ADVANCED ENGINEERING MATERIALS
www.aem-journal.com

Enabling Digital Continuity in Virtual Manufacturing for Eco-Efficiency Assessment of Lightweight Structures by Means of a Domain-Specific Structural Mechanics Language: Requirements, Idea and Proof of Concept

Martin Rädel, Andreas Schuster, Felix Schoenitz, and Jean Lefèvre*

Virtual product development involves a wide range of specialized assessment tools, each tailored to specific physical phenomena and modeling requirements. A formalized, yet flexible, way to establish interfaces between these tools is required for their efficient use in automated virtual product development processes. This document presents a novel approach for the interaction of different assessment capabilities in a virtual product development process based on a solver-agnostic domain-specific language (DSL) for computational structural mechanics problems. The proposed system enables seamless integration with various assessment capabilities and solvers. The DSL is designed to handle multiple numerical implementation methods, including finite element method, peridynamic, and others. The approach utilizes a hierarchical data model, separating mathematical, physical, mechanical, and simulation data into distinct containers, facilitating modularity and reusability. The DSL is derived from code in a Java-based backend Java mechanics suite and provides interfaces to external tools and solvers through plugins. Using this approach, it is shown how tools and data from different solver ecosystems and data formats can be seamlessly interacted in the structural assessment for the optimization of eco-efficiency key performance indicators in the manufacturing of lightweight structures.

1. Introduction

1.1. Motivation

Simulation-based virtual product development is the future value driver for a faster time-to-market with reduced development risks, improved product performance, reduction of engineering costs, and tailoring to improved eco-efficiency, among others.[1] However, as the number of available simulation capabilities grows, so does the complexity of integrating them into coherent development processes.

A central challenge lies in establishing digital continuity: ensuring that different product development steps and phases are interoperable and can seamlessly exchange information. Existing tool coupling strategies often lack flexibility and scalability, particularly when partners employ different solvers or modeling approaches.

This article addresses the need for a formalized yet adaptable method to connect diverse assessment tools in computational structural mechanics (CSM). We introduce a domain-specific language (DSL) that abstracts solver-specific details while enabling modular data exchange. The DSL is designed to integrate multiple numerical approaches, including the finite element method (FEM) and peridynamic (PD).

The novelty of this work lies in the development of a hierarchical data model that separates mathematical, physical, mechanical, and simulation aspects. This separation enhances clarity, reusability, and consistency across different abstraction levels. The approach is implemented in a software library and extended through plugins to interface with external tools. To demonstrate feasibility, we present a case study on the eco-efficiency assessment of a composite manufacturing process.

1.2. Application

An example of such a process is the Virtual Product House (VPH) end-to-end (E2E) process shown in **Figure 1**, which covers the digital design,[2] virtual representations of the manufacturing process steps[3] as well as the structural and systems testing of

M. Rädel
German Aerospace Center
Institute of Lightweight Systems
Department Structural Mechanics & Virtual Product House
38108 Braunschweig & Bremen, Germany
E-mail: martin.raedel@dlr.de

A. Schuster, F. Schoenitz, J. Lefèvre
German Aerospace Center
Institute of Lightweight Systems, Department Structural Mechanics
38108 Braunschweig, Germany

The ORCID identification number(s) for the author(s) of this article can be found under https://doi.org/10.1002/adem.202501777.

DOI: 10.1002/adem.202501777

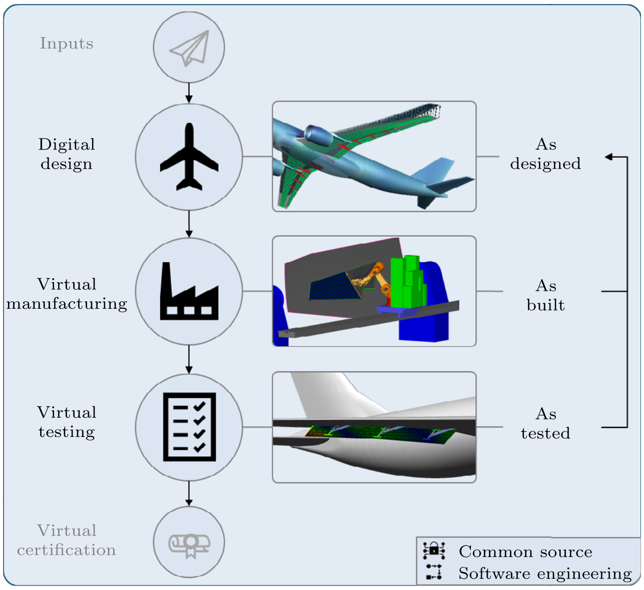

Figure 1. Representation of a VPH E2E process for a moveable example: digital continuity along the product development process and feedback capabilities to the design.

components and systems.[4,5] Such processes are also used to optimize processes with high resource requirements in terms of their eco-efficiency and at the same time ensure the requirements for mechanical component quality.

An exemplary implementation of the VPH E2E process is set up as a loose tool coupling via the workflow orchestration tool remote component environment[6] following the common source architecture.[7] Each partner can contribute its assessment capabilities in its own tool ecosystem. This creates challenges in the exchange of information in interfaces between assessment capabilities. Special care has to be taken to make models and data interoperable.

1.3. Use Cases

The approach is presented for the development of lightweight structures for mobility applications, e.g., a composite shell and a composite multispar moveable based on CSM simulations. The moveable is part of a reference research aircraft configuration, the eXternal research forum 1 (XRF1), see **Figure 2**b. The design investigations and manufacturing concept derivation are explained in ref. [8].

The moveable is created from dry preforms and is manufactured with a combined injection and curing process in the autoclave. It consists of a skin, spars, and a foam trailing edge, see Figure 2a. The skin preforms are created by automated fiber placement of dry fibers[9] and the spars are preformed by hand layup of fabric material on manufacturing cores followed by a diaphragm consolidation in an oven. The entire package is inserted into a resin transfer molding (RTM) injection mold. The injection as well as the following curing process is carried out in an autoclave[10] as substitution of a press at the German aerospace center (DLR) site in Stade.

Afterward virtual structural testing, e.g., damage tolerance simulations in accordance with,[11] is carried out. Finally, combined structural and systems testing is performed. Therein, all information is used to investigate nominal and failure cases in a cosimulation of actuation system and moveable structure in a multibody simulation.[12]

1.4. Requirements

Figure 3 shows some exemplary applications and solvers that are used during the processing of the use case story. This landscape might get even more heterogeneous as more assessment steps are considered. An additional challenge in the connection of the evaluation steps lies in the fact that although each individual step depicts the same physical reality of interest, different methods and idealizations are used to implement the specific solutions as shown in **Table 1**.

Even if one stays within FEM-based solutions, the phenomena to be modeled in the respective problem of an assessment step may require different modeling.

This raises the question of how to ensure that a complete simulation, consisting of model and result data, can be interconnected between different levels of abstraction in the form of idealizations between calculation steps in an E2E chain. Therein, separation of concerns and single responsibility principles should be applied, which leads to a distinction between data modeling and functionality implementation. The solution should be able to handle multiple numerical implementations of CSM theories like FEM and PD.[13] The solution should be designed to be independent of any specific solver, and therefore license, to be

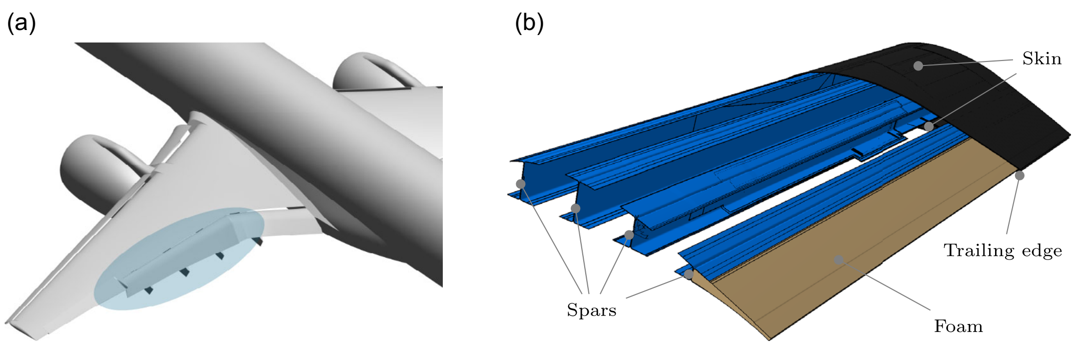

Figure 2. XRF1 moveable and structural concept: Basis for the requirement analysis of an E2E process: a) moveable location and b) multispar structure.

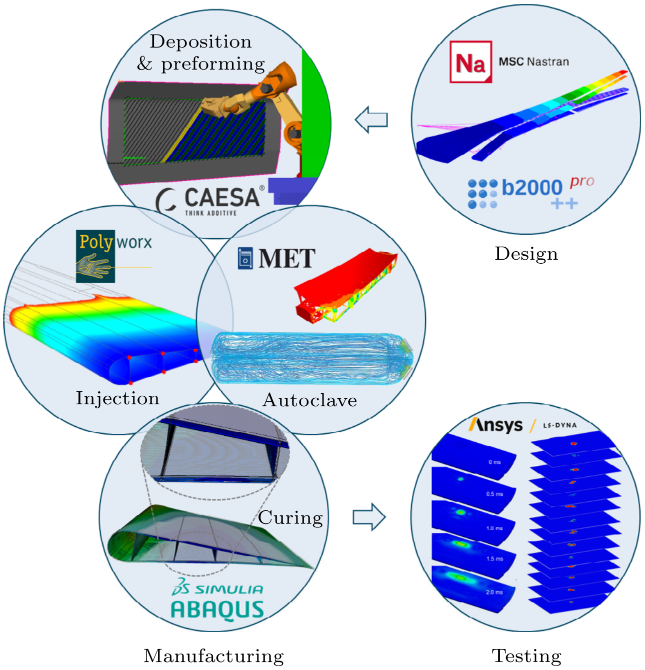

Figure 3. Simulation steps in the use case simulation process as reference for the requirement analysis.

applicable throughout the partners in a VPH process. However, there should be mechanisms to store solver-specific information and software to interact with the data directly. However, it can also prove to be advantageous if data can be provided independently of a programming language in order to be able to provide further functionalities based on this in the partner group. It is therefore advisable to use a standard if one is available. It also makes it easier to interact with data if the information is machine-readable if stored in a file format.

2. Language

2.1. Design Philosophy

The proposed approach introduces a solver-agnostic DSL that formalizes the representation of CSM problems. The design philosophy centers on creating a flexible yet rigorous data model that supports multiple simulation methods while maintaining clarity and modularity. As shown in **Figure 4**, two scenarios were analyzed: a direct coupling of evaluation steps (Figure 4a) and the use of a DSL as a connecting element (Figure 4b).

Although direct bidirectional coupling between two solvers is a worst-case scenario and not particularly relevant for the example in Figure 3, the benefits of using an internal data standard to represent CSM data as a DSL are still obvious, especially for a

Table 1. Exemplary assessment steps for moveable use case.

Step	Solver	Method	Idealization	Discretization	Remark
Design	Nastran, b2000++ pro	FEM	2.5D, layered	GFEM, mixed	–
Deposition	CAESA TapeStation	–	Points	–	Gaps/overlaps → KDF
Injection	RTM-Worx[25]	FEM	2.5D, smeared	Unstructured	–
Autoclave	Virtual autoclave[10]	FVM	Surface	Point cloud	–
Curing	Abaqus	FEM	3D, stacked	Structured	–
DT testing	Abaqus, LS-Dyna	FEM	3D, stacked	Structured	With interlaminar behavior

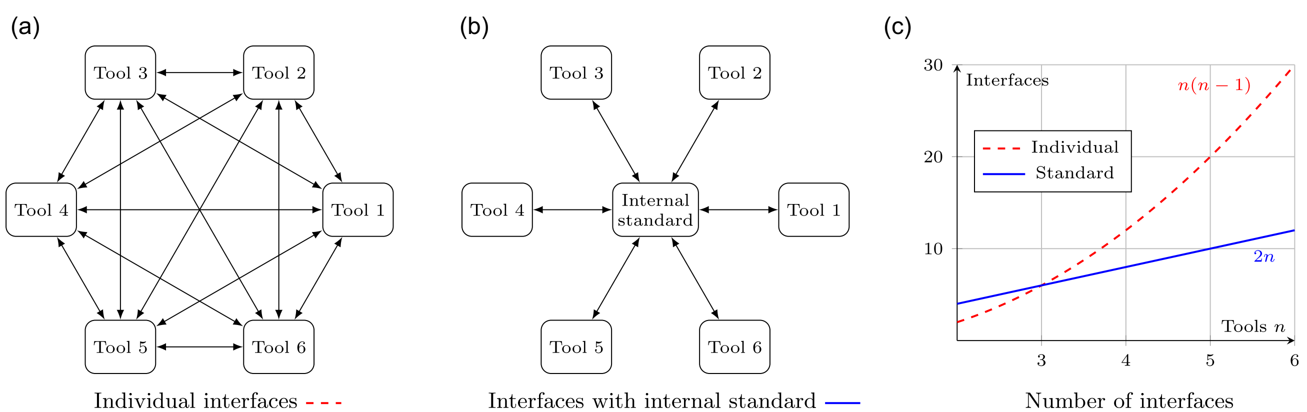

Figure 4. Evaluation of different approaches toward interfacing for simulation tool communication: a) individual interfaces, b) interfaces with internal standard, and c) number of interfaces.

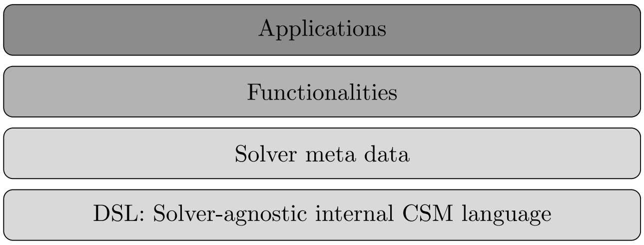

Figure 5. Layer-based architecture approach.

rising number of tools involved in an assessment chain. This advantage is further enhanced if one considers the provision of functionalities based on the data. In the case of direct coupling, there is a high risk of multiple repetitive implementations that only differ in terms of access and the provision of solver-specific information. If the data itself is under control as a DSL, a reusable and expandable ecosystem of functions can easily be implemented based on the data.

After analyzing the problem at hand, a multilevel architecture as shown in **Figure 5** proved to be expedient.

While the lower two levels represent data, once solver-agnostically and, additionally, in *meta*-languages of the respective solver formats, the third level can provide functions based on the solver-agnostic description, e.g., input and output (I/O) from and to specific solvers, data manipulation for probabilistic analyses or mapping capabilities. These functional units can be combined to public applications in the fourth layer. In the course of this publication, the focus is on the lowest level.

2.2. Data Model

2.2.1. Data Types

In a first step it is analyzed what type of information is required to assure data continuity in VPH processes. As shown in **Figure 6**, the data in a CSM simulation typically consist of three types of information: the first describes the mechanical and physical reality of interest. This information may include definitions of constituents, e.g., materials, composites with their respective characteristics, cross sections, and other. This may include a distinction w.r.t. the applied idealization, e.g., mass definition for properties in $\mathbb{R}^0$, spring or damper definitions in $\mathbb{R}^1$, or material definitions including characteristics necessary for a $\mathbb{R}^1$, $\mathbb{R}^2$, or $\mathbb{R}^3$ idealization.

The next type of information describes the data necessary for the numerical implementation of the solution method, so, e.g., for an FEM analysis this includes the discretization in the form of nodes and elements and, on the results side, nodal, element, and integration point values. For a meshfree approach, e.g., in PD, this might be collection points and the respective states.

The final information type contains an assignment of the mechanical properties to a region of the model, e.g., an assignment of a mechanical property for a domain of elements for an FEM analysis as well as loads and boundary conditions (BCs) on the discretization and, finally, a description of the solution method.

During the course of this publication, the data connected with the numerical implementation, so mainly discretization and result data, are referred to as heavy data, following the

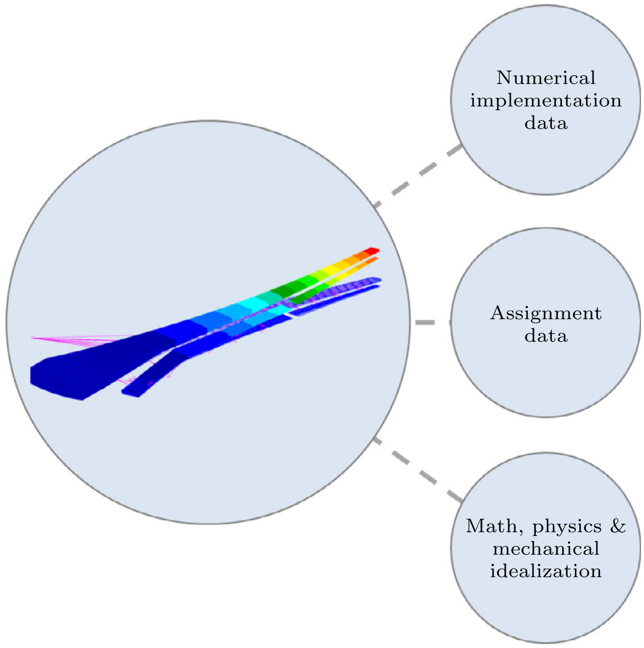

Figure 6. Evaluation of CSM simulation models and identification of general data types.

eXtensible data model and format (XDMF) notation, as the amount of data can be huge for large models or transient analyses with a lot of time steps. It is usually not expected that a user directly interacts with the raw data. Thus, binary storage schemes are probably preferred.

The two bottom types of information are called light data. This is the data a user is likely to manipulate. It is therefore preferred if this data can be accessed from a human-readable description, e.g., an American standard code for information interchange (ASCII) format.

2.2.2. Available Standards and Libraries

A screening for available data standards was initially performed to see whether there are approaches for a DSL for CSM simulations.

Numerical Implementation Data: Different attempts have been made to standardize the information with respect to numerical solution techniques. Some key representatives will be discussed here and the usability for the present approach will be examined with regard to certain usability criteria. The focus is on cross-tool open-source solutions that try to represent the complete topology of a numerical model including sets and other items required and potentially have available interfaces to a multitude of other software. Standards that are used purely for visualization are not taken into account, nor are available specialized formats for open-source solvers.

In the field of structural mechanics, the classical continuum mechanic (CCM) is a widely used solution method in conjunction with a mesh-based discretization based on the implementation in the FEM. Numerous data formats were developed over the years. A selection is shown in **Table 2**.

Table 2. FEM discretization data standards (selection).

Format	Base format	Languages	License
EXODUSII	NetCDF	C/C++/Fortran	Proprietary
GAMBIT	ASCII	–	Proprietary
MOAB	HDF5	Any	LGPL3
Silo	Own binary	C/Fortran	BSD3
VMAP	HDF5	Any	Proprietary
MED	HDF5	Any	LGPL

A more in-depth evaluation of the concepts and features of these standards can be found in Appendix A. In general, it can be found that most formats are stored in a binary format, as a result of the potentially large amount of data. Basically, all the aforementioned standards for mesh-based applications may be applicable for use as the representation of the discretization-dependent information in FEM. Due to the number of potential interfaces to other FEM codes, VMAP is chosen as the initial representation for discretization-dependent information in the present approach. The lack of direct visualization capabilities for VMAP is overcome by an interface to XDMF standard which is understood by visualization libraries like ParaView. In the meantime, a ParaView plugin for VMAP is available.

There are some software libraries out there that deal with the conversion of mesh data, e.g., meshio (https://github.com/nschloe/meshiohttps://github.com/nschloe/meshio)[14] and other applications like MeshFEM (https://github.com/MeshFEM/https://github.com/MeshFEM/MeshFEM), PyMesh (https://pymesh.readthedocs.io/en/latest/https://pymesh.readthedocs.io/en/latest/), or Meshes.jl (https://github.com/JuliaGeometry/Meshes.jlhttps://github.com/JuliaGeometry/Meshes.jl). As the names suggest, they are capable to interact with FEM mesh information.

Complete Simulation Data: None of the investigated data standards and libraries describe or use the whole simulation. The focus is on the discretization-dependent information for models and results, also in VMAP. However, additional (light data) information about materials, composites, offsets, loads, initial and BCs as well as the analysis are necessary for the use case to be implemented. For the problem at hand, a potential solution would be to use the proprietary format of a commercial solver for the model, e.g., a solver input file format for ABAQUS, ANSYS, NASTRAN, or LS-Dyna. There are open-source software packages for the interfacing with specific solvers available, e.g., PyAnsys (https://docs.pyansys.com/https://docs.pyansys.com/) for Ansys[15] or pyNastran (https://pynastran-git.readthedocs.io/en/latest/https://pynastran-git.readthedocs.io/en/latest/) for Nastran.[16] However, after careful consideration, it was found that for some model aspects a direct transfer between solvers is not possible. A new implementation of a data format allows the consideration of these aspects. Plugins for the consumption from and export to commercial solver must then remedy these data inconsistencies. Additionally, these formats are limited to a single CSM theory, like FEM or PD for the respective solver. Therefore, a novel approach is developed that covers all the required aspects.

2.2.3. Concept

From Figure 6 it can be seen that multiple levels of abstraction and the associated data are required to create a simulation model. For the present approach, it is decided to separate math, physics, mechanical, and simulation data into separate, reusable, and expandable containers, as shown in **Figure 7** and store them in the new hierarchical common structural mechanics format (CoSMo). Doing so brings clarity, modularity, and reusability to the DSL. In the containers, a modular language vocabulary is created which is based on the level of assumptions and theories up until the level of the respective container. A physics application, e.g., a module that calculates the International Civil Aviation Organization standard atmospheric model, only requires math and phys container data. If you want to interact with an analytical solution for a mechanics problem, e.g., the classical laminate theory[17] or first-order shear deformation theory (FSDT), you can reuse the data from math and phys containers, e.g., for a material definition, and add the material or composite layup definitions in the mech container.

By isolating mathematical objects from physical entities and further separating these from mechanical data, a clear hierarchy

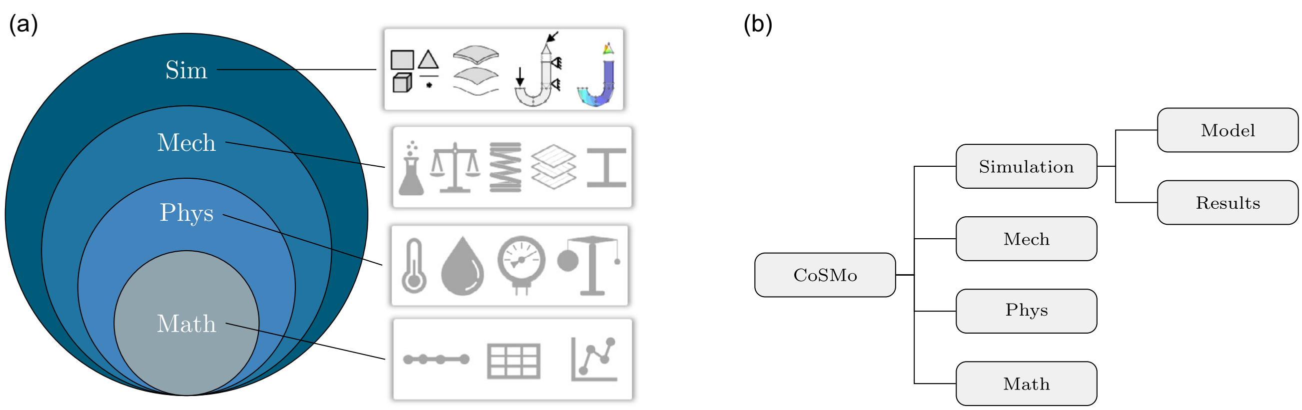

Figure 7. An approach toward reusable containers for abstraction levels in simulation data: a) approach and b) CoSMo hierarchy.

of dependencies is enforced and the single source of truth principle is inherently implemented.

This information can be used in model generators, interfaces to analytical solutions, or the interaction with simulation models and results in external solvers. The data description remains the same—dependent of the level of abstract of a CSM theory—only the in- and output to a specific functionality changes. The data therefore remain highly reusable and are an excellent choice for the storage of data in databases.

Math, Phys, and Mech Data: To build an input deck for CSM models several mathematical, physical and mechanical entities are required. **Figure 8** gives an overview about the top-level elements in the respective containers. In the math container discretization biases used in modeling, curve definitions, e.g., for a later use for the application of transient loads and BCs, or tables for field variables are stored.

The phys container holds information about state variables of base quantities like densities, e.g., for material or atmospheric definitions, temperatures for mechanical BCs, or the dependency of material parameters. These state variables can be combined to different implementations of conditions, e.g., to describe the environmental conditions during the measurement of material parameters during testing.

The mech container holds information for use in mechanical theories. These are dependent of the idealization of the underlying mechanical abstraction level and theory. For example, it is possible to store cross sections and materials to represent a structure in a beam theory and associate a member, e.g., a rod or a beam, in $\mathbb{R}^1$ in a 2D or 3D realm and define an offset and make it available in a property for assignment in a model.

2.2.4. Simulation Data

Model Data: On model level an attempt is made to recreate all information required to create input decks for specific solvers. Thus, the structure in **Figure 9** shows the structure for a model consisting of an assembly of parts with geometric and CSM theory items and the subcases that include the loads and BCs a model is subjected to and the respective solution technique consisting of an analysis and the associated output.

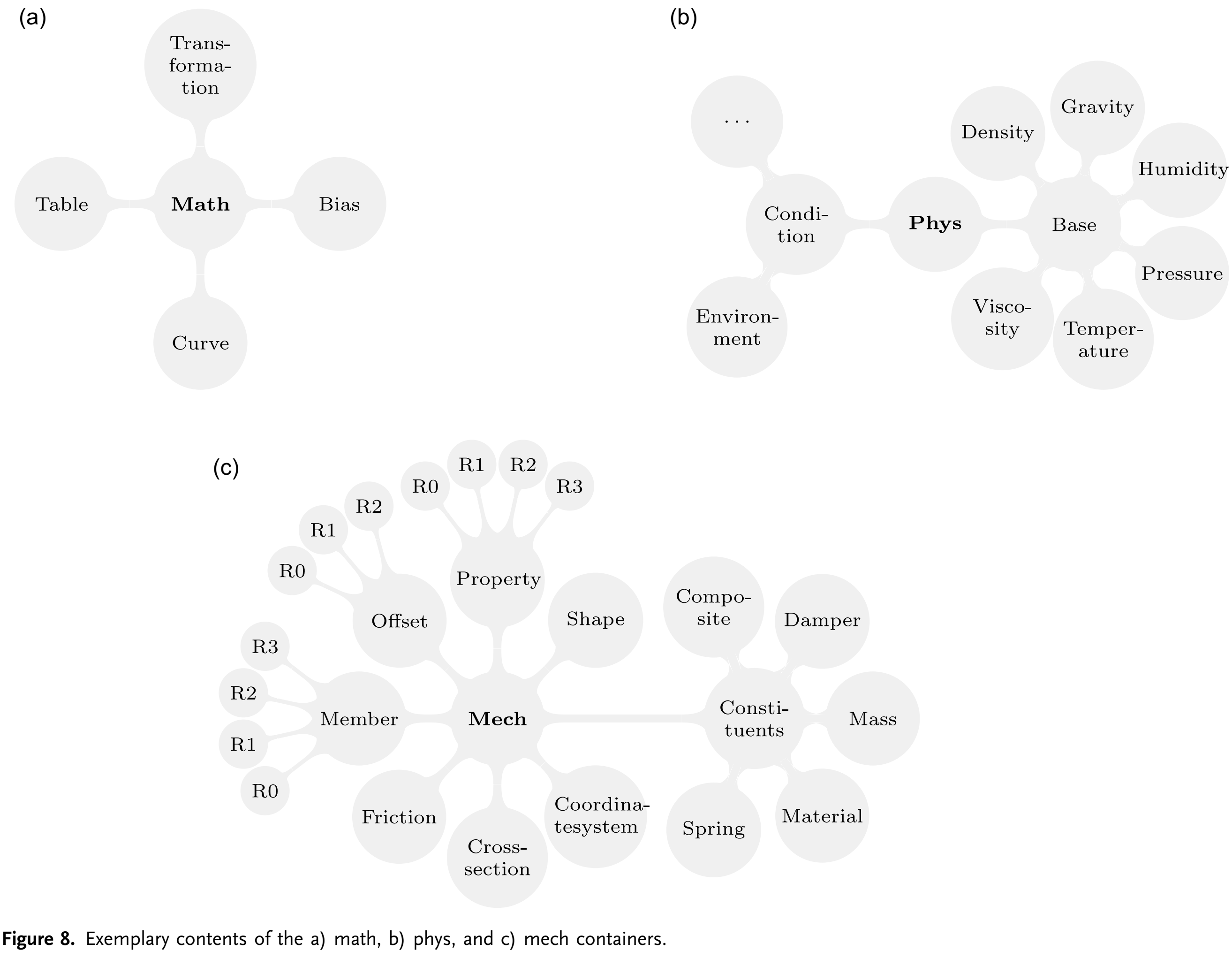

Figure 8. Exemplary contents of the a) math, b) phys, and c) mech containers.

ADVANCED ENGINEERING MATERIALS
www.aem-journal.com

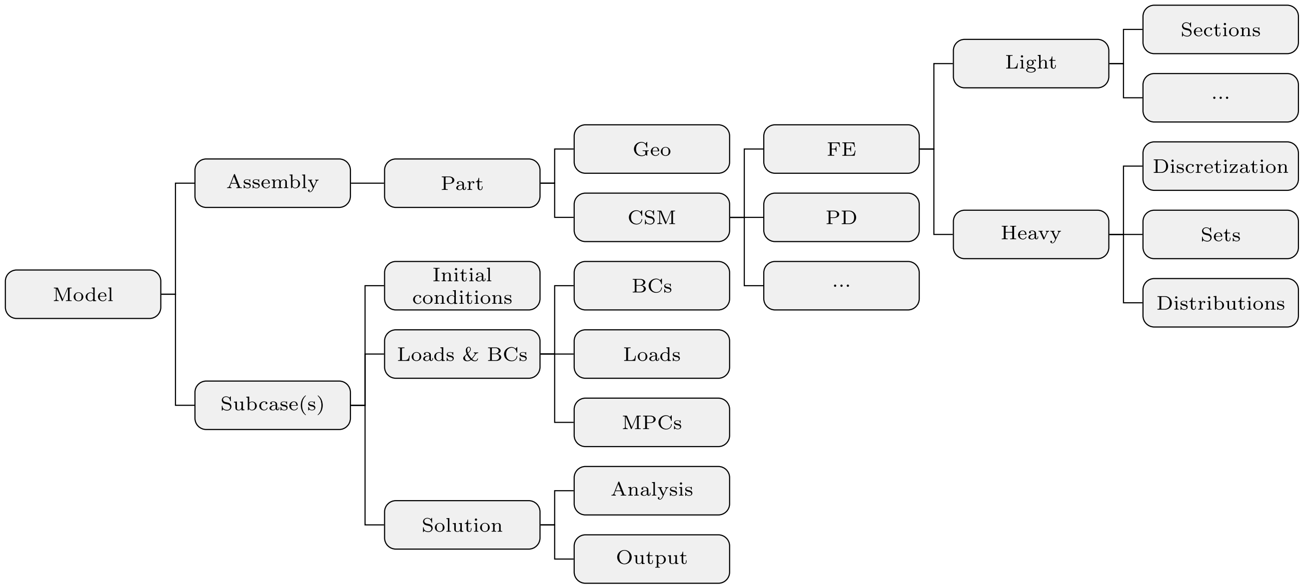

Figure 9. A simplified representation of the developed model hierarchy representing an assembly of parts and subcases it is subjected to.

The CSM container is setup to store information model domains with different numerical implementations of theories. Thus, a part can consist of regions represented by finite elements (FEs) or PDs or others as required for.[18] This capability is a major extension compared to classical data descriptions which are usually limited to one numerical implementation.

This model container can be filled from external models or implementations of model generators that use or generate the DSL vocabulary during model creation.

Result Data: In automated simulation-based evaluation processes, it is necessary, on the one hand, to be able to create models and provide them to a solver. On the other hand, it is necessary to be able to interact with the calculation results of the solver. Therefore, the calculation results from different implementations of numerical solution strategies are part of the DSL.

The type of results depends on the analysis selected in the model and therefore the mathematical solution method. The structure shown in **Figure 10** attempts to take this and the applicability to several CSM theories into account.

For a mathematical solution based on a perturbation problem, like linear eigenvalue analysis for stability or modal behavior, results, e.g., eigenvalues, are defined for the whole model. Transient or nontransient analyses define time steps and associated states. Therein, integral results for the whole model, like energies, can be stored as well as domain-specific quantities associated to the respective assembly object. To allow the consideration of multiple CSM theories, containers for FEM and PD domains exist. This structure makes it possible to implement efficient postprocessing methods and result data queries.

2.2.5. Implementation

Implementing a DSL within a software library offers a powerful way to encapsulate domain knowledge and provide expressive, concise interfaces for specific tasks. This allows developers to write code that closely reflects real-world operations or workflows within that domain, making the software easier to use and more aligned with the needs of its users.

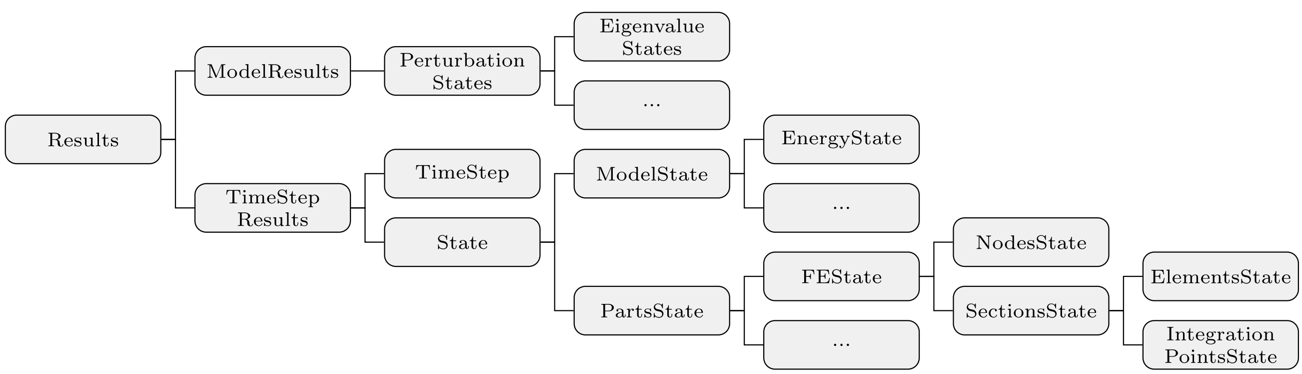

Figure 10. A simplified representation of the result data hierarchy applicable to general solution types.

To maximize interoperability and automation, it is often beneficial for the DSL input and output to be in a machine-readable format, such as JavaScript object notation (JSON), extensible markup language (XML), or YAML ain't markup language (YAML). These formats enable seamless integration with other tools, easier testing, and support for automated processing pipelines. Moreover, they provide a standardized way to represent structured data, reducing ambiguity and increasing robustness across system boundaries.

A practical approach to generate machine-readable formats is through serialization of annotated code. By leveraging annotations, developers can mark relevant constructs in the source code, which are then automatically processed and transformed into structured representations. This approach reduces boilerplate code, ensures consistency, and allows DSL vocabulary items to be introspected or exported with minimal manual effort, thereby streamlining the development and integration of DSL-based systems.

An object-oriented programming (OOP) approach is chosen with an implementation in the Java-based backend Java mechanics suite (jMeS). The DSL vocabulary is implemented as data transfer objects (DTOs) or plain old Java objects (POJOs). These have no other task than to store information. No functionality is associated with them directly. Jakarta (https://jakarta.ee/specifications/annotations/https://jakarta.ee/specifications/annotations/) and Jackson (https://github.com/FasterXML/jackson-annotationshttps://github.com/FasterXML/jackson-annotations) annotations are used to annotate classes and attributes. Objects can be

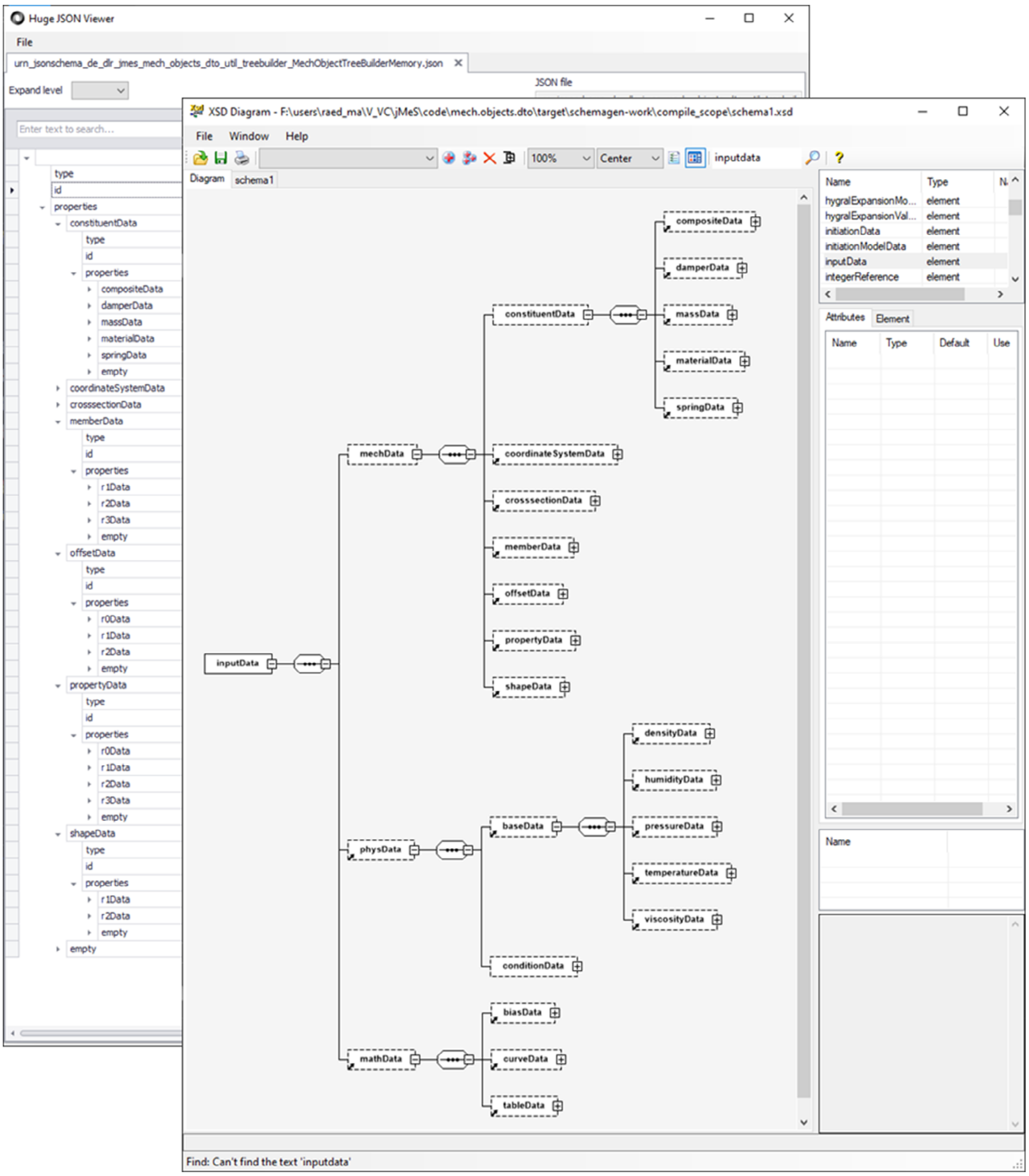

Figure 11. Adding standardization via wire format schemas: XML and JSON schema example of DSL mech-level data.

referenced by assigned universally unique identifiers (UUIDs) which are labeled by the @XmlID annotation. The notation @XmlIDRef is used on attributes for the references.

Jakarta Marshaller and Unmarshaller are used to serialize the containers to and deserialize them from XML information, e.g., strings or stream implementations, e.g., files. A similar task is performed by ObjectMappers in Jackson for JSON and YAML formats.

During code compilation XML schema definition (XSD) and JSON schema definitions of the containers are automatically created, see **Figure 11**. These schemas can be used to easily create the container and DTO code in multiple OOP programming languages, e.g., with CodeSynthesis (https://www.codesynthesis.com/products/xsd/https://www.codesynthesis.com/products/xsd/) to C++ or generateDS (https://pypi.org/project/generateDS/https://pypi.org/project/generateDS/) for Python.

For easier integration into python-based functionalities, the result data model is also implemented in an in-house python package.[19]

3. Ecosystem

The DSL is available for interaction as code in the software library jMeS. However, to be able to use it properly in the VPH E2E processes an ecosystem of functionalities is required around the DSL implementations. To realize this, the approach from Figure 5 is followed. In the course of the present publication the focus is on interfacing with third party tools. Following a similar approach as presented for the core DSL, OOP meta language implementations of external tool data are created. These mimic the data structures used for specific solvers and other tools. Finally, plugins are written to convert the external data into the internal data format and export them vice versa. Also, a possibility is created to store the solver-agnostic data. An example for a model definition is shown in **Figure 12**.

Using the approach from Section 2.2.4, the internal DSL data are deserialized to a JSON representation. During this conversion, the heavy data are externally stored in to a binary VMAP representation that is referenced in the light data file. This approach is implemented in a modular way, so that other external representations for the discretization can be added if necessary.

Using this approach, interfaces to the solvers mentioned in Table 1 are created as plugins to the DSL functions. From that, adapters can be created that couple two specific external tools directly by converting the input file format to the DSL and converting the DSL representation to the desired target tool format. Additionally, triangulation of $\mathbb{R}^2$ data in $\mathbb{R}^3$ realm and tetrahedralization of point cloud data is implemented, as the complete approach presented is dependent of discretized data. Particular attention is given to ensuring that the approach includes the capabilities to align data from different coordinate and unit systems. Also, interfaces to mapping tools are provided. In the present approach, MpCCI Mapper (https://www.mpcci.de/en/mpcci-software/mapper.htmlhttps://www.mpcci.de/en/mpcci-software/mapper.html) is used.

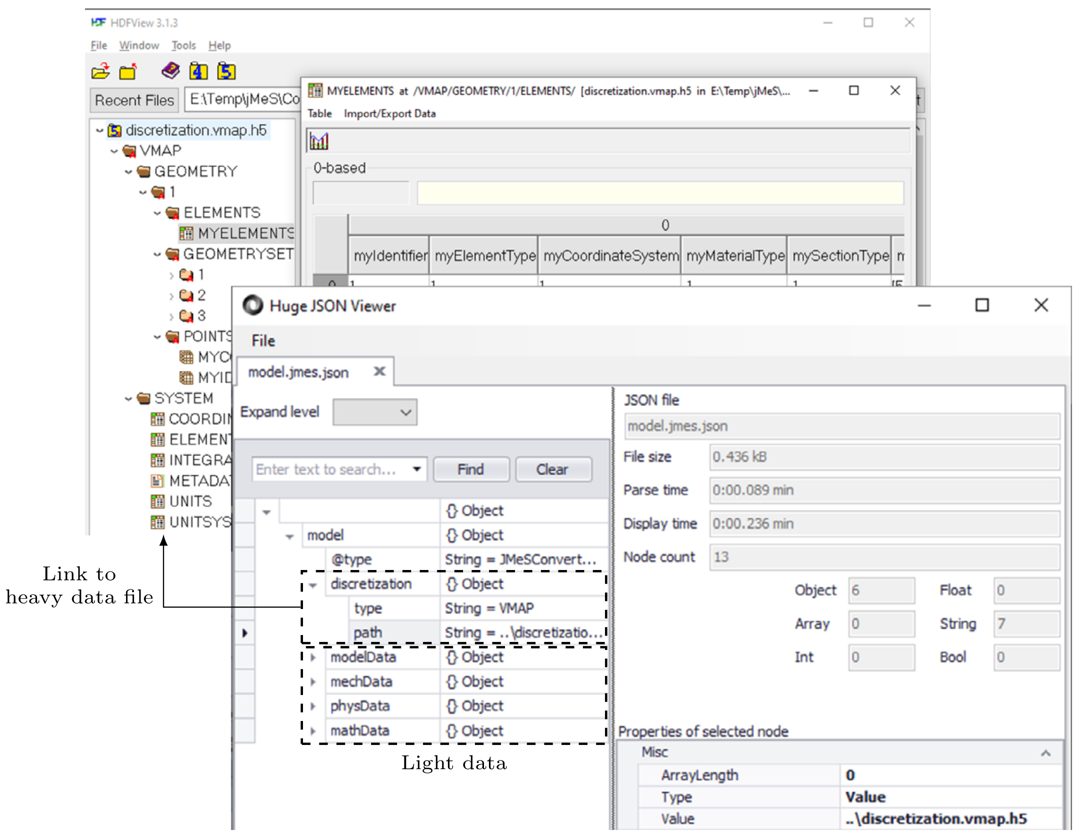

Figure 12. Implementation of a bi-file representation of simulation data: solver-agnostic model file export example.

4. Application Example

The functionality of the presented approach is demonstrated by linking the simulative representation of the production of a carbon fiber reinforced plastic (CFRP) shell. The process is part of the consideration of ecological efficiency KPIs of manufacturing processes in the virtual product development of lightweight structures using resource-intensive manufacturing processes in the VPH. The aim is to develop manufacturing processes that are intelligently adapted to the component behavior but do not affect the performance of the component. The behavior of the component during the manufacturing process is checked as part of the virtual prediction using numerical sensors. In this case, the curing of the component is monitored via the direct operating costs and process-induced deformations.

As shown in **Figure 13**, the shell is first deposited using dry fiber placement (DFP). Injection then takes place in the open mold in the autoclave before the shell is subsequently cured. In the example, the injection and curing steps as well as the ambient conditions in the autoclave are simulated as BCs during curing.

Coupling the simulation steps is necessary because the curing of the resin does not begin at the start of the curing simulation, but naturally starts during injection. As this process also takes a certain amount of time, the final state of the degree of curing at the end of injection must be taken into account as the initial condition for the curing simulation. In addition, injection and curing take place in an autoclave. The ambient conditions in the autoclave must therefore be considered as BCs for an accurate prediction of the component quality. As a simplification, the influence on the injection is neglected. The turbulent flow conditions in the autoclave are integrated by means of the transient temperatures and heat transfer coefficients (HTCs) on the top and bottom of the shell. These are obtained by means of simulations from the virtual autoclave.[10]

The complete assessment process based on the DSL implementation and its functionalities is shown in **Figure 14**. Initially, the injection simulation is carried out in RTM-Worx. This creates a result in a solver-specific ASCII solver file format. The resulting degree of cure (DoC) values are to be mapped upon the curing simulation model. For this purpose, an adapter is written that converts the injection results to the internal DSL representation and then converts this to the VMAP format.

For the present task, the mapping target is defined in an Abaqus INP input file. Here also an adapter is created that converts the INP to the internal data format and uses the same plugin as before to convert the model to VMAP. Afterward the mapping is performed. The mapping results, in VMAP format, are converted to the internal representation. Here, the internal representation of the target model is used and the DoC is added to the target model as an initial condition. The same basic process is performed for the temperature and HTC data from the virtual autoclave, which are added as BCs to the model after triangulation of point-cloud data on the virtual autoclave component surfaces. Another converter is then used to output the modified target model, including initial and BCs, back as an INP file. The results of some of these steps can be seen in **Figure 15** for initial testing data.

Here, it can be seen that the developed process, based on a DSL for CSM, demonstrates functionality and enables the coupling of data from different data sources and idealizations. Calculations are currently being carried out on the basis of the developed approach in order to determine the quantitative effects of the process. As part of the investigations, various autoclave cycles and their effects on the energy and media consumptions on the one hand and the structural performance of the

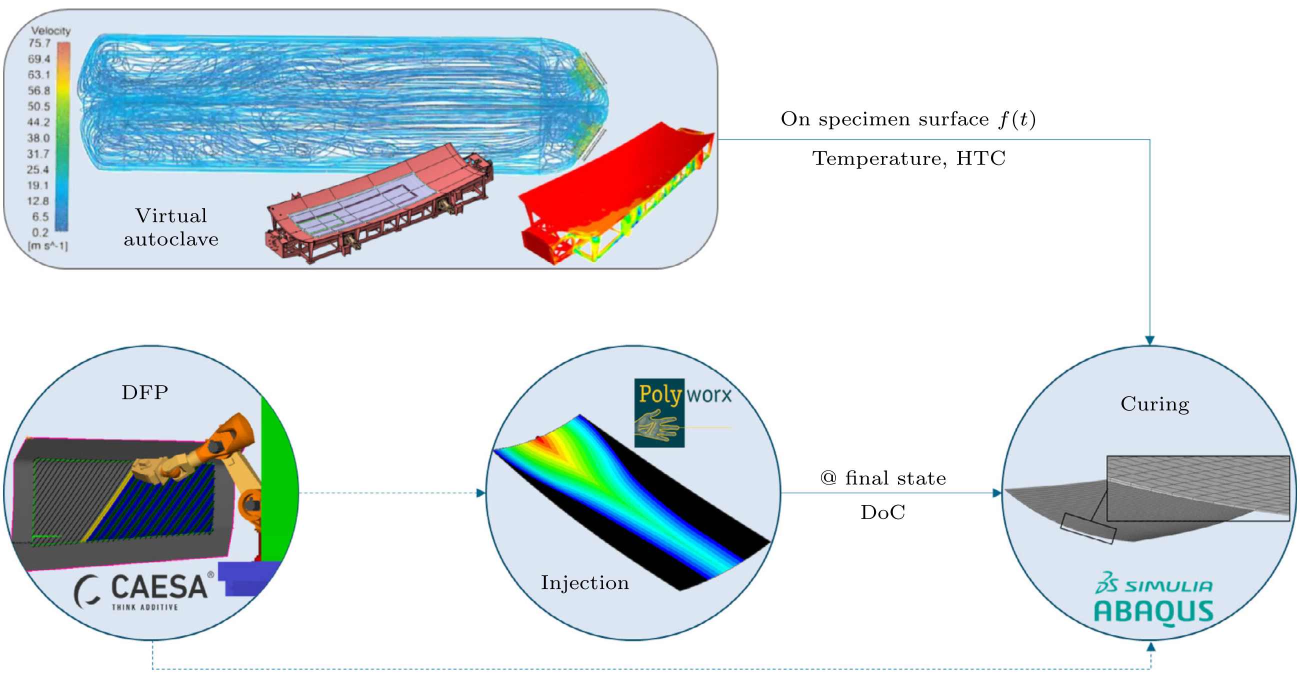

Figure 13. Application of the presented approach in the PredictECO use case.

ADVANCED ENGINEERING MATERIALS
www.aem-journal.com

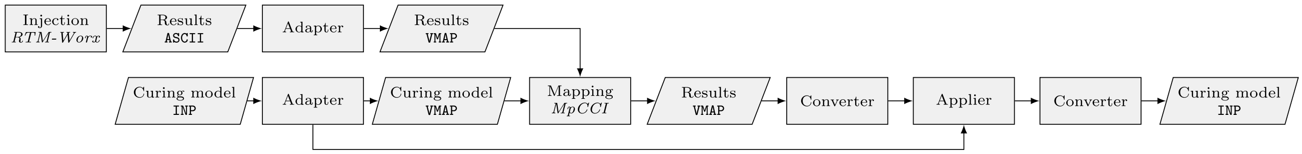

Figure 14. Process flow for interaction of injection and curing simulation.

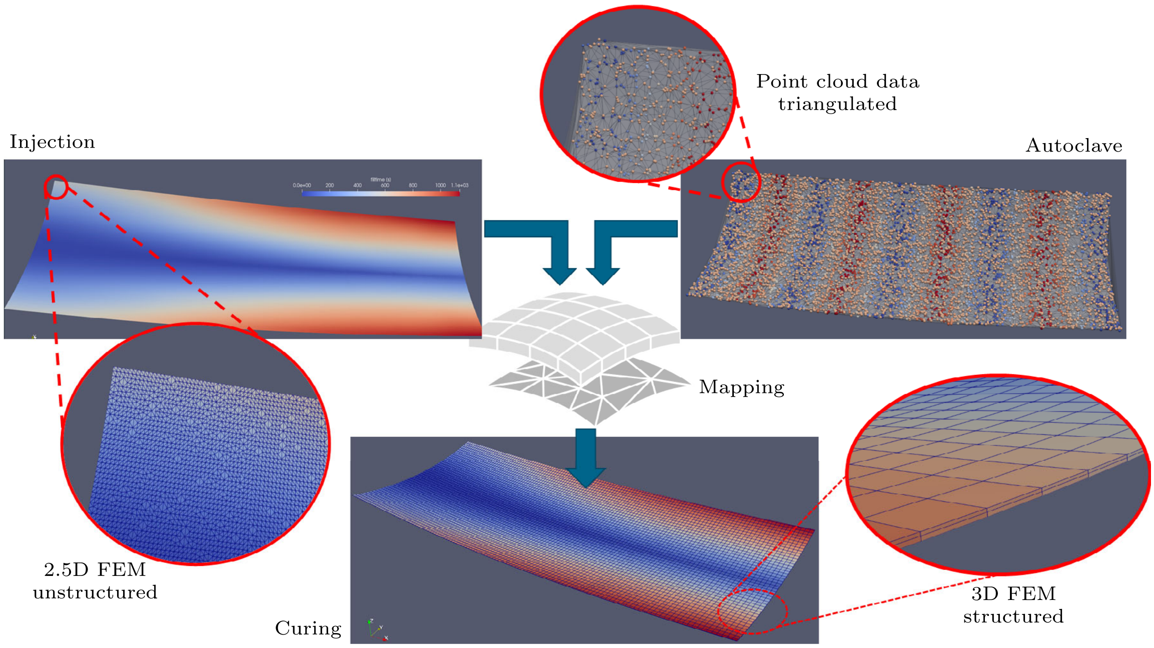

Figure 15. Example for the interoperability of model and result data for the PredictECO use case: connecting different model idealizations and result data origins using a combination of DSL and mapping methods.

component with regards on requirements on the other are being carried out.

5. Conclusion

In a world of ever more and more capable assessment tools for more aspects in virtual product development, the real challenges for efficient and potentially automated virtual product development processes are interfaces to connect these functionalities. This is especially relevant in a field where the physical reality of interest and the phenomenon under investigation determine how a structure is idealized for each assessment step.

The proposed approach presents a significant advancement in integrating simulation-based virtual product development processes, enabling seamless integration with various assessment capabilities and solvers. The DSL's hierarchical data model provides modularity and reusability, while the use of plugins allows for easy adaptation to new tools and solvers. The application of the proposed system to simulate the production process of a CFRP shell demonstrates its potential for improving ecological efficiency in manufacturing processes. This work showcases the feasibility of the proposed approach, highlighting its benefits for virtual product development in the field of lightweight structures.

While the examples used in this publication use models on an industry-relevant scale in terms of component model size, further investigations about the scalability of the solution, e.g., to mega-degree of freedom-problems are necessary. Early investigations show that in-memory performance is suitable while proper care has to be taken with respect to the implementation for file ex- and import.

Appendix

Standards and Libraries for Numerical Implementation Data

As an extension to Section 2.2.2, we provide additional information about existing approaches.

The Sandia National Laboratories developed EXODUS[20] and EXODUS II[21] as successors to older approaches like GENESIS and the TAPE formats. It offers access via C, C++, and Fortran and has a lightweight Python module. It is part of Sandia National Laboratories Engineering Analysis Code Access System and offers interfaces to the Sandia tool ecosystem like CUBIT, Sierra, and Peridigm.

The GAMBIT neutral file is the data standard for the GAMBIT software for the creation of mostly computational fluid dynamics (CFD) meshes used in FLUENT or now Ansys FLUENT but also allows the storage of typical FE topologies. It is represented by an ASCII file which makes parsing and writing easier than for binary formats. It not only consists of the mesh topology but, in contrast to the other formats, also allows the storage of discretization-dependent BCs.

The Argonne national laboratory (ANL) develops mesh-oriented database (MOAB)[22,23] as the representation of mesh data for their project SIGMA. Internally, MOAB uses an hierarchical data format (HDF5) container with a certain standardized information hierarchy. HDF5 has the advantage that interaction libraries for multiple programming languages, platforms, etc. exist.

Similar to the previously mentioned institutions, the Lawrence Livermore National Laboratory also developed their own standard Silo[24] for the interaction with mesh data inside their tool suite. Silo has an internal data structure with libraries for C and Fortran.

The VMAP standard was the result of a project that had the goal to develop a new interface standard for integrated virtual material modeling in manufacturing industry. It is represented by a certain HDF5 hierarchy. The native library is written in C and offers interfaces to a multitude of programming languages via simplified wrapper and interface generator. In contrast to the other standards, it offers reference publications of external wrappers for the use of VMAP in various commercial FE codes. VMAP has an active standards community.

The MED file format is the mesh representation for the SALOME platform, the numerical simulation platform of french électricité de France (EDF), and commissariat ‘a l’energie atomique (CEA). As MOAB MED represents a special hierarchy inside a HDF5 container.

For the finite volume method used in CFD, similar approaches exist. CFD general notation system has proven to be the de-facto standard in that field.

For mesh-free methods used to solve the CCM, specialized formats exist, like h5hut or H5Part for point-wise data with an application programming interface for C/C++/ and Fortran. Internally, HDF5 is used. However, other schemes could also be used to store the point-wise data for particle-based methods as well as, e.g., for PD. The PD solver peridigm uses EXODUSII for example and uses point-wise element topologies.

Acknowledgements

The research for this work was carried out partially in the project Virtual Product House im EcoMaT: Startprojekt Multifunktionale äußere Steuerungsfläche (grant in EFRE-Bremen 2014-2020), funded by the German federal state of Bremen and the European Regional Development Fund ERDF, project Virtual Product House im ECOMAT 2.0: “Virtuelle Produktenwicklung für klimafreundliche Flugzeuge” (grant number LU-RAFO4001A, funded by the German federal state of Bremen and the European Regional Development Fund ERDF) and the project PredictECO (grant number 01MN23014B, funded by the German federal ministry for economic affairs and climate action BMWK). The authors gratefully acknowledge the financial support. The authors thank Dominic Bertling for the injection simulation, Hakan Ucan for the virtual autoclave results, and Robert Hein for the support with the process simulation. Furthermore, the authors would like to thank the German aerospace center (DLR) for providing computational resources on the high-performance computing (HPC) system computer for advanced research in aerospace (CARA).

Open Access funding enabled and organized by Projekt DEAL.

Conflict of Interest

The authors declare no conflict of interest.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Keywords

computational structural mechanics, domain-specific language, interoperability, virtual product development

Received: July 4, 2025
Revised: September 30, 2025
Published online:

[1] A. F. Ragani, J. P. Stein, R. Keene, I. Symington, *Tech. Rep.*, McKinsey & Company & NAFEMS, NY, USA **2023**.
[2] F. Lange, A. S. Zakrzewski, M. Rädel, R. W. Hollmann, K. Risse, *Deutscher Luft- Und Raumfahrtkongress (DLRK)*, Bremen **2021**.
[3] M. Rädel, D. P. P. Delisle, D. Bertling, R. Hein, T. Wille, *Deutscher Luft- Und Raumfahrtkongress (DLRK)*, Bremen **2021**.
[4] R. W. Hollmann, A. Schäfer, O. Bertram, M. Rädel, *Deutscher Luft- Und Raumfahrtkongress (DLRK)*, Bremen **2021**.
[5] M. Rädel, R. W. Hollmann, A. Schäfer, A. Schuster, M. Alder, F. Lange-Schmuckall, *Deutscher Luft- Und Raumfahrtkongress (DLRK)*, Dresden, Germany **2022**.
[6] B. Boden, J. Flink, N. Först, R. Mischke, K. Schaffert, A. Weinert, A. Wohlan, A. Schreiber, *SoftwareX* **2021**, *15*, 100759.
[7] F. Dressel, A. Doko, *Deutsche Gesellschaft Für Luft- Und Raumfahrt - Lilienthal-Oberth E.V.* **2021**.
[8] A. S. Zakrzewski, F. Lange-Schmuckall, M. Rädel, R. W. Hollmann, *New Results In Numerical And Experimental Fluid Mechanics XIV* (Eds: A. Dillmann, G. Heller, E. Krämer, C. Wagner, J. Weiss), Springer Nature, Switzerland **2024**, pp. 196–205.
[9] C. Krombholz, F. Kruse, M. Wiedemann, *J. Large-Scale Res. Facil.* **2016**, *2*, 1.
[10] A. Tripmaker, W. Fröhlingsdorf, H. Ucan, M. Bludszuweit, *5. Fachkongress Composite Simulation*, Hamburg, Germany **2016**, https://www.composite-simulation.de/.
[11] European Union Aviation Safety Agency (EASA), “Certification specification cs-25: Large aeroplanes, amendment 28,” Certification Specification CS-25 Amendment 28, EASA, Cologne, Germany,12 **2023**.
[12] R. W. Hollmann, A. Schäfer, O. Bertram, M. Rädel, *CEAS Aeronautical J.* **2022**, *13*, 979.
[13] M. Rädel, C. Willberg, D. Krause, *Composites Part B: Eng.* **2019**, *158*, 18.
[14] N. Schlömer, *Meshio: Tools for mesh files* **2025**.
[15] A. Kaszynski, *Pyansys: Pythonic interface to MAPDL* **2021**.
[16] S. Doyle, *PyNastran: A Python-Based Interface Tool For Nastran’s File Formats* **2024**.
[17] M. Rädel, A. Hauffe, K. Wolf, *Deutscher Luft- Und Raumfahrtkongress DLRK*, Bremen, Germany **2011**.

[18] C. Willberg, J.-T. Hesse, A. Pernatii, *SoftwareX* **2025**, *31*, 102168.
[19] A. Schuster, M. Rädel, J. Lefèvre, *VMAP Userforum* **2025**.
[20] W. C. Mills-Curran, A. P. Gilkey, D. P. Flanagan, *Exodus: A Finite Element File Format For Pre- And Postprocessing, Sandia Report SAND87-2997*, Sandia National Laboratories, Albuquerque, New Mexico, USA **1988**.
[21] V. R. Y. Larry, A. Schoof, *Exodus Ii: A Finite Element Data Model," Sandia Report SAND92-2137*, Sandia National Laboratories, Albuquerque, New Mexico, USA **1992**.
[22] T. J. Tautges, R. Meyers, K. Merkley, C. Stimpson, C. Ernst, *MOAB: A mesh-oriented databaseSAND2004-1592*, Sandia National Laboratories, Albuquerque, New Mexico, USA **2004**.
[23] V. Mahadevan, I. Grindeanu, R. Jain, P. Shriwise, P. Wilson, *Moab V5.2.1* **2020**.
[24] LLNL, *Silo User's Guide Version: 4.10*, Lawrence Livermore National Laboratory, Livermore, CA **2014**.
[25] D. Bertling, R. Kaps, E. Mulugeta, *CEAS Aeronautical J.* **2016**, *7*, 577.

