of the complexity of requirements management in construction projects. Eng Constr Archit Manag 21:505–531
14. Jallow AK, Demian P, Anumba CJ, Baldwin AN (2017) An enterprise architecture framework for electronic requirements information management. Int J Inf Manage 37:455–472
15. Talebi S, Koskela L, Tzortzopoulos P, Kagioglou M (2020) Tolerance management in construction: a conceptual framework. Sustainability 12:1039
16. Rausch C, Talebi S, Poshdar M, Li B, Schultz C (2022) Tolerance management domain model for semantic enrichment of BIMs. Autom Constr 141:104394
17. Ebert C, Jastram M (2012) ReqIF: seamless requirements interchange format between business partners. IEEE Softw 29:82–87
18. Object Management Group. Requirements Interchange Format (2016)
19. Deutsches Institut für Normung e.V. DIN EN 13369 Allgemeine Regeln für Betonfertigteile (Beuth Verlag GmbH) (2018)
20. Deutsches Institut für Normung e.V. DIN EN 14992 Betonfertigteile - Wandelemente (Beuth Verlag GmbH) (2012)
21. Kosse S, Vogt O, Wolf M, König M, Gerhard D (2022) Digital twin framework for enabling serial construction. Front Built Environ 8:864722
22. Wheatcraft LS, Ryan MJ, Dick J (2016) On the use of attributes to manage requirements. Syst Eng 19:448–458
23. German Geotechnical Society (2013). Empfehlungen für den Entwurf, die Herstellung und den Einbau von Tübbingringen in Taschenbuch für den Tunnelbau 2014, edited by G. G. Society (Wiley-VCH Verlag GmbH), pp 17–121
24. Bundesministerium für Digitales und Verkehr. Zusätzliche Technische Vertragsbedingungen und Richtlinien für Ingenieurbauten (ZTV-ING) (2022)
25. Admin-Shell-io/Aasx-Package-Explorer. https://github.com/admin-shell-io/aasx-server. Accessed 30 Sep 2022