Simulation Data Management

RECOMMENDATION

Integration of Simulation and Computation
in a PDM Environment (SimPDM),
PSI 4, Version 2.0

ProSTEP iViP

Abstract

This recommendation is intended to standardize the integration of simulation and computation data in a product data management environment (PDM environment) in the form of simulation data management (SDM) and to define the communication processes between a simulation data management system (SDM system) and CAE systems as well as between an SDM system and other data management systems within the PDM environment.

With this purpose in mind, the recommendation defines use cases and functional requirements for simulation data management and a metadata model to describe simulation and computation data. The metadata model is modularly designed to allow a custom-tailored solution for simulation data management. Furthermore this recommendation describes necessary functionalities, clustered towards use cases. Functionality and use cases are modular, too. This means that it is possible to customize SimPDM to suit the specific requirements of an enterprise.

The recommendation does not define an interchange data format between different CAE applications.

The recommendation consists of several documents:
- The main recommendation document (the document in hand)
- Annex A: Glossary (included in the main document)
- Annex B: Business process diagrams (simulation data management for MBS, FEM and CFD)
- Annex C: Core data management functionality
- Annex D: System communication functionality
- Annex E: Data model

The annexes B, C, D, E are available for download as separate documents on the websites of the ProSTEP iViP Association (www.prostep.org) and VDA (www.vda.de).

Disclaimer

ProSTEP iViP Recommendations (PSI Recommendations) are recommendations that are available for general use. Anyone using these recommendations is responsible for ensuring that they are used correctly.

This PSI Recommendation gives due consideration to the prevailing state-of-the-art at the time of publication. Anyone using PSI Recommendations must assume responsibility for his or her actions and acts at their own risk. The ProSTEP iViP Association and the parties involved in drawing up the PSI Recommendation assume no liability whatsoever.

We request that anyone encountering an error or the possibility of an incorrect interpretation when using the PSI Recommendation should contact the ProSTEP iViP Association (psi-issues@prostep.com) immediately so that any errors can be rectified.

Copyright

This recommendation is also published by the VDA, with the same title and version.

This recommendation has been developed and is supported by the VDA and the ProSTEP iViP Association.

Acknowledgment

Our thanks go to all the companies and their staff who were actively involved in drafting this recommendation and for the many constructive suggestions received. The following companies and research institutes were involved:

Altair Engineering GmbH, Audi AG, BMW AG, CADFEM Gesellschaft für computerunterstützte Konstruktion und Berechnung mbH, Daimler AG, Das Virtuelle Fahrzeug Forschungsgesellschaft mbH, DiK TU Darmstadt, Dr. Ing. h.c. F. Porsche AG, :em engineering methods AG, IBM Deutschland GmbH, IMI Universität Karlsruhe, MAN Turbo AG, MDTVision GmbH, MSC.Software GmbH, PTC GmbH, PDTec AG, PROSTEP AG, Schaeffler KG, Siemens PLM Software, T-Systems Enterprise Services GmbH, Volkswagen AG.

Table of Contents

Figures

Abbreviations, definitions, references

Abbreviations, definitions:

See Annex A for a list of relevant abbreviations, terms and definitions.

References:

- VDA Recommendation: VDA 4967 Integration of Simulation and Computation in a PDM Environment, Version 2.0, 2008
- ProSTEP iViP Association: SimPDM Implementation Guideline, 2009
- ProSTEP iViP Association White Paper: Simulation Data Management, 2008
- Association for Standardization of Automation and Measuring Systems: Open Data Services, Version 5.1.1, July 2006
- NAFEMS: Project AUTOSIM. Homepage: www.autosim.org, September 2007
- SIMDAT: Grids for Industrial Product Development. Homepage: www.scai.fraunhofer.de/simdat.html, September 2007
- VIVACE Project: Aeronautical Collaborative Design Environment with associated Processes, Models and Methods. Homepage: www.vivaceproject.com, September 2007
- DIN: PAS 1013: MechaSTEP- STEP data model for simulation data of mechatronic systems. Beuth-Verlag, Berlin, 2001
- Fischer, M., Sachers, M.: CC8 Recommended Practices, Version 1.3, December 2002

1 General

1.1 Preamble

An increasing number of companies have introduced and are using product data management systems (PDM systems) to control engineering data storage and to manage engineering workflows. In particular, the integration of product structure / bill of materials and CAD data has already reached an advanced level. In contrast to this, the integration of simulation and computation data and results in the product data management environment have not reached the same level of evolution. One of the resulting disadvantages is that the results of virtual product validation often fail to comply with the ongoing maturity of product design and development. Other disadvantages are the length of time required to gather data for use in the simulation and the low frequency of data reuse.

To increase the efficiency of virtual product validation and decrease the time to market, companies are striving to improve the synchronization and integration of the technical (CAD) engineering process and the virtual product validation process (simulation and computation). This implies an integration of simulation and computation data in the PDM environment.

The objective of the joint VDA and ProSTEP iViP Association Project Group SimPDM was to develop a common technical solution concept and recommendations for the integration of simulation and computation data (CAE) in a PDM environment. The function of the project group was to gather experiences and requirements from process users, system vendors and research institutes involved in the areas of product data management and simulation/computation. The recommendation is the official document presenting SimPDM project group results.

Within this recommendation, the general term for all data management systems within the PDM environment is xDM system. This term is used to refer to the application for the management of CAD data, CAE data, testing data or any other data related to product development processes. Applications used for the management of simulation and computation data are specifically called SDM systems within this recommendation.

1.2 Objectives of the recommendation

This recommendation serves to standardize the integration of simulation and computation data in a product data management environment (PDM environment) in the form of simulation data management (SDM) and to define the communication processes between an SDM system and CAE systems as well as between an SDM system and other data management systems in a PDM environment.

For this purpose, the recommendation defines use cases, functionalities and functional requirements for simulation data management as well as a metadata model to describe simulation and computation data. The metadata model is modularly designed to allow a custom-tailored solution for simulation data management.

The recommendation does not define an interchange data format between different CAE applications.

The recommendation does not predefine or prefer any kind of data and system integration technology or any kind of commercially available data management systems and system integration facilities.

The recommendation shall be understood as an open and commonly available requirement specification for the implementation of simulation data management and for the integration of simulation and computation data in a PDM environment, defined and accepted by the members of SimPDM project group.

1.3 Compatibility with preceding versions

The document is version 2 of the recommendation for the integration of simulation and computation in a PDM environment (SimPDM). It completely covers the scope of version 1 of the recommendation, although it may slightly vary in content and structure. In addition to version 1, version 2 also contains an extended data model (post processing issues added) and a detailed and formal description of core simulation data management functionality and synchronization and integration functionality.

1.4 Document structure

Chapter 2 introduces business use cases for simulation data management whose requirements have been investigated. Annex B provides scalable process diagrams for each business use case in a large readable format. Chapter 3 introduces the detailed requirements for the integration of simulation and computation in a PDM environment. Chapter 4 describes core data management functionalities. Annex C describes the core functionalities on a more detailed level. Chapter 5 describes system communication functionalities. Annex D describes the integration and communication functionalities on a more detailed level. Chapter 6 introduces the SimPDM metadata model at an overview level. The detailed description of class attributes and relationships between classes is provided by a set of external HTML documents as mentioned in and referenced by Annex E. Annex A provides an auxiliary glossary.

2 Business use cases and processes

This chapter introduces business use cases and processes for simulation data management.

2.1 Introduction

For the purpose of the recommendation, the requirements of the following exemplary business use cases have been investigated:

- Crash simulation (by applying FEA)
- Load case simulation (by applying MBS)
- Design verification (by applying CFD)

Generally considered, all three business use cases above have a similar general workflow from the start to the end of the simulation. The overall steps of this general workflow are as shown in Figure 1.

This general workflow may be processed in iterative loops if the results are not sufficiently sophisticated or appropriate.

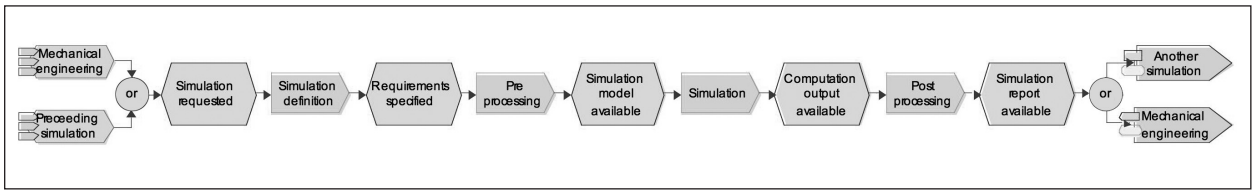


Figure 1: Simulation business process

Depending on the applied simulation task, the business use cases may vary slightly in detail depending on activities to create simulation domain specific information, for example, and may be configured to meet context-specific process requirements.

The following subsections describe each of the three exemplary business use cases in detail. The process diagrams for the use cases are each presented on a single page in Annex B.

2.2 Crash simulation (by applying FEA)

A crash simulation of an entire vehicle or parts of a vehicle is actually performed by a finite element analysis (FEA), which uses the numerical technique termed 'finite element method' (FEM). The process diagram given in Annex B1 illustrates the data and workflow of a crash simulation.

Process overview

Typically, a crash simulation business use case starts with selecting an original CAD geometry (CAD model). The geometry is simplified in one or more steps and a derived geometry is created and applied to the requirements of the simulation task. This process step is performed for each part of the simulation amount, taking into account the specific requirement for the part. A FEM mesh is created for each derived CAD file. The meshes are assembled into modules continuing right through to the assembly of complete vehicles. The meshes defining the topology are supplemented by parameters defining the scaling and the properties of the topology and linked to the load case. This results in a simulation model or input deck which is the input for the solver. The original analysis result data may be processed by certain and specific templates to obtain the key result, i.e. the result which represents the actual solution to the simulation task. The key result of a crash simulation may be improved by re-executing the simulation with updated / optimized input data.

2.3 Load case simulation (by applying MBS)

The recognition of load cases considering, for example, generally expected roadway conditions, is a precondition for several design processes, e.g. the dimensions of bearings, damper elements or similar. A load case simulation may be performed by a multi-body simulation (MBS) or by a finite element analysis (FEA). The process diagram given in Annex B2 illustrates the data and workflow of a load case simulation by applying a multi-body simulation.

Process overview

Typically, a load case simulation starts by gathering the required data from the engineering domain. The actual preprocessing begins with the selection of subsystems which are linked to a template file defining the topology of the system. Subsystems are assembled into modules; modules may be assembled to form larger systems or even complete vehicles. The modules are supplemented with parameters defining the scaling and the properties of the topology and are linked to the load case. The result is an assembly. The assembly may be specifically applied to the simulation task by linking a maneuver to the assembly to complete the input information for the simulation (input deck), which is the input for the actual solver. The original analysis result data may be processed by certain and specific templates to gain the key result, i.e. the result which represents the actual solution to the simulation task. The results of a load case simulation may be improved by re-executing the simulation with updated / optimized input data.

2.4 Design verification (by applying CFD)

A fluid simulation is applied, for example, to examine the functional design verification for the air cooling unit and is performed by computational fluid dynamics (CFD). The process diagram in Annex B3 illustrates the data and workflow of a fluid simulation.

Process overview

The fluid simulation is slightly similar to the crash simulation since it is based on a similar modeling method, i.e. the finite volume elements. Unlike the crash simulation, the fluid simulation is not based on meshed rigid or flexible bodies (mechanical parts), but on meshed volumes, for example, the volumes of air flowing within the tubes and pipes of a cooling unit. The surrounding parts are determined by their boundary geometry and by their thermodynamic influence on the fluid. The original analysis result data may be processed by certain and specific templates to gain the key result, i.e. the result which represents the actual solution to the simulation task. The results of the design verification may be improved by re-executing the simulation with updated / optimized input data.

3 Requirements for simulation data management

This chapter summarizes the requirements for simulation data management systems.

3.1 Introduction

The recommendation specifies a metadata model which serves as the basis for the customizing of an xDM system to allow the management of simulation and computation data. The realization of the metadata model is applied to the so-called macro level of the xDM system.

The basic xDM system architecture consists of two levels. Figure 2 shows the micro and macro level of the xDM system. Metadata are stored at the macro level, whereas the micro level is used for application data management.

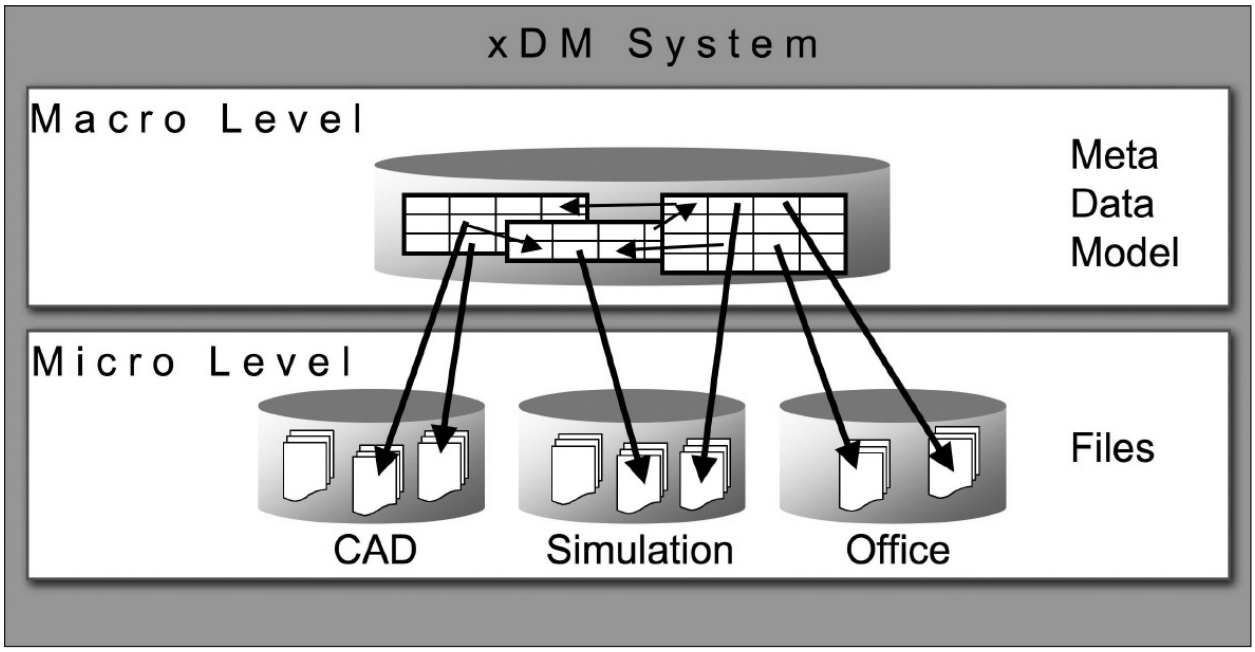


Figure 2: Macro and micro level in the xDM system

Data describing application data are termed metadata. Metadata refer to the application data documents. Metadata can contain identifying, describing, and organizational and status-oriented information as well as information displaying changes over time or information specific to the creating system. Usually, metadata are managed in a separate database termed the metadatabase.

Metadata include, among others:
- the author of a document
- date of creation
- check-in
- path to the physical file
- release degree, etc.

Application data are the actual data described by the metadata. Application data may include a 3D CAD model, a specification, a computation model or a simulation result.

The product metadata model defined at the macro level of the xDM system is termed a meta model because it defines the relations between metadata. It also defines which application data are managed by the xDM system, e.g. data with load cases, boundary conditions and results. The physical data of the meta model objects are saved at the micro level. The metadata of the macro level are connected logically to the physical data of the micro level as seen in Figure 2.

The core of the meta model is the possibility to display the different structures of different systems. Besides the bill of material structure (often termed product structure), the depiction of the CAE model structure is essential. The links between the different structures, i.e. their elements and attributes, shall also be defined in the meta model.

3.2 Requirements

The following subsections describe the requirements of simulation data management, taking both metadata model requirements and functional requirements into consideration, i.e. which actions shall be available in an SDM system.

3.2.1 Expandability

Two different characteristics of expandability shall be considered with regard to specifying and implementing an SDM system:

- Expandability of the metadata model within an SDM system
- Expandability of the stored simulation and computation data

3.2.1.1 Metadata model expandability

The implemented metadata model shall be flexible enough to allow changes to stored simulation and computation data at any time. Expandability applies to:

Elements

The elements defined by the metadata model shall be expanded, if required, without losing existing elements or structures.

> *Example: A new damper element is available in the MBS. The new element shall be represented either by existing elements within the metadata model or by a new element to be defined.*

Attributes

Element attributes shall be expanded, if required, at any time without losing content of existing attributes.

> *Example: A new attribute state is to be defined for an existing element rigid body. Some attributes already exist, such as name, mass, centre of gravity, etc. The existing attributes shall not be deleted when the new attribute state is defined.*

Domains

The domains already considered by the metadata model shall be expanded, if required, at any time with new domains.

> *Example: The MBS, FEA and CFD domains already considered shall be expanded by adding domain "Analytical Methods".*

3.2.1.2 Expandability of stored simulation and computation data

Simulation and computation data already stored in the SDM system shall be expanded by further elements, structures or parameters. Furthermore, the CAE model structures shall also be expanded by a new definition of system borders.

> *Example: Within an SDM system, a CAE model structure of a gearbox is represented. To meet the expandability requirement, it shall be possible to expand the gearbox structure at any time with new elements and structures to expand the system border up to a complete power train. Elements of the existing gearbox structure shall be retained.*

3.2.2 Reproducibility

An executed simulation or computation shall be reproducible at any time, if required, from within an SDM system. This requirement is closely related to the versioning requirement (see chapter 3.2.4).

The SDM system shall be able to freeze the state of relevant objects and relations related to a specific simulation or computation. Among others, the following objects and their relations shall be stored reproducibly in an xDM system:

- The simulation model (in the form of a CAE model structure)
- Constraints
- Start conditions
- Solver settings
- The input deck, identification of the master input deck
- The hardware environment used to process the computation (NB: the processors floating comma accuracy might influence the computation result)
- Results

In the following, the entirety of these objects is called analysis.

Within an SDM system, it is unnecessary to store the complete development history of input decks and models. One of the functions of the SDM system is the reproducible documentation of the above mentioned objects at the time of simulation. The versioning requirement (see chapter 3.2.4) ensures clear object identification together with its version.

> *Example: The computation of Oct. 5th, 2006, shall be reproduced. The simulation engineer reproduces the state of computation from that date within the PDM system (though progress was made). The PDM system filters the CAE model structure and shows all valid versions of the specified objects of Oct. 5th, 2006.*

3.2.3 Base lining

Base lining is analogous to the freezing of a certain configuration, i.e. the current version state of all the objects mentioned in chapter 3.2.2 and their relations. Development may continue and create new versions of these projects. However, when the base line is recalled, the stored configuration becomes available again. Base lining may be performed by setting off an attribute (cf. Figure 3).

Creation of defined base lines within an xDM system is necessary to release computations. It is therefore possible to relate design base lines and computation base lines.

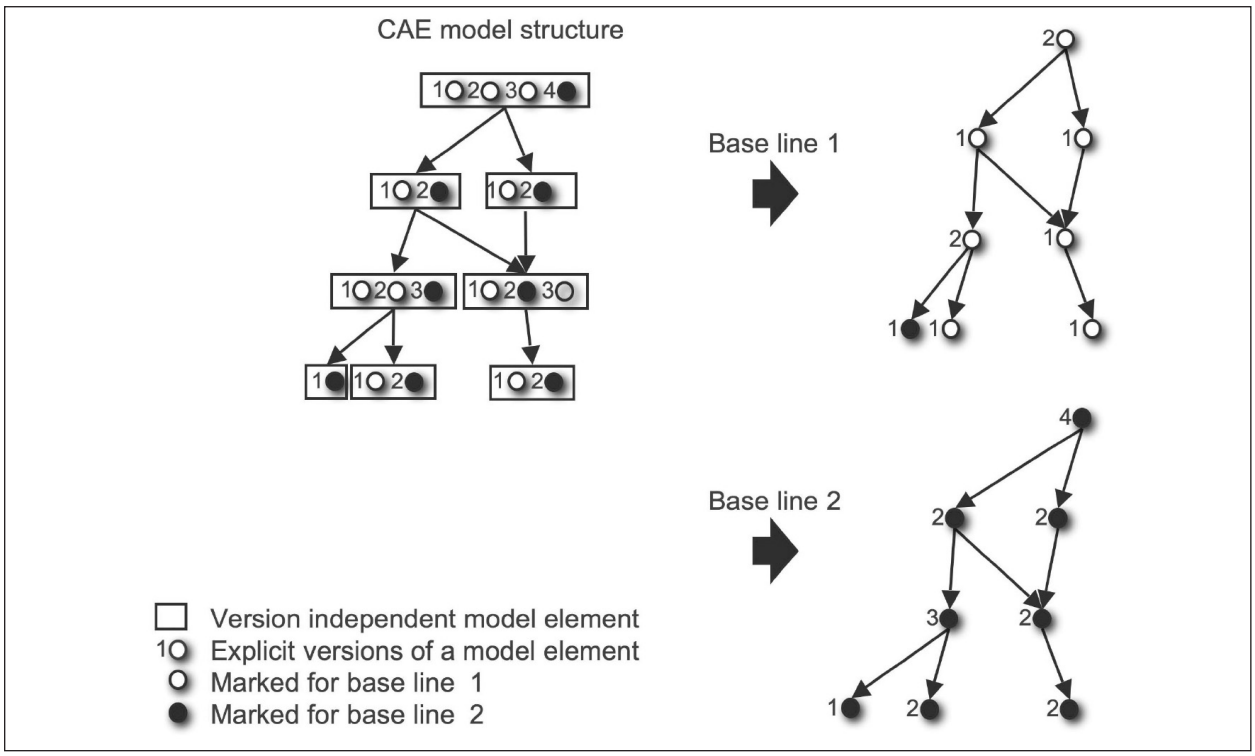


Figure 3: State creation of a CAE model structure

Example: The design engineer defines a product structure base line (e.g. by freezing) and gives a notification to the simulation engineer to start his computation based on the product structure. In the course of time different version states of the CAE model structure and its related objects are created. The simulation engineer can create computation base lines independently from the design engineer. One of these base lines can be released and can be related to a product structure state.

3.2.4 Storability in versions

All objects within an SDM system describing a simulation or computation must be storable in separate versions. The simulation engineer is not bound to the versions of the product structure.

Example: The simulation engineer receives a defined state of the product structure (e.g. gearbox). Based on this state, he starts his computation and his own versioning. After the end of a computation its version state is frozen and linked to the product structure (cf. Figure 4).

The SDM system shall be able to handle its own conventions for computation versioning independently from design.

Example: In Figure 4 the computation versions are named a.0, a.1, a.2, etc. In Design, the versions are numbered 1.0, 1.1, 2.0, 2.1, etc. consecutively. Computation version a.0 refers to design version 2.0. The computation result has version number a.2 but also refers to design version 2.0.

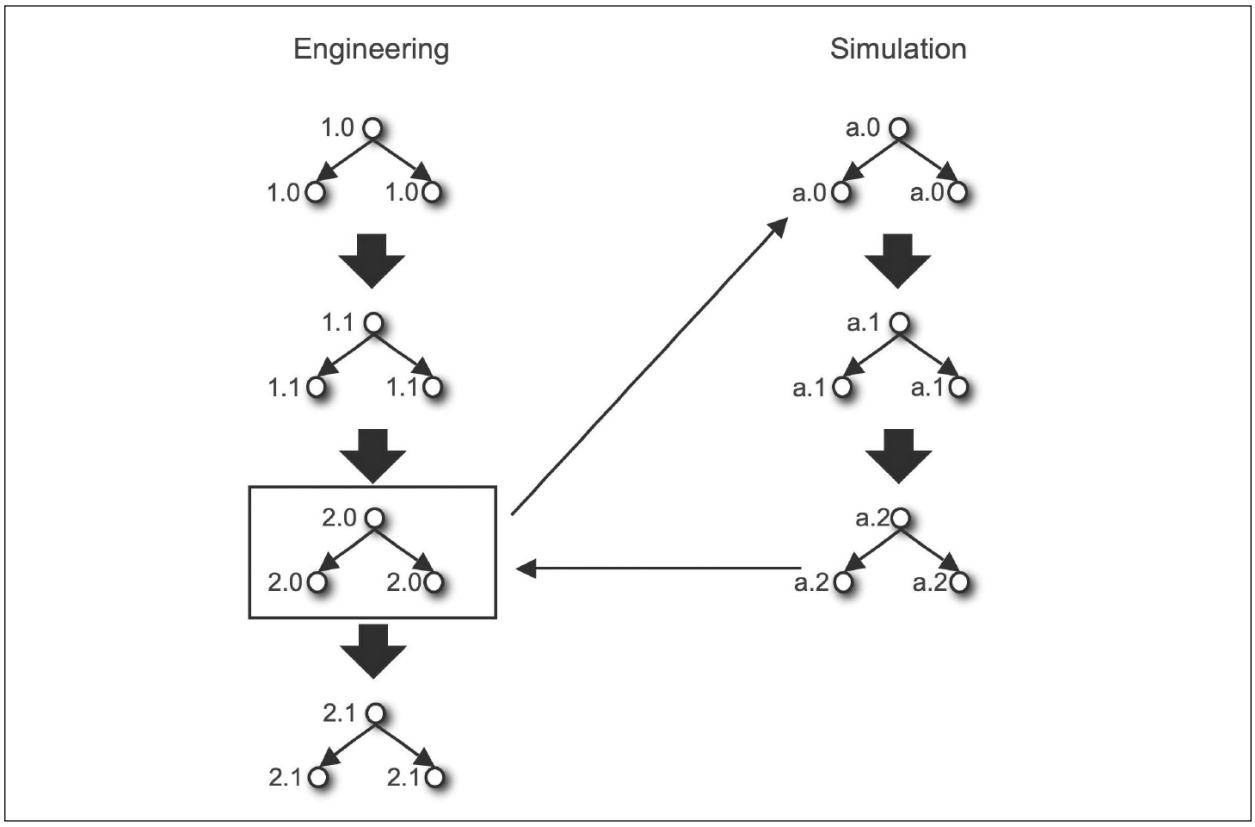


Figure 4: Different versioning mechanisms for design and computation (inner/outer Loop)

3.2.5 Configurability

Two different configurability characteristics require consideration to specify and implement an SDM system:

Configurability as an extension

Within simulation and computation, configurability is understood as the possibility to extend the computation model manually, i.e. the simulation engineer shall be able to add partial models to his computation model manually. A consequence may be that the basic geometry information of the computed state is not completely represented in the backbone PDM system.

> *Example: A simulation engineer is to perform the crash analysis for the type series SP200 in an early phase of product development. He obtains all the necessary information provided by the design engineer from the PDM system. In the early phase there is no information about the new type series engine. The computation engineer shall be able to search for the engine of the predecessor type series SP100 within the PDM system and then add that information to his computation model. The simulation engineer used a state of the type series SP200 with an engine of type series SP 100. This configuration is not represented in a PDM system. In this case, it is not possible to return results to a PDM system.*

Configurability as the possibility of combination

Configurability is also the possibility of combinations, i.e. the simulation engineer shall be able to assemble and run analyses with alternative combinations of models, partial models, load cases, solver settings, etc. The CAE model structure of a certain simulation variant must always remain free of variants. Nevertheless there may be a configurable CAE model structure stored in the SDM system.

> *Example: A simulation engineer is to build up a computation model. He can then carry out various analyses by combining the model with different load cases.*

The configurability of the product structure within a backbone PDM system, e.g. variant control, is not covered by the configurability requirement.

3.2.6 Manipulability/changeability

For simulation engineers, there shall be two main domains within a PDM system that allow manipulation / changes.

CAE model structure

The simulation engineer shall be able to manipulate the CAE structure, i.e. by adding, deleting elements, restructuring, adding new structures, changing elements versioning elements, etc., assuming he possesses the appropriate authorization. See also 3.2.1.2

PDM product structure in early phases

In certain situations, especially in early phases, the product structure may not yet be defined. Nevertheless simulation tasks may already be in process. In this case, the simulation engineer may be authorized to define a product structure or its attributes. For instance, this may be the optimization of a position. Consequently, the simulation engineer shall be able to define a change to the product structure or attributes, assuming he possesses the appropriate authorization. Changes are based on computation results like new positioning or mass information. The simulation engineer shall not be permitted to delete or restructure elements of the product structure.

> *Example: In an early phase of product development, an MBS simulation is to be conducted to specify the position of interfaces and therefore the position of the product module. This position information shall be supplied from an MBS system, i.e. the CAE model structure, to the product structure.*

It is not part of the recommendation to specify a notification service for design engineers. As a result, the functionality of the PDM shall be used.

3.2.7 Variable granularity

It shall be possible to define the CAE model structure within an SDM system with sufficient flexibility to be able to vary the granularity depending on a specific computation tasks.

> *Example: For the simulation of a complete vehicle, an MBS system is defined with only a few elements. These elements shall be related to the product structure and to the real units. Many 1:n relations are created, i.e. n product structure elements correspond to one MBS model element. For a detailed simulation of a module, e.g. a front suspension, CAE model structure granularity is usually much finer, possibly as much as a cardinality of 1:1, i.e. one product structure element corresponds to one MBS model element. The MBS model may even be more finely defined than the product structure, e.g. the front suspension may be composed of two rigid bodies.*

Variable granularity also refers to the CAE model structure management. It shall be possible to manage files with simulation data content (black box concept) as well as manage the simulation data content itself (the parameter management concept) by addressing parameters within the contents.

> *Example: In an early phase of development, several car body variants are analyzed for their eigenfrequencies. For this purpose, several parameters such as sheet metal thickness, material properties, etc. are varied and analyzed. For design verification, a documented computation of a crash analysis is required, whereas it is suitable to manage the meshed car body geometry as a black box file. In this case, access to nodes and elements exceeds system performance.*

3.2.8 Relations between product structure elements

Although the CAD related product structure and the element structure defined within CAE models are very different from each other, it shall be possible to recognize which simulation data element refers to which element within the CAD related product structure. Thus, the CAE model structure elements shall be related to the appropriated CAD related product structure elements.

> *Example: Within the front suspension product structure, there are five units with five article codes. In the MBS model, only one rigid body is to be used for the front suspension for the overall simulation. Therefore, five relations are defined from the product structures which all refer to the same rigid body front suspension. At these relations, additional information is to be defined within the PDM system, such as performing actions (addition of masses by an external program), change notifications, depending relations, validity of relations, etc.*

3.2.9 Synchronization points

CAD product development and CAE engineering perform their work in parallel workflows. As they are both working on the same issue – the product to be developed– they need to update each other several times during development. These are synchronization points within the development process, which can occur more frequently than milestones.

Therefore it shall be possible to define synchronization points at which CAE model structure and CAD product structure are synchronized. At these points, frozen states of both structures are created and related to each other.

The synchronization point definition usually refers to a base line and to a certain point within the workflow.

3.2.10 Parametrical storage and management

The simulation or computation data stored within an SDM system and consisting of the objects mentioned in chapter 3.2.2 shall be storable and manageable parametrically within the PDM system.

> *Example: Parametrical storage and management may apply to a model, its elements, a load case or any other CAE model structure model element. For example, an MBS model can be computed with new parameters, e.g. rigid body's masses. In another example, the variable parameters are limited to the load case definition, such as the alteration of load cases.*

3.2.11 View generation

If simulation and computation data are stored within the same data management system as the CAD (mechanical engineering) related product data, a view generation concept may be practical to provide a focus on different product structures, for example the mechanical engineering view or the simulation view on product data / product structure (Figure 5).

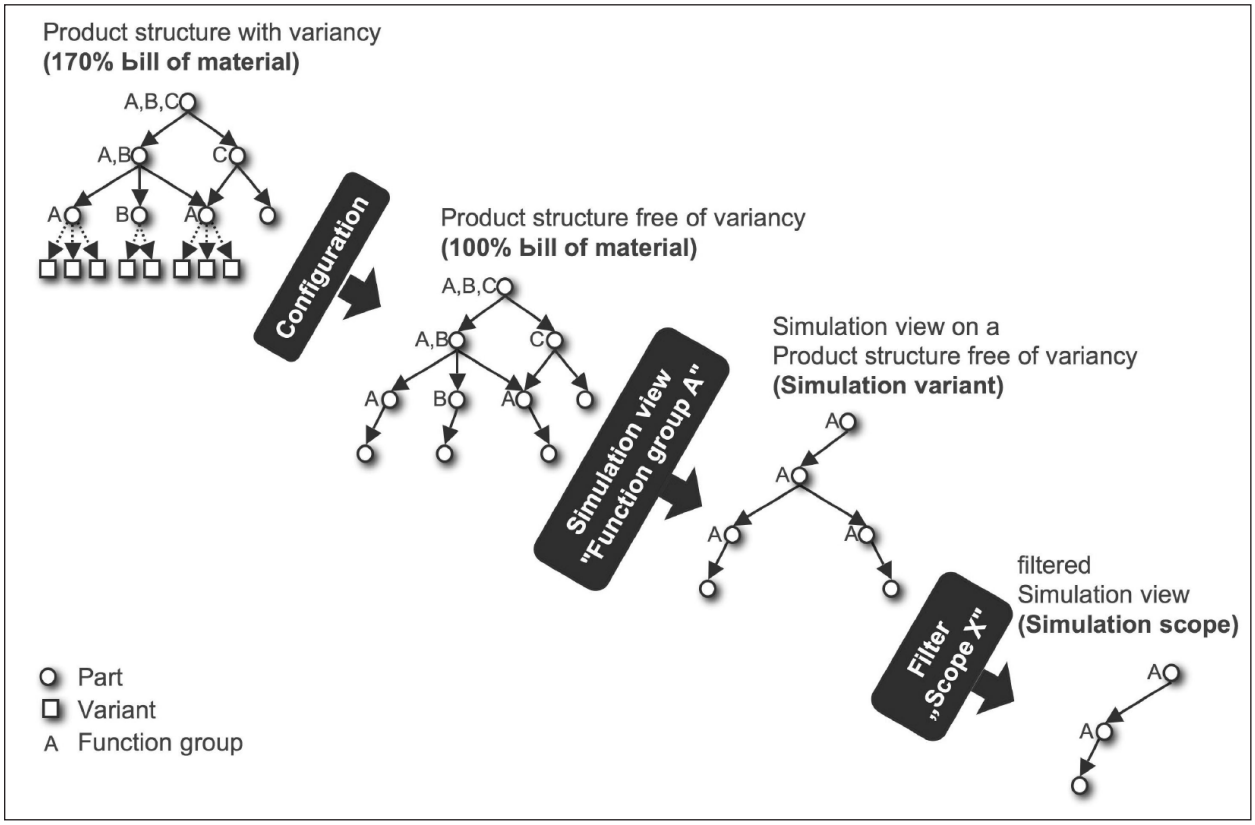


Figure 5: Views and filters

3.2.12 Filter

A filter definition may prove practical to provide different simulation tasks with specifically defined simulation amount based one simulation variant (configuration).

> *Example: After selection of a computation variant (e.g. right-hand-drive vehicle, type series ABC), computation filters are to be defined to start the computation. These computation filters shall allow the selection of the computation data (e.g. front suspension) depending on the computation job (e.g. behavior in acceleration).*

It shall be possible to define these filter functionalities across domains, i.e. a computation filter for a computation data shall be used for the FEA, CFD as well as for the MBS model structure.

4 Core data management functionality

This chapter describes the core functionality required for simulation data management clustered as SDM Use Cases.

4.1 Introduction

Managing simulation and computation data requires standard functionalities. These include user management, role management, file management and workflow and process management. These functionalities are not covered or redefined by the recommendation since they could be assumed and considered as base data management application functionalities. This chapter describes the specific core functionalities required for simulation data management, assuming the existence of the standard functionalities mentioned.

The described functionalities are clustered in SDM use cases. The SDM use cases are modular. Depending on specific application context requirements, not all functionalities of the SDM use case are necessary in each context. Moreover, not all SDM use cases are necessary for a certain application.

Within the description of core functionalities, processes and activities concerning versioning are not mentioned. Nevertheless, versioning (and all related tasks, like version propagation) is an important task. As these tasks are enterprise specific, this recommendation does not describe any versioning processes, not even as a reference process. The explicit versioning of simulation data and the management of version information for simulation data is taken into consideration by the data model.

The following list contains the necessary roles for simulation data management:

- CAE integration interface: Item of software allowing communication between a CAE tool and the simulation data management system
- Key user: User with a central role in a certain department because of his specific design knowledge
- Load case library administrator: User who administrates the content of a load case library. He is not necessarily involved in simulation process itself
- Model library administrator: User who administrates the content of a model library. Not necessarily involved in the simulation process itself
- Other databases, e.g. a PDM system: Database which stores information regarding product or product development. The database contributes or receives the information. It does not necessarily perform the simulation process automatically
- Simulation engineer: User who performs the simulation process itself
- xDM integration interface: Item of software allowing communication between a simulation data management system and other xDM systems within the product development environment

The following definition of use cases does not claim to be complete. It covers the main issues. In addition, the given functionalities do not claim to be complete. They also cover the main issues. Generally, the metadata model is the fundamental base definition for all described functionality.

The order of the required use cases and functionalities listed below does not imply a mandatory order for the use cases or functionalities to be performed by a data management system. The use cases are listed in alphabetical order.

4.2 Use case Administration Management

4.2.1 Use case purpose

Administration management comprises the definition and the attachment of administrative data, such as the date of creation, within a SimPDM-compatible simulation data management system.

The purpose of administration management is to provide additional administrative data for information nodes in the data management system. The management of administrative data is not actually specific simulation data management functionality. It is common data management functionality. It supports, for example, the traceability of manipulations or the controlling of accessibility and changeability of simulation and computation data.

4.2.2 Actors involved

- Simulation engineer

4.2.3 Required functionality

- Create new administrative data
- Delete existing administrative data

See Annex C for a detailed description of the required functionality.

4.2.4 Workflow description

Since the description of the required functionality in the annex already takes into consideration the fact that data can not exist without the administrated information node, a detailed workflow description is not needed at this point.

4.2.5 Notes and remarks

The functionality of the use case Administration Management might be covered by general system functionality and is not simulation data management specific.

4.3 Use case Analysis Classification Management

4.3.1 Use case purpose

Analysis classification management comprises the functionality for managing analysis types and analysis type hierarchies within a SimPDM-compatible simulation data management system.

The definition of analysis types and analysis type hierarchies is an optional concept of simulation data management and provides the possibility to classify analyses by an analysis type specification. For example, an analysis type called 'crash' may be defined to classify all kinds of crash simulations using the finite element analysis method. Another analysis type may be called 'front crash', rear 'crash' or 'side crash' Since all crash simulations might use the same solver and might have identical setting options but different crash barriers applied as different load case models and even a front crash simulation might be applied with or without passengers, i.e. performed with or without additional passenger models, it might be a good idea to specify analysis type hierarchies as illustrated in Figure 6.

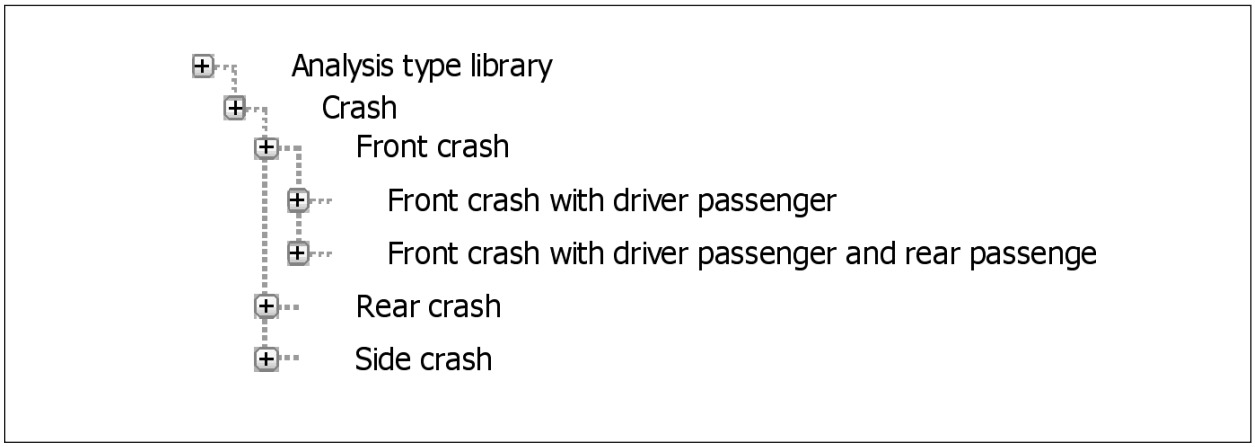


Figure 6: Analysis type hierarchy for an analysis classification concept

The analysis classification may be useful for supporting the application of standard processes, the definition and application of default settings or the application of predefined configurations, which are common specific types or subtypes of analyses.

4.3.2 Actors involved

- Key user

4.3.3 Required functionality

- Create new analysis type
- Delete existing analysis type
- Create new analysis type hierarchy
- Delete existing analysis type hierarchy

See Annex C for a detailed description of the required functionality.

4.3.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Extending analysis type classification system	Step 1: Create new analysis type Step 2: Create new analysis type hierarchy Note 1: If step 2 is left out, the new analysis type is a root type by default
Reducing analysis type classification system	Step 1: Delete existing analysis type Step 2: Delete existing analysis type hierarchy if the analysis type is not a root type Note 1: Step 1 and step 2 are interrelated steps and can not be performed separately if the analysis type to be deleted is not a root type Note 2: If the analysis type to be deleted has subtypes, all the subtypes are also deleted and step 1 and step 2 have to be applied on all subtypes Note 3: The deletion of information may be restricted by system functionality or object dependencies
Moving analysis type within the classification system	Step 1: Delete existing analysis type hierarchy Step 2: Create new analysis type hierarchy Note 1: Step 1 and step 2 are either interrelated steps, or the analysis type which has been moved becomes a root type by default Note 2: If the analysis type to be moved has subtypes, all the subtypes are also moved Note 3: The changing of information may be restricted by system functionality or object dependencies

4.3.5 Notes and remarks

Although the SimPDM data model does not provide an explicit approval and applicability concept, it is conceivable that system functionality has to be used for approving and withdrawing analysis types.

4.4 Use case Analysis Definition Management

4.4.1 Use case purpose

Analysis definition management comprises the functionality for creating, versioning and classifying analyses within a SimPDM-compatible simulation data management system.

The purpose of analysis definition management is to provide project task driven management of all simulation and computation data. All simulation and computation data relating to one specific simulation task can be accessed – assuming sufficient access rights have been granted – by browsing down from the analysis node. This includes all model and topology information, output requests, load cases, external files, results, setting, etc. for all versions of the analysis.

4.4.2 Actors involved

- Simulation engineer

4.4.3 Required functionality

- Create new analysis
- Delete existing analysis
- Create new analysis version
- Delete existing analysis version
- Define new analysis version dependency
- Delete existing analysis version dependency
- Classify existing analysis
- Delete existing analysis classification

See Annex C for a detailed description of the required functionality.

4.4.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Creating analysis definition management information	Step 1: Create new analysis Step 2: Create new analysis version Step 3: Classify existing analysis Step 4: Create new analysis version dependency Note 1: Step 1 and step 2 are interrelated steps, and step 1 can not be performed without step 2 Note 2: Step 2 can only be performed alone if an initial version already exists Note 3: Step 3 and step 4 are optional and can be performed alternatively and at any time later on – assuming that sufficient rights for making changes have been granted
Deleting analysis definition information	Step 1: Delete existing analysis version dependency, if applicable Step 2: Delete existing analysis version as often as desired or applicable for a specific analysis Step 3: Delete existing analysis classification Step 4: Delete existing analysis, if desired Note 1: The deletion of information may be restricted by system functionality or object dependencies

4.4.5 Notes and remarks
No further notes and remarks available

4.5 Use case Document Management

4.5.1 Use case purpose
Document management comprises the functionality for managing files as documents within a SimPDM-compatible simulation data management system and associating documents with information nodes.

Many simulation and computation data can be stored in digital files, especially data which are encoded in an application system specific data format. Like PDM systems, the SimPDM concept for simulation data management provides the management of external files containing simulation and computation data. In this case, the data management system not only manages accessibility to the external file but also the association of the external file with information nodes in the data management system.

The overall concept is that a managed information node can have zero or more than zero documents and that a document can have zero or more than zero references to external files.

4.5.2 Actors involved
- Load case library administrator
- Model library administrator
- Simulation engineer

4.5.3 Required functionality
- Create new document
- Delete existing document
- Create new document association
- Delete existing document association
- Attach external file reference
- Delete external file reference

See Annex C for a detailed description of the required functionality.

4.5.4 Workflow description
The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Creating document management information	Step 1: Create new document Step 2: Attach external file reference Step 3: Create new document association Note 1: Step 2 and step 3 may be used alternately, step 2 can be left out
Deleting document management information	Step 1: Delete external file reference Step 2: Delete existing document Note 1: The deletion of information may be restricted by system functionality or object dependencies

4.5.5 Notes and remarks

The functionality of the use case Document Management might be covered by general system functionality and is not simulation data management specific. Only the association of documents to other simulation data management objects is simulation data management specific.

4.6 Use case Load Case Definition Management

4.6.1 Use case purpose

Load case definition management comprises the functionality for creating and assigning loads / load cases.

Within SimPDM, this use case differentiates between detailed and non-detailed load case definitions. In the case of non-detailed load case definitions, load cases are defined within an attached document and assigned to a certain analysis. In the case of detailed load case definitions, it is possible to represent load cases as metadata within the SDM system and, it is also possible to support a load case library, e.g. stored in an xDM system.

Detailed load case definition is realized with the help of the SimPDM property concept.

4.6.2 Actors involved

- CAE integration interface
- Other databases, e.g. dummy or barrier database
- Simulation engineer
- Load case library administrator

4.6.3 Required functionality

- Create new load
- Delete existing load
- Assign property to load
- Delete property assignment from load
- Assign load to a model element
- Delete load assignment from model element
- Assign a model as load (e.g. dummy, barrier)
- Delete load model assignment from load
- Assign load to an analysis
- Delete load assignment from analysis
- Add another detailed load to an analysis
- Delete a detailed load assignment

See Annex C for a detailed description of the required functionality.

4.6.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Assigning a detailed load case to an analysis	Step 1: Create new load Step 2: Assign property to load Step 3: Assign load to an analysis Step 4: Assign load to a model element Note 1: Step 2 and Step 4 can be performed more than once within the workflow.
Assigning a non-detailed load case to an analysis	Step 1: Assign load to an analysis Note 1: The load case information is stored within a document. Please refer to the appropriate use case for further information.
Defining a detailed load for a library	Step 1: Create new load Step 2: Assign property to load Step 3: Assign a model as load (e.g. dummy, barrier) Note 1: Step 2 and Step 3 can be performed more than once within the workflow. Note 2: Step 3 is optional. It is used when the load case includes a model (e. g. a dummy, a barrier or a test track)

4.6.5 Notes and remarks

It is important to use the functionality with the right granularity, depending on the implemented SimPDM granularity.

4.7 Use case Model Assembly Management

4.7.1 Use case purpose

Model assembling comprises the functionality for defining model hierarchy relationships and the related positioning of models within a SimPDM-compatible simulation data management system.

The purpose of model assembly management is to provide management of model structures on the data management level, where each single model within the structure may either be stored as an external file or managed on a topology level within the data management system.

4.7.2 Actors involved

- Simulation engineer
- Load case library administrator
- Model library administrator

4.7.3 Required functionality

- Create new model assembly relationship
- Delete existing model assembly relationship

See Annex C for a detailed description of the required functionality.

4.7.4 Workflow description

Since the description of the required functionality in the annex already takes the dependencies of model assembly structures into consideration, a detailed workflow description is not needed at this point.

4.7.5 Notes and remarks

No further notes and remarks available

4.8 Use case Model Configuration Management

4.8.1 Use case purpose

Model configuration comprises the functionality regarding CAE model configuration in the SimPDM system.

Two main tasks can be identified for model configuration. First of all, an abstract product structure must be defined and managed. Then the procedure of configuration itself must be performed. The first point includes all tasks involved in creating an abstract product structure, which comprises a product, its components and solutions. Assignment of solutions to product models – which represents a sure and concrete solution – is also part of the first point. Configuration itself includes the selection of solutions for a certain product setup as well as the assignment and storage of the selected solutions as a configuration.

Within product development, configuration is an important topic. Several good solutions already exist. Therefore, this is not a major use case within SimPDM. SimPDM only provides some basic mechanisms upon which a configuration system might be based. For example, SimPDM does not provide any configuration logic.

4.8.2 Actors involved

- Simulation engineer
- CAE integration interface
- Other databases, e.g. a PDM system

4.8.3 Required functionality

- Create abstract product
- Delete abstract product
- Create product component
- Delete product component
- Create component hierarchy
- Delete component hierarchy
- Create solution
- Delete solution
- Assign model to solution
- Delete model assignment from solution
- Create configuration within SimPDM
- Assign configuration to BOM
- Delete configuration assignment from BOM
- Add solution to configuration
- Delete solution from configuration
- Delete configuration
- Assign configuration to analysis
- Delete configuration assignment from analysis

See Annex C for a detailed description of the required functionality.

4.8.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Defining an abstract product structure	Step 1: Create abstract product Step 2: Create product component Step 3: Create component hierarchy Step 4: Create solution Step 5: Assign model to solution Note 1: Step 2 to Step 5 can be performed more than once within the workflow; Step 3 is related to two product components and Step 5 is related to a solution.
Creating a configuration for a certain simulation	Step 1: Create configuration within SimPDM Step 2: Add solution to configuration Step 3: Assign configuration to analysis Note 1: Step 2 can be performed more than once.

4.8.5 Notes and remarks

The functionality of the use case Model Configuration Management might be covered by general system or third party system functionality and is not simulation data management specific.

4.9 Use case Model Definition Management

4.9.1 Use case purpose

Model definition management comprises the functionality for creating and versioning models, as well as defining model version dependencies within a SimPDM-compatible simulation data management system.

The purpose of model definition management is to manage simulation model master data within the data management system. The topological definition of a model is not covered by the use case, since the topological definition may be stored as a digital file outside the data management system.

4.9.2 Actors involved

- Simulation engineer
- CAE Integration interface
- Model library administrator
- Load case library administrator

4.9.3 Required functionality

- Create new model
- Delete existing model
- Create new model version
- Delete existing model version

- Connect model version to analysis version
- Disconnect model version from analysis version
- Define new model version dependency
- Delete existing model version dependency

See Annex C for a detailed description of the required functionality.

Use case application	Workflow
Creating model definition management information	Step 1: Create new model Step 2: Create new model version Step 3: Create new model version dependency Step 4: Connect model version to analysis version Note 1: Step 1 and step 2 are interrelated steps, and step 1 can not be performed without step 2 Note 2: Step 2 can only be performed alone if an initial version already exists Note 3: Step 3 is optional
Deleting analysis definition information	Step 1: Disconnect model version from analysis version, if applicable Step 2: Delete existing model version dependency, if applicable Step 3: Delete existing model version, as often as desired or applicable for a specific analysis Step 4: Delete existing model, if desired Note 1: The deletion of information may be restricted by system functionality or object dependencies

4.9.4 Workflow description
The following table contains only major workflows; it does not provide a complete list of workflows.

4.9.5 Notes and remarks
No further notes and remarks available

4.10 Use case Output Specification Management

4.10.1 Use case purpose
Output specification management comprises the functionality for specifying the requested output of an analysis within a SimPDM-compatible simulation data management system.

This is important for tracing elements of interests. Output specification identifies the elements of interest before the simulation run. It places a kind of marker on certain elements so they can be used in post processing.

4.10.2 Actors involved
- Simulation engineer
- CAE Integration interface

4.10.3 Required functionality

- Create new output specification
- Delete existing output specification
- Define new group of interesting elements
- Delete group definition of interesting elements

See Annex C for a detailed description of the required functionality.

4.10.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Defining an output specification	Step 1: Create new output specification Step 2: Define new group of interesting elements Note 1: Step 2 may be performed more than once

4.10.5 Notes and remarks

No further notes and remarks available

4.11 Use case Parameter Association Management

4.11.1 Use case purpose

Parameter management comprises defining simulation and computation data by means of individual parameters and their unit. The purpose of parameter management is to provide the granular description of simulation data such as model element properties or model properties. Since SimPDM supports both the document level and the parameter level for the management of simulation and computation data, parameter management is an optional concept.

4.11.2 Actors involved

- Simulation engineer
- xDM integration interface
- Other database, e.g. a PDM system

4.11.3 Required functionality

- Create new parameter
- Delete existing parameter
- Add a parameter unit
- Delete a parameter unit

See Annex C for a detailed description of the required functionality.

4.11.4 Workflow description

Since the description of the required functionality in the annex already takes into consideration the fact that parameter information can not exist without an information node to which the parameter is assigned, a detailed workflow description is not needed at this point.

4.11.5 Notes and remarks
No further notes and remarks available

4.12 Use case PDM Information Derivation Management

4.12.1 Use case purpose
Geometry derivation comprises the functionality relating to the CAD-PDM data already stored in the SimPDM system.

Usually the domains CAD and CAE work closely together. They share a lot of data, e.g. geometry information or weld point lists. It is absolutely necessary that traceability is ensured, i.e. to know which information coming from CAD domain was used in a specific analysis. Unfortunately, CAD information often cannot be used in the form that it is provided by the CAD domain. A certain data derivation is often necessary, e.g. geometry simplifications for FEM meshing.

Scope of this use case is to support the traceability of incoming data from PDM systems and their derivation into a CAE-applicable manner. Therefore, SimPDM differs with regard to CAD geometry data, bill of materials and connection elements. It is possible to represent and trace this information and its derived children within SimPDM.

4.12.2 Actors involved
- Other databases, e.g. a PDM system
- CAE integration interface
- Simulation engineer

4.12.3 Required functionality
- Create new CAD PDM information
- Delete CAD PDM information
- Derive CAD PDM information
- Delete CAD PDM information derivation
- Assign BOM to analysis
- Delete BOM assignment from analysis
- Create BOM parts
- Deleting BOM parts
- Assign part to model
- Delete part assignment from model
- Specify CAE model of shape representation
- Delete CAE model specification from shape representation
- Assign shape representation to CAE model
- Delete shape representation assignment from CAE model
- Assign connection element list to analysis
- Delete connection element list assignment from analysis
- Create connection element
- Delete connection element

See Annex C for a detailed description of the required functionality.

4.12.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Deriving CAD data	Step 1: Create new CAD PDM Information Step 2: Derive CAD PDM Information Step 3: Assign shape representation to CAE model Note 1: Step 1 and Step 2 may be performed more than once. Note 2: Performing Step 3 more than once (on different derived shape representations) means that different CAE models are based on the same geometry derivation chain.
Deriving a bill of material	Step 1: Create new CAD PDM information Step 2: Derive CAD PDM information Step 3: Assign BOM to analysis Step 4: Create BOM parts Step 5: Assign part to model Note 1: Step 1 and Step 2 may be performed more than once Note 2: Performing Step 3 more than once (on different derived bill of materials) means that different analyses are based on the same bill of material derivation chain. Note 3: Step 4 and Step 5 are optional depending on the implemented SimPDM granularity. They can be both performed more than once.
Deriving a connection element list	Step 1: Create new CAD PDM information Step 2: Derive CAD PDM information Step 3: Assign connection element list to analysis Step 4: Create connection element Note 1: Step 1 and Step 2 may be performed more than once Note 2: Step 4 is optional depending on the implemented SimPDM granularity. It can be performed more than once.

4.12.5 Notes and remarks

No further notes and remarks available.

4.13 Use case Post-Processing Management

4.13.1 Use case purpose

Post-processing comprises the functionality for analyzing and reporting simulation and computation results. Within SimPDM, the generated post-processing data can be managed in different levels of granularity.

4.13.2 Actors involved

- Simulation engineer
- CAE integration interface
- Key user

4.13.3 Required functionality

- Create analysis result
- Delete analysis result
- Create a computation output (result of first order)
- Delete a computation output (result of first order)
- Create a key result (result of second order)
- Delete a key result (result of second order)
- Create a report (result of third order)
- Delete a report (result of third order)
- Create a detailed post processing object
- Delete a detailed post processing object
- Create a template
- Delete a template

4.13.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Creating a report Creating a post-processing template	Step 1: Create analysis result Step 2: Create a computation output (result of first order) Step 3: Create a key result (result of second order) Step 4: Create a detailed post processing object Step 5: Create a report (result of third order) Step 6: Create a detailed post processing object Note 1: Step 4 and Step 6 are optional for use with detailed granularity Note 2: Step 3 and step 4 may be performed multiple times for a certain analysis Note 3: As a report is usually based on more than one result and analysis, Step 1 to 4 may be performed multiple on different analysis.
	Step 1: Create a template Note 1: This task is not daily business. It is usually done by experienced users.

4.13.5 Notes and remarks

It is important to use the functionality with the right granularity, depending on the implemented SimPDM granularity.

4.14 Use case Property Definition Management

4.14.1 Process purpose

Property definition comprises the functionality for creating and versioning properties and assigning property constraints.

4.14.2 Actors involved

- CAE integration interface
- Other databases, e.g. material database
- Simulation engineer

4.14.3 Required functionality

- Create new property set
- Delete property set
- Create new property set version
- Delete existing property set version
- Create new property set version relationship
- Delete existing property set version relationship
- Create new property set structure
- Delete existing property set structure
- Create a property
- Delete a property
- Create property relationship
- Delete existing property relationship
- Create property constraint
- Delete existing property constraint
- Assign property set version to model version
- Delete property set version from model version
- Connect property set version to model element
- Disconnect property set version to model element

See Annex C for a detailed description of the required functionality.

4.14.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Building up a property set	Step 1: Create new property set Step 2: Create new property set version Step 3: Create new property set structure Step 4: Create new property Step 5: Create Parameter Step 6: Apply use case Document Management Step 7: Create property relationship Note 1: Step 1 and step 2 are interrelated steps and step 1 can not be performed without step 2 Note 2: Step 2 can only be performed alone if an initial version already exists Note 3: Step 3 and step 7 are optional Note 4: Step 5 and step 6 are optional depending on the subtype of PROP.property used

4.14.5 Notes and remarks
No further notes and remarks available

4.15 Use case Setting Definition Management

4.15.1 Use case purpose
Setting definition management comprises the functionality for creating and assigning settings.

Within SimPDM, this use case differentiates between detailed and non-detailed setting definitions. In the case of non-detailed setting definitions, settings are defined within an attached document and assigned to a certain analysis. In the case of detailed setting definitions, it is possible to support a setting library.

It is possible to support analysis steps with the specialized setting SETT.analysis_step. This is necessary whenever different succeeding analyses are computed within a single analysis run, e.g. a FEM simulation of different forming steps at a manufacturing simulation.

4.15.2 Actors involved
- Simulation engineer
- Other databases
- CAE integration interface

4.15.3 Required functionality
- Create new setting
- Delete existing setting
- Assign setting to an analysis
- Delete setting assignment from analysis
- Create analysis step
- Delete analysis step
- Add another detailed setting to an analysis
- Delete a detailed setting assignment

See Annex C for a detailed description of the required functionality.

4.15.4 Workflow description
Since the description of the required functionality in the annex already describes a workflow, especially the functionality of assigning a setting to an analysis, a detailed workflow description is not needed at this point.

4.15.5 Notes and remarks
It is important to use the functionality with the right granularity, depending on the implemented SimPDM granularity.

4.16 Use case Topology Element Definition Management

4.16.1 Use case purpose
Topology element definition management comprises the functionality for managing topological model elements and assigning properties to model elements.

The definition of model elements can be used to define elements for a predefined element library or to create completely new or user-defined elements. The distinction between the element node itself and its properties supported by the SimPDM packages property definition management and property association management is established to allow company, context or application-specific definitions of topological elements. For instance, different applications or different companies may define different numbers and different kinds of attributes for a spring. But all the different attribute specifications are supported by the data model. The workflow and element dependencies can be coupled with the TOPO.spring entity, and the attributes of the spring are defined by entities from the SimPDM package PROP.

4.16.2 Actors involved

- Simulation engineer
- Element library manager
- CAE integration interface

4.16.3 Required functionality

- Create new topological element
- Delete existing topological element
- Create topological element relationship
- Delete topological relationship
- Include model element into model
- Delete model element from model

See Annex C for a detailed description of the required functionality.

4.16.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Creating topological element information	Step 1: Create new topological element Step 2: Connect property set version to model element Step 3: Include model element in model
Deleting topological element information	Step 1: Delete model element from model Step 2: Disconnect property set version from model element Step 3: Delete new topological element

4.16.5 Notes and remarks

It is important to use the functionality with the right granularity, depending on the implemented SimPDM granularity.

4.17 Use case Topology Structure Definition Management

4.17.1 Use case purpose

Topology definition comprises the functionality for defining dependencies between model elements and assigning model elements to models.

Topological structure definition management is used to define the topological structure in the and to connect two different models with each other topologically.

4.17.2 Actors involved

- Simulation engineer
- CAE integration interface

4.17.3 Required functionality

- Define interface element
- Delete interface element
- Create topological model connection
- Delete topological model connection
- Assign a surface to a set
- Delete a surface from a set

See Annex C for a detailed description of the required functionality.

4.17.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Creating topological model connection	Step 1: Create topological model connection between elements in the same model
Creating topological cross-model connection	Step 1: Define interface element in model A and in model B Step 2: Apply use case Model Assembling Management Note 1: Step 1 and step 2 may be alternated

4.17.5 Notes and remarks

The topological structure of a model can either be defined by applying this use case or using an external modeler and assigning the model file as an external file by applying the use case Document Management.

It is important to use the functionality with the right granularity, depending on the implemented SimPDM granularity.

5 System communication functionality

This chapter introduces the general SimPDM concepts for synchronization management and CAE connection management. While synchronization management describes the synchronization of parameters, files and properties between different databases of product development (e.g. PDM system or material database and SDM system), CAE connection management describes the integration of a CAE application with a SDM system (e.g. data transfer between SDM system and a specific CAE application).

In order to support the reference processes specified within SimPDM, both aspects shall be regarded as if they were complementary (cf. Figure 7):

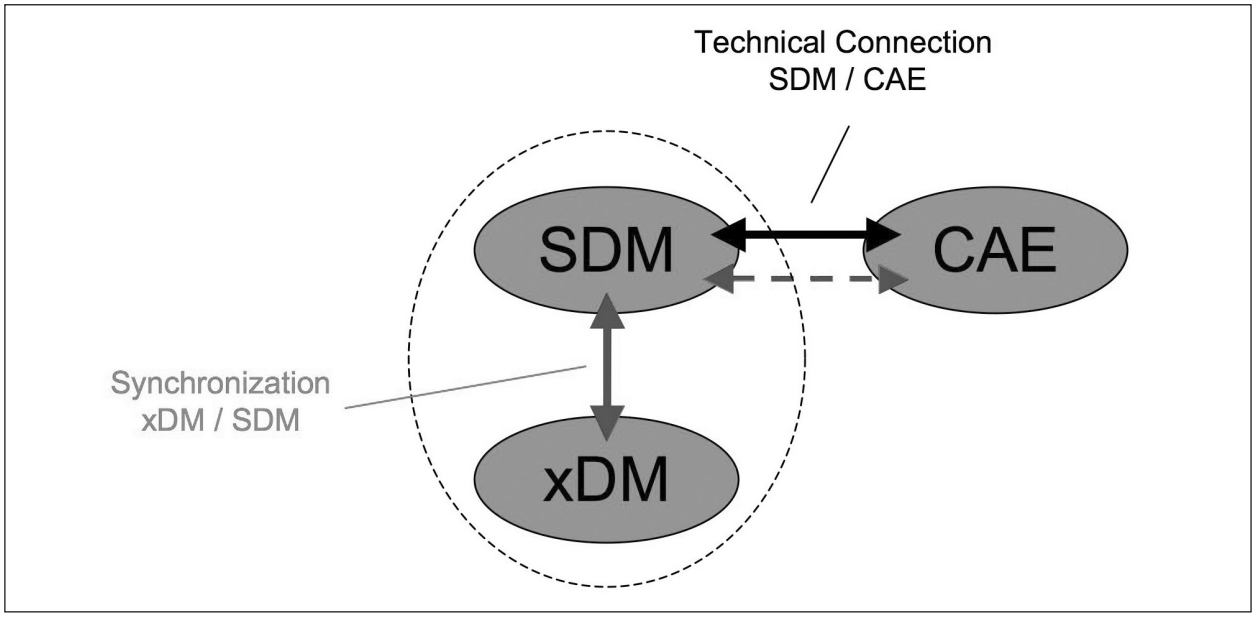


Figure 7: Overview

Processes and activities concerning versioning are not mentioned in the description of system communication functionalities. Nevertheless, versioning (and all related tasks, such as version propagation) is an important task. As these tasks are highly enterprise specific, this recommendation does not describe any versioning processes, even as a reference process. Generally, versioning is possible with SimPDM since several classes are provided in the SimPDM metadata model for this purpose.

The definition of use cases in chapter 5.3 and chapter 5.4 does not claim to be complete. It covers the main issues. In addition, the given functionalities do not claim to be complete. They also cover the main issues. Generally, the metadata model is the fundamental base definition for all described functionality. The order of the required use cases and functionalities listed there does not imply a mandatory order for the use cases or functionalities to be performed by a data management system.

5.1 System communication purpose

Ideally, the CAE simulation process is supported by a simulation data management (SDM) system which, on the one hand, stores and manages the relevant information for the simulation process and, on the other hand, controls the simulation process by launching the specific CAE applications (pre processor, solver, post processor) while transferring the relevant data sets.

In general, the SDM system represents the basic data repository for the whole simulation process. The following information content is typically stored and managed by the SDM system:

- Information emerging from the design process (e.g. CAD geometry to build a simulation model, parameters and properties) and
- Information generated during the simulation process (e.g. FE model, input deck for a solver run or output deck for the generation or derivation of key results).

The SDM information is typically stored in a structural way similar to folders in a file system. Furthermore, SDM systems shall provide additional basic functionality (see chapter 3).

This section specifies concepts and a general approach for the
- Synchronization of parameters between xDM (e.g. product structure in a PDM system) and SDM systems (CAE model structure), and the
- Technical connection between SDM systems and CAE applications (e.g. input and output deck).

Figure 8 shows the general concept of both parameter synchronization between xDM and SDM systems and the connection of CAE applications from a system point of view.

The figure shows some typical information to be transferred between the participating systems during the simulation process. In order to provide the CAE applications with current data, the relevant information (e.g. parameters, files, properties) has to be synchronized between xDM and SDM. For example:
- Product structure information including metadata (ID, name, description, version, revision),
- Positioning information (local transformation matrix) of single parts or assemblies relative to their upper-level assembly,
- Mass data (calculated/estimated weight), local center of gravity (CoG), local moment of inertia (MoI),
- CAD geometry.

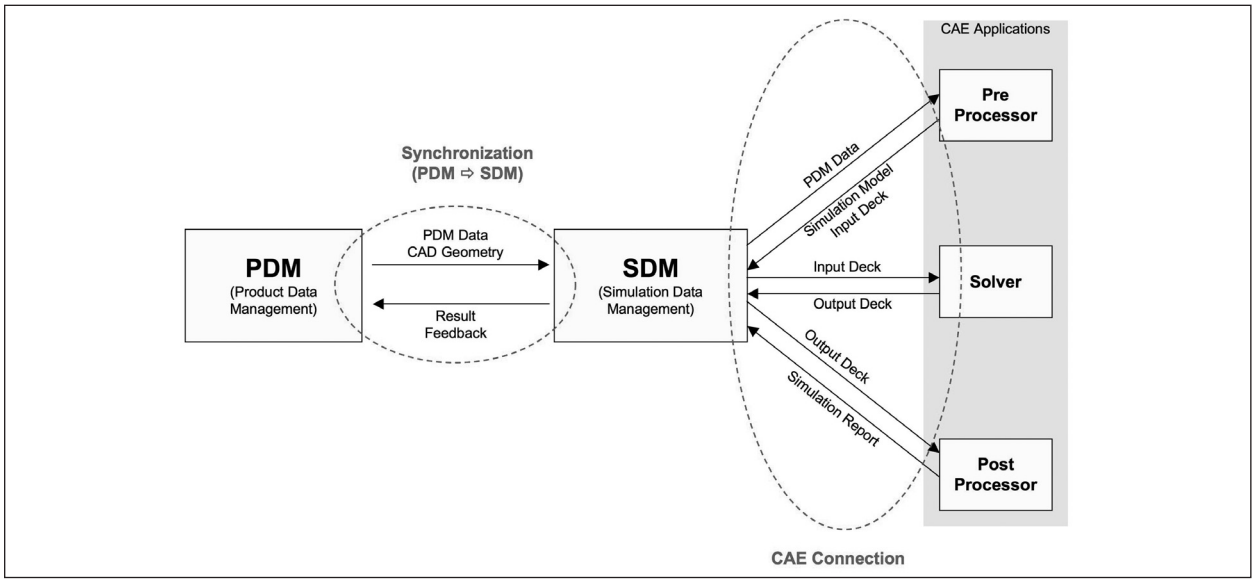


Figure 8: The "Big Picture" (system view)

Ideally, parameter synchronization also covers the (automatic) provision of feedback from the simulation run and transfer back to the xDM system.

With regard to SDM-CAE connection, file-based information exchange plays an important role as the information needed is usually transferred in a native format (e.g. solver-specific input deck, simulation report).

5.1.1 Synchronization management

In order to achieve an effective, integrated simulation process, the objective is to (automatically) provide the SDM system with up-to-date parameters relevant for the simulation. This kind of information is typically stored in xDM systems, e.g. the product structure itself plus additional parameters such as meta information (part ID, version), weight, centre of gravity, moment of inertia, etc. (see above). Information may also be stored in additional systems (e.g. material database). The provision of parameters can be supported automatically or at least semi-automatically (with the necessary user interaction).

With regard to xDM integration, there are several established data exchange standards available that may be used to support the synchronization.

See also chapter 5.2.1.

5.1.2 CAE connection management

Concepts for the connection of CAE and SDM systems are specified based on the SimPDM metadata model. This includes the evaluation of existing technologies regarding data exchange, data integration and system integration in order to support SimPDM processes. Based on defined concepts and identified technologies, the necessary system interfaces are specified in order to read and/or write SimPDM-specific data. Additionally, it defines the way in which the SimPDM metadata model could be used with regard to the interface specification (metadata exchange format, metadata model for SDM systems, etc.).

With regard to CAE connection, there are no established standards available; data transfer is usually 100% native.

See also chapter 5.2.2.

5.1.3 Role of the SimPDM metadata model

This document also provides recommendations concerning the role of the SimPDM metadata model.

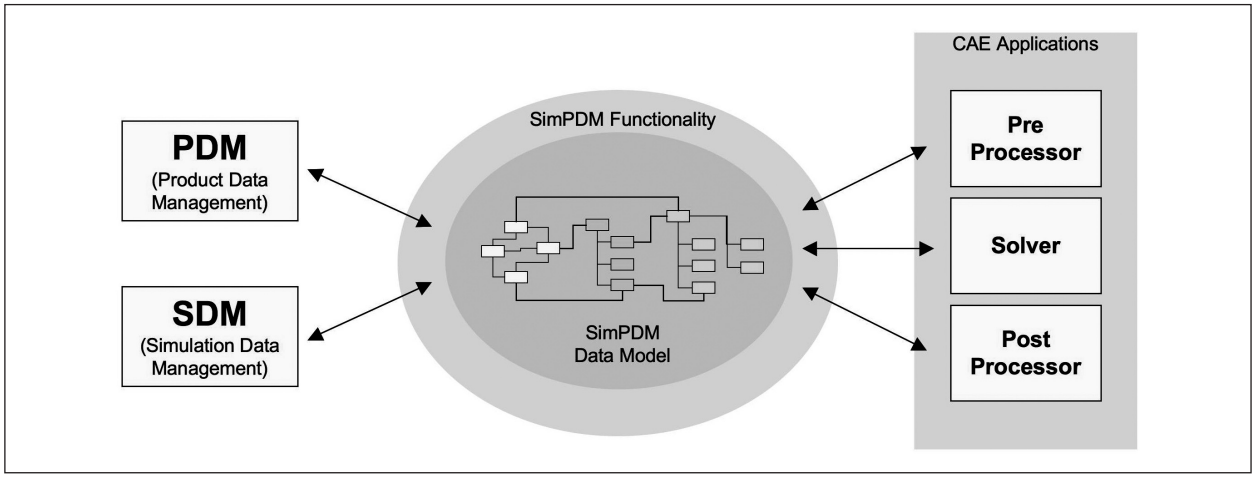


Figure 9: SimPDM metadata model and SimPDM functionalities

The SimPDM metadata model can be used in different ways. On the one hand, a SDM system may provide business logic based on the metadata model; this may be realized by the definition or setup of a SDM system based on the SimPDM metadata model, or by an extension of an existing PDM/EDM system with parts of the SimPDM metadata model. Most xDM systems on the market provide functionality to extend the basic metadata model by specific types.

In general, there are 3 possible options:

- Option 1: 'SimPDM' SDM system, ideally providing business logic based on the SimPDM metadata model (as a whole or a subset of it). This option would be the ideal case as the SDM can represent all information based on the SimPDM metadata model.

- Option 2: 'Extended' PDM system covering the relevant SDM aspects. In this case, an "out-of-the-box" PDM system is used and its original metadata model is extended and customized by SimPDM object types to cover SimPDM aspects.

- Option 3: 'Integrated' PDM/SDM system: In this case, both PDM and SDM are managed in the same system.

Regarding the SimPDM approach, options 1 and 2 are the most relevant to support SimPDM processes.

5.1.4 Information granularity

The SimPDM metadata model shall be the basis for information exchanged between PDM and SDM systems as well as data transfer between SDM system and CAE applications. This can be realized in different granularities:

- All information is represented by data according to the SimPDM metadata model (e.g. load cases, nodes of a FE model etc.)

- SimPDM is used to represent metadata in addition to native files to be exchanged (e.g. metadata describing a solver specific input deck). This typically leads to information redundancy. In this case, consistency between native and metadata has to be ensured (see also chapter 5.2.2.4).

The SimPDM metadata model is NOT foreseen as a format for an information exchange between specific CAE applications.

5.2 General implementation aspects

5.2.1 Synchronization management

The parameter synchronization process could either be supported by a specific external application or directly in the simulation data management environment (SDM system). Furthermore, access may be either online (e.g. using Web Services) or offline (e.g. by physical file exchange).

5.2.1.1 Online integration

Ideally, the SDM system and xDM system are connected by a socket connection. In this case, the synchronization process is controlled via a specific user interface (cf. Figure 10). This solution represents a 100% online integration supported by SimPDM-specific Web Services and typical web-based xDM interfaces. Direct online access to the involved systems is supported by specific interfaces, e.g.:

- SDM system: 'on SimPDM adapted' services based on the SimPDM metadata model and synchronization functionality (see Annex D)
- xDM system: standardized WebService interface (e.g. OMG PLM Services).

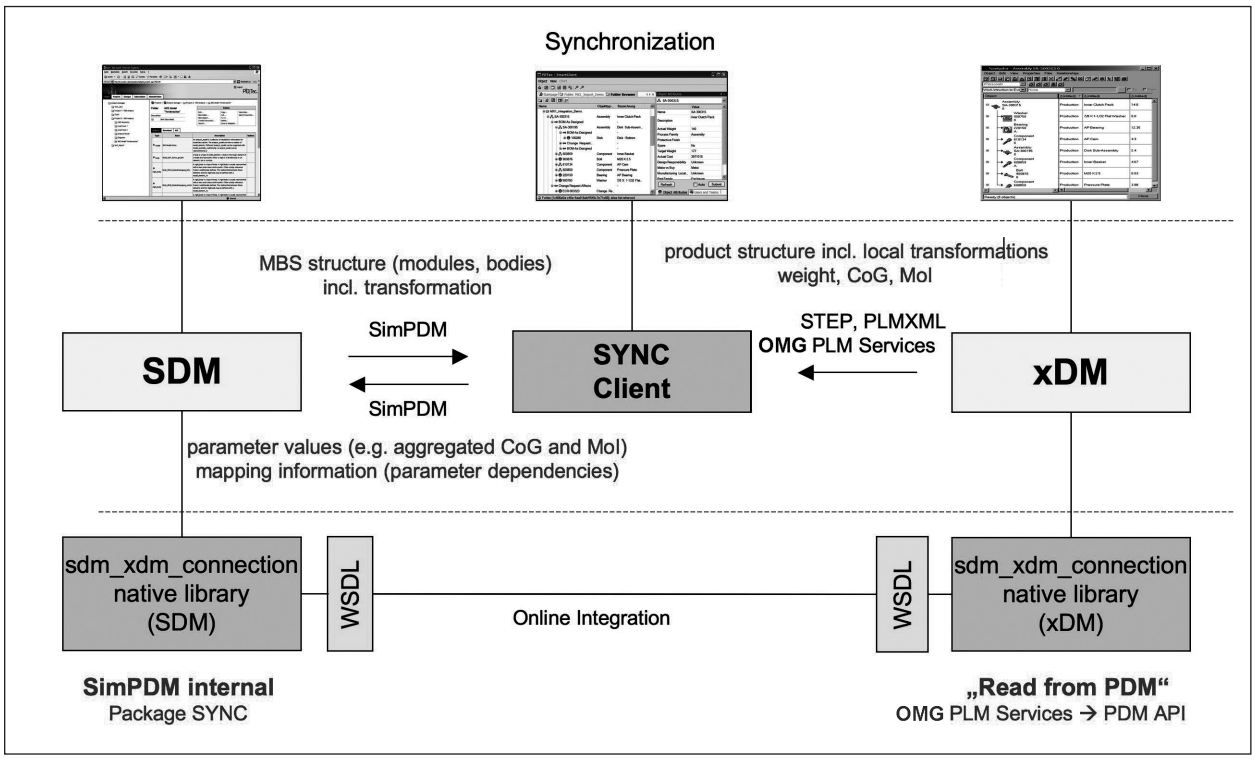


Figure 10: Client GUI and online integration

In the case of parameter synchronization, the synchronization process is performed based on defined dependencies between parameters from the SDM and xDM structures. Based on these parameter dependencies, parameters are 'mapped' and parameter values are calculated. Both dependencies and values must be available in the SDM system.

Information is read from PDM (e.g. via standardized interfaces or as OMG PLM Services). This could be meta information concerning product structure, local transformations, weight, CoG and MoI from objects within the PDM structure.

On the SDM side, the necessary information is represented based on the SimPDM metadata model (a specific part of the metadata model is defined for the representation of synchronization information).

For details concerning the synchronization process see chapter 5.3.

5.2.1.2 Offline integration

Information transfer between SDM and xDM may also be supported by file-based data exchange using a specified (standardized) format and predefined system interfaces, e.g.:

- SDM system: SimPDM interface exporting an output file according to the SimPDM metadata model.

- xDM system: standard interface designed for PDM systems (e.g. STEP AP214 CC6, PLMXML).

The information content to be transferred is the same; the only difference is that there is no online connection between SDM and xDM available.

5.2.2 CAE connection management

With regard to CAE communication, online and offline integration is also possible in general. But in this case, there are no established standards; data transfer is usually 100% native. Thus, in the scope of SimPDM, additional metadata shall be transferred based on an extra data set according to the SimPDM metadata model. As the metadata represents information that is also part of the native file, this leads to data redundancy.

5.2.2.1 SDM interfaces

The SDM system shall provide a SimPDM interface that reads/writes SimPDM-compliant data in XML format. Furthermore, the interface shall be capable of reading/writing files to the hard disk or a digital vault. This may be either an offline interface (data exchange) or online interface (e.g. SimPDM-compliant Web Services).

With regard to the SDM system approach, there are two different scenarios possible (cf. Figure 11):

- SDM system is 100% SimPDM compatible (option 1, 'SimPDM' SDM system); in this case, no data conversion is necessary during import.
- SDM system is covering only a subset of SimPDM (option 2, 'Extended' PDM system or option 3, 'Integrated' PDM/SDM system); in this case, a data conversion is necessary during import.

The SimPDM interface of the SDM system shall perform the consistency check part 3 as described in chapter 5.2.2.4. It checks the incoming metadata and assigns it to existing data nodes or creates new nodes.

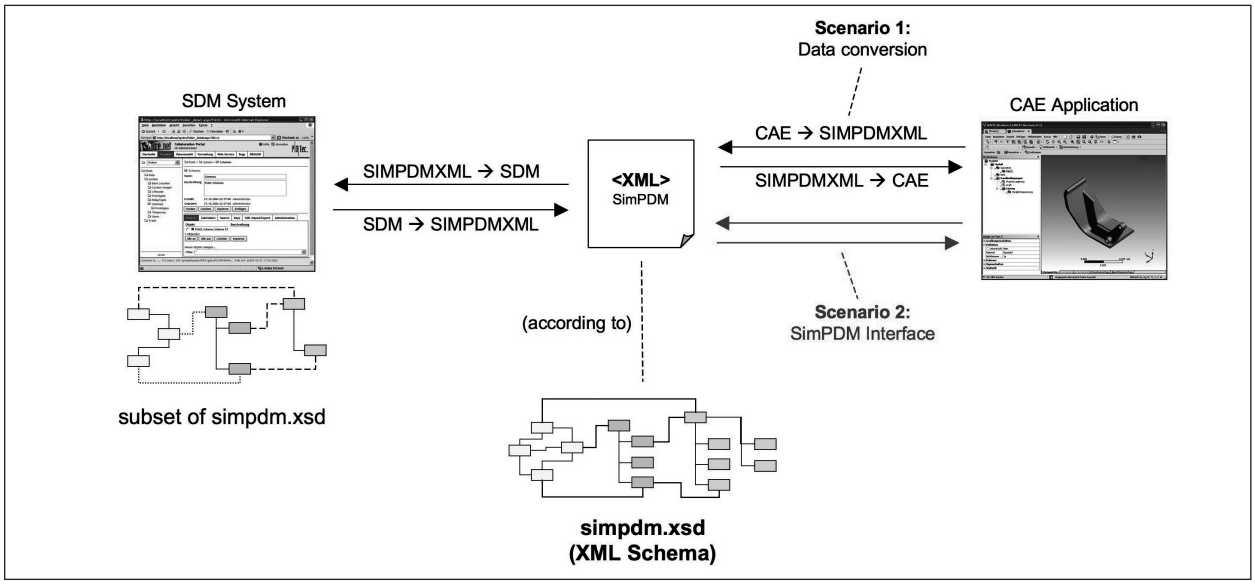


Figure 11: Data transfer process SDM / CAE

5.2.2.2 CAE interfaces

The CAE application shall provide a SimPDM interface that reads/writes SimPDM-compliant data. This could be realized by two different approaches (cf. Figure 11):

Scenario 1: Direct API programming

SimPDM-compliant information is generated by converting native file content. This is typically done by a processor. Conversion functionality may be programmed either by original vendor or by third-party developers.

Critical issues may be release changes and API license problems.

Scenario 2: SimPDM interface

A SimPDM-specific interface is provided by the CAE system vendor. In this context, it is important to note that the SimPDM interface is not foreseen as a CAE-CAE interface.

5.2.2.3 Metadata vs. native data

While exchanging information between SDM systems and CAE applications, there are a number of different possibilities concerning the exchange format. Simulation data may be completely described by a set of data according to the SimPDM metadata model. But, in most cases, not all information is represented in that way; in particular regarding FEM, a lot of information is probably exchanged as native file (e.g. solver specific input deck).

Generally speaking, SimPDM supports both extremes:

- All information is represented as SimPDM-compliant metadata or

- Information is represented by native files plus some meta information in SimPDM format describing the information content.

In order to cover all cases, there are two possible ways of exchanging information.

Case 1: Data transfer

Meta information is exchanged in a container sent from the SDM system to the CAE application in analogy to PDM-CAD integration approaches (cf. Figure 12). This may lead to redundant information (see section on consistency check for details). Meta information is exchanged in a format according to the SimPDM metadata model.

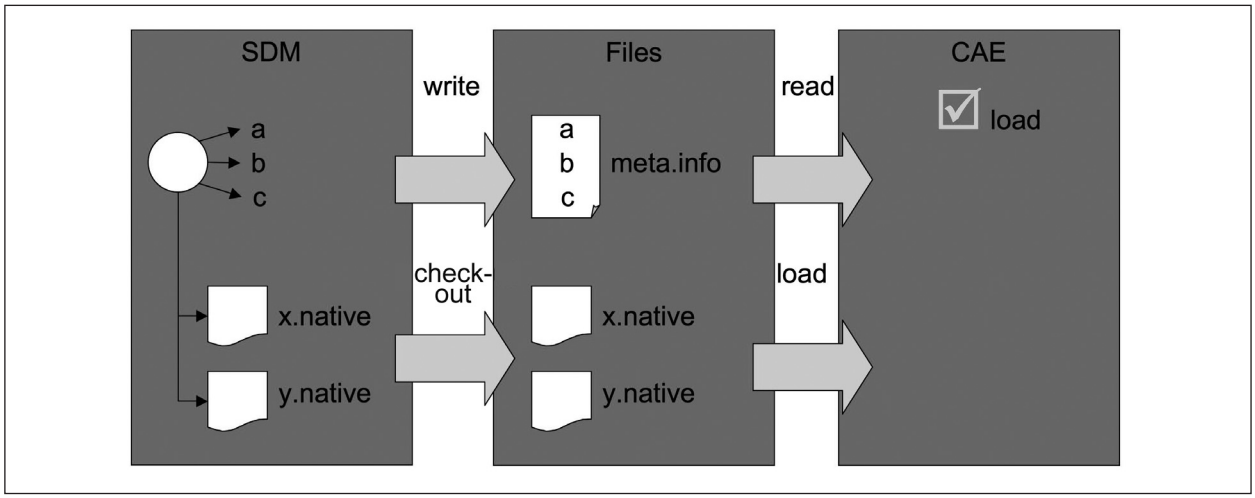


Figure 12: Transfer of metadata between simulation data management and CAE

Case 2: Database integration

The CAE application is directly linked with the SDM database (cf. Figure 13). Information is read directly from the SDM database and is processed in the CAE tool. This solution may cause difficulties if the SDM system and the CAE application are from different vendors. Furthermore, consistency problems may arise in this case (e.g. if some content is stored in the SDM system and other content in the CAE application).

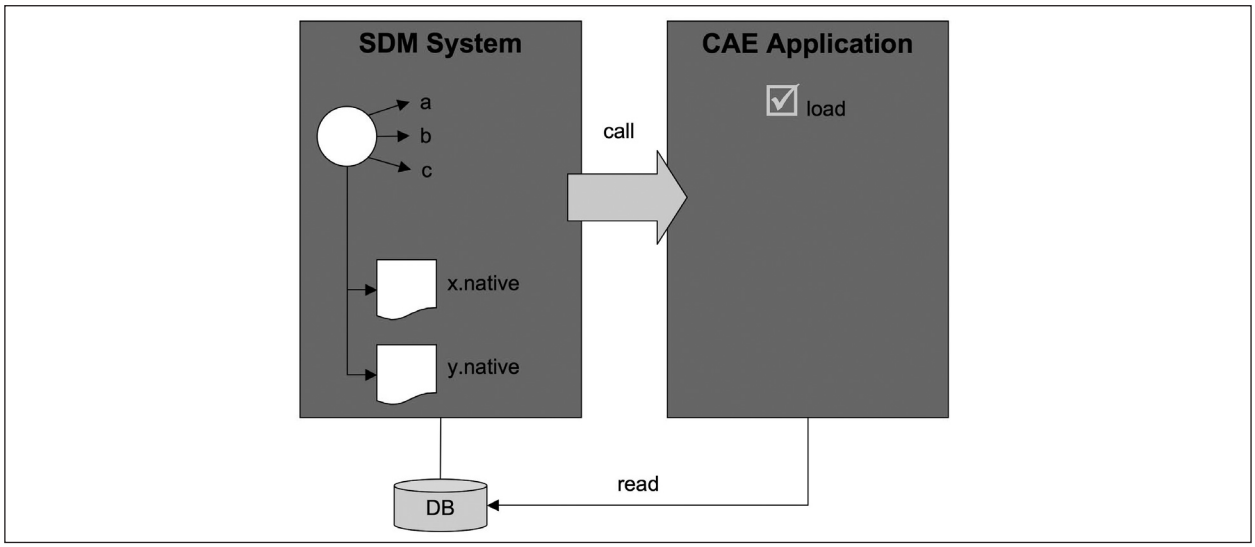


Figure 13: Direct link between the SDM database and the CAE application

5.2.2.4 Consistency check

Due to data manipulation in a specific system, the information content of the metadata and physical files may not be consistent. Therefore, data consistency must be checked during import/export of information between SDM and CAE (cf. Figure 14). Data inconsistency may be caused by

- Changes concerning metadata in the SDM system before checking out a data set for a CAE system or
- Updated native file content in a CAE application.

Conflicts may occur if the data sets are inconsistent.

If the SDM system is defined as the master system (as it is the usual case), changes to meta information are binding while exporting information for a task in a specific CAE system. Thus, the native file content must be updated during import into the CAE application. If information is processed by a CAE application, an updated set of meta information shall be created before checking information back into the SDM system.

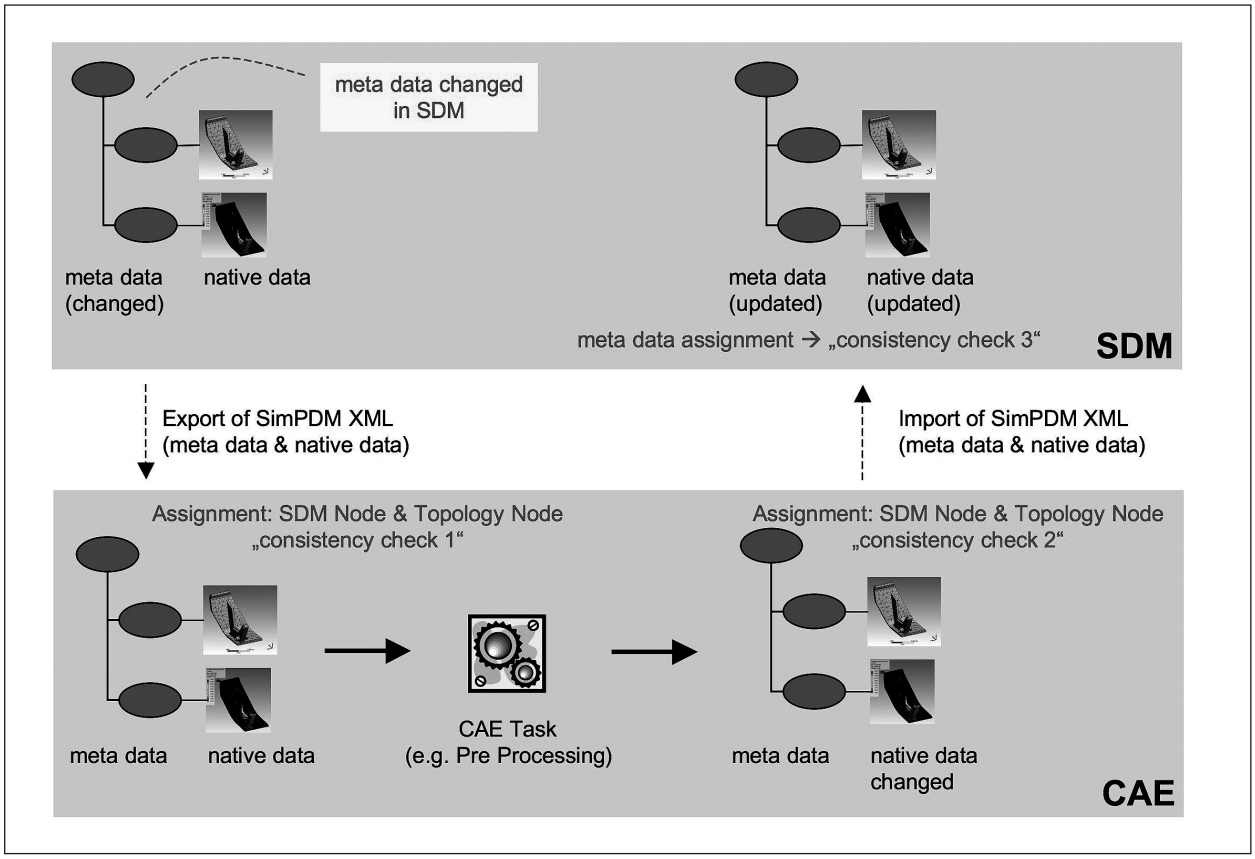


Figure 14: Sequence of consistency checks

A distinction can be made between three different types of consistency check:

Consistency check 1 (to be performed in CAE application on data import)

The aim is to ensure data consistency when work with the CAE application starts. Metadata emanating from SDM may have been changed but these changes do not affect the native data (e.g. in case of weight updates). Therefore, the CAE application shall check on import whether the metadata set is still consistent with the native data file. In case of differences, SimPDM metadata is the master data set.

Consistency check 2 (to be performed in CAE application on data export)

The aim is to ensure data consistency when work with the CAE application ends. Native data emanating from the CAE application has changed. SimPDM metadata shall be updated at the corresponding metadata nodes (e.g. transformation matrix changes or creation of new nodes). Therefore, the CAE application shall generate a new SimPDM metadata set with a corresponding level of detail. In case of differences, native data is the master data set. The native data file is the data set on which the CAE application works. Therefore, no special actions are required here.

Consistency check 3 (to be performed in SDM system on data import)

The aim is to ensure data consistency within the database when work with the CAE application has been finished and the data is re-imported into the SDM system. The CAE application has created new SimPDM-compliant metadata and native data sets. On database import, a consistency check shall assign the imported data to existing or new nodes within the database.

Changes (either concerning meta information or native files) always lead to new versions in the simulation data management system. Inconsistencies shall be checked and reported during import and export.

5.3 Use case synchronization management

This section describes the basic concepts for synchronization management between xDM and SDM systems. In this section, xDM is seen as a data management system from a different discipline, usually mechanical design (then PDM).

5.3.1 Use case purpose

The approach involving (either automatic or semi-automatic) synchronization between the xDM and CAE worlds provides a major benefit concerning simulation data management. This section describes general application cases and provides recommendations for technical implementation.

Generally, the following synchronization steps shall be performed:

1. Set and initialize external parameters, files or properties: Definition of external parameter objects (e.g. from xDM product structure) as basis for an assignment to SDM internal parameters, files or properties.
2. Connect and define dependencies: Assignment of external parameters to SDM parameters (mapping) OR assignment of an external file or external property. Definition of dependencies between internal and external parameters as basis for aggregation (this is only valid for parameter synchronization).
3. Synchronize: Aggregation of information as input data for the simulation process based on functional or mathematical dependencies. This step is only valid for parameter synchronization.
4. Update: Update the information within SDM based on updated values.

Initial mapping shall be performed manually, e.g. assignment of parameters and values such as CAE elements (MBS bodies, geometry for FE models, materials, etc.) and xDM elements. Update shall be possible based on the already defined dependencies.

Results of the synchronization:

- Aggregated parameter values: e.g. aggregated centre of gravity and moment of inertia and
- Mapping information: assignment of SDM and PDM parameters/files (as basis for updates).

Both items of information shall be saved and made available in the SDM system. In the case of a change in the xDM system, synchronization is performed based on the initial assignment (e.g. when the product structure is changed). It shall be possible to access the mapping automatically from the SDM system. Synchronization and consistency checks are initiated by the SDM system.

SimPDM must support the following cases:

Design status “Design Validation”

In this case, a simulation task is started based on specific milestones during the design process. Thus, the design status to be evaluated (e.g. a so-called 100% car) is available in the xDM system.

Design status “CAE Driven Development”

In this case, a simulation task is started based on bilateral agreements between the design and simulation departments. This is typically the case at an early design stage.

The design stage to be evaluated is usually not available in the xDM system and may be fixed individually at a specific point of time.

5.3.2 Actors Involved

- Design engineer
- Simulation engineer
- PDM interface
- Other databases, e.g. (Material DB)

5.3.3 Required functionality

Parameter synchronization:

- Set and initialize external parameter
- Connect external parameter and define dependencies
- Delete external parameter
- Evaluate parameter availability
- Synchronize parameter
- Update external parameter

File synchronization:

- Define and initialize external file synchronization
- Disable external file synchronization
- Evaluate file availability
- Update external file

Property synchronization:

- Define and initialize external property synchronization
- Disable external property synchronization
- Evaluate property availability
- Update external property

See Annex D for a detailed description of the required functionality.

5.3.4 Workflow description

5.3.4.1 Parameter synchronization

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Set and initialize external parameter	Step 1: Objects of type BASE.external_parameter are defined as the basis for parameter synchronization. These objects represent the external parameters (e.g. from PDM) that have to be assigned to internal SDM parameters Step 2: The necessary elements in the xDM structure are identified Step 3: Meta information from xDM is imported into the SDM environment, e.g. via STEP AP214 CC6 or PLMXML ("offline") or PLM Services ("online"). Exemplary information content: • Product structure metadata (ID, name, version, etc.). • Local transformation matrix between part and its next higher assembly • Weight information (e.g. real weight, calculated weight, estimated weight). • Local center of gravity (according to local coordinate system) • Local moment of inertia (according to local coordinate system). Step 4: Parameter values are written to external parameter objects

Use case application	Workflow

Connect external parameter and define dependencies	Step 5: External parameters (from xDM structure) are assigned to internal parameters (from CAE structure). Exemplary information content:
	• MBS structure (modules, bodies) incl. transformations.
	• Transformation matrix between inertial coordinate system (root system of the product) and the coordinate system of each model (location of bodies related to inertial system).
	• Internal parameter values to be aggregated:
	  • center of gravity for MBS body
	  • moment of inertia for MBS body
	Step 6: Parameter dependencies between internal and external parameters from the PDM structure and SDM structure are defined as the basis for aggregation (defining functional or mathematical dependencies between parameters)

Evaluate parameter availability	Step 7: Availability checks have to be performed, e.g. • Assignment completeness (are all parameters assigned and are all elements considered?) • Information availability (are all assigned parameters available, e.g. weight, MoI, CoG, …) • Version check (are the correct versions available?)
Synchronize parameter	Step 8: Synchronized and aggregated values are calculated based on the assignment of the parameters and the dependencies. This function is called automatically as soon as the parameter dependencies are changed Step 9: Aggregated parameter values are written to objects. This may lead to new versions
Update external parameter	(In case of a change in xDM/CAE) Step 10: Updated parameter values are transferred to the SDM system Step 11: Synchronization is initialized automatically based on initial assignment and defined dependencies Step 12: Updated values are aggregated and written to objects

5.3.4.2 File and property synchronization

The following table contains only major workflows; it does not provide a complete list of workflows.

Use case application	Workflow
Define and initialize external file/property synchronization	Step 1: Objects are defined as references to external files or properties. These objects represent the external files/properties (e.g. in PDM) Step 2: The necessary elements in the external systems are identified Step 3: Meta information is imported into the SDM environment Step 4: Values are written to the objects referencing the external file/property
Evaluate file/property availability	Step 5: Availability checks have to be performed, e.g. • File availability (is the assigned external file available?) • Property availability (is the assigned external property available?) • Version check (is the correct version available?)
Update external file/property	In case of a change in xDM/CAE: Step 6: Updated values are transferred to the SDM system Step 7: Updated values are written to objects

5.3.5 Notes and remarks

Synchronization functionality includes parameter synchronization as well as file synchronization and property synchronization.

5.4 Use case CAE connection management

This section describes the basic concepts for the connection of the SDM and CAE application.

5.4.1 Process purpose

Depending on the application context, it shall be possible to exchange various items of information within the scope of granularity and content:

- Different simulation domain (e.g. MBS, FEM, CFD).
- Application context, simulation task (e.g. crash simulation, NVH).

Depending on the phase in the CAE process under review, various items of information are needed and shall be exchanged and/or provided:

- Preprocessing: all information shall be represented. This may lead to data redundancy (meta information and native files) and possible inconsistencies.
- Solving: Mainly file management is needed here. A link shall be stored to assign input data (input deck) with output data (output deck).
- Post processing: in analogy to the solver, mainly file management is needed here.
- Furthermore, key results shall be representable. This may lead to data redundancy (meta information and native files) and possible inconsistencies.

5.4.2 Actors involved

- Simulation engineer
- PDM interface
- CAE interface

5.4.3 Required functionality

- SDM data export (metadata, native data) for CAE application
- CAE data import (metadata, native data) from SDM data base
- CAE data export (metadata, native data) for SDM data base
- SDM data import (metadata, native data) from CAE application

See Annex D for a detailed description of the required functionality.

5.4.4 Workflow description

The following table contains only major workflows; it does not provide a complete list of workflows.

System	Task / Description
SDM system	Step 1: Define simulation task • Post processing • Solver run • Post processing Step 2: Define information content to be transferred to CAE (simulation data) • Metadata • Native data Step 3: Identify CAE application (preprocessor, solver etc.). • Build connection to CAE application • Check-out (export) simulation data and initialize a CAE application. • Check status (approved, in work, etc.). • Transfer data to CAE application. • Launch CAE application. • Load data (e.g. input deck in the case of a solver run).
CAE system	Step 4: Perform import consistency check (part 1). Metadata changed in simulation data management system may be inconsistent with native data. Step 5: Process simulation task in CAE application. Step 6: Save native data, metadata update may be necessary (consistency check part 2).
SDM system	Step 7: Check in (import) results to SDM system (metadata, native data). • Metadata in SimPDM format. • Application-specific native data. Step 8: Perform consistency check (part 3). • Synchronize single nodes. • Save information as new version.

It is assumed that the user is primarily working with the SDM system and the simulation process is controlled by the SDM system. If the user is primarily working with the CAE application, the SDM system is called from the CAE system; the other steps remain the same as above.

5.4.5 Notes and remarks

No further notes and remarks available.

6 SimPDM metadata model

This chapter contains an introduction to the SimPDM metadata model, its modular structure and how to reach an adapted level of granularity.

6.1 Introduction

The approach of the SimPDM metadata model is a description of simulation metadata. Simulation metadata include:

- administration information
- simulation model structure information
- topological model definition
- properties and parameters
- solver settings
- load case definitions
- original input information
- intermediate information such as, for example, meshes
- post processing information, i.e. simulation results and reports

The SimPDM metadata model defines data objects which are needed to describe this information and the relations between different items of information.

The purpose of the SimPDM metadata model is to support the data management processes concerning the simulation data and to manage the configuration of input data (input deck) for the solver systems at data management level, i.e. from the preprocessors. Therefore the metadata model can be used as a high level application metadata model for simulation data management systems, as well as a data transfer format definition the communication between a simulation data management system and a CAE application.

The SimPDM metadata model is not supposed to substitute the input decks for the solver systems. Neither is it focused on the data conversion of different preprocessor data formats, or communication between preprocessors.

6.2 Generic and modular approach

Since the data structures of CAE applications on the market are very specific and the SimPDM metadata model is not focused on certain tools, the SimPDM metadata model is based on a generic design. In addition, the definition of topological elements may apply differently in different companies or simulation domains and this also encourages a generic metadata model design. As an example, a spring as a topological model element is not hard-coded by SimPDM with a certain number of parameters such as spring coefficient, length or diameter. The generic approach allows the definition of spring parameters using the SimPDM property concept. The spring coefficient, the length and the diameter are defined as separate and independent properties and are assigned to a spring object that is grouped as a property set. This allows the definition of a precise quantity of element parameters which are needed to meet corporate and domain specific requirements.

Figure 15 illustrates the generic approach by means of a spring definition. Instead of a hard-coded spring definition with a fixed number of predefined spring parameter, SimPDM defines a bar spring by means of a number of administration metadata. The actual technical parameters of the spring are defined separately and independently, grouped as a set of properties and assigned to the spring object. This flexibility allows different derivations of a spring definition. For example, the definition of a "diameter" parameter may be applied optionally or not, depending on the simulation requirements or the spring definition provided by the solver used.

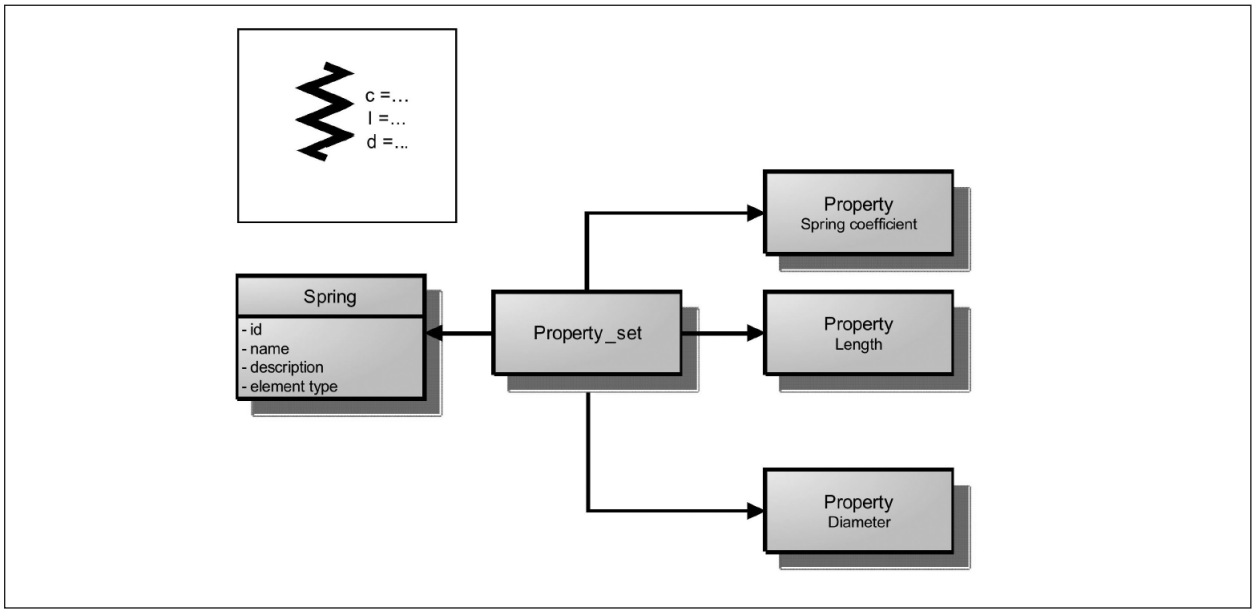


Figure 15: Generic definition of topology model elements

Different companies do not necessarily have the same requirements regarding the granularity of the managed data. For example, companies or departments focused on dynamic systems such as chassis or similar may need to configure and manage their simulation models with a high granularity, including the explicit management of single parameters or the versioning of property sets for certain topological model elements. Other companies may be seeking a version-controlled management of simulation models, files containing simulation input information and analysis results to enhance the reusability of simulation data and the tracking of simulation processes. Considering this wide range of requirement constellations, the SimPDM metadata model has a modular structure that comprises a mandatory basic module and a number of satellite modules. The basic module (called the SimPDM.BASE package) is independent from the satellite modules (packages called SimPDM.TOPO, SimPDM.PROP, SimPDM.CONF, SimPDM.SETT, SimPDM.LOAD, SimPDM.CAD, SimPDM.SYNC and SimPDM.POST).

The packages mentioned are defined by SimPDM metadata model. SimPDM classes and relations are described in chapter 6.4.

SimPDM.BASE

The SimPDM.BASE package contains the analysis definition and management, model structure management, output specification, post processing data management and document management for analysis results. For a detailed description of the package, please refer to the file "BASE.html" in the HTML documentation (cf. Annex E).

SimPDM.CAD

The SimPDM.CAD package contains all classes which are required for the CAD PDM database referencing in terms of, for example, bill of materials, connection point list and shape representation. For a detailed description of the package, please refer to the file "CAD.html" in the HTML documentation (cf. Annex E).

SimPDM.CONF

The SimPDM.CONF package contains the configuration of the model structure for an analysis run, i.e. an analysis version regarding a specific variant of a product to be simulated. For a detailed description of the package, please refer to the file "CONF.html" in the HTML documentation (cf. Annex E).

SimPDM.LOAD

The SimPDM.LOAD package contains the specification of load case definitions. For a detailed description of the package, please refer to the file "LOAD.html" in the HTML documentation (cf. Annex E).

SimPDM.POST

The SimPDM.POST package contains mechanism for the detailed post processing data management. This is mainly by using existing mechanism from the packages SimPDM.CAD, SimPDM.PROP and SimPDM.TOPO. For a detailed description of the package, please refer to the file "POST.html" in the HTML documentation (cf. Annex E).

SimPDM.PROP

The SimPDM.PROP package contains the definition, management and assignment of properties and property sets. For a detailed description of the package, please refer to the file "PROP.html" in the HTML documentation (cf. Annex E).

SimPDM.SETT

The SimPDM.SETT package contains the specification of solver and environment settings. For a detailed description of the package, please refer to the file "SETT.html" in the HTML documentation (cf. Annex E).

SimPDM.SYNC

The SimPDM.SYNC package contains all classes which are required for the parameter synchronization between different data management systems and/or simulation disciplines. Synchronization is fully automated processable. For a detailed description of the package, please refer to the file "SYNC.html" in the HTML documentation (cf. Annex E).

SimPDM.TOPO

The SimPDM.TOPO package contains the topological structure definition of simulation models. For a detailed description of the package, please refer to the file "TOPO.html" in the HTML documentation (cf. Annex E).

6.3 Package dependencies

A company which does not need to manage topology structures, properties or settings may only implement the SimPDM.BASE package. Other companies which need to consider detailed topological structure management may implement additional satellite modules. Figure 16 shows the modular structure of the SimPDM metadata model and the dependencies between the packages. It also indicates that the SimPDM.BASE package has only minor dependencies on the other packages.

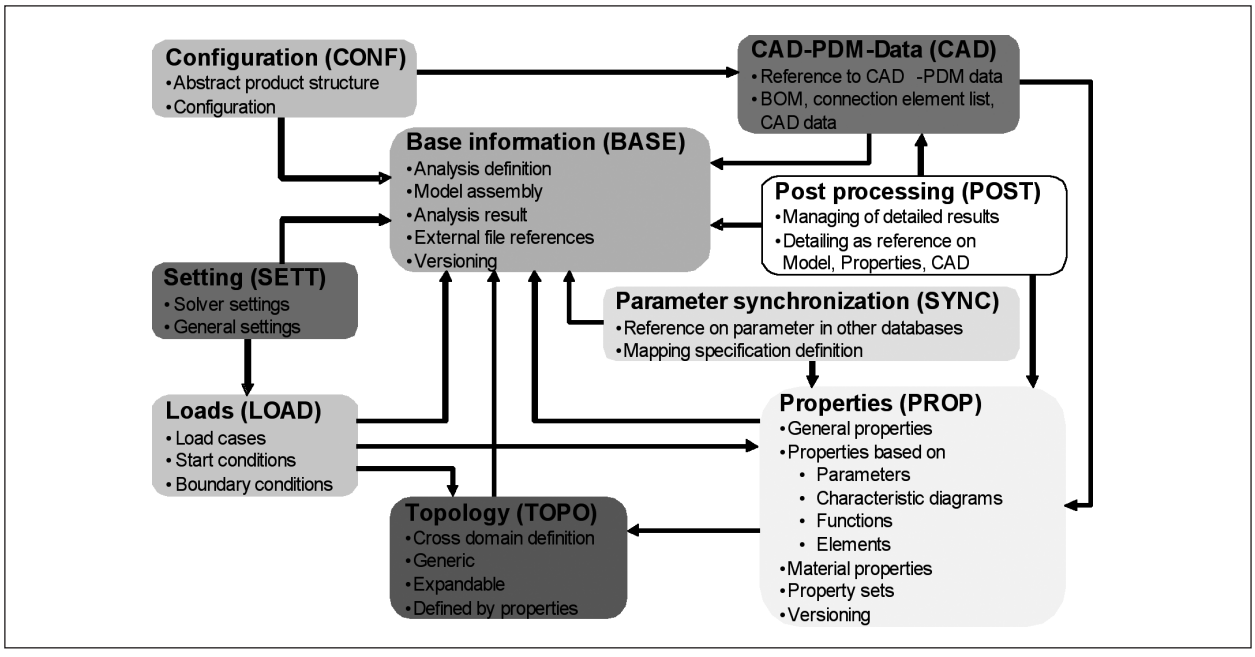


Figure 16: Structure of the packages

In principle, the satellite modules can be combined individually. In practice, the following list of combinations may be of interest.

SimPDM.BASE
Version controlled and file based management of original simulation input data such as template files and property files, intermediate simulation data such as derived geometry, meshes and assembly files, as well as generated input decks, model structures, output specifications and simulation results, and reports.

SimPDM.BASE, SimPDM.CONF
In addition to the basic functionality, this package combination allows the dynamic configuration of the model structure for an analysis version regarding the product variant. Related to this functionality is the feasibility to structure the product into components and component families.

SimPDM.BASE, SimPDM.TOPO, SimPDM.PROP
In addition to the basic functionality, this package combination allows the management of detailed model topologies and properties of topological elements. In principle, it is possible to use the SimPDM.TOPO package without SimPDM.PROP, but this is not practical.

SimPDM.BASE, SimPDM.CONF, SimPDM.TOPO, SimPDM.PROP
In addition to the management of topology structure information, this package combination allows the dynamic configuration of the model structure for an analysis version regarding the product variant. Related to this functionality is the feasibility to structure the product into components and component families.

SimPDM.BASE, SimPDM.TOPO, SimPDM.PROP, SimPDM.LOAD, SimPDM.SETT
In addition to the basic functionality, this package combination allows the management of detailed model topologies and properties of topological elements and solver settings. This combination also allows the definition of load cases. A managed definition of load cases is only practical if the load cases, including start and boundary conditions, are related to the effected model elements. This implies the usage of the SimPDM.TOPO and SimPDM.PROP packages.

SimPDM.BASE, SimPDM.CONF, SimPDM.TOPO, SimPDM.PROP, SimPDM.LOAD, SimPDM.SETT
In addition to the management of settings and load cases, this package combination allows the dynamic configuration of the model structure for an analysis version regarding the product variant. Related to this functionality is the feasibility to structure the product into components and component families.

Enhancement by SimPDM.CAD (applicable for all combinations)
The SimPDM.CAD package can be used optionally with all reasonable combinations. It allows the additional management of function model information emanating from CAD-PDM environment (BoM, Connection element list, geometry).

Enhancement by SimPDM.POST (applicable for combinations including SimPDM.BASE and preferable SimPDM.TOPO, SimPDM.PROP and/or SimPDM.CAD)
The SimPDM.POST package allows the management of detailed post processing data. Therefore the mechanisms of other SimPDM packages (SimPDM.TOPO, SimPDM.PROP, SimPDM.CAD) are used. These packages must be also implemented according to the required mechanism. Use of the SimPDM.POST package is optional.

Enhancement by SimPDM.SYNC (applicable for all combinations)
The SimPDM.SYNC package allows parameter synchronization and mapping definitions between the CAE-SimPDM and CAD-PDM environments.

6.4 Class and relation description SimPDM
For a detailed description of the SimPDM metadata model, please refer to Annex E and the SimPDM HTML documentation.

Annex A: Glossary

Analysis
An analysis is a simulation process which is associated with a certain simulation model, a certain model structure, certain solver settings, an output specification, a defined input deck and a certain hardware environment. Analysis can also be understood as the result of a simulation process with all related information for the interpretation of the result. In this case, the analysis consists of a certain simulation model, a certain model structure, certain solver settings, an output specification, a defined input deck and certain hardware environment information.

Attribute
An attribute represents a characteristic of a specific object. Each attribute has an assigned value. The value might be empty if the attribute is specified as an optional attribute.

CAD geometry
CAD geometry is a geometric product description. A CAD geometry might be the original CAD geometry produced by the mechanical product engineering or derived geometry generated by simulation engineers specifically for simulation purpose.

CAE
Computer Aided Engineering. Method used for virtual product validation to ensure certain product behavior without testing a real product or prototype. CAE is used within several different disciplines.

CAE structure
The CAE (model) structure describes the structure dependencies of a product description from the view point of CAE processes, for example the hierarchical structure of simulation models.

Class
A class is an information container that consists of a particular kind of metadata. A class may have a one or more instances at runtime. This instance of a class is called an object.
Computational fluid dynamics (CFD)
Computational Fluid Dynamics (CFD) is a method used for the simulation and problem solving of fluid flows (e.g. air) which uses the numerical methods and algorithms of fluid mechanics.

Computation
A computation is the mathematical or internal computer representation of a simulation.

Configuration
A configuration concerning a PDM structure represents a certain combination of variants which provides a basis for the computation. The computation configuration whose CAE structure is variant-free is described by a specific input deck and hardware environment combination. Optionally, a certain combination may be chosen or the configuration inside the CAE structure independent of the PDM.

Finite element analysis (FEA)
Finite element analysis (FEA) is a computer simulation technique used in engineering analysis. It uses a numerical technique called the finite element method (FEM).

Granularity
The level of detail of the CAE model structure and associated information within the context of simulation data management is called granularity.

Input deck
An input deck is a group of solver-specific files which contains a simulation model, a load case and solver settings.

Load case
A load case represents an operating state of the product by defining possible loads and constraints on the model.

Mapping
Mapping is the process of describing data element dependencies between two distinct data models. The term "mapping" is also used for the transformation of information between two distinct data models by transformation rules which refer to the dependencies between the two data models. Example of mappings are the transformation of information between CAD, PDM and CAE structures.

Metadata
Metadata are data which describes data. An item of metadata may describe a content item, or a collection of data including multiple content items. Metadata are used to facilitate the understanding, use and management of data. The metadata required for effective data management vary with the type of data and context of use.

Metadata model
A metadata model defines classes which provide a semantic information structure to describe real data.

Multi-body simulation (MBS)
Multi-body simulation (MBS) is a method used for the simulation of mechanical systems (e.g. vehicles) especially for solving kinematical problems or dynamic forces simulation.

Object
An object is an instance of a class. It is a container for real data.

Package
A package is a group of related classes within the metadata model. It is used to define modules within the data model to allow user and function-driven adaptation of the meta model.

Parameter
A parameter is a variable element in a system of mathematical relations. A parameter's value influences the behavior of the system.

Parameter synchronization
Parameter synchronization is a process by which parameters of different disciplines are updated. In most cases, a parameter from depending systems are updated by values of parameters from reference systems.

Product
A product is an object which can be sold to a customer. A service product is not included in the scope of SimPDM.

Product structure
A product structure is a description of the hierarchy and the components of a product.

Property
A property is a characteristic of a real object.

Service
A service is a function provided by a system that can be used by other systems by means of a definitive service description and a communication technology supported by service communication interfaces.

Simulation
A simulation is a method of analyzing the behavior of a system or the system itself. A simulation requires a simulation model. A simulation is actually performed by a computation.

Simulation data management (SDM) system
Simulation Data Management (SDM) is a data management system with emphasis on managing simulation data. Functionality and data storage are adapted to this specific case.

Simulation domain
A simulation domain supports a specific discipline. It may be an MBS, FEA, dimensioning, CFD, etc.

Simulation model
A simulation model is an implementation of the abstract model of a real system.

Solver
A solver is program code which solves the computation. Depending on the program code architecture, parameters are set which are required for the internal mathematical code description. For example, increasing accuracy will increase computation time.

Use case
A use case comprises a set of processes and activities which are supposed for a specific business goal.

xDM system
xDM system is a general term for data management systems independent of a specific discipline. This could be the backbone PDM (product data management) or EDM (engineering data management) system of a company, a material database or one of several concurring CAD system specific data management systems of a company, also referred to as a TDM (team data management) system. Therefore, an SDM system is, in this sense, an xDM system configured for the management of simulation data.

Annex B: Process diagrams

The detailed process diagrams are available as independent documents at the websites of the ProSTEP iViP Association (www.prostep.org) and the VDA (www.vda.de).

Annex B.1: Process diagram for crash simulation

Annex B.2: Process diagram for load case simulation

Annex B.3: Process diagram for design verification

Annex C: Core data management functionality

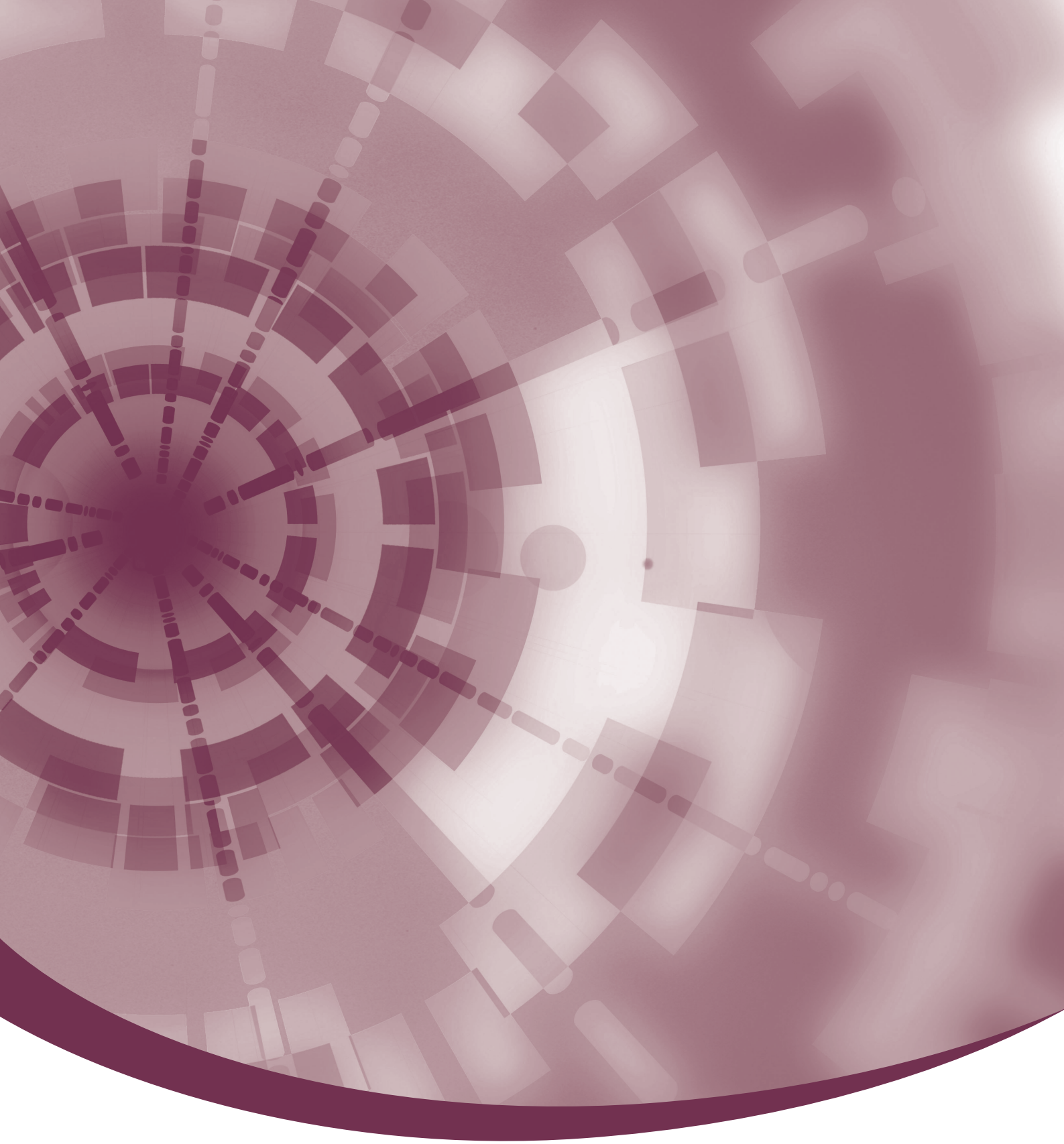
A detailed description of the SimPDM core functionality is available as an independent document at the websites of the ProSTEP iViP Association (www.prostep.org) and the VDA (www.vda.de).

Annex D: System communication functionality

A detailed description of the SimPDM system communication functionality is available as an independent document at the websites of the ProSTEP iViP Association (www.prostep.org) and the VDA (www.vda.de).

Annex E: SimPDM metadata model

A detailed description of the SimPDM metadata model is available as HTML documentation at the websites of the ProSTEP iViP Association (www.prostep.org) and the VDA (www.vda.de).



ProSTEP iViP Association
Dolivostraße 11
64293 Darmstadt
Germany

Tel. +49-6151-9287336
Fax +49-6151-9287326

psev@prostep.com
www.prostep.org

ISBN 978-3-9811864-9-9

Version 2.0, November 2008