Available online at www.sciencedirect.com

ScienceDirect

Procedia CIRP 145 (2026) 55–60

www.elsevier.com/locate/procedia

19th CIRP Conference on Computer-Aided Tolerancing (CAT 2026)

Towards Seamless Integration: Embedding Advanced Geometric Tolerance Models in System-level Simulations

Christoph Steinmann[a,*], Jens Schirmer[a], Jens Lienig[a]

[a]*Institute of Electromechanical and Electronic Design, Dresden University of Technology, Helmholtzstr. 18, 01069 Dresden, Germany*

* Corresponding author. Tel.: +49-351-463-32169 ; *E-mail address:* christoph.steinmann@tu-dresden.de

Abstract

Geometric tolerances significantly influence the behavior of technical systems. However, tolerance effects are typically assessed in isolation and only weakly connected to system-level simulations used in model-based systems engineering (MBSE). As a result, tolerance-induced interactions across physical domains often remain invisible until late development stages. To address this gap, this paper proposes a general approach for embedding executable geometric tolerance simulations into system-level models using the Functional Mock-up Interface (FMI). Rather than focusing on data exchange alone, the approach enables tolerance models to actively participate in transient system simulations, allowing their effects to propagate across coupled subsystems. The main contribution is a method-independent concept for Functional Mock-up Units (FMUs) and a corresponding generation workflow that encapsulate complex tolerance simulations behind a standardized interface. A case study demonstrates how tolerance-induced forces can be efficiently integrated into a multi-domain system simulation without exposing internal model complexity.

Peer-review under responsibility of the scientific committee of the 19th CIRP Conference on Computer-Aided Tolerancing (CAT 2026)

Keywords: Geometric Tolerancing; ISO GPS; Functional Mock-up Interface; FMI; FMU; System-level Simulation; MBSE; Surrogate Modeling; Metamodel

1. Introduction

Model-based systems engineering (MBSE) provides a formal, model-centered framework for managing the increasing complexity of multidisciplinary technical systems [3, 20]. As manufacturing tolerances influence system-level performance, reliability, and robustness, it becomes desirable to integrate tolerance analysis directly into system models. In particular, geometric tolerances as defined by geometric product specification standards (ISO GPS) are a key factor for functional fit and product reliability.

At the same time, geometric tolerance simulation methods remain highly heterogeneous, ranging from analytical and kinematic models, through calculations in transformed tolerance spaces, to Monte Carlo and finite element analysis (FEA) based approaches [13, 4]. This methodological diversity is accompanied by a wide range of tools [15], which limits interoperability and hinders the consistent integration of tolerance effects into MBSE development workflows (cf. Figure 1).

Existing standards such as QIF and STEP/AP242 mainly support structured data exchange of tolerance information but

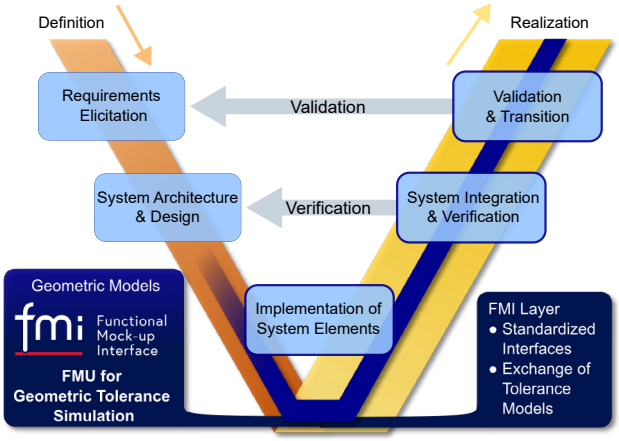

Fig. 1. The V-model presented in [20] outlines the processes for model based product development. The stages at which our approach integrates into that workflow are highlighted in dark blue. By embedding tolerance models into transient system simulations based on FMI, tolerance effects can be accounted for consistently from detailed design through system integration. This allows the impacts to be monitored throughout the development process.

Peer-review under responsibility of the scientific committee of the 19th CIRP Conference on Computer-Aided Tolerancing (CAT 2026)
10.1016/j.procir.2026.03.140

do not provide standardized representations of executable tolerance models [15]. Especially in later phases of the MBSE process with detailed geometry and physics-based system models available, standardized *computational* tolerance models are required to provide simulation results directly at system-level. This enables the consideration of tolerance effects throughout different product development steps as visualized in Figure 1.

Basic tolerance models based on vector-loops have already been encapsulated in Functional Mock-up Units (FMUs) [19]. This paper extends that foundation by addressing the integration of complex geometric tolerance simulations, which remains challenging due to their specialized mathematical formulations and computational workflows.

System-level integration of tolerance effects is essential, as cross-domain dependencies often only become apparent through multi-domain coupling. Following the V-model for the design of mechatronic systems [20], tolerance considerations should therefore be consistently linked from system requirements to detailed design and system validation (Figure 1), for which applicable workflows are required.

Against this background, this paper investigates how complex geometric tolerance simulations can be encapsulated into FMUs to support their use within MBSE workflows. The main contributions of this paper are:

- a method-independent FMU concept for executing geometric tolerance models including *transient behavior*,
- a *generalized workflow* for generating such FMUs from *arbitrary tolerance models*,
- and a *case study* illustrating the approach using an FEA-based tolerance simulation to demonstrate the *geometric influences on system behavior* in an openly provided system model [18].

2. State of the art and conceptual background

The following subsections summarize the conceptual foundations required for integrating geometric tolerance analysis into MBSE workflows.

2.1. Model-based system simulation

In this work, model-based system simulation refers to physics-based, equation-oriented system models rather than descriptive architecture models such as SysML. They are typically formulated in an acausal manner, where connections represent physical interactions rather than predefined signal flow. Modeling languages such as Modelica® or Simscape™ by MathWorks® implement this paradigm using object-oriented, multi-domain components.

The mathematical foundation of such models is a set of index-1 differential–algebraic equations (DAEs),

$$\begin{aligned} \dot{x}(t) &= f(t, x(t), y(t)) \\ 0 &= g(t, x(t), y(t)) \end{aligned} \tag{1}$$

where $\dot{x}$ denotes differential states and g represents algebraic constraints. Simulation tools transform these acausal equations into executable representations via symbolic manipulation, causalization, and numerical integration. Although system models are acausal by construction, solvers impose a computational causality during equation sorting and solving [14]. This distinction becomes relevant when system components are exported as FMUs, which expose a causal input–output interface.

Modelica-based tools such as Dymola®, SimulationX®, and OpenModelica, as well as MATLAB®/Simscape™, provide modeling environments, code generation, and numerical solvers tailored to these DAE-based system descriptions. Such transient simulations can be used to investigate the dynamic behavior and optimize technical multi-domain systems. They thus form a basis for integrating tolerance effects into MBSE workflows.

2.2. Tolerance modeling approaches

A wide range of tolerance modeling approaches has been proposed to describe and analyze the effect of geometric deviations in mechanical assemblies [4, 6]. These approaches differ substantially in mathematical formulation [7], modeling assumptions, and suitability for integration into physics-based system simulations.

DAE-compatible approaches: The vector loop method [5] formulates tolerance accumulation as algebraic relations over closed kinematic chains. Matrix-based TTRS models [9] encode small translational and rotational deviations using homogeneous transformation matrices, while the Jacobian–torsor approach [8] maps torsor deviations to functional measures via linear Jacobian relations. These approaches can be expressed as algebraic constraints and are therefore well suited for DAE-based system simulation (Equation 1).

Restricted or indirectly compatible approaches: Parametric variation models introduce tolerances as explicit parameter intervals [11] and can be directly integrated into equation-based modeling environments. However, they need to solve contact problems, which is usually done by specific numeric geometry solvers. Polytope models [4, 12] represent tolerance zones as bounded convex sets and use set-based operations for tolerance stack-up analysis. While powerful for worst-case verification, these formulations rely on global set operations (e.g. Minkowski sums) rather than local state equations, and therefore do not naturally align with time-continuous DAE systems. The skin model shape concept [17] represents tolerances as discrete surface deviations and is used for virtual assembly, contact analysis, and inspection-oriented simulations. Due to its inherently high-dimensional and contact-driven formulation, such methods typically require preprocessing, numerical contact solvers, or surrogate models before they can be coupled to time-continuous system-level simulations.

2.3. FMI standard and FMUs

The Functional Mock-up Interface (FMI) is a tool-independent standard for exchanging dynamic models across heterogeneous simulation environments [2]. Although originating in equation-based system simulation, FMI is not restricted to this type of model; its purpose is the modular exchange of be-

havioral models in a unified container format. An FMU is such a container, packaged as a ZIP archive comprising an XML description of variables, interfaces, and static metadata as well as a compiled executable implementation of the model (Figure 2). FMI thereby separates a well-defined interface from the internal model representation.

FMI supports two primary interaction modes: *Model Exchange*, in which the importing tool provides the numerical solver and integrates the DAE system, and *Co-Simulation*, where the FMU encapsulates its own solver and interacts through discrete input–output exchanges at communication points. With FMI 3.0 a third mode for scheduled execution was introduced, which is similar to model exchange. FMUs are inherently causal artifacts: they map inputs to outputs. Consequently, FMUs behave as computational components rather than structural elements in a larger system simulation equation system.

The widespread adoption of FMI can be attributed to its standardized mechanism for packaging and exchanging models of dynamic system behavior, enabling interoperability, integration across tools, reproducible Co-Simulation results, and systematic deployment.

2.4. Extensions of FMI

The FMI ecosystem is extended through layered standards that add optional, domain-specific capabilities while preserving a simple base standard. These extensions follow the mechanism defined in FMI, in which additional metadata or artifacts are embedded in dedicated subfolders of the FMU without altering the base interface. A relevant example for geometric tolerances is the Common Standard for Uncertainty Quantification, currently under development [1], which introduces a unified schema for uncertain parameters, probability distributions, sampling procedures, and statistical outputs. This enables machine-interpretable uncertainty propagation across FMI-based workflows.

Complementary work includes the emerging *fmi-ls-ref* specification [10], which provides structured references to external specification artifacts such as STEP or QIF files and supports traceability in MBSE contexts.

3. Concept of FMUs for geometric tolerances

The diversity of geometric tolerance modeling approaches (subsection 2.2), their different assumptions, intended applications and tool availability limit model reuse across MBSE workflows. Prior studies identified method complexity as one factor hindering the broader industrial adoption of tolerance simulation [21]. FMI provides a practical mechanism to encapsulate executable tolerance models as FMUs and to reuse them in system-level simulation environments.

A key challenge arises from the partial incompatibility between DAE-based system simulation and tolerance models based on different mathematical foundations. FMI mitigates this limitation by supporting Co-Simulation, which allows tolerance models to be packaged together with dedicated solvers. The following subsections introduce a generic FMU interface concept for geometric tolerance models and a corresponding workflow for generating FMUs from arbitrary tolerance simulations.

3.1. FMU specification for geometric tolerance models

A generic, method-independent FMU interface for geometric tolerance models is proposed in Figure 2. The internal tolerance computation remains method-specific, while the external interface follows FMI semantics.

Static quantities include parameters p, representing sampled geometric realizations within tolerance fields T_{ni}, and internal result variables w, representing functional requirements FR_{mj}. Tolerance zones (T_n) and FRs (FR_m) can be defined using more than one parameter via the indices i and j, depending on the method utilized. Parameters typically vary between tolerance samples but remain constant within a transient system simulation run. Transient quantities such as forces F and displacements x may additionally be exchanged in assembly or contact simulations.

Since FMI requires a clear separation between inputs and outputs, the role of each variable must be explicitly defined for a given FMU. If different interpretations are required, separate FMUs should be provided. Detailed specification data such as datum definitions or tolerance types is spcific to the implemented method and thus handled internally to preserve interface generality.

The concept primarily targets Co-Simulation. If a tolerance model is DAE-compatible or can be transformed accordingly, Model Exchange is also possible. Tolerance zone widths are treated as uncertainty specifications and are therefore not exposed at the FMU interface, but expected to be handled via layered standards for uncertainty quantification.

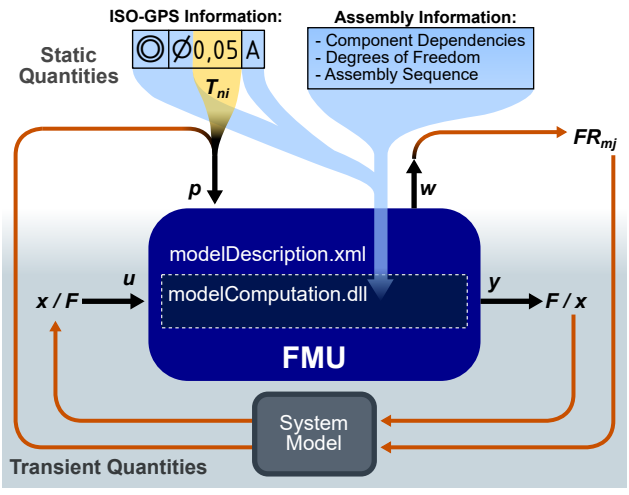

Fig. 2. Input and output data flow of our proposed FMU concept with static and transient quantities. Here, transient variables such as position x and force F are mapped to the transient input u and output y. Static tolerance parameters (p) and internal variables (w) are assigned to a specific realization within the tolerance field T_{ni} and the functional requirement FR_{mj}. All additional information is incorporated into the internal implementation of the model computation.

3.2. Workflow for FMU generation of tolerance models

The proposed workflow for generating FMUs from generic geometric tolerance models comprises three main steps, as illustrated in Figure 3. It starts from an arbitrary tolerance analysis model, as introduced in subsection 2.2, which evaluates geometric tolerances using either commercial software or custom implementations.

In the first step, the tolerance model is treated as a black box that maps tolerance-related inputs to corresponding outputs. In a second step, the original model is sampled to construct a metamodel, for example in the form of a polynomial response surface or a Gaussian process. This metamodel approximates the input–output behavior of the original tolerance model while significantly reducing computational effort and improving compatibility with system-level simulation environments.

When constructing such metamodels, potential requirements for derivative evaluation must be considered. Accordingly, the response surface must provide sufficient smoothness and differentiability, depending on the intended simulation use case. Alternative metamodeling approaches like neural networks may also be employed, provided that these requirements are met.

In the final step, the metamodel is packaged as an FMU following the interface specification introduced in subsection 3.1, enabling standardized integration into system simulations. For tolerance models that are already DAE-compatible and computationally efficient, the metamodeling step may be omitted, although performance implications must then be evaluated carefully. The workflow is demonstrated in the following case study.

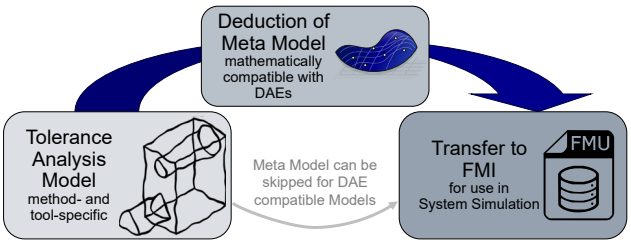

Fig. 3. Workflow: Starting from an arbitrary tolerance analysis model, an FMU can be generated using a metamodel. For DAE-compatible approaches, a direct transition is also possible.

4. Case study

In a previous study, the direct generation of FMUs from static DAE-compatible tolerance models has already been demonstrated [19]. The generalized workflow proposed in this paper, however, is intended to be applicable to more complex tolerance modeling approaches as well. To this end, the following case study is based on a FEA model with transient quantities. Since FEA models are based on partial differential equations and contacts between components cause additional nonlinear effects that must be resolved by specialized numerical solvers, this example falls into the group of models that are not directly DAE-compatible.

The focus of the case study is on tools with existing FMI support, rather than manual FMI implementations as used previously. This is intended to demonstrate that the proposed workflow can be realized using established FMI toolchains and integrated seamlessly into system-level simulations.

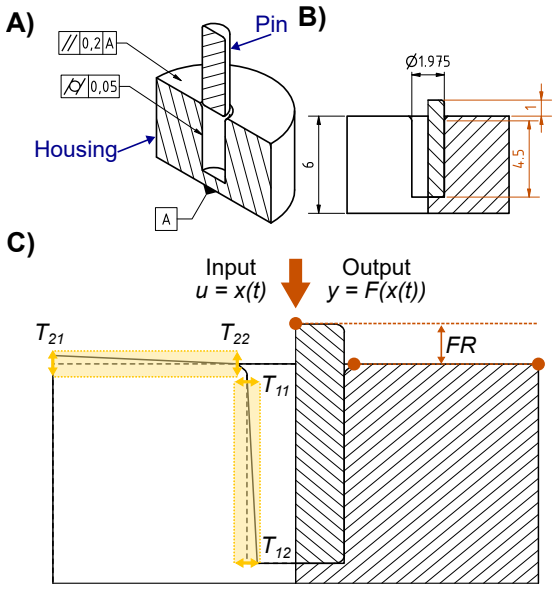

Fig. 4. Setup of the case study with its components and the geometric tolerances assigned to the housing (A) as well as nominal dimensions of the housing (B). Subfigure C shows the relevant variables and parameters at their specified location: Position $x(t)$ is the input for which force $F(x(t))$ and distance FR are evaluated. The tolerances defined in A are represented by the parameters T_{ni}.

4.1. Setup and objectives

The case study considers a press-fit assembly of a steel pin into a polymer housing shown in Figure 4. The pin is assumed ideal, while geometric deviations are assigned to the housing, including a cylindricity tolerance of the bore and a parallelism tolerance of the top surface (Figure 4 A).

The joining process is driven by a prescribed displacement $x(t)$. Contact interactions generate a reaction force $F(x(t))$, which is evaluated as transient output. For a compact representation, each tolerance zone is described by two parameters T_{ni}, which capture both translation and tilt deviations of the respective surfaces. As a static result, the final distance between the pin and the upper surface of the housing is evaluated as the functional requirement FR.

4.2. Implementation of the tolerance FMU workflow

The tolerance simulation is implemented as an FEA, following the general ideas of FEA-based tolerance modeling discussed in [16], where more complex assemblies than the one here are also considered. This choice makes it possible to carry out all steps of the proposed workflow in a transparent way. At the same time, it allows for the demonstration of a complex tolerance modeling approach that cannot easily be implemented as a DEA-based FMU in its original form.

The study is set up in ANSYS® Workbench™ 2025R1 using a two-dimensional axisymmetric model for computational efficiency. The tolerance parameters are embedded in the para-

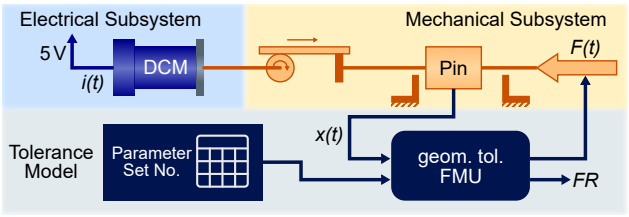

Fig. 5. Schematic of the system model with a DC-motor powering the assembly process of the pin, which is influenced by assembly forces. The force $F(t)$ is generated by the geometric tolerance model inside the FMU based on the current position $x(t)$ and a set of tolerance parameters. These parameters also influence the motor current $i(t)$ through the system coupling.

metric geometry definition as shown in Figure 4 C to capture the geometric effects according to the assigned tolerances.

The assembly is simulated as an insertion of the pin driven by a displacement boundary condition $x(t)$. Contact interactions between pin and housing are resolved within the FEA model, and appropriate boundary conditions are defined to ensure a physically consistent representation of the joining process. Output quantities are the reaction force $F(x(t))$ and the minimal distance FR between both parts top surfaces.

To generate the FMU, the tolerance parameter space is swept using ANSYS® optiSLang® and a 4-step full factorial design of experiment, resulting in 1024 simulation samples. The response surfaces are built by moving least squares for the force values (RMSE: 5.2 N) and linear regression for FR (RMSE: 32 μm). They ares used as metamodels and exported as an FMU using the integrated FMI export functionality. While the underlying tolerance model is simplified, the FMU-based integration procedure remains applicable to more detailed models.

4.3. Results and integration into the system model

The exported FMU is integrated into a system model using SimulationX® and the Modelica Standard Library (Figure 5). The Modelica-based system model with the FMU is available online [18]. A minimal setup is considered, in which a DC-motor drives the pin into the housing and provides the displacement $x(t)$ as the FMU input. The resulting force $F(x)$ and functional requirement FR are evaluated for exemplary tolerance parameter sets of different simulation runs listed in Table 1. For these runs, the performance advantage of the approach can also be evaluated on standard desktop hardware (AMD Ryzen 7 PRO 8840U, 64GB RAM). The FEA has an average compute time of 59 s, while the system model needs 77 μs, which represents a computation time improvement by a factor of 766.

Table 1. Results of different geometrical variations for distance FR and maximum force F_{max}. The relative deviations of the metamodel from the FEA refer to the nominal value of 1 mm for FR and the maximum force of 278 N for F_{max}.

	Tolerance Inputs				Results	
No.	T_{11}	T_{12}	T_{21}	T_{22}	FR	F_{max}
1	0 μm	0 μm	0 μm	0 μm	0.97 mm (−3.5 %)	174 N (−0.8 %)
2	−25 μm	−25 μm	100 μm	0 μm	0.92 mm (+1.3 %)	278 N (−0.1 %)
3	25 μm	25 μm	0 μm	−100 μm	1.01 mm (+1.0 %)	1.1 N (+0.4 %)
4	18 μm	−12 μm	30 μm	10 μm	0.94 mm (−3.2 %)	160 N (−0.6 %)
5	−15 μm	20 μm	30 μm	60 μm	0.92 mm (−3.3 %)	150 N (−0.8 %)

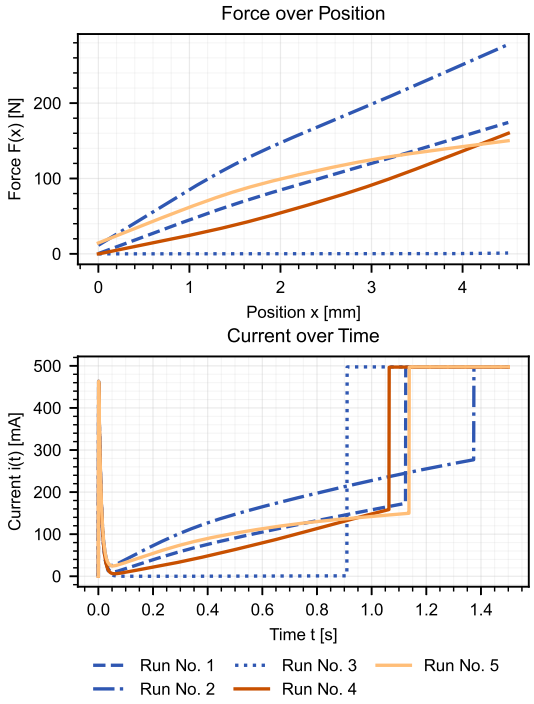

Fig. 6. Results of force output $F(x)$ of the FMU on the top and transient current values $i(t)$ in the DC-motor sub-model for parameter sets in Table 1. Both trajectories and the process duration are affected.

The resulting transient force profile is shown in Figure 6. This is a direct result of the metamodel inside the FMU. To demonstrate the possibilities of the system model, the electric current in the DC-motor model is also monitored. This quantity can now be used to investigate the influence of the geometric tolerances on the motor current as well as the process duration. The results illustrate how different tolerances lead to distinct force and current responses and variations in the functional requirement while maintaining efficient system-level simulation.

5. Discussion

A central strength of the approach is its method independence. The FMU interface and associated workflow do not impose assumptions on the internal tolerance modeling technique, allowing different classes of tolerance models, from analytical formulations to complex numerical simulations, to be integrated uniformly. From the system simulation perspective, tolerance models are accessed solely through their input–output behavior, supporting reuse across development phases and organizational boundaries while decoupling system-level modeling from tolerance-specific implementation details.

The distinction between static tolerance parameters and transient variables becomes less critical when surrogate-based FMUs are used at system-level. Although time-varying tolerances, such as wear or lifetime effects, are not explicitly addressed, the presented concept establishes a foundation on which such extensions could be built.

The use of metamodels as FMU cores introduces an additional abstraction layer. While surrogate-based integration improves computational efficiency and enables smooth interaction with DAE-based system simulators, it also introduces approxi-

mation errors depending on sampling strategy, model complexity, and surrogate type. In the presented study this is particularly evident in the reduced accuracy of the FR prediction (see Table 1). For the demonstrated transient use case this limitation is acceptable, but it highlights the need for tailored surrogate modeling strategies when higher accuracy is required.

The implementation effort of the proposed approach is closely linked to the creation of metamodels. In critical tolerance scenarios, this can involve considerable computational costs due to the required sample size, especially when high-dimensional tolerance spaces or complex contact models are taken into account. Although the theoretical framework is flexible and extensible, scalability to such cases cannot yet be fully investigated with the demonstrative case study. Further research is needed to address this possible limitation of the approach.

While many classical tolerance analysis methods are not naturally compatible with DAE-based system simulation, FMI (especially in Co-Simulation mode) also offers a general integration path by allowing entire classes of tolerance models to be executed with dedicated solvers or external wrappers. Such integrations without surrogate modeling, however, require substantial implementation effort for each tolerance modeling approach. These scenarios would also demand careful consideration of computational cost and solver coupling.

From a practical tooling perspective, the case study indicates that the approach is currently most feasible when combining tolerance models and metamodeling environments with FMI export capabilities within a single toolchain, as exemplified by the optiSLang®-based approach. For most commercial tolerance analysis tools, FMU generation would still require dedicated development effort. In addition, emerging layered standards for uncertainty quantification are still under development and not yet available for practical use. These standards are intended to complement the FMU specification by providing structured meta-information on tolerances and statistical evaluation, representing a relevant aspect for future work.

6. Conclusion and outlook

This work demonstrates that geometric tolerance simulations can be systematically integrated into system-level models by means of the Functional Mock-up Interface. By introducing a method-independent FMU interface and a corresponding generation workflow, tolerance models can be treated as executable, reusable components within MBSE environments, independent of their internal modeling approach. This enables tolerance-induced effects to be considered directly at system-level, where their impact often emerges only through interaction with other domains.

The presented case study shows that this integration can be achieved using tools with existing FMI support. As a result, our approach contributes toward closing the gap between detailed tolerance analysis and system-level simulation, supporting consistent design decisions across development stages.

Future work should focus on enhanced standardization and improved tool support. Particular attention should be given to uncertainty representation and to expanding native FMI integration in geometric tolerance simulation tools. Additionally, the scalability of the proposed approach to more complex scenarios should be systematically investigated to ensure its applicability in a broader range of industrial use cases.

References

[1] Bajand, A., Larsson, L.V., Buffoni, L., Nahodovic, E., Hällqvist, R., Lenord, O., Olsson, H., Otter, M., Vandamme, A., Pop, A., 2025. Towards a common standard for uncertainty quantification, in: Proceedings of the 16th int. Modelica Conf., Linköping University Press. pp. 671–680.

[2] Blochwitz, T., Otter, M., Arnold, M., Bausch, C., Clauß, C., Elmqvist, H., Junghanns, A., Mauss, J., Monteiro, M., Neidhold, T., Neumerkel, D., Olsson, H., Peetz, J.V., Wolf, S., 2011. The functional mockup interface for tool independent exchange of simulation models, in: Proceedings of the 8th int. Modelica Conf., Linköping University Press. pp. 105–114.

[3] Busch, A., Dreisbach, R., Popielas, F., et al., 2026. What do simulation engineers need to know about (model-based) systems engineering. Engineering Modelling, Analysis and Simulation 3.

[4] Cao, Y., Liu, T., Yang, J., 2018. A comprehensive review of tolerance analysis models. The Int. Journal of Advanced Manufacturing Technology , 3055–3085.

[5] Chase, K.W., Gao, J., Magleby, S.P., 1995. General 2-d tolerance analysis of mechanical assemblies with small kinematic adjustments. Journal of Design and Manufacturing , 263–274.

[6] Chen, H., Jin, S., Li, Z., Lai, X., 2014. A comprehensive study of three dimensional tolerance analysis methods. Computer-Aided Design , 1–13.

[7] Dantan, J.Y., Homri, L., 2024. Overview of mathematical concepts for tolerance analysis. Procedia CIRP 129, 73–78.

[8] Desrochers, A., Ghie, W., Laperrie‘re, L., 2003. Application of a unified jacobian—torsor model for tolerance analysis. Journal of Computing and Information Science in Engineering 3, 2–14.

[9] Desrochers, A., Riviere, A., 1997. A matrix approach to the representation of tolerance zones and clearances. The Int. Journal of Advanced Manufacturing Technology 13, 630–636.

[10] FMI Project, 2025. FMI Layered Standard REF. URL: https://github.com/modelica/fmi-ls-ref. accessed: 2 December 2025.

[11] Gupta, S., Turner, J.U., 1993. Variational solid modeling for tolerance analysis. IEEE Computer Graphics and Applications 13, 64–74.

[12] Homri, L., Teissandier, D., Ballu, A., 2015. Tolerance analysis by polytopes: Taking into account degrees of freedom with cap half-spaces. Computer-Aided Design 62, 112–130.

[13] Hong, Y., Chang, T., 2002. A comprehensive review of tolerancing research. International Journal of Production Research 40, 2425–2459.

[14] Janschek, K., 2011. Mechatronic systems design: methods, models, concepts. Springer Science & Business Media, Berlin.

[15] Roth, M., Goetz, S., Dharmaraj, M.R.R., Wärmefjord, K., Söderberg, R., 2025. On the potential of the quality information framework (qif) standard driving the interoperability in variation simulation. Proceedings of the Design Society 5, 2541–2550.

[16] Roth, M., Kopatsch, J., Wärmefjord, K., Söderberg, R., Goetz, S., 2026. Linking model-based definition and non-intrusive finite element analysis for automated variation simulation. Computer-Aided Design 191, 104003.

[17] Schleich, B., Anwer, N., Mathieu, L., Wartzack, S., 2016. Status and prospects of skin model shapes for geometric variations management. Procedia CIRP 43, 154–159.

[18] Steinmann, C., 2026. System model and FMU used in this work. URL: https://github.com/IFTE-EDA/TolFEA2SystemModel.

[19] Steinmann, C., Wrede, K., Schirmer, J., Lienig, J., 2025. Integration of geometric tolerance analysis in system simulations via functional mock-up units, in: Modelica Conferences, pp. 765–774.

[20] VDI/VDE Society Measurement and Automation, 2021. VDI-2206; Development of Mechatronic and Cyber-Physical Systems. 2021-11-00 ed., VDI-Verlag, Düsseldorf.

[21] Walter, M.S.J., Klein, C., Heling, B., Wartzack, S., 2021. Statistical tolerance analysis—a survey on awareness, use and need in german industry. Applied Sciences 11, 2622.