PDM Workbench

PDM Workbench
Release 18.0.5
for Aras Innovator

Release Notes

Copyright

Contact

T-Systems International GmbH
Business Unit PLM
Fasanenweg 5
70771 Leinfelden-Echterdingen
Germany

https://plm.t-systems.net/en-DE/pdm-workbench

☎ +49 40 30600 5544
✉ +49 3915 80125688

E-Mail: *cmi_support@t-systems.com*

Trademarks

CATIA is a registered trademark of Dassault Systèmes.

Aras is a registered trademark of Aras Corporation.

Names of other products mentioned in this manual are used for identification purpose only and might be trademarks of their companies.

Preface

About this Document

This document provides general release information for the PDM Workbench.

Related Documents

The following manuals contain information about installation, administration, usage, and customization of the PDM Workbench:

Manual Title	Version
PDM Workbench Installation & Administration Manual	*18.0*
PDM Workbench User Manual	*18.0*

Your Comments are Welcome

Please feel free to tell us your opinion; we are always interested in improving our publications. Mail your comments to:

T-Systems International GmbH
Business Unit PLM
Fasanenweg 5
70771 Leinfelden-Echterdingen
Germany

Email: *cmi_support@t-systems.com*

Table of Contents

Table of Figures

CHAPTER 1

Platform Support

Supported Platforms

CATIA V5 Version V5-6R2021, V5-6R2022, V5-6R2023, and V5-6R2024

(other revisions on request):

CATIA V5 Client V5-6R2021	Windows 10 (64Bit), Windows 11 (64Bit)
CATIA V5 Client V5-6R2022	Windows 10 (64Bit), Windows 11 (64Bit)
CATIA V5 Client V5-6R2023	Windows 10 (64Bit), Windows 11 (64Bit)
CATIA V5 Client V5-6R2024	Windows 10 (64Bit), Windows 11 (64Bit)

Important notice:

CATIA V5-6R2014 SP2 has been retracted by Dassault Systèmes and is not supported. Please use SP3 instead.

CATIA V5-6R2022. Minimum service pack: SP1

Server Installation of Aras Innovator:

(Other service packs on request)

Aras Innovator Server 12 SP09	Windows Server 2019, Windows Server 2016, Windows Server 2012
Aras Innovator Server 22 to 31	Windows Server 2022, Windows Server 2019, Windows Server 2016

T-Systems licman21 license manager:

| T-Systems licman21	Please refer to the Licman documentation.

CHAPTER 2

Functional Changes

Dynamic Dialog Functionality

It is possible now to have user dialogs which are not defined in a static way in the PWBSchema.xml file, but defined dynamically in a custom Aras Innovator server method. This makes it possible to dynamically define at runtime both the dialog attribute definitions and their values.

Configuration

A custom C# server method which is called by the connector and which returns the dialog information depending on the content of the call has to be defined.

A sample method for this, "Pwb_Sample_DynDlgUpdServerMethod", exists.
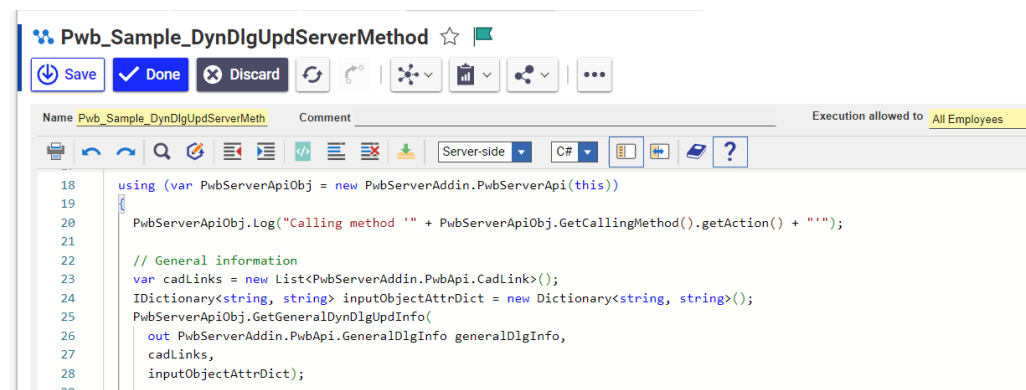
Picture 1: Sample dynamic dialog custom server method

This method has to be defined in the 'PWBSchemata / PWBSchema / dynDlgServerMethod' XML attribute in the PWBSchema.xml file, as shown in this example:

```
<PWBSchemata>

    <PWBSchema system="Aras" customization="Aras" serverConfig=""
    displayName="NLS_System" visibleLength="15" allowedLength="64"
    UseBomPartStructure="false"
    dynDlgServerMethod="Pwb_Sample_DynDlgUpdServerMethod" >

    <!-- <PWBSchema system="Aras" customization="Aras"
    serverConfig="" displayName="NLS_System" visibleLength="15"
    allowedLength="64" UseBomPartStructure="false" > -->
```

The XML code in green shows the regular definition without the dynamic dialog functionality commented out.

The dialog definitions in the PWBSchema.xml file are the same as before, except they contain no attribute definitions, since the dialog content is requested from the Aras Innovator server dynamically:

```
<object name="/CAD/Mechanical/Part" displayName="NLS_CATPart"
icon="CATPart" isDefaultFor="CATPart">


  ...


  <!-- Define custom context action(s) with dialog-->

  <!-- If using dynamic dialogs (dynDlgServerMethod defined) use
  empty dialog definition for custom action -->

  <customContextAction name="Pwb_Sample_ContextAction"
  usedIn="PdmWindow|QueryDialog" confirm="false"
  dialog="Pwb_Sample_ContextActionDynDlg" />

  <customContextAction name="Pwb_Sample_DynContextPromote"
  usedIn="PdmWindow|QueryDialog" confirm="false"
  dialog="Pwb_Sample_ContextPromoteDlg" />


  <!-- If not using dynamic dialogs use dialog definition using
  attributes -->

  <!--

  <customContextAction name="Pwb_Sample_ContextAction"
  usedIn="PdmWindow|QueryDialog" confirm="false"
  dialog="Pwb_Sample_ContextActionFixedDlg" />

   -->

  ...

  <!-- Dialog definitions without any attributes: -->

  <form name="Pwb_Sample_ContextActionDynDlg" />

  <form name="Pwb_Sample_ContextPromoteDlg" />


  <form name="Pwb_Sample_ContextActionFixedDlg">

    <formAttribute name="item_number"
    displayName="NLS_item_number" widgetType="SingleLineEditor"
    mode="output" ... />

    ...

    <formAttribute name="classification"
    widgetType="SingleLineEditor" mode="output" visibleLength="15"
    required="false" />

  </form>
```
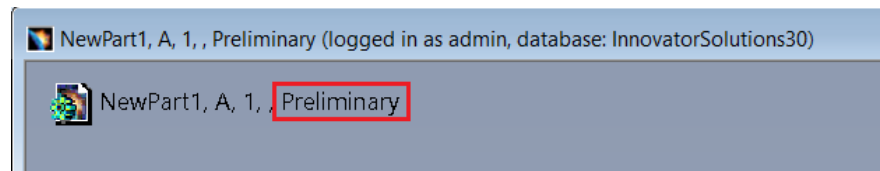
Usage

An example is shown with the custom context action method
'Pwb_Sample_ContextPromoteDlg'. The starting point is a CAD document in the
'Preliminary' lifecycle state:

Picture 2: CAD document in 'Preliminary' state

The user clicks on the custom action "Custom Promote":

Picture 3: "Custom Promote" custom action

A dialog appears with the currently available target lifecycle states based on the current 'Preliminary' lifecycle state of the CAD document:
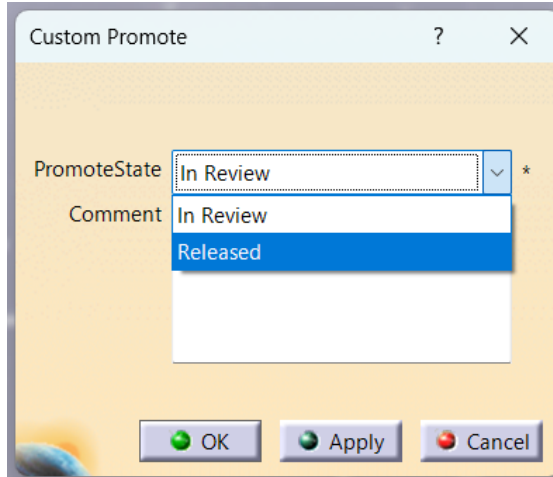
Picture 4: Lifecycle states in dialog based on 'Preliminary'

These are the same states that are available in the Aras Innovator web client:

Picture 5: Lifecycle states in browser based on 'Preliminary'

If the user clicks on "Custom Promote" the CAD document is promoted to the selected lifecycle state:
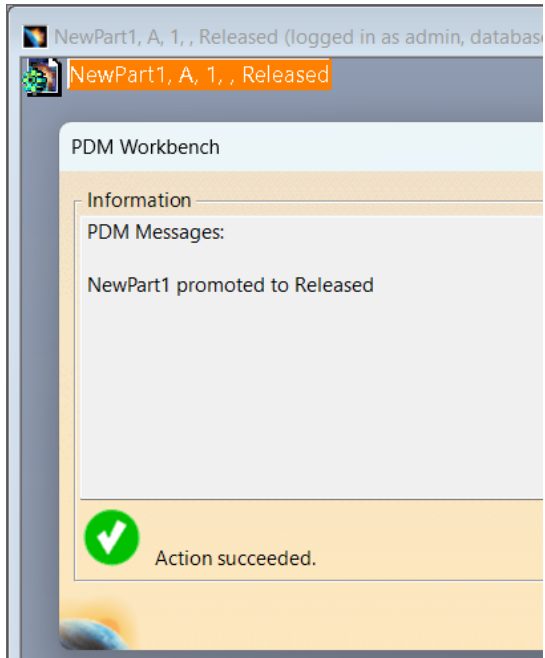
Picture 6: Result of "Custom Promote" action

Selecting "Custom Promote" again now shows different target lifecycle states based on the different current lifecycle state 'Released':
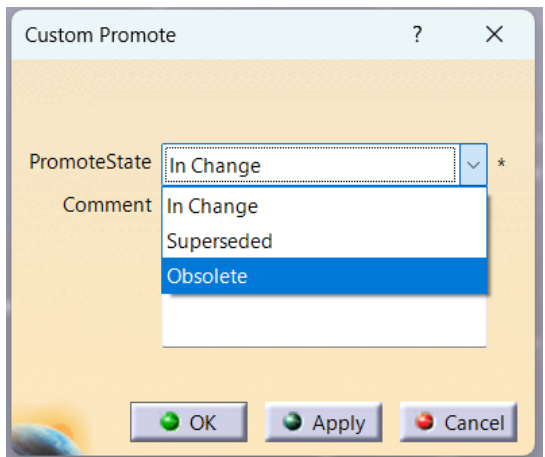
Picture 7: Lifecycle states in dialog based on 'Released'

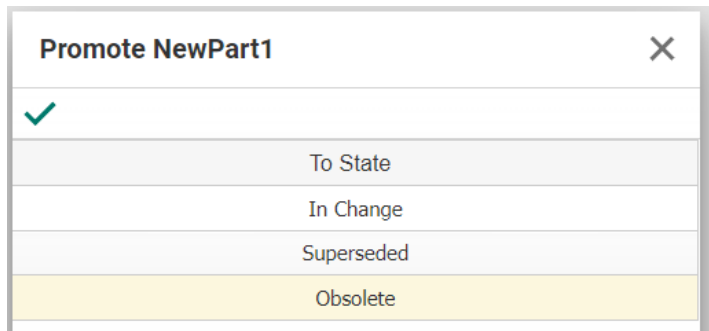Again, this is the same as in the Aras Innovator web client.

Picture 8: Lifecycle states in browser based on 'Released'

New Version 2.0 of the Connector Server API

public class **PwbServerApi**
  Member of **PwbServerAddin**
Summary:
Main class for PWB server API functionality.

New methods:

public **void SyncToBomApi**(
string *Type*, **string** *Id*, out **System.Collections.Generic.List<string[]>** *Messages*)
Summary:
Synchronizes a CAD Structure to BOM, that is, updates a structure of 'Part BOM' and 'BOM Instance' relation
from a 'CAD Structure' and 'CAD Instance' structure as the input.
Parameters:
Type: Type of selected object. Currently only 'CAD' is supported.
Id: Aras id of selected object.
Messages: User messages: List of [MessageText][MessageType].

public **string GetDrawingType**()
Summary:
Returns the connector drawing type string. By default it is "/CAD/Mechanical/Drawing", but it can be
configured by defining the server setting "CATDrawingType".
Returns:
The connector drawing type string.

public **void GetGeneralDynDlgUpdInfo**(
out **PwbServerAddin.PwbApi.GeneralDlgInfo** *GenlDlgInfo*,
[**System.Collections.Generic.List<PwbServerAddin.PwbApi.CadLink>** *CadLinks* = null],
[**System.Collections.Generic.IDictionary<string, string>** *InputObjectAttrDict* = null])
Summary:
Class for general data which is not associated to one particular dialog (primary or secondary dialog).
Parameters:
GenlDlgInfo: General CAD attribute information.
CadLinks: CAD link types and target document ids from the CAD session.
InputObjectAttrDict: Attributes of a PDM object which may be passed to the method, like the item being
duplicated. Does not always contain data, depending on the use case.

public **PwbServerAddin.PwbApi.Dlg GetDynDlgFromClient**(
PwbServerAddin.PwbApi.DlgOrdinal *DialogOrdinal*)
Summary:
Retrieves the primary or secondary dialog object which has been sent from the client. 'null' if it doesn't exist.
The secondary dialog only exists in the 'PdmUpdate' context, if a Part item is being created in addition to
the CAD item.
Parameters:
DialogOrdinal: 'Primary' or 'Secondary'.
Returns:
The dialog if it exists, or 'null' otherwise.

public **PwbServerAddin.PwbApi.Dlg CreateNewDynDlg**(
PwbServerAddin.PwbApi.DlgOrdinal *DialogOrdinal*, **string** *ClassStr*, **string** *FormStr*)
Summary:
Creates a new dialog object to be returned to the client.
Parameters:
DialogOrdinal: The primary dialog, or the secondary (the Create dialog of the Part item in 'Update PDM').
ClassStr: Corresponds to <object name="..." in the PWBSchema.xml file.
FormStr: Corresponds to <form name="..." in the PWBSchema.xml file.

public **void SetDynDlgOutputInfo**(

PwbServerAddin.PwbApi.DlgOrdinal *DialogOrdinal*, **PwbServerAddin.PwbApi.Dlg** *DialogToReturn*)
Summary:
Defines the dialog to be returned as the primary or secondary dialog to the client.
Parameters:
DialogOrdinal: Primary or secondary.
DialogToReturn: The dialog object.

public **bool** **IsCadDoc**(**string** *PdmClass*)
Summary:
Returns whether the class string refers to a CAD document.
Parameters:
PdmClass: The class string, as defined in the PWBSchema.xml file, for example "/CAD/Mechanical/Assembly".
Returns:
'true' if it is a CAD, otherwise 'false'.

public **bool** **IsPart**(**string** *PdmClass*)
Summary:
Returns whether the class string refers to a Part item.
Parameters:
PdmClass: The class string, as defined in the PWBSchema.xml file, for example "/Part/Assembly".
Returns:
'true' if it is a Part item, otherwise 'false'.

public **void SetAutonameValue**(**string** *autonameValue*)
Summary:
Sets the string which is set as the automatically generated item number of a new CAD or Part item.
Parameters:
autonameValue: The item number of the new item to be created.

PwbServerAddin.PwbApi.GeneralDlgInfo

Summary:

Class for general data which is not associated to one particular dialog (primary or secondary dialog).

public **Aras.IOM.Item GetDialogInputItem**(
PwbServerAddin.PwbApi.InputItem *inputItemType*)
Summary:
Retrieves a specific item which has been passed as input. The input item type 'CorrespondingPdmObj' exists for the dialog context 'ContextAction', and 'DuplicateSourceFile' and possibly 'DuplicateSourcePart' exist for the dialog context 'Duplicate'.
Parameters:
inputItemType: Can be 'CorrespondingPdmObj', 'DuplicateSourceFile', or 'DuplicateSourcePart'.
Returns:
The item which corresponds to the input item type if it exists, 'null' otherwise.

public **System.Collections.Generic.List<Aras.IOM.Item> GetDialogInputItemList**(
PwbServerAddin.PwbApi.InputItemList *inputItemListType*)
Summary:
Returns a list of items. Currently the only valid input value is 'BomPartsInSession' for the dialog context 'Create'.
Parameters:
inputItemListType: Only 'BomPartsInSession' is valid.
Returns:
The list of Part items in the CAD session, if any exist.

public **System.Collections.Generic.IList<string> GetUdpNames**()
Summary:
A list of the user-defined property names.
Returns:
The names in a string list.

---

public **string** **GetUdpValue**(**string** *name*)
Summary:
The value of a specific user-defined property.
Parameters:
name: The UDP name.
Returns:
The UDP value corresponding to the name.

public **string** **CadComponentName** { get; }
Summary:
The CAD component name (Context=Create).

public **string** **CadDefinition** { get; }
Summary:
The CAD definition (Context=Create).

public **string** **CadDescriptionReference** { get; }
Summary:
The CAD reference description (Context=Create).

public **string** **CadFileName** { get; }
Summary:
The CAD file name (Context=Create).

public **string** **CadFileType** { get; }
Summary:
The CAD file type, for example "CATDrawing".

public **string** **CadNomenclature** { get; }
Summary:
The CAD nomenclature (Context=Create).

public **string** **CadPartNumber** { get; }
Summary:
The CAD part number, usually mapped to the Innovator item number (Context=Create).

public **string** **CadPsnSpecType** { get; }
Summary:
The internal CAD specification type (Context=Create).

public **string** **CadRevision** { get; }
Summary:
The CAD revision (Context=Create).

public **PwbServerAddin.PwbApi.DlgContext** **Context** { get; }
Summary:
The context in which the server method is called. Possible values are 'Create', 'PdmUpdate', 'Duplicate', and 'ContextAction'.

public **string** **ContextProduct** { get; }
Summary:
Context product, needed for specific use cases.

public **string** **ContextProductId** { get; }
Summary:
Context product ID, needed for specific use cases.

public **string** **OriginatedFromCad** { get; }
Summary:
For 'Context=Duplicate' the ID of the original CAD item.

public **string** **OriginatedFromPart** { get; }
Summary:
For 'Context=Duplicate' the ID of the original Part item, if it exists.

public class **CadLink**
    Member of **PwbServerAddin.PwbApi**
Summary:
Information about internal CAD links.

public **string** **CadId** { get; set; }
Summary:
The ID of the CAD document where the link points to.

public **string** **CadType** { get; set; }
Summary:
The type, including the classification, of the CAD document the link points to.
Example: "/CAD/Mechanical/Part".

public **string** **Type** { get; set; }
Summary:
The type string, corresponds to the classification of the 'CAD Structure' relation.

public class **Dlg**
    Member of **PwbServerAddin.PwbApi**
Summary:
Class representing dialog information which is dynamically returned to the user.

public **string** **Key** { get; }
Summary:
The dialog key. Only "Primary" or "Secondary" are currently valid.

public **string** **Class** { get; }
Summary:
The dialog class. Corresponds to the object name in the PWBSchema.xml file. Example values are
"/CAD/Mechanical/Part" or "/Part/Component".

public **string** **Form** { get; }
Summary:
The dialog form. Corresponds to the form name in the PWBSchema.xml file. Example values are "Create" or
"Register".

public **string** **ChangedAttribute** { get; }
Summary:
The name of the changed attribute that triggered the call of the server method.

public **string** **NewAttributeValue** { get; }
Summary:
The new value of the changed attribute that triggered the call of the server method.

public **PwbServerAddin.PwbApi.CorrespondingPartAction** **Action** { get; }
Summary:
The 'CorrespondingPart' action.

public **string** ActionData { get; }
Summary:
Optional additional 'CorrespondingPart' action data, like for example a part type, or an item ID.

public **System.Collections.Generic.IDictionary<string, string>** InputValueDict { get; }
Summary:
Dictionary containing the dialog attribute input values.

public **System.Collections.Generic.IDictionary<string, string>** InputTypeDict { get; }
Summary:
Dictionary containing the dialog attribute input widget types, for example "String", "StringList" or "Boolean".

public **void BeginDialog**()
Summary:
Has to be called before adding information to the dialog.

public **void AddDlgAttrWidget**(
System.Collections.Generic.IDictionary<string, string> *xmlAttrs*,
[**System.Collections.Generic.List<PwbServerAddin.PwbApi.Dlg.ListValueEntry>** *values* = null])
Summary:
Adding information about one attribute widget to the dialog.
Parameters:
xmlAttrs: Dictionary containing the dialog XML attributes. Example (like in the PWBSchema.xml file):
name="item_number" widgetType="ComboBox" mode="update" visibleLength="15" required="false"
listViewRelevant="true" entryAllowed="true"
values: List containing the values of a list widget.

public **void AddActionDlgAttrWidget**(
System.Collections.Generic.IDictionary<string, string> *xmlAttrs*,
System.Collections.Generic.List<PwbServerAddin.PwbApi.Dlg.ActionListValueEntry> *actionValues*)
Summary:
Adds a widget performing a specific action to the dialog. Currently only "name"="PwbCorrespondingPart" is
allowed.
Parameters:
xmlAttrs: Dictionary containing the dialog XML attributes
actionValues: Currently only the actions NoPart, CreateCorrespondingPart, QueryForCorrespondingPart,
RelateCorrespondingPart, FirstPartInSession, and DisplayMessage are allowed. Starting with PWB 18.0.1 also
support the action: CustomNoPart.

public **void AddAttributeValues**(
System.Collections.Generic.IDictionary<string, string> *StringAttrValues*,
System.Collections.Generic.IDictionary<string, System.Collections.Generic.List<string>>
StringListAttrValues)
Summary:
Adding values to the dialog to be returned to the user.
Parameters:
StringAttrValues: Values of "String" type attributes.
StringListAttrValues: Values of "StringList" type attributes.

public **void AddControlSettings**(
System.Collections.Generic.Dictionary<PwbServerAddin.PwbApi.ControlSetting, string>
ControlSettings)
Summary:
Adding control setting which trigger specific actions on the client. Currently only NoPart,
CreateCorrespondingPart, QueryForCorrespondingPart, RelateCorrespondingPart, FirstPartInSession, and
DisplayMessage are allowed.
Parameters:
ControlSettings: A dictionary containing control settings and an optional data string.

public **void EndDialog**()
Summary:
Has to be called after adding the last piece of information to the dialog.

Improved Design Table Functionality

When design table files are created in PDM they can now be renamed according to the CATIA file that they are related to. It is now also possible to relate one design table document to multiple CATPart documents.

Configuration

By default the first design table document will be named like the CAD document it is related to, and additional design table documents will be named like the CAD document plus "_" plus an index (1, 2, 3, etc.).

It is possible to customize this behavior by setting a custom method for the server setting 'CustomMethod_PreProcCreDlgAttrs'. A sample method for this ('Pwb_Sample_PreProcDlgAttrs') exists.
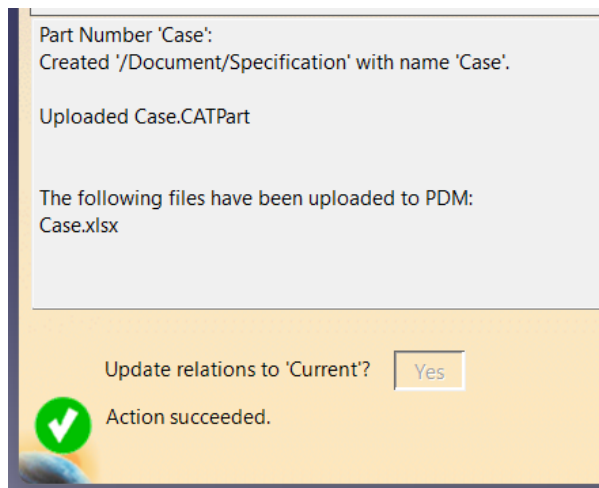
Usage

In this example two design tables are related to the file 'Case.CATPart'.

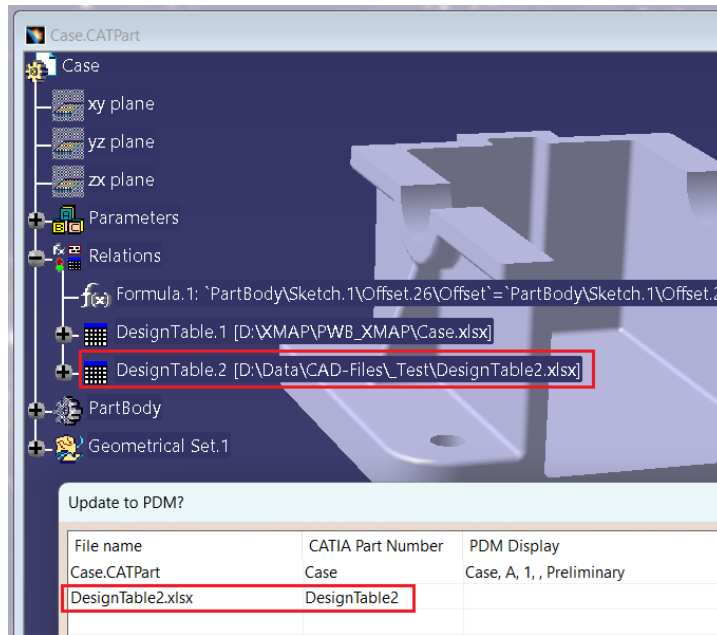The first design table file is originally named 'DesignTable1.xslx':
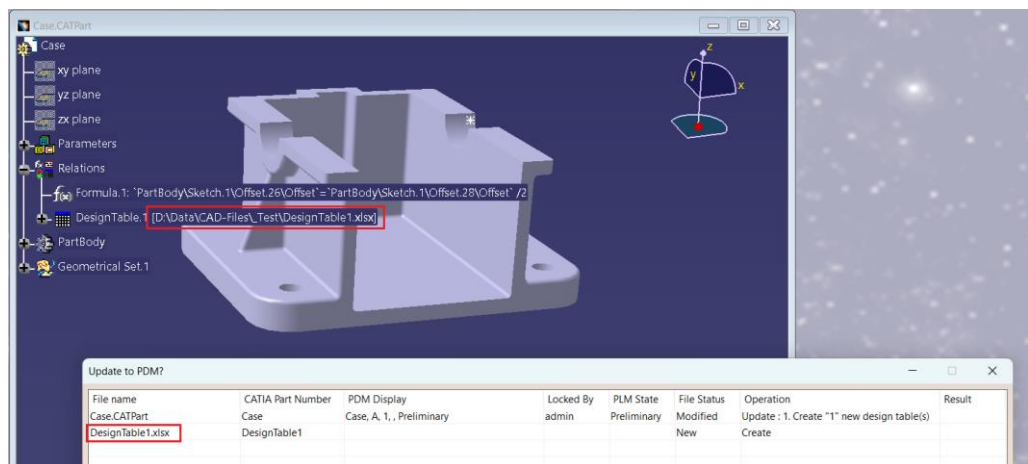
Picture 9: Adding first design table

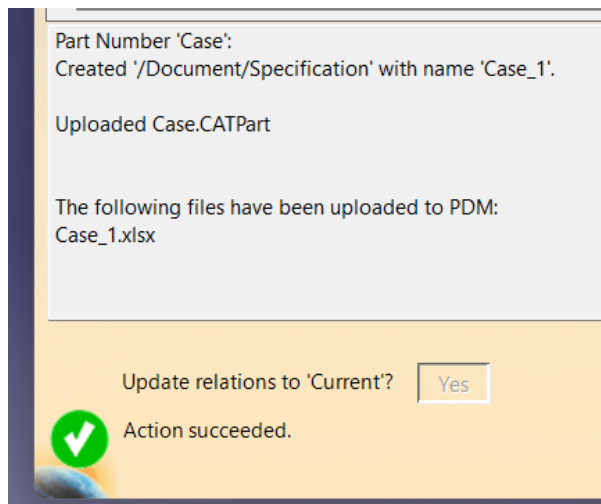PDM update renames the design table file to 'Case.xslx':

Picture 10: Renamed first design table file.

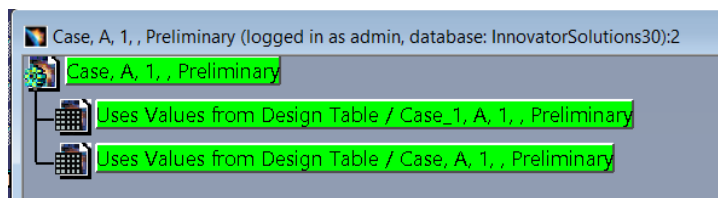A second design table file ('DesignTable2.xslx) is related:

Picture 11: Adding second design table

PDM update renames the design table file to 'Case_1.xslx':
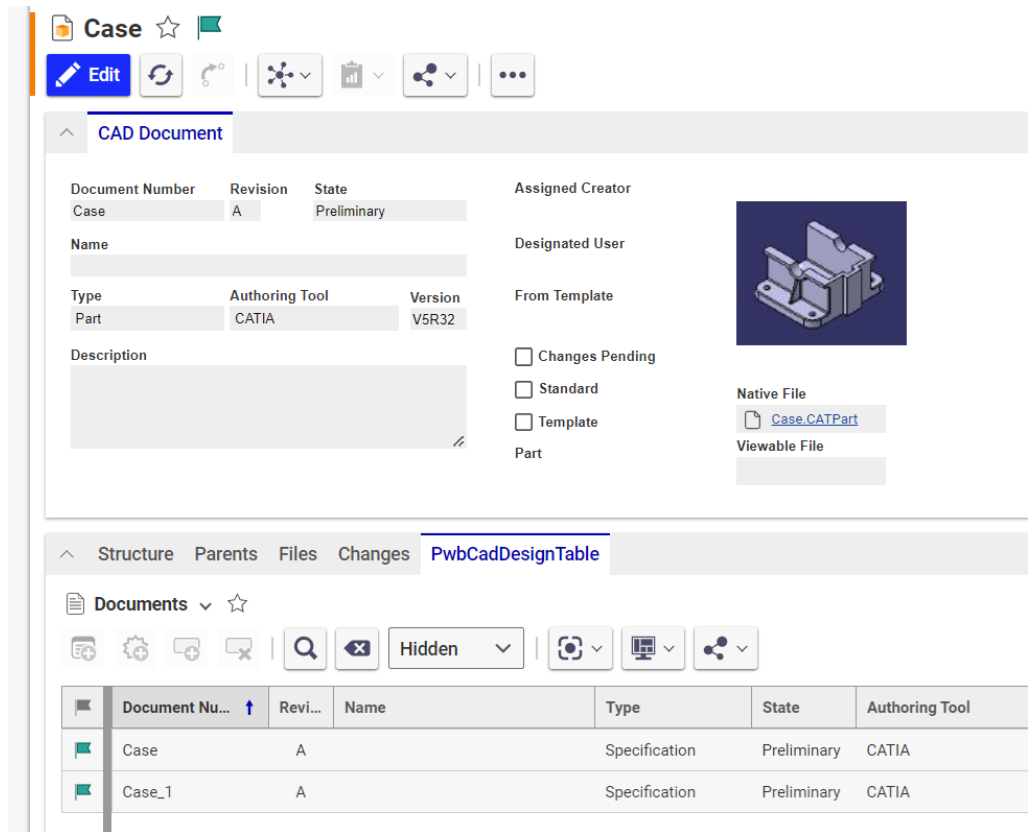

Picture 12: Renamed second design table file.

Both design table documents are related in PDM:

Picture 13: Display of related design tables in the PWB structure window

Picture 14: Display of related design tables in the Aras Innovator web client

"Synchronize CAD to BOM" Command in the Aras Innovator Web Client

This functionality which was previously only available in the query result and the PDM structure window in CATIA is now available in the Aras Innovator web client too:
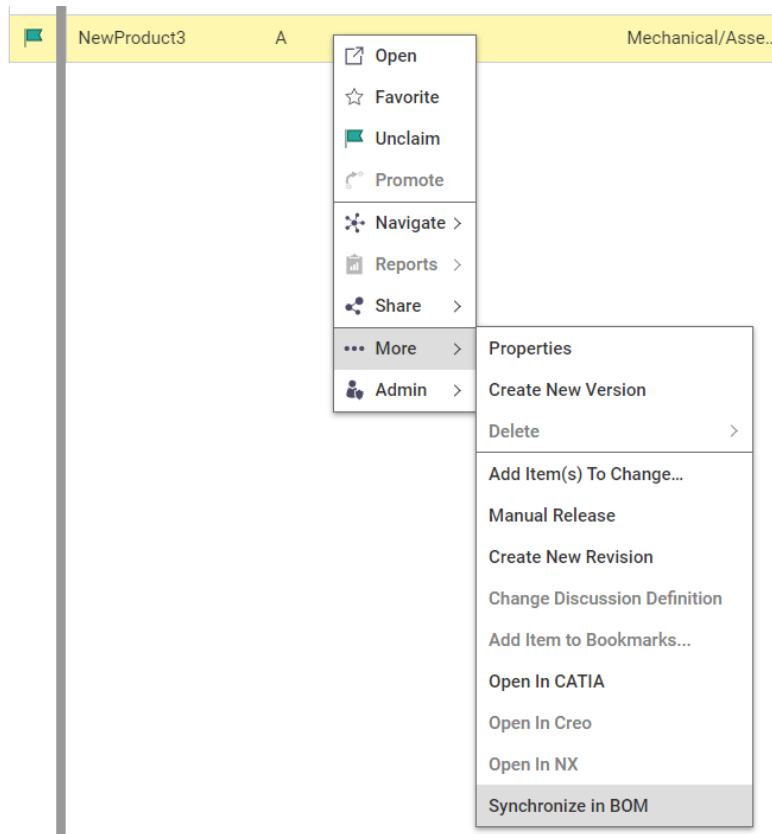
Picture 15: "Synchronize in BOM" in Aras Innovator web client

Configuration

No specific configuration is necessary.

Client/Server "UseBomPartStructure" Setting Compatibility Check

There is an optional attribute in the PWBSchema tag of the Schema file:
UseBomPartStructure=true|false

If it exists then, during login, it is checked against the 'UseBomPartStructure' setting on the server. If the values do not fit login is not possible.
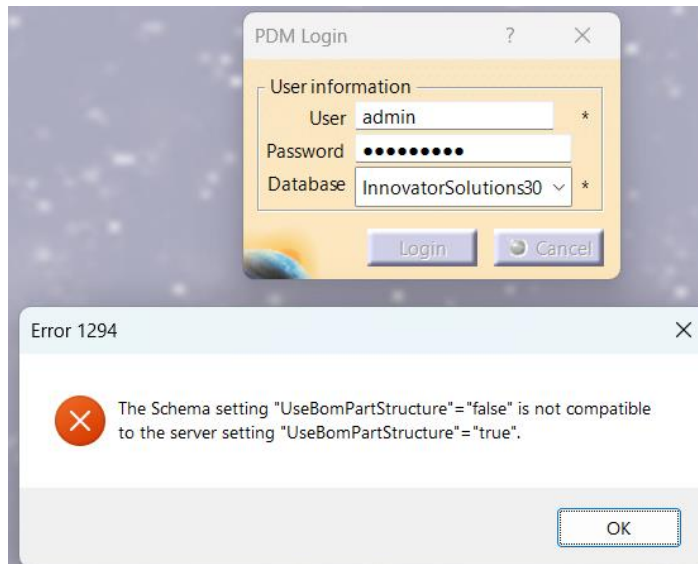
Configuration

Here the value is set to 'false' in the PWBSchema file:

```
<PWBSchemata>

  <PWBSchema system="Aras" customization="Aras" ...
  UseBomPartStructure="false" >

  ...
```

Usage

If the setting on the server is 'true' the user will get this error when trying to log in:

Picture 16: Configuration error at login

Query Performance CAD Structure mode with Part attribute mapping (PWB 18.0.2)

Starting with PWB 18.0.2, when using attribute mapping from a related Aras Part to the file in CATIA, use a set based expand to get the related Parts during query.
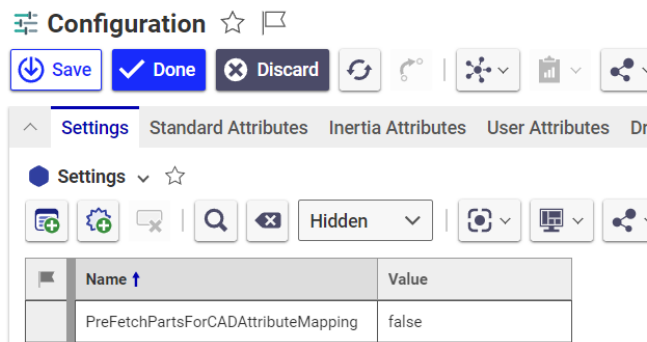
The feature is only relevant if you are in CAD Structure mode (UseBomPartStructure=false) and you have mappings from Part to Cad in "Standard Attributes", "User Attributes" or "Drawing Attributes":

Configuration
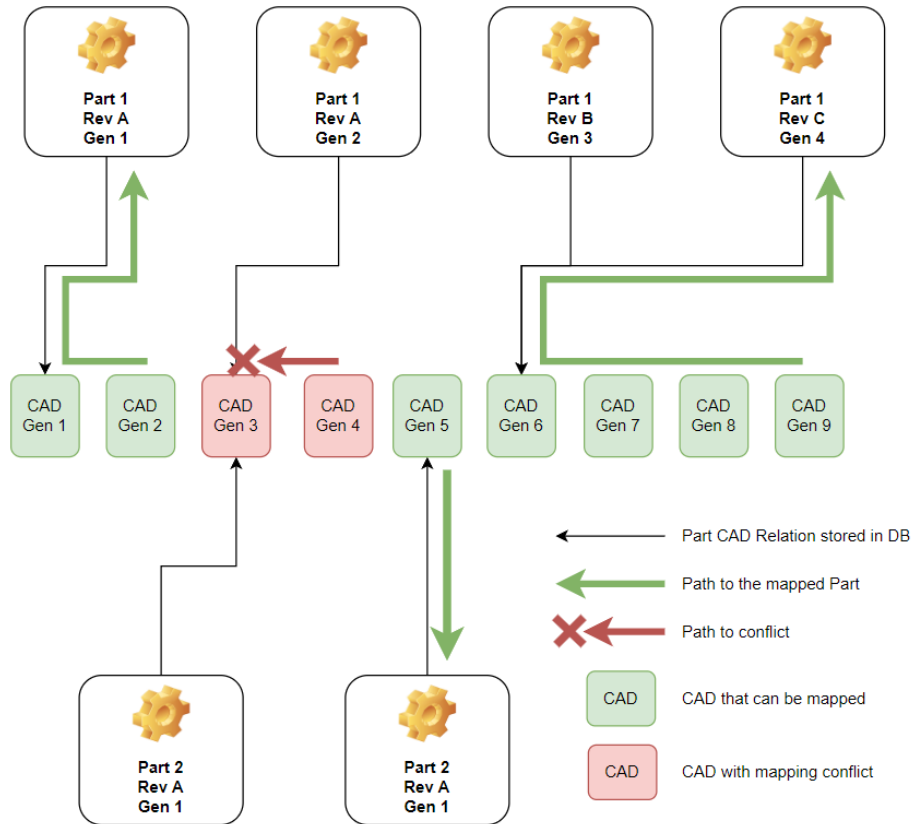
This behavior can be disabled with the PWB configuration setting:

PreFetchPartsForCADAttributeMapping = false (default value: true)

When activated, the set based expand uses the following behavior to get the related Part for a specific generation of each CAD:

If a generation of a CAD is attached to two different Parts, there is no attribute mapping available for this generation of the CAD.

If a generation of a CAD is attached to multiple generations of the same Part, the most recent generation of the Part is used.

If a generation of a CAD is not directly attached to a Part, use the Part of the last generation before the inspected one which is attached to the Part.

CHAPTER 3

Data Model Changes

The following data model changes are introduced in this release of the PDM Workbench.

Added Classes

None.

Added Forms

None.

Changed Server Methods

Added Sample Server Methods

"Pwb_Sample_DynDlgUpdServerMethod", configured in the PWBSchema.xml file.

"Pwb_Sample_AutonameMethod", configured in the PWBSchema.xml file.

"Pwb_Sample_ContextAction", configured in the PWBSchema.xml file.

"Pwb_Sample_SyncCadToBom", configured in the server setting "CustomMethod_SyncCadToBom'.

"Pwb_Sample_CadTreeIconInfo", configured in the server setting "CustomMethod_CadTreeIconInfo".

"Pwb_Sample_PostProcNewItem", configured in the server setting "CustomMethod_PostProcNewItem".

"Pwb_Sample_ ReconnectAtUpdate", configured in the server setting "ReconnectAtUpdateMethod".

"Pwb_Sample_PostProcUpdateInfo", configured in the server setting "CustomMethod_PostProcUpdateInfo".

"Pwb_Sample_CheckPostProcUpdate", configured in the server setting "CustomMethod_CheckPostProcUpdate".

"Pwb_Sample_PostProcUpdate", configured in the server setting "CustomMethod_PostProcUpdate".

"Pwb_Sample_PreProcCadStructure", configured in the server setting "CustomMethod_PreProcessCadStructure".

"Pwb_Sample_PreProcDlgAttrs", configured in the server setting "CustomMethod_PreProcCreDlgAttrs".

"Pwb_Sample_PreProcQryDlgAttrs", configured in the server setting "CustomMethod_PreProcQryDlgAttrs".

CHAPTER 4

Bug Fixes

Version 18.0.1

- Catia Crash with PWBAlignCadAndBomDialog
- CATIA crash when multiple times pressing close to SyncToBom Dialog
- Dynamic dialogs support CustomNoPart action
- Dynamic dialogs: The 'CadFileName' of a child CATPart is of the instance, not the reference.
- Legacy Aux File Create Order incompatible with dynamic dialogs
- Drawing create dialog is presented multiple times
- CreateItem does not work if CustomMethod_PostProcNewItem is not defined
- Manual change of Partname causes PWB Error message

Version 18.0.2

- PwbCheckStandardPartPermissions PwbCheckTemplateFilePermissions always use PwbConfigurationItemName
- Custom Action Dialog does not check if required attributes are filled
- Neighbourhood feature & CAD structure with different revisions opening
- Not possible to open as for visualization after open as current (improve message for user)
- Replace CGR tries to create new object
- Fixed a problem with the query performance if the attribute mapping of part attributes to CAD in CAD structure mode is switched on.

Version 18.0.3

- PwbCheckStandardPartPermissions
- Fixed an error in a specific case of the query performance problem when the attribute mapping of part attributes to CAD in CAD structure mode is switched on.

Version 18.0.4

- CATIA will in some cases stop responding with Client setting "Load with Links" activated

- Cannot re-save CATProducts which have dependencies with empty instances
- Update Property does not work in PSN window
- Mapped Part Attributes if no Part is related
- "Open related drawings" expand links of Drawing AsSaved
- Use Configured Expand mode for related 3D Files of Drawing

Version 18.0.5

- The CATIA description was changed to one line during PDM update.
- CATIA stopped responding in some cases when expand multi-level "where used" was called in the PDM structure window.
- In the BOM Sync in the Innovator web client pressing "No" still executed the sync instead of cancelling it.