

Simulation Data Management Annex D: System Communication functionality

Integration of Simulation and Computation in a PDM-Environment (SimPDM)

Version 2.0, November 2008

Status: Released (22)



Abstract

This document is Annex D of ProSTEP iViP Recommendation PSI 4. This annex contains the technical description of SimPDM system communication functions.

The SimPDM 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 meta data model to describe simulation and computation data. The meta data model is modularly designed to allow a custom-tailored solution for simulation data management. Furthermore this recommendation describes necessary functionality and its clustering towards use case. Functionality and use cases are modular, too. So it is possible to customize SimPDM to 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
- 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 (the document in hand)
- Annex E: Data model

The Annexes B, C, D, E are available for download as separated documents on the websites of ProSTEP iViP e.V (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

- I. All rights to this PSI Recommendation, in particular the copyright rights of use, and sale such as the right to duplicate, distribute or publish the recommendation, remain exclusively with the ProSTEP iViP Association and its members.
- II. This PSI Recommendation may be duplicated and distributed unchanged, for instance for use in the context of creating software or services.
- III. It is not permitted to change or edit this PSI Recommendation.
- IV. A suitable notice indicating the copyright owner and the restrictions on use must always appear.

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, dem engineering methods AG, IBM Deutschland GmbH, IMI Universität Karlsruhe, MAN Turbo AG, MDTVision GmbH, MSC.Software GmbH, PTC GmbH, PD Tec AG, PROSTEP AG, Schaeffler KG, Siemens PLM Software, T-Systems Enterprise Services GmbH, Volkswagen AG.

Contents

Table of Contents

1 Introduction	1
1.1 Scope and Purpose	1
1.2 Notation.....	1
1.2.1 Function Calls.....	1
1.2.2 Usage of PLM Services.....	1
1.3 Additional Remarks.....	2
2 Functional description of SimPDM use cases (synchronization and connection).....	2
2.1 Functionality of use case Synchronization Management (Parameter, File, Property)	2
2.1.1 Set and initialize external parameter.....	2
2.1.2 Connect external parameter and define dependencies	3
2.1.3 Delete external parameter.....	4
2.1.4 Evaluate parameter availability	4
2.1.5 Synchronize parameter	4
2.1.6 Update external parameter	5
2.1.7 Define and initialize external file synchronization	6
2.1.8 Disable external file synchronization.....	6
2.1.9 Evaluate file availability	7
2.1.10 Update external file	7
2.1.11 Define and initialize external property synchronization.....	8
2.1.12 Disable external property synchronization	8
2.1.13 Evaluate property availability	9
2.1.14 Update external property.....	9
2.2 Functionality of use case CAE Connection Management	10
2.2.1 SDM data export (meta data, native data) for CAE application	10
2.2.2 CAE data import (meta data, native data) from SDM data base	11
2.2.3 CAE data export (meta data, native data) for SDM data base	11
2.2.4 SDM data import (meta data, native data) from CAE application	12

1 Introduction

1.1 Scope and Purpose

The use cases defined by the recommendation are described by their general purpose within the context of simulation data management with regard to their value progress. This annex describes the atomic data management system functions which are needed to perform the use case, for instance, 'Set external parameter' as an atomic function of the use case Synchronization Management. Each function itself is described by the start state and precondition as well as the end state. Each described function requires operations on data objects or the creation of new data objects. These operations are listed for each function and directly refer to the SimPDM data model. The operations are named by a formal code. Last but not least additional notes and remarks are provided where applicable.

This annex contains the technical description of SimPDM system communication functions.

1.2 Notation

This section gives an overview of additional used notation for SimPDM system communication issues. It bases on the notation given in the Annex C defining SimPDM core data management functionality. These are also valid here.

1.2.1 Function Calls

Function calls are defined to initiate functions that are defined for specific object types. A function call is described by the following formal code

- Call.<package name>.<class name> (<function name>)

Examples are

- Call.SYNC.parameter_synchronization (check_availability)
- Call.SYNC.parameter_synchronization (synchronize)

1.2.2 Usage of PLM Services

1.2.2.1 PLM Services Use Cases

PLM Services Use Cases are used to specify access to the following external systems:

- PDM system providing parameters, properties or files to be synchronized with information managed in the SDM system during the simulation process.
- Any additional system involved in the process. As an example, this could be a material database that provides specific properties necessary for a specific simulation step.

The access to an external system using PLM Services is described by the following formal code:

- <PLM_Services> <Use Case>

Examples are

- <PLM_Services> Authorization
- <PLM_Services> Start Node Identification
- <PLM_Services> Browse down Product Structure
- <PLM_Services> Download Single Digital File (OID, version_OID, external_file_location)

In the scope of this document, detailed PLM Services queries (generic queries, specific queries etc.) defined in the PLM Services Computational Model are not considered like item_query, property_query, etc.

1.2.2.2 PLM Services Communication Concept

The specification of SimPDM functionality does not prescribe a specific communication concept. Generally, communication can be handled in different ways:

- The PLM Services communication concept (defined in the scope of the Computational Model) could be used to support SDM-CAE connection management. However, the usage of the PLM Services communication concept for the exchange of SimPDM data postulates an extension of the PLM Services Informational Model.
- If PLM Services are not used here, an independent communication concept has to be implemented in order to support SDM-CAE connection management. In the scope of this recommendation, it is defined which functions are needed in general in order to define SimPDM specific communication functionality.

The communication steps are described by the following formal code:

- <SimPDM_connection> (<function>)

Examples are

- <SimPDM_connection> (open)
- <SimPDM_connection> (read_data_set)
- <SimPDM_connection> (check_data_set)

1.3 Additional Remarks

The document does not claim the completeness of meta data model operations. Especially the change operations for attribute values are only described in a few functions where the attribute value is significant.

The order of the required operations listed in this document does not imply a mandatory order for the operations to be performed by a data management system.

Generally, the meta data model is the fundamental base definition for all described functionality. The required operations directly refer to the meta data model.

2 Functional description of SimPDM use cases (synchronization and connection)

2.1 Functionality of use case Synchronization Management (Parameter, File, Property)

See SimPDM recommendation for a general description of the use case Parameter Synchronization Management.

2.1.1 Set and initialize external parameter

2.1.1.1 Start state and preconditions

At least one instance of the following class does exist:

- BASE.parameter

2.1.1.2 Required operations

- create.SYNC.external_parameter
- <PLM_Services> Authentication/Start-Up of Session
- <PLM_Services> Authorization
- <PLM_Services> Start Node Identification
- <PLM_Services> Browse down Product Structure
- <PLM_Services> Download Meta Data (OID, value, unit)
- Set.SYNC.external_parameter.external_id
- Set.SYNC.external_parameter.value

- Set.SYNC.external_parameter.unit

2.1.1.3 End state and post condition

A new instance of the class SYNC.external_parameter exists and is initialized.

2.1.1.4 Notes and remarks

This use case describes the assignment of parameters from a PDM product structure to CAE parameters available in the SDM system. Based on the information from PDM, a SimPDM parameter can be synchronized.

This use case assumes that the external parameter is read from an external system (e.g. PDM system). This could as well be any kind of system involved in the simulation process. Details concerning data access (export logic, system API etc.) of the external system are not considered.

In the scope of this recommendation, all security and authentication aspects concerning an external PDM system are not considered.

Concerning PLM Services, only high level use cases are named.

2.1.2 Connect external parameter and define dependencies

2.1.2.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.parameter
- SYNC.external_parameter

2.1.2.2 Required operations

- Create.SYNC.parameter_synchronization
- Connect.SYNC.parameter_synchronization_parameter
- Connect.SYNC.parameter_synchronization_external_parameter
- Set.SYNC.parameter_synchronization.function_definition

2.1.2.3 End state and post condition

A new instance of the class SYNC.parameter_synchronization exists and is connected to

- BASE.parameter
- SYNC.external_parameter

Dependencies are defined for the parameter to be synchronized.

2.1.2.4 Notes and remarks

This function describes the assignment and definition of parameter dependencies from PDM structure and SDM structure as input information for the simulation process (defining functional or mathematical dependencies between parameters).

Since a parameter in the SDM system may depend on several external parameters from another data management system, the relationship connect.SYNC.parameter_synchronization_external_parameter may have to be initiated more than once. SYNC.parameter_synchronization.function_definition describes the mathematical dependency between the parameter in SDM and the external parameters.

This function does not cover the update of a parameter value.

2.1.3 Delete external parameter

2.1.3.1 Start state and preconditions

An instance of the class SYNC.external_parameter exists and is assigned to an instance of the class BASE.parameter by an instance of the class SYNC.parameter_synchronization.

2.1.3.2 Required operations

- Disconnect.SYNC.parameter_synchronization_external_parameter
- Delete.SYNC.external_parameter
- Disconnect.SYNC.parameter_synchronization_parameter
- Delete.SYNC.parameter_synchronization

2.1.3.3 End state and post condition

Both SYNC.external_parameter and SYNC.parameter_synchronization are no longer available and no longer assigned to BASE.parameter.

2.1.3.4 Notes and remarks

The deletion of the SYNC.external_parameter and SYNC.parameter_synchronization is described with regards to the data model specification. Nevertheless, the deletion of objects might specifically be restricted by data management system restrictions.

2.1.4 Evaluate parameter availability

2.1.4.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.parameter
- SYNC.external_parameter
- SYNC.parameter_synchronization

2.1.4.2 Required operations

- Call.SYNC.parameter_synchronization (check_availability)

2.1.4.3 End state and post condition

It is ensured that all information necessary for the synchronization is available.

2.1.4.4 Notes and remarks

This function covers the following checks:

- Assignment completeness (are all parameters assigned and are all elements considered?)
- Information availability (are all assigned parameters available, e.g. weight, Mol, CoG, ...)
- Version check (are the correct versions available?)

2.1.5 Synchronize parameter

2.1.5.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.parameter
- SYNC.external_parameter
- SYNC.parameter_synchronization

2.1.5.2 Required operations

- Call.SYNC.parameter_synchronization (synchronize)

2.1.5.3 End state and post condition

The parameter is synchronized based on the input information and the defined dependency:

- SYNC.external_parameter.value
- SYNC.parameter_synchronization.function_definition

Result is an updated value of the internal parameter:

2.1.5.4 Notes and remarks

Synchronized and aggregated values are written to BASE.parameter. Possibly, this leads to the creation of a new version of the object superior to the parameter. As this is handled in different ways in the companies, there is no detailed process defined in the scope of this recommendation.

This function is called automatically as soon as the attribute BASE.parameter_synchronization.function_definition is changed.

Additionally, the function is called automatically when an update for external parameter values is available.

2.1.6 Update external parameter

2.1.6.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.parameter
- SYNC.external_parameter
- SYNC.parameter_synchronization

At least one value of an assigned object BASE.external_parameter has been changed.

2.1.6.2 Required operations

- <PLM_Services> Change Notification
- <PLM_Services> Authentication/Start-Up of Session
- <PLM_Services> Authorization
- <PLM_Services> Start Node Identification
- <PLM_Services> Browse down Product Structure
- <PLM_Services> Download Meta Data (OID, updated_value)
- Set.SYNC.external_parameter.external_id
- Set.SYNC.external_parameter.value
- Call.SYNC.parameter_synchronization (check_availability)
- Call.SYNC.parameter_synchronization (synchronize)

2.1.6.3 End state and post condition

The BASE.external_parameter link is updated based on the input information.

2.1.6.4 Notes and remarks

In the case of a change in the external system, the function BASE.parameter_synchronization (synchronize) is automatically 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 simulation data management system. Synchronization and availability checks are initiated by the SDM system.

Concerning PLM Services (access of PDM system), details concerning the change notification action are not considered.

The following actions are considered:

- Synchronisation of parameter based on updated attribute values
- Error handling in the case of missing information
- Comparison of parameter values
- Writing of updated parameter values
- Creation of new parameter or parameter version

2.1.7 Define and initialize external file synchronization

2.1.7.1 Start state and preconditions

At least one instance of the following classes does exist and the respective attributes are defined:

- BASE.reference_to_file
- BASE.reference_to_file.file_id
- BASE.reference_to_file.file_version_id
- BASE.reference_to_file.location

2.1.7.2 Required operations

- Create.SYNC.file_synchronization
- Connect.SYNC.file_synchronization_reference_to_file
- Set.SYNC.file_synchronization.external_id
- Set.SYNC.file_synchronization.external_version_id

2.1.7.3 End state and post condition

A new instance of the class SYNC.file_synchronization exists and is connected to

- BASE.reference_to_file

2.1.7.4 Notes and remarks

The following attributes represent the SimPDM internal identification of the existing file (available in SDM system):

- BASE.reference_to_file.file_id
- BASE.reference_to_file.file_version_id
- BASE.reference_to_file.location

The following attributes identify the external file to be synchronized:

- SYNC.file_synchronization.external_id
- SYNC.file_synchronization.external_version_id

2.1.8 Disable external file synchronization

2.1.8.1 Start state and preconditions

An instance of the class SYNC.file_synchronization exists and is assigned to an instance of the class BASE.reference_to_file.

2.1.8.2 Required operations

- Disconnect.SYNC.file_synchronization_reference_to_file

- Delete.SYNC.file_synchronization

2.1.8.3 End state and post condition

SYNC.file_synchronization is no longer available and no longer assigned to BASE.reference_to_file.

2.1.8.4 Notes and remarks

The deletion of the SYNC.file_synchronization is described with regards to the data model specification. Nevertheless, the deletion of objects might specifically be restricted by data management system restrictions.

2.1.9 Evaluate file availability

2.1.9.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.reference_to_file
- SYNC.file_synchronization

2.1.9.2 Required operations

- Call.SYNC.file_synchronization (check_availability)

2.1.9.3 End state and post condition

It is ensured that all information necessary for the file synchronization is available.

2.1.9.4 Notes and remarks

This function covers the following checks:

- File availability (is the assigned external file available?)
- Version check (is the correct file version available?)

2.1.10 Update external file

2.1.10.1 Start state and preconditions

At least one instance of the following classes does exist:

- BASE.reference_to_file
- SYNC.file_synchronization

2.1.10.2 Required Operations

- <PLM_Services> Change Notification
- <PLM_Services> Authentication/Start-Up of Session
- <PLM_Services> Authorization
- <PLM_Services> Start Node Identification
- <PLM_Services> Browse down Product Structure
- Call.SYNC.file_synchronization (update)

2.1.10.3 End state and post condition

The file_synchronization link is updated based on the updated information. By default, this does not include the file transfer itself. Optionally, file transfer could be performed.

Update external file includes an update of the following attributes:

- SYNC.file_synchronization.external_id

- SYNC.file_synchronization.external_version_id

2.1.10.4 Notes and remarks

In the scope of this recommendation, all security and authentication aspects concerning an external PDM system are not considered.

Concerning PLM Services, only the high level use cases are named. Detailed queries are not considered.

In the case of a change in the SDM system, file_synchronization (update) is performed based on the initial assignment (e.g. when a new version of a referenced external file is created). It shall be possible to access the mapping automatically from the simulation data management system.

Concerning PLM Services (access of PDM system), details concerning the change notification action are not considered.

The following actions are considered:

- Update of respective attributes
- Error handling in the case of missing information

2.1.11 Define and initialize external property synchronization

2.1.11.1 Start state and preconditions

At least one instance of the following classes does exist and the respective attributes are defined:

- PROP.property_set_version
- PROP.property_set_version.id
- PROP.property_set_version.version_id

2.1.11.2 Required operations

- Create.SYNC.property_synchronization
- Connect.SYNC.property_synchronization_property_set
- Set.SYNC.property_synchronization.external_property_definition_id.

2.1.11.3 End state and post condition

A new instance of the class SYNC.property_synchronization exists and is connected to

- PROP.property_set_version

2.1.11.4 Notes and remarks

The following attributes represent the SimPDM internal identification of the existing property (available in SDM system):

- PROP.property_set_version.id
- PROP.property_set_version.version_id

The following attribute identifies the external property to be synchronized:

- SYNC.property_synchronization.external_property_definition_id

2.1.12 Disable external property synchronization

2.1.12.1 Start state and preconditions

An instance of the class SYNC.property_synchronization exists and is assigned to an instance of the class PROP.property_set_version.

2.1.12.2 Required operations

- Delete.SYNC.property_synchronization
- Disconnect.SYNC.property_synchronization_property_set

2.1.12.3 End state and post condition

SYNC.property_synchronization is no longer available and no longer assigned to PROP.property_set_version.

2.1.12.4 Notes and remarks

The deletion of the SYNC.file_synchronization is described with regards to the data model specification. Nevertheless, the deletion of objects might specifically be restricted by data management system restrictions.

2.1.13 Evaluate property availability

2.1.13.1 Start state and preconditions

At least one instance of the following classes does exist:

- PROP.property_set_version
- SYNC.property_synchronization

2.1.13.2 Required operations

- Call.SYNC.property_synchronization (check_availability)

2.1.13.3 End state and post condition

It is ensured that all information necessary for the synchronization is available.

2.1.13.4 Notes and remarks

This use case covers the following checks:

- Property availability (is the assigned external property available?)
- Version check (is the correct property version available?)

2.1.14 Update external property

2.1.14.1 Start state and preconditions

At least one instance of the following classes does exist:

- PROP.property_set_version
- SYNC.property_synchronization

2.1.14.2 Required Operations

- <PLM_Services> Change Notification
- <PLM_Services> Authentication/Start-Up of Session
- <PLM_Services> Authorization
- <PLM_Services> Start Node Identification
- <PLM_Services> Browse down Product Structure
- <PLM_Services> Download Meta Data (OID)
- Call.SYNC.property_synchronization (update)

2.1.14.3 End state and post condition

The property_synchronization link is updated based on the updated information. The according data itself is not transferred. Update external property includes an update of the following attribute:

- SYNC.property_synchronization.external_property_definition_id

2.1.14.4 Notes and remarks

This use case assumes that the external property is read from an external system (e.g. material database). This could as well be any kind of system involved in the simulation process. Details concerning data access (export logic, system API etc.) of the external system are not considered.

PLM Services are used to define access to an external system (e.g. material database).

In the scope of this recommendation, all security and authentication aspects concerning access on an external system (example: material database) are not considered.

In the case of a change in the SDM system, parameter_synchronization (update) is performed based on the initial assignment (e.g. when a new version of a property is created). It shall be possible to access the mapping automatically from the simulation data management system.

Concerning PLM Services (access of PDM system), details concerning the change notification action are not considered.

The following actions are considered:

- Update of respective attributes
- Error handling in the case of missing information

2.2 Functionality of use case CAE Connection Management

See SimPDM recommendation for a general description of the use case SDM-CAE Connection Management.

2.2.1 SDM data export (meta data, native data) for CAE application

2.2.1.1 Start state and preconditions

Meta data available in the SDM system has to be exported to a CAE application at a specified point of time in the simulation process.

2.2.1.2 Required operations

- <SimPDM_connection> (open)
- <SimPDM_connection> (query)
- <SimPDM_connection> (write_data_set)
- <SimPDM_connection> (check_completeness)
- <SimPDM_connection> (export_data_set)
- <SimPDM_connection> (close)

2.2.1.3 End state and post condition

A SimPDM data set is written and exported for a specific CAE application.

2.2.1.4 Notes and remarks

On database export, no consistency check is performed. In order to export a consistent data set, the completeness of the data to be exported has to be checked by a specific service.

2.2.2 CAE data import (meta data, native data) from SDM data base

It is possible that meta data related to native data has been changed within the SDM system. This can lead to inconsistency between meta data and native data sets available in the SDM system.

Since a consistency check is not performed by the SDM data export function this consistency check has to be performed during the CAE data import.

2.2.2.1 Start state and preconditions

A data set (meta data and native data) has been exported by the SDM system and has to be imported by a specific CAE application. Meta data emanating from the SDM system may have been changed while the respective native data set remains unchanged.

2.2.2.2 Required operations

The CAE application checks on import whether meta data are still consistent with the respective native data. In case of differences, changes concerning meta data are binding while exporting information for a task in a specific CAE system as the SDM system represents the master system. Thus, the native file content must be updated during import into the specific CAE application.

In the scope of the SimPDM recommendation, this check is called consistency check 1.

SimPDM data remains unchanged while native data is changed by the CAE application.

2.2.2.3 End state and post condition

Both meta data and native data are imported to CAE and both data sets are consistent.

2.2.2.4 Notes and remarks

This function assumes that a connection between SDM and CAE system is available and established. This may be either an online or offline connection.

Details concerning data access (import and export logic, system API etc.) of the involved systems are not considered. Additionally, security and authentication aspects are not considered.

As this function does not affect changes on data available in the SDM system, no detailed function specification is given. Functions have to be implemented for the specific CAE application.

2.2.3 CAE data export (meta data, native data) for SDM data base

The aim is to ensure data consistency when work with the CAE application ends.

2.2.3.1 Start state and preconditions

A data set (meta data and native data) is exported by the CAE system in order to be imported by the SDM system. Native data emanating from the CAE application may have been changed while the respective meta data set remains unchanged.

2.2.3.2 Required operations

SimPDM meta data has to be updated at the corresponding meta data nodes, e.g. transformation matrix changes or creation of a new node. Therefore, the CAE application exports a consistent SimPDM data set (both meta data and native data) with a corresponding level of detail.

Native data remains unchanged while meta data is changed respectively updated by the CAE application.

In the scope of the SimPDM recommendation, this check is called consistency check 2.

2.2.3.3 End state and post condition

Both meta data and native data are exported from CAE and both data sets are consistent.

2.2.3.4 Notes and remarks

This function assumes that a connection between SDM and CAE system is available and established. This may be an online or offline connection.

Details concerning data access (import and export logic, system API etc.) of the involved systems are not considered. Additionally, security and authentication aspects are not considered.

In case of differences, native data are the master data set. Native data are the data set on which the CAE application works. Therefore, no special operations are required on the SDM system.

As this function does not affect changes on data available in the SDM system, no detailed function specification is given. Functions have to be implemented for the specific CAE application.

2.2.4 SDM data import (meta data, native data) from CAE application

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.

2.2.4.1 Start state and preconditions

The CAE application has created new SimPDM and native data to be imported into the SDM system.

2.2.4.2 Required operations

- <SimPDM_connection> (open)
- <SimPDM_connection> (read_data_set)
- <SimPDM_connection> (check_data_set)
- <SimPDM_connection> (import_data_set)
- <SimPDM_connection> (close)

Furthermore the incoming meta data has to be assigned to the right existing objects respectively to new objects. As this is a very company and process specific task – especially on creating new versions – there is no detailed definition of required operations. Specific business logic gives directives whether to create new versions.

In the scope of the SimPDM recommendation, this check is called consistency check 3.

2.2.4.3 End state and post condition

Both meta data and native data are imported into the SDM system and both data sets are consistent.

2.2.4.4 Notes and remarks

On database import, a consistency check shall assign the imported data to existing or new nodes within the database. In order to import a consistent data set, the completeness of the data to be exported has to be checked by a specific service.

It is essential that structures are written to the database correctly.

Changes (either meta information or native files) can lead to new versions in the simulation data management system.

Inconsistencies are checked and reported during import.