CUSTOMER

Configuration Guide

SAP Engineering Control Center 5.2

DOCUMENT HISTORY

⚠ **Note**

Before you start the implementation, make sure you have the latest version of this document.

This guide is valid as of SAP Engineering Control Center 5.2.7.0

The latest version is available on SAP Service Marketplace at https://help.sap.com/viewer/p/SAP_ENGINEERING_CONTROL_CENTER

Version	Date	Change
1.0	2019-05-29	Initial creation
2.0	2019-07-26	• New section added "Configuring Tabs in the Object Browser"
3.0	2019-11-21	• "Configuration Files" section optimized • "Functions for searching from the clipboard" section optimized • "Preference variables for configuring the search from the clipboard" section optimized
4.0	2020-07-02	• “Displaying Imported Documents in the Last Used Documents Container“ section optimized
5.0	2021-04-29	• Note added to "VE Client Converter" section • "Generic Call of Transactions and RFC-Enabled SAP Function Modules" section optimized • "Startup Procedure" section optimized • "Generic Call of Transactions and RFC-Enabled SAP Function Modules" section optimized • "Defining Filters for Windows" section optimized • New "Controlling Automatic Expansion of Containers" section added • "Configuring Previews" section optimized • New "Searching for Documents in the Folder Structure" section added
6.0	2021-07-08	• "Function for Transferring Values from Document to Material" section added • "Searching with SAP Search Helps" section added • "Configuring Additional Input Fields for the Document Creation Dialog" section optimized

Version	Date	Change
7.0	2022-07-14	• Chapter „Configure CADEnvironmentChecker“ deleted. Described in default.txt • „Configuration files“ optimized • New Chapter „Edit description field for parameters in macro editor“ added • „Restricting S_RFC Authorizations for SAP ECTR Users“ optimized • „Configuring Tabs in the Object Browser“ optimized • Chapter „Attribute transfer“ section „Function Modul CAD_GET_TITLE_BLOCK_DATA call via from_sap“ optimized • Chapter “System Environment” section “plm_setenv.bat” optimized • Chapter “Generic Call of Transactions and RFC-Enabled SAP Function Modules” section “Access to environment and preference variables” added • Chapter “Supress BOM items” added
8.0	2023-10-23	• Chapter “Activate or deactivate line numbers in macro editor“ added • Chapter „Generic Call of Transactions and RFC-Enabled SAP Function Modules“ optimized • Chapter “Preference variables for configuring the search from the clipboard” enhanced • Chapter “Configuration Files” optimized

TABLE OF CONTENTS

1 OVERVIEW OF THE SOLUTION

The following graphic provides an overview of a solution using SAP ECTR. SAP ECTR connects your CAD system to your SAP system. The elements highlighted in pink relate directly to SAP ECTR.

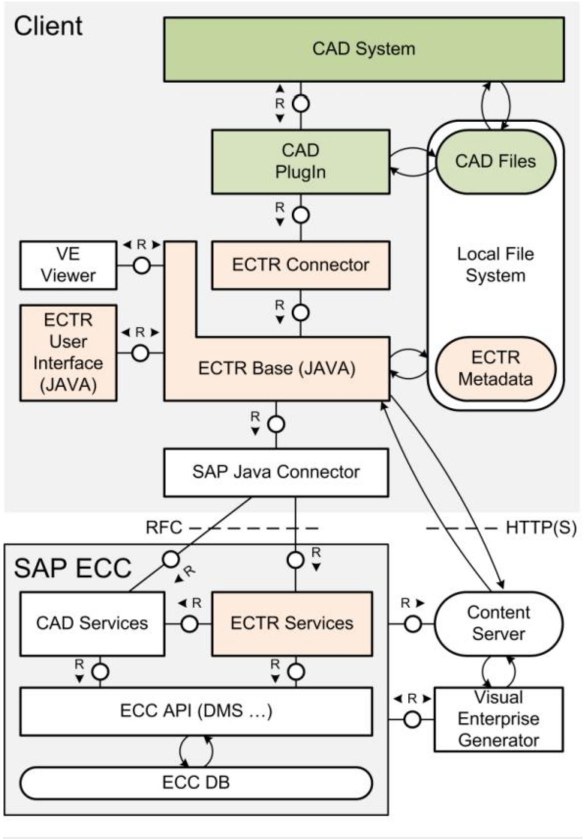


Figure 1: Overview of the Solution

1.1 System Landscape

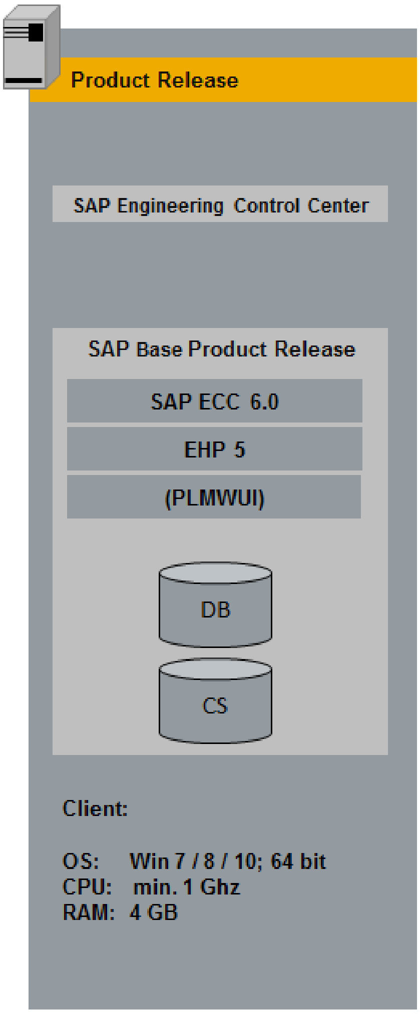


Figure 2: Exemplary System Landscape

1.2 Implementation and Configuration Information

This configuration guide assists you with the configuration of your SAP system. You configure the system in SAP customizing. This guide provides a description of all required steps in the implementation process, SAP default settings for Customizing as well as system configuration activities. Customizing activities and the corresponding documentation are broken down by functional aspects.

2 SAP ENHANCEMENT

You can make customer-specific adjustments in the SAP system.

2.1 Business Configuration Sets

The following Business Configuration Sets are supplied with SAP ECTR:

- /DSCSAG/ECTR_BC_BASE1
- /DSCSAG/ECTR_BC_BASE2
- /DSCSAG/ECTR_BC_BASE_CDESK
- /DSCSAG/ECTR_BC_ECN_STATUS
- /DSCSAG/ECTR_PDF_PIC
- /DSCSAG/ECTR_TXT
- /DSCSAG/ECTR_WRD_XLS_PPT

Structure Business Configuration Sets

/DSCSAG/ECTR_BC_BASE1

- DMS: Maintain data carrier (vault, front end, archive...)
  - Data carrier type `PC`
- Start Processing for Documents
  - ECTR
- Define Document Types
  - Define document types
    - `ANY` - SAP ECTR any
    - `AUX` - SAP ECTR auxiliary
    - `FOL` - SAP ECTR folder
  - Define language-dependent descriptions
    - `EN - ANY` - SAP ECTR any
    - `EN - AUX` - SAP ECTR auxiliary
    - `EN - FOL` - SAP ECTR folder
  - Define document status
    - Predefined status network for the document types `ANY`, `AUX`, and `FOL`
  - Display language-dependent document statuses
    - Status texts for the predefined statuses
  - Define object links
    - Assigns allowed object links to the predefined document types `ANY`, `AUX`, and `FOL`.
- Define Workstation Application
  - Define workstation application
    - `ECF X *` SAP ECTR container file
    - `EPG X *` SAP ECTR preview
    - `EPJ X *` SAP ECTR preview
    - `EVI X *` SAP ECTR viewing meta file
  - Define workstation application in network
    - `PC - EPG`
    - `PC - EPJ`

/DSCSAG/ECTR_BC_BASE2

- Define Data Carrier

  - Define data carrier type "server, front end"
    - `PC`
  - Identify frontend computers
    - `Default - PC`

/DSCSAG/ECTR_BC_BASE_CDESK

- Define CAD Fields used for Explosion Scenario
  - Define CAD Fields used for Explosion Scenario
    - `RES4`
    - `SORTF`

/DSCSAG/ECTR_BC_ECN_STATUS

- Define Statuses for Change Master Records
  - Change Master Status
    - `10` - Collecting Phase
    - `20` - Change Phase
    - `35` - Valid / Productive
    - `60` - Declined

/DSCSAG/ECTR_PDF_PIC

- Define Document Types
  - Define document types
    - `PDF`
    - `PIC`
  - Define document status
    - Predefined status network for the document types `PDF` and `PIC`
  - Display language-dependent document statuses
    - Status texts for the predefined statuses
  - Define object links
    - Assigns allowed object links to the predefined document types `PDF` and `PIC`.
- Define Workstation Application
  - Define workstation application
    - `PDF X PDF`
    - `PIC X PIC`
  - Define workstation application in network
    - `PC - PDF`
    - `PC - PIC`

/DSCSAG/ECTR_TXT

- Define Document Types
  - Define document types
    - `TXT`
  - Define document status
    - Predefined status network for the document type `TXT`
  - Define object links
    - Assigns allowed object links to the predefined document type `TXT`.
- Define Workstation Application
  - Define workstation application
    - `TXT X * text files`
  - Define workstation application in network
    - `PC - TXT`

/DSCSAG/ECTR_WRD_XLS_PPT

- Define document types
  - Define document types

        - `DOC` Word
        - `PPT` PowerPoint
        - `XLS` Excel
    - Define document status
        - Predefined status network for the document types `DOC`, `PPT`, and `XLS`
    - Display language-dependent document statuses
        - Status texts for the predefined statuses
    - Define object links
        - Assigns allowed object links to the predefined document types `DOC`, `PPT`, and `XLS`.
- Define Workstation Application
    - Define workstation application
        - `PPT X Powerpoint`
        - `WRD X A2 Word`
        - `XLS X * Excel`
    - Define workstation application in network
        - `PC - PPT`
        - `PC - WRD`
        - `PC - XLS`

For general information and notes on activation, see SAP Help Portal.

2.2 Customizing

You call SAP customizing by choosing transaction `SPRO` → `SAP Reference IMG`.

2.2.1 Data Carrier and Workstation Applications

To change the workstation applications, go to SAP Customizing and choose

`Cross-Application Components` → `Document Management` → `General Data` → `Define Workstation Application`

If the workstation applications listed below do not exist, you have to make the following entries as the minimum:

Workstation application	**Description**	**Suffix**	**File format**
ECF	SAP ECTR-Container File	`ecf`	`*.ecf`
EPG	SAP ECTR-r Preview	`gif`	`*.gif`
EPJ	SAP ECTR-Preview	`jpg`	`*.jpg`
TXT	Text files	`Txt`	`*.txt`
XML	XML	`xml`	`*.xml`

If you want to define additional workstation applications, repeat this process.

2.2.2 Settings for the SAP Knowledge Provider (KPro)

To use the KPro, set up the following preference variables in the `default.txt` file. Definition of the default storage category:

```
plm.kpro.storageCateg.default = <storage_category>
```

Example

```
plm.kpro.storageCateg.default = Z_PLM
```

If you wish to use a special combination of document type, workstation application and another storage category, you can create additional entries in the `default.txt` file according to the following schema:

```
plm.kpro.storageCateg.<document_type>.<workstation_application> = <storage_category>
```

Example

```
plm.kpro.storageCateg.DRW.UGI = Z_PLM_2
```

Note

KPro is required in order to create documents with originals in SAP ECTR.

2.2.3 Preventing Two Active Content Versions

In SAP customizing, you can set it up so that a content version is created for each save and display. A content version is not a document version. Here, the latest version of the content is set to active and loaded when the document is opened again. If communication problems occur during save and display, two content versions can be marked as active in exceptional cases. In that case, opening the document is not possible due to the unclear assignment. However, the creation of such duplicate content versions is to be prevented.

The DSC-Report (SE38) /DSCSAG/DMS_KPRO_DOUBLE_REPAIR enables the administrator to repair the double content versions. The useless content version can be easily deactivated with the report.

Note

In order to prevent the emergence of two active content versions, proceed as described in Note `1403231` and put in the method `SET_KPRO_UPDATE_MODE` of BAdIs `DOCUMENT_MAIN01` the flag `SET_UPDATE_MODE` to `'X'`.

2.3 Documents

2.3.1 Defining Document Types

You make the required settings for the document type in this operation. You can describe documents language-specifically, define the document status and specify object links here. You make these settings in the Customizing under:

`Cross-Application Components` → `Document Management` → `Control Data` → `Define Document Types`

For each operation, online documentation is available directly in the maintenance dialog.

Defining a New Document Type

You can define a new document type by copying the existing document type DRW.

The preset document type is:

`AUX` = for the `Recently Used Documents` folder in SAP Engineering Control Center

Adjust these document type.

Specifying Attributes for a Document Type

When you set up the attributes for a document type, keep the following in mind:

- You have selected `KPro Storage` (document storage using Knowledge Providers).

- The "file size" value is only checked if originals are not stored using a Knowledge Provider. Hence this value is not relevant here.

2.3.2 Defining the Status

A status must be defined for each document type. As the minimum, the two statuses AC (`zugänglich` or `accessible`) and IA (`in Arbeit` or `in Work`) are required. The settings in the preference files must also be checked. The status settings must be the same.

Creating a New Status

You can maintain the description of the new status in the `Define Document Status` dialog. You can use `New Entries` to define a new status entry for a document type here.

Note

Document status "`L`" is not currently supported by SAP ECTR.

Defining Status Transitions

In SAP PLM, the change from one status to the next is only possible by means of predefined status transitions. You define these status transitions by double-clicking the table row of a document status and opening the maintenance dialog of this document status entry.

You define the status transition by entering the respective predecessor status for the current document status in the input fields for the predecessors.

Define the following status transitions:

Document status	Input field	Predecessor status
AC	Predecessor 1	AA
AC	Predecessor 2	IA
IA	Predecessor 1	AC
IA	Predecessor 2	AA
FR	Predecessor 1	AC

The status network consists of all the status transitions for a document type in SAP PLM.

Note

In the SAP system, you can define additional statuses for a status network, which have neither a predecessor nor a successor status. The SAP GUI offers this status as a target status for any status and you can switch to this status whenever you wish. Normally, SAP ECTR assumes that there is a continuous status network and filters out these statuses. Hence, they are not offered as a target status. The display of these statuses in the SAP ECTR status change dialog can be set using the following preference variable in the `systems-default.txt` file:

```
# Status change for statuses without predecessor

plm.document.statuschange.allowStatusWithoutPredecessors = true
```

2.3.3 Maintaining Object Types and Class Types

In this IMG activity, you can maintain new classifiable object types and class types for an object type and data that depends on the class type, .e.g. class status and application views. This is done in Customizing under:

`Cross-Application Components → Classification System → Classes → Maintain Object Types and Class Types`

2.4 Scheduling Report /DSCSAG/OBSERVER_CLEAR

1. For the SAP-side configuration, you have to schedule report `/DSCSAG/OBSERVER_CLEAR`. The report clears the `Observer` table. This can be used to ensure that the table does not become too large. Define the settings for execution in the background in transaction `SE38`.
2. Enter the report name and schedule the report periodically.
3. Set the desired interval.

2.5 Restricting S_RFC Authorizations for SAP ECTR Users

To restrict the "`S_RFC`" authorizations for SAP ECTR users, you need to change two rules in the authorization data by using transaction `PFCG`.

1. In transaction `PFCG`, choose the Basis authorization role for SAP ECTR users.
2. Change the authorization object `S_RFC` in the role.
3. In the `RFC_NAME` field, replace the value `*` with the following values:

Rule1

Activity	16	ACTVT
RFC object type	`FUGR`	`RFC_TYPE`
Name (whitelist) of the RFC object	`/DSCSAG/* ; BAPT;  CADR; CDESK*; CLBP; CVBAPI; ITOB_BAPI_EQ; ITOB_BAPI_FL; RFC_METADATA; SYSU`	`RFC_NAME`

If the value `/DSCSAG/*` is not allowed, you need to add the following entries:

- `/DSCSAG/ADV_ASSEMBLY`
- `/DSCSAG/ADV_ASSEMBLY1`
- `/DSCSAG/ADV_MAIL`
- `/DSCSAG/ADVOBJ`
- `/DSCSAG/AUDIT1`
- `/DSCSAG/AUTH`
- `/DSCSAG/BSPLM`
- `/DSCSAG/CDESK`
- `/DSCSAG/CHANGE`
- `/DSCSAG/CHANGE_L1`
- `/DSCSAG/CHANGE_MGT`
- `/DSCSAG/CLASS`
- `/DSCSAG/CLASS_L1`
- `/DSCSAG/CLONE`
- `/DSCSAG/CONSISTENCY`
- `/DSCSAG/CONVCUST`
- `/DSCSAG/DOCUMENT`
- `/DSCSAG/DOCUMENT_L1`
- `/DSCSAG/DOCUMENT_SEARCH`

- /DSCSAG/DOCUMENT2
- /DSCSAG/DOCUMENT2_L1
- /DSCSAG/ENHANCEMENT
- /DSCSAG/EQUIPMENT_TLM
- /DSCSAG/EXCHANGE
- /DSCSAG/FACTORY
- /DSCSAG/FACTORY_TLM
- /DSCSAG/FOLDER
- /DSCSAG/FOLDER_L1
- /DSCSAG/LEG001
- /DSCSAG/LEG002
- /DSCSAG/MASTERMODELL
- /DSCSAG/MASTERMODELL_L1
- /DSCSAG/MATERIAL
- /DSCSAG/MATERIAL_L1
- /DSCSAG/MIGRATION
- /DSCSAG/MIGRATION_TOOLS
- /DSCSAG/MM_PURCHASING
- /DSCSAG/MONITORING
- /DSCSAG/MSIT
- /DSCSAG/NOTIFICATION
- /DSCSAG/PLM
- /DSCSAG/PLM_ADV_ASSEMBLY
- /DSCSAG/PLM_BOM
- /DSCSAG/PLM_BOMC
- /DSCSAG/PLM_CLASS_EXCHANGE
- /DSCSAG/PLM_CR
- /DSCSAG/PLM_ECR
- /DSCSAG/PLM_L1
- /DSCSAG/PLM_PSMA
- /DSCSAG/PLM_PSMS
- /DSCSAG/PLM_WORKFLOW
- /DSCSAG/PLM_WUI_ACM
- /DSCSAG/PLM_WUI_BASE
- /DSCSAG/PMA
- /DSCSAG/PMS
- /DSCSAG/PRODUCTION
- /DSCSAG/PROJECT_SYSTEM
- /DSCSAG/QM01
- /DSCSAG/REFILE
- /DSCSAG/REFILEGT
- /DSCSAG/RENAME_R
- /DSCSAG/RENAME_REPAIR
- /DSCSAG/SALES_ORDER
- /DSCSAG/SMARTHELP
- /DSCSAG/SMARTSEARCH
- /DSCSAG/SMARTSET
- /DSCSAG/TABLE_EDIT
- /DSCSAG/TEXT
- /DSCSAG/UPDATE
- /DSCSAG/UPDATE_L1
- /DSCSAG/UTILS
- /DSCSAG/UTILS2
- /DSCSAG/VERSION_BOM
- /DSCSAG/CV_ITERATIONS
- /DSCSAG/WORKFLOW46

If the value `CDESK*` is not allowed, you need to add the following entries:

- `CDESK_SRV`
- `CDESK_SRV_ADM`
- `CDESK_SRV_COLL`
- `CDESK_SRV_MBOM`
- `CDESK_SRV_SEARCH`
- `CDESK_SRV_UI`

Rule2

Activity	16	ACTVT
RFC object type	`FUNC`	`RFC_TYPE`
Name (whitelist) of the RFC object	`BAPI*; RFCPING; SYSTEM_ATTACH_GUI`	`RFC_NAME`

Note

When preference variables are loaded, SAP ECTR can determine whether the value of the preference variable was redefined or whether a preference variable was relocated. SAP ECTR delivers the original value, the original file, the new value, and the new file. This function can be defined using the environment variable `TRACE_PREF_CHANGES = true` in the file `plm_initialize.bat`.

To customize the installation, you need to make settings in various files. These files are generally located in the following directory:

`%PLM_INSTDIR%\customize\config`

Note

Environment variable `%PLM_INSTDIR%` contains the path to the SAP ECTR installation directory. It is set when SAP ECTR is started.

Note

These directories are created with the prefix "`template`-" during a patch installation:

`%PLM_INSTDIR%\template-customize`

When the SAP PLM integration is started for the first time, directories with the prefix "`template-`" are copied to directories without prefixes.

1. Start by maintaining the settings files.
2. Start SAP Engineering Control Center by calling up `dsc_StartPlm.exe` in the Windows environment.

Note

Updates/patches only contain files from "`template`" directories. Customer settings are retained.

The following files establish the required system configuration and start SAP ECTR.

The call structure of the start scripts is as follows, starting with the installation directory `%PLM_INSTDIR%`:

```
│
├─ \basis\sys\win\bin\dsc_StartPlm.exe        Starts SAP ECTR either on its own
│                                             or with another application.
│
├─ \customize\config\
│      │
│      ├─ plm_initialize.bat                  Initializes SAP ECTR
│      │
│      ├─ SAPlogon.txt                        Specifies the SAP logon
│      │                                      parameters.
│      └─ plm_setenv.bat                      Sets the environment variables for
│                                             SAP ECTR.
│
```

`└─ \applications\ugs\`	For the NX application, for example.
`   ├─ basis\sys\win\bin\start_application_ugs.exe`	Starts the application.
`   └─ customize\config\plm_setenv.bat`	Sets additional environment variables.

3.1 Startup Procedure

When SAP ECTR is started, initialization is controlled by the `plm_initialize.bat` file.

This initialization file contains the following default settings which are required before logging on to the SAP system:

- Logon language
- Name of the file with SAP logon parameters
- Name of the directory in which the local copy of the installation can be stored

The file is located in the following location:

`%PLM_INSTDIR%\customize\config\plm_initialize.bat`

For more information, refer to the actual file.

dsc_StartPlm.exe

`Path: %PLM_INSTDIR%\basis\sys\win\bin\`

- Processes the transferred start parameters
- Starts SAP ECTR as well as the application
- Initializes the communication between SAP ECTR and the application

Start Parameters

A list of available start parameters can be found here:

`%PLM_INSTDIR%\basis\sys\win\bin\command-help\dsc_StartPlm.txt`

plm_initialize.bat

`Path: %PLM_INSTDIR%\customize\config\`

The system calls the script when SAP ECTR is started. The script defines the environment variables for initializing the SAP logon.

For more information, refer to the actual file.

3.2 System Environment

Preference files are text files containing the variables that control the behavior of SAP ECTR. You can adjust the preference files to the respective environment; these files are kept in the following location:

`%PLM_INSTDIR%\customize\config`

Basic Settings

Maintain the basic settings (e.g. environment variables, directories) of SAP PLM integration in file `plm_setenv.bat`. These default settings are read after logging on to the SAP system.

The following information is available:

- Local working directory

- Order of the preference files
- Directory for previews

This information can be used within the basic settings.

The file is located in the following location:

`%PLM_INSTDIR%\customize\config\plm_setenv.bat`

For more information, refer to the actual file.

Note

Please note that some of the preference variables described in this documentation already exist in the relevant preference files, whereas others must be explicitly added to the end of the file.

plm_setenv.bat

`Path: %PLM_INSTDIR%\customize\config\`

This script defines the system variables which control the settings and responses of SAP Engineering Control Center. On start, a role-dependent `plm_setenv_<role>.bat` is loaded. The role is supplied either by SAP role definitions or by using parameter`"-I"`.

For more information, refer to the actual file.

Note

If changes are made in the file it must be noted that these do not delay the start of SAP ECTR. The runtime of the file must not be higher than the parameter set in the preference variable `plm.start.script.maxWaitTime`. This variable can be found here: `%PLM_INSTDIR%\customize\config\SAPlogon.txt`

3.3 Basic System Properties

system-default.txt

`Path: %PLM_INSTDIR%\basis\config\`

SAP ECTR processes the preference variables of this file to control basic system properties. The basic system properties are:

- Function call for reading user details
- Basic observer settings
- Maintenance of additional change number fields in SAP ECTR
- Dictionary key for the container-object link (only the latest version of the document)
- Check release and environment
- Display class on desktop as class or folder
- Disable SAP warnings/error messages
- Document type priority list for adding standard parts with SAP classification
- Avoid calling unnecessary modules when changing documents in the Object Browser
- Hide document placeholder in the change number

For more information, refer to the actual file.

3.4 Preference Files

The `default.txt` file contains settings for the SAP system and SAP PLM integration. SAP logon details are maintained in the `SAPlogon.txt` file.

Logon-Specific Preference Variables

Once logon is completed, the following additional information is available in the initialization file `%PLM_INSTDIR%\customize\config\plm_setenv.bat`:

Environment Variable	Description
`PLM_ROLE`	User role as defined in SAP
`PLM_SYSTEM`	SAP system to which the user is logged on
`PLM_CLIENT`	Client to which the user is logged on
`PLM_USER`	SAP user name
`PLM_LANGUAGE`	User's default logon language

This information can be used to further customize the installation. You can configure it so that certain administrative functions are only accessible to users with certain roles.

To do this, additional settings files can be called using the file `plm_setenv.bat`:

Example

```
set PLM_PREFFILES=layout.txt;default_bu1.txt;menu_bu1.guidef
if "%PLM_ROLE%" EQU "Z_PLM_ADMIN" (
   set PLM_PREFFILES=%PLM_PREFFILES%;admin.txt
)
```

You can also use this mechanism to personalize installation paths or make other user-specific, SAP system-specific or logon language-specific settings.

If you want to start SAP ECTR with user-specific preference files, amend your `%PLM_USER%.txt` file in the `plm_setenv.bat` file at the relevant position next to `PLM_PREFFILES`.

After logon, SAP ECTR calls up an additional settings file:

`%PLM_INSTDIR%\customize\config\R3_<SAP_SYSTEM_ID>.txt`

Example

`R3_D01.txt`

Once the user is logged on to an SAP system D01, this file assumes highest priority. Settings contained in this file are not overwritten.

Microsoft Office-Specific Preference Variables

You can customize the Microsoft Office integration by using the following files:

- `default.txt`
- `plm_setenv.bat`
- `.ini file`

The preference files contain the descriptions of the individual parameters.

SAPlogon.txt

Path: `%PLM_INSTDIR%\customize\config\`

SAP ECTR uses the preference variables defined in this file to configure the SAP logon dialog.

For more information, refer to the actual file.

options.xml

Path: `%PLM_INSTDIR%\customize\config\`

User-specific reference variables that control the behavior of SAP ECTR can be defined in this file. You assign the preference variables by choosing `Configuration Options -> Preferences` in the system bar.

For more information, refer to the actual file.

3.5 Configuring the Tab Pages in the Options Menu

You can show and hide the tab pages in the Options menu. This is controlled using preference variable `plm.optionGui.tabOrder` in the `systems-default.txt` file.

Possible values are:

Value	Tag pages
Gen	`General`
CLS	`Classification Settings`
BOM	`Bill of Material`
UI	`UI`

3.6 Distribution of Configurations via the SAP System

You can distribute the SAP ECTR configurations via the SAP system.

For a detailed description, refer to the SAP Community WIKI:

https://wiki.scn.sap.com/wiki/display/PLM/How+to+configure+the+distribution+of+SAP+ECTR+configurations+via+SAP+system

4 LAYOUT AND GUI RESPONSE

4.1 Specifying a Default Layout

When started for the first time, SAP ECTR does not yet have a layout, and therefore reads the layouts from the `system-layouts.txt` file.

Users can personalize the layout of SAP ECTR as they wish. You can define an admin layout that strictly defines the layout settings and prevents personalization by users. Admin layouts are defined in the file `%PLM_INSTDIR%\customize\config\system-layouts.txt` . You use preference variable `plm.om.admin.layout.initial` to determine which admin layout is read from this file. Assign the preference variable the name of the desired admin layout. The name of the admin layout is determined in `system-layouts.txt`, using the character string between the beginning of the line and the first period. Please note that the admin layout is loaded the first time SAP ECTR is started.

4.2 Menus

Menus and toolbars are configured in the files `menu.guidef` and `menu_macros.txt`.

The files are located in the directory:

- `%PLM_INSTDIR%\customize\config`

4.2.1 Configuration Files

Link to Function Groups

Various DTypes, which can be processed by the same application, can be summarized into `function groups`. Documents within the same function group use identical layouts, icons and context menus in SAP Engineering Control Center.

Layout settings can be adopted from other function groups, which speeds up and facilitates the layout creation.

You are assigned to a group through function groups.

For documents, assignment to function groups is controlled using DTypes. For folders, this is controlled using preference variables.

Menu Definitions

Menu entries are maintained in the files `menu_macros.txt` and `menu.guidef`. Recurring function sequences in menus are defined as macros in the file `menu_macros.txt`, which thereafter can only be referenced by the menu itself.

The selection of the menu file is configured using environment variable `PLM_PREFFILES` in configuration file `plm_setenv.bat`.

Example

```
set PLM_PREFFILES=layout.txt;default_bu1.txt;menu_bu1.guidef
```

You can specify multiple menu definition files. Files that are located further back in the list defined in `PLM_PREFFILES` are added to the previous definitions.

Structure of the File menu_macros.txt

A question mark indicates the start of the macro definition. This is followed by the identifier of the menu without numbering. This identifier is only required once. A macro definition can contain a reference to another macro.

Menu flyouts are defined by indentation, which must be created using spaces.

Tabs are not allowed in the menu file. SAP ECTR uses dashes (`---`) to represent hyphens.

Example of the Definition of a Macro Reference

```
# ****************************************************************************
# Macro Definitions
# ****************************************************************************

? DOC_MASTER_TOOLBAR_DESK     = fnc.doc.change.multi
                              = fnc.doc.status.change
                              = fnc.doc.new.version
                              = fnc.mat.create.multi
                              = ----------------------------
                              = fnc.sap.note.send
```

Structure of the File menu.guidef

A + sign in column 1 indicates the start of a menu definition. This is followed by the identifier of the menu without numbering. This identifier is only required once and does not need to be repeated in each line of the menu.

Menu flyouts are defined by indentation, which must be created using spaces.

Tabs are not allowed in the menu file. SAP Engineering Control Center uses dashes (`---`) to represent hyphens.

Example of including a new function in the context menu of folders

1. Search for a function that you want to insert into the context menu of folders. The easiest way to find a function name is in the dictionaries.
2. Open the ectr.txt file in the installation directory.

   `basis` → `dictionary` → [your language version] → `ectr.txt`

   Besides the function name, the system name of the function is also displayed here. The system name always begins with `fnc.` Copy the system name of the function.
3. Open `customize` → `config` → `menu.guidef`
4. Navigate to the `Normal Folder Definitions` section.
5. Here, you can change the context menu and the toolbar for folders and containers and adjust the sequence.

   The `om.popup.menu.FLD` entry describes the context menu of folders and containers.
6. Enter the required function in the following form:

```
+  om.popup.menu.FLD                  = fnc.fld.setActive
                                      = [your function]
```

Example of the Inclusion of a Macro Reference in the Menu

```
+  om.custom.tool.DOC.STN.ctx_desk    = ? DOC_MASTER_TOOLBAR_DESK
                                      = ----------------------------
                                      = fnc.object.refresh
```

Example of the Inclusion of Environment Variables in a Macro Reference in the Menu

Environment variables can be used in macro references in menus.

In the `menu.guidef`, a definition could be as follows:

```
+   om.popup.menu.DOC.STM                = ? DOC_GENERAL
                                         = -----------------------------
                                         = ? DOC_DOC
                                         = ? %DOCMAT_MACRO%
```

The relevant macro definition in the `menu_makros.txt` could be as follows:

```
? DOC_MAT                        = mnu.flyout.doc_mat
                                 = fnc.mat.create.multi
                                 = fnc.doc.allocate.material
                                 = -----------------------------
                                 = fnc.mat.display
                                 = fnc.mat.change
                                 = fnc.mat.create
                                 = -----------------------------
                                 = ? DOC_MAT_WUI
```

Finally, you define the environment variable in the `plm_setenv.bat`. To do this, use the value `PLM_ROLE`. You also use this value to name the compiled menu files. If a different logic is used, value changes are not made to the environment variable. Problems can arise because the menu file has not been decompiled.

```
set DOCMAT_MACRO=DOC_MAT_%PLM_ROLE%
```

Note

To comment out a line, insert a hash # at the start of the line.

SAP Role-Dependent Menu Files

In SAP ECTR, menu files can be defined according to the SAP role. For each logon to the SAP system, SAP ECTR determines the roles which have been assigned to the user in the SAP system. SAP ECTR checks these roles against a list of roles previously defined using a preference variable. When a match of roles is determined, this role is retained as the user role for the current SAP ECTR session.

The list of roles is defined in the configuration file `%PLM_INSTDIR%/customize/config/SAPlogon.txt` using the variable `sap.logon.userrole.list`.

The role with the most privileges is at the top of the list, followed by roles with fewer privileges in descending order, e.g.:

```
sap.logon.userrole.list = SAP_ADMIN; SAP_USER_EDIT; SAP_USER_SHOW
```

If there is no match between SAP roles and those on the list, “`FULL`” is used as a role.

In SAP ECTR configuration files, the role is available in two ways:

- Using environment variable `%PLM_ROLE%`
- In the preference files, using the placeholder `@PLM_ROLE@`

The role can be used for the definition of role-dependent configurations.

Example

Example application of the variable `%PLM_ROLE%` in `plm_setenv.bat` in order to dynamically load a role-dependent menu definition file:

```
set PLM_PREFFILES = system-default.txt; system-layouts.txt; default.txt;
%PLM_ROLE%_menu.guidef
```

Note

In the SAP system, you can use characters that are invalid for Windows file names when you name a role:

```
< > : " / \ | ? *
```

Before assigning them to the environment variable `%PLM_ROLE%`, SAP ECTR replaces these characters with `'_'`. A role in SAP with a name such as `CATEGORY:USER` is assigned to the environment variable `%PLM_ROLE%` as `CATEGORY_USER`.

Hide Functions

Using the file `menu_suppress.txt`, you can hide specific functions. The file is located in the following directory:

```
%PLM_INSTDIR%\customize\config
```

In this blacklist, you can use `!` to mark specific functions as `whitelist` entries.

Example

```
# suppressed
fnc.doc.change


# not suppressed
!fnc.doc.change.user.defined
```

This configuration sets a global lock on all functions that start with `fnc.doc.change`, but allows the function `fnc.doc.change.user.defined`.

4.2.2 System Bar Menu

Menu entries are defined according to the following schema: `<Identification of the menu entry> = <Identifier of function to be executed when menu entry is clicked>` Menu entries for the system bar, toolbars and context menus are identified as follows:

```
om.<value>.menu.<sequence_number>
```

The default tags for `<value>` are as follows:

<value>	Description
`file`	`System` drop-down menu
`edit`	`Edit` drop-down menu
`help`	`Help` drop-down menu
`options`	`Administration` drop-down menu

SAP ECTR automatically generates the `Window` drop-down menu. This menu contains all the contexts (Desktop, Object Browser, etc.), which are listed in preference variable `plm.om.context.enabled` in `menu.guidef`.

Other user-defined menus with custom tags can be used for `<value>`. To alert SAP ECTR that these additional menus are in use, the following preference variable is set in the file `menu.guidef`:

```
om.menu.additionals = <value>; <value>.
```

You can control the order of menu entries using sequence numbers. When making changes, be sure that number sequencing is uninterrupted.

4.2.3 Toolbars

Toolbars for Each Window

```
om.custom.tool.<context>.<sequence_number>
```

The default tags for `<context>` are as follows:

<context>	Description
ctx_desk	Desktop
ctx_ass	Assembly Window
ctx_ob	Object browser
ctx_clh	Classification Window
ctx_so	SAP Office
ctx_clone	Clone window
ctx_bom	Bill of Material context
ctx_vis	SAP 3D Visual Enterprise Viewer

The only exception is the `Active List` window. Here the structure is as follows:

```
om.activelist.tool.<object_type>.<sequence_number>
```

The object type can be a material `MAT` or a document `DOC`. Sequence numbers can also consist of two figures. This is used to define submenus.

Recommendation

In the toolbar, you can set the distance between individual components using the following preference variables in the file `menu.guidef`. This is specified in pixels:

```
om.toolbar.insets.top=4
om.toolbar.insets.left=4
om.toolbar.insets.bottom=4
om.toolbar.insets.right=4
```

Customizing a Window Toolbar

The complete context menu of the selected object is normally displayed as a toolbar e.g. in the Object Browser. Where context menus are extensive this means that the toolbar won't have sufficient space for all functions.

For better oversight in a context toolbar, you can create your own menu definitions. Your menu definitions are dependent on the currently selected object.

This can be done to various degrees of specificity:

```
om.custom.tool.<object_type>.<function_group>.<context>.<sequence_number>

om.custom.tool.<object_type>.<context>.<sequence_number>

om.custom.tool.<object_type>.<function_group>.<sequence_number>

om.custom.tool.<object_type>.<sequence_number>
```

Placeholder	Description
`<object_type>`	Object type identifier, e.g. `DOC`, `MAT`, `CHANGE`
`<function_group>`	Function group
`<context>`	Context identifier which is displayed in the toolbar. You can find a list of available contexts in the file `menu.guidef` under `UI Context Definitions.`
`<sequence_number>`	Sequence number for the toolbar entry, starting with 1. The number is omitted in the new notation of menu definitions.

The Object Browser searches for menu definitions in the order shown above. When a matching menu definition is found, the menu definition processes and defines the appearance of the toolbar.

When an explicit toolbar definition cannot be found, the previous response applies, with the context menu displayed as a toolbar. The following search order applies to the context menu definition:

```
om.popup.menu.<context>.<object_type>.<function_group>.<sequence_nummer>

om.popup.menu.<context>.<object_type>.<sequence_number>

om.popup.menu.<object_type>.<function_group>.<sequence_number>

om.popup.menu.<object_type>.<sequence_number>
```

Response for Multiple Selections

The following applies if all objects belong to the same type and function group:

```
om.custom.tool.<object_type>.<function_group>.<context>.<sequence_number> =...
```

The following applies if objects belong to the same type but different function groups:

```
om.custom.tool.<object_type>.<context>.<sequence_number> =...
```

The following applies if objects belong to different types:

```
om.custom.tool.<context>.<sequence_number> =...
```

4.2.4 Context Menus

Context Menus for Objects

```
om.popup.menu.<object_type>.<function_group>.<sequence_number>
```

The following applies:

Placeholder	Description
`<object_type>`	Identifier of the object type, e.g. DOC, MAT, CHANGE, etc.
`<function_group>`	Optional: Identifier of the function group
`<sequence_number>`	Sequence number for the toolbar entry, starting with 1. The number is omitted in the new notation of menu definitions.

Configuring Context Menu Entries

In SAP ECTR you can display additional menu items in context menus. You can configure these additional menu entries using the following menu definitions:

```
om.popup.menu.children.<parent-object>              = <Identifier of function to be
executed when menu entry is clicked>
om.popup.menu.children.<parent-object>.<object_type> = <Identifier of function to be
executed when menu entry is clicked>
```

The `<parent-object>` in this example denotes the object link container. Using `<object_type>` you can make these menu definitions object type-dependent, e.g. for materials.

To remove the supplied OMFs from the context menu in a certain window for a certain object, use these menu definitions:

```
+ om.popup.menu.children.OLINKS     = fnc.delete.object.links

+ om.popup.menu.children.OLINKS.MAT = fnc.delete.object.links
                                    = fnc.mat.link.mark
```

The first line indicates that the menu entry `Delete Object Link` is to be added to the context menu for all objects. The second line, however, determines that menu entry `Tag material link as exclusive` can also be selected for materials.

Note

Make sure that menu definition entries with a greater degree of specificity, in this example `om.popup.menu.children.OLINKS.MAT`, are executed after menu definition entries with a lower degree of specificity.

Suppressing Context Menu Entries

In SAP ECTR you can deactivate object-dependent context menu entries at the context level. This applies for instance in the `SAP Office Inbox` context with the functions `Cut` and `Delete`, because these functions cannot be applied in SAP Office.

The following menu definitions are used for this purpose:

```
om.popup.menu.<context>.<object_type>.excludeListForChildren = <Identifier of function to
be executed when menu entry is clicked>;<Identifier of function to be executed when menu
entry is clicked>
```

Placeholder	Description
`<context>`	Indicates a context in which you can change the context menu, e.g. `SAP Office Context`
`<object_type>`	Denotes an object in whose context menu functions are suppressed, e.g. a message in `SAP Office Context`
`<Identifier of function to be executed when menu entry is clicked>`	Denotes the functions to be suppressed

Example

```
om.popup.menu.ctx_so.SO_MSG.excludeListForChildren = fnc.clpbrd.cut;
                                                     fnc.clpbrd.delete
```

4.3 Display Response of SAP ECTR on Activating an Application

You can allow the response to be automatically determined using the connected screens.

When SAP ECTR is initialized, the SAP system determines how many screens are connected. The data is written to the environment variable %`PLM_DISPLAY_DEVICES%`.

4.4 The SAP GUI Window is Supposed to Appear in the Foreground

Program `ActivateSAPguiWindow` is called after transactions in the foreground. The program ensures that SAP help windows in the background are displayed without an entry in the task bar of the system and in the foreground. If an SAP GUI window is in the foreground, the call of this function has no effect. The function can be called even if SAP Windows is in the background.

To activate the function, you have to set the following preference variables:

- `plm.control.sapguiToFront=true`
- `plm.control.sapguiToFront.text.<LANG>.<MODULE_NAME>=window text`

4.5 Defining the Colors of the Activity Indicator

In SAP ECTR, the activity indicator is located at the bottom right and resembles a dice icon. It provides information about activities that are executed in the background. Define the color of the activity indicator in the `default.txt` file using the following variable:

```
system.color.activity.indicator.active
system.color.activity.indicator.inactive
```

4.6 Enhancements to Objects in the Application Structure

Within the application structure, SAP ECTR allows you to define a type-specific distinction with separate icons and functions.

Procedure

In the file `%PLM_INSTDIR%\customize\config\default.txt`, set the following preference variable:

`structure.view.<applType>.generic.object.type = XXX`

This preference variable defines the property key of the generic object (in the CAD system).

The value of this preference variable is used as part of the object type in the application structure.

Example

In the CAD system, the object has the following attribute:

`P3 = C`

In SAP Engineering Control Center, the object has the following attribute:

`structure.view.<applType>.generic.object.type = P3`

Application Structure

In the application structure, this produces a generic object with object type:

`cuapob_c`

This object can be configured separately because it is a new object.

Icons

`structure.view.generic.object.c.icon = <icon>`

Status Icons

```
plm.structure.node.cuapob_c.icon.1 = <icon>
plm.structure.node.cuapob_c.icon.2 = <icon>
```

Name

`structure.view.generic.object.c.name = <name>`

4.7 Fill description for parameters in the macro editor

With the following dictionary entry you can fill the individual parameter descriptions for the suggestions lists in the macro editor:

`plm.macro.suggestion[.<context>.]field.<field>.description`

Example

`plm.macro.suggestion.doc_fields.field.<field>.description`

This is possible for the following parameters:

Search: `doc_fields, mat_fields, ecn_fields,`

Filter: `doc_filter_fields, mat_filter_fields,` Classification (`cls`), ActivityLog (`activity_log_fields`)

Transform (`TRANSFORM_SET_WITH_PARAMETERS`): `where_used, onlinks, expand_structure`

4.8 Activate or deactivate line numbers in macro editor

Line numbers in the macro editor can be activated or deactivated using the following preference variable :

`plm.macro.editor.show.line.numbers`

The default value is `true`.

5 DATA SELECTION AND EDITING

5.1 Configuring Containers

The display and order of the containers follow the instructions below. By default, settings can be made for all document types or separately for some document types. Each container has a unique name. The following container types can currently be used to configure the containers on documents:

Container Type	Name
Part family template	`FAMILY_TEMPLATE`
Corresponding dependent documents	`MASTERMODEL`
Linked objects	`OLINKS`
Versions	`VERSIONS`
Originals	`ORIGINALS`
Usages	`WHEREUSED`
Components	`COMPS`
External references	`WLINKS`
Part family	`FAM_MEMBER`
Document part	`DOCPARTS`
Usage in classes	`CLASSES`
Export / import	`EXCHANGE`
Work items	`OBJ_WI`
Order of the templates	`SOURCEDOC`
Document alternatives	`DOCALTERNATIVES`
All associated engineering records in which the object is used as a change item	`AL_ERP_CP`

Container Priority Rules

```
1. plm.om.DOC.containers.fgroup.<FunctionGroup>  = <list of containers>
2. plm.om.DOC.containers.<document type> = <list of containers>
3. plm.om.DOC.containers = <list of containers>
```

Processing takes place from top to bottom and therefore from specific to unspecific. The last stage therefore offers a fall-back item for the container definition for all documents that could not be assigned earlier.

Displaying the Authorization Folder on the DIR

The following preference variable generates a sub-container for authorization folders in the where-used list:

```
plm.om.DOC.containers.WHEREUSED.<TYPE> = WHEREUSED;AUTH_RECORDS
```

Example

```
plm.om.DOC.containers =
DOCPARTS;OLINKS;VERSIONS;ORIGINALS;COMPS;WHEREUSED;CLASSES;EXCHANGE

plm.om.DOC.containers.UGM =
DOCPARTS;OLINKS;VERSIONS;ORIGINALS;COMPS;WHEREUSED;CLASSES;EXCHANGE; FAM_MEMBER;
WLINKS

plm.om.DOC.containers.fgroup.STN =
MASTERMODEL;DOCPARTS;OLINKS;VERSIONS;ORIGINALS;COMPS;WHEREUSED;CLASSES;EXCHANGE;
WLINKS

plm.om.DOC.containers.WHEREUSED.UGM = WHEREUSED;AUTH_RECORDS
```

Displaying Containers with Template Documents

Display the sequence of the templates from which the document was created:

```
sap.bapi.replace./DSCSAG/DOC_SEQUENCE_GET = <Z Module>
```

5.1.1 Setting up the Sequence of the Root Containers in the Desktop

The containers and their sequence can be determined using the listing of container key words in preference variable `plm.om.desktop.rootcontainers`.

Keyword	This keyword influences…
`con_fol`	`normal folders` (as defined in `plm.om.desktop.rootFolders`)
`con_lu`	`last used container`
`con_tmp`	`Object list container`
`con_smart`	`SmartList container`
`con_sess`	`Session directory`

Example

```
plm.om.desktop.rootcontainers = con_lu; con_tmp
```

The `Last used documents` container is displayed first and then a container with object lists.

Keyword `con_fol` is a placeholder for the folders that are defined in preference variable `plm.om.desktop.rootFolders`. By using this preference variable, you can define several folders at the top level. Container `con_tmp` cannot be hidden. If the container is not defined, it automatically appears as the last container in the desktop window.

5.1.2 Object Links Container on the Document or Material

This following preference variables make it possible to configure the `Object links` container on the document or material. For objects, you can display and change the classification in a tab page in the Object Browser.

Preference Variable	Description
`plm.om.DOC.containers... = ...;OLINKSONLY;...`	Configuration of the new container for display in SAP ECTR.
`plm.olink.fgroup.<TABLE_TYPE> = <function group>`	If you want to display different properties for material and document object links in the layout manager, you have to define the respective function group.

Example

```
plm.olink.fgroup.MARA = MARA
plm.olink.fgroup.DRAW = DRAW
```

Define icon for objects.

You can use `$(TYPE)` as a placeholder. The table type of the referenced object is mapped. You can use the symbol icon table to map the resulting icon name to existing icons.

```
plm.olink.icon.1 = <Iconname>
```

Example

```
plm.olink.icon.1 = tree.olink.obj_$(TYPE)
```

Deactivating the changing of objects in the Object Browser. The default value is set to `true`, i.e. activated:

```
plm.olink.editAllowed = true
```

5.1.3 Displaying Imported Documents in the Last Used Documents Container

After importing, documents can be displayed in the desktop in container `Last Used Documents`.

In the `default.txt` file, set the following preference variable:

```
plm.document.import.addToLastUsedFolder = true/false
```

5.1.4 Controlling Automatic Expansion of Containers

The following preference variable controls whether the container may be expanded automatically:

```
plm.browser.allow.autoexpand.[<ROOT_OBJECTS>].<CONTAINER_TYPE> = true(default) / false
```

For dynamic containers such as SmartContainers or GroupContainers, `<CONTAINERTYPE>` consists of two components:

- `ObjectType` of the container
- `ID` of a macro or a group

5.2 Configuring Previews

For documents, the preference variables `plm.preview.wsappl` (general) and `plm.preview.wsappl.aux.<document_type>` (for specific `document types`) provide information about which WSAPPL is used as the preview. This is a priority list and can therefore contain multiple WSAPPLs. For example: `plm.preview.wsappl.aux = EPG; EPJ; JPG; GIF; PIC`.

For previews of the characteristic value, the document must be named as follows: `<char_preview-characteristic value name>.<file extension>`

If you want to attach originals as a preview via drag&drop, you can use the following preference variables to define the assignment of file extension to WSAPPL when storing. Or, as always, you can use the DType settings.

```
plm.document.originals.extension.WsAppl.<extension>.<document_type> = <WSAPPL>
```

5.2.1 Folder

Please note that to set the previews, you need to set the preference variables from the previous section.

To make all previews the same size, the `picConvert.exe` program scales them to a fixed size.

Depending on the preview format, the file extension of the preview is changed. The preference variable `plm.preview.extension` specifies the valid file extensions. If the current file has an extension that is not yet included, the following preference variable is used: `plm.dnd.preview.exchange.format`.

The name of the file extension is changed to the extension defined there.

To enable SAP ECTR to recognize the previews, the following preference variables must be set in the `default.txt` file:

```
plm.document.originals.extension.WsAppl.jpg.<document type> = EPJ

plm.document.originals.extension.WsAppl.gif.<document type> = EPJ

plm.document.originals.extension.WsAppl.png.<document type> = EPJ
```

5.2.2 Tile View in the Object Browser

You can set the size of the previews for the folder contents in the thumbnails view of the Object Browser via the following preference variable:

Preference variable	Description
`obr.panel.FolderBrowser.btnSize = <width>x<height>`	The value of this variable defines the width and the height of the previews in pixels (e.g. 400x200). If the previews exist in different sizes, they are scaled.

5.2.3 Determining Previews for Materials in the Object Browser

In SAP ECTR, you can display previews for materials in the `Object Browser` by accessing previews in the linked documents.

Previews are stored as additional originals of documents.

The following preference variable in the `default.txt` file defines the list of document types that are used for the preview:

```
plm.om.MARA.preview.doctypepriolist=<DocType>;<DocType>
```

The following priority logic applies here:

1. The system first takes all documents from the object links that have a preview. Documents without a `"2OE Flag"` are ignored.
2. This is followed by the preference variable with the priority list from left to right, e.g.:

   `plm.om.MARA.preview.doctypepriolist=UMM; UGM; UGD`
3. If two or more documents are left over, the system checks whether one of them has a release status. The released document is preferred.
4. If both or neither have been released, the system checks the versions. If the versions are identical, the lexicographically higher version is preferred.
5. If the documents are different, no specific document is selected. In this case, the first document is taken.

Note

For performance reasons, only previews that are in the preview directory are read; there is no current status check. You can determine the preview images either by means of a refresh (F5) or by selecting the object by applying the logic explained above.

5.2.4 *Previews for Classes*

To display previews for a class, SAP ECTR must be able to distinguish the previews from the navigation images for the class document. This is achieved by defining a separate workstation application (e.g. `PIC`). The Basic Data tab of the class displays only the originals with this workstation application. For performance reasons, the maximum size of the previews is about 300x300 pixels.

You need to maintain the following preference variables in the `default.txt` file:

Preference variable	Description
`plm.clh.basicdata.preview.wsappls = <WSAPPL>`	Defines the workstation application of the previews, e.g. `PIC`.
`plm.clh.preview.allow.dnd = true/false`	Specifies whether or not a preview may be dragged and dropped onto a class.
`plm.document.originals.extension.WsAppl.<Extension>.AUX = <WSAPPL>`	Specifies the workstation application that receives a preview with a specific extension when dragging and dropping onto a class.
`obr.object.indication.ob.cls.previewSize = <value>`	Defines the size of the previews. Set a value for the variable to specify that the preview window takes half of the available space. For example, if the screen resolution is 1280x1024, the corresponding value would be 450.
`plm.charact.gui.show.characteristic.preview = true/false`	Specifies whether or not the characteristic preview is visible. If it is `<false>`, the preview for the characteristic is not visible in the value help.

5.2.5 *Lock Preview Settings*

You can prevent users from setting the size of previews in SAP ECTR themselves.

The following preference variable can be used to remove or make the relevant layout function available in the `Object Browser`.

Preference variable	Description
`obr.options.disablePreviewSizeField = true/false (default: false)`	This setting works for documents and materials as well as for previews in the Classification window.

5.2.6 *Previews for Characteristics and Values*

If you open the help for a characteristic by choosing F4, you can display a preview.

In the value help, information about the characteristic value is provided in the area on the right. This area is often collapsed at first and you have to click the small arrow on the right to expand it. If you do not select a characteristic value, it does not contain any information. If you select a characteristic value for which an image or long text is available, the window opens automatically and displays the content. If you select multiple characteristic values, the information for the first value you clicked remains in the area. If no image or long text is available for the value, nothing is displayed.

The following preference variable can be used to assign the previews for characteristics:

```
plm.charact.preview.wsappls = <e.g. PIC;JPG;EPJ>
```

The following preference variable can be used to assign the previews for classes:

```
plm.clh.basicdata.preview.wsappls = <e.g. PIC;JPG;EPJ>
```

When a characteristic is created in the system, it is linked to a document to which all the images for the characteristic are attached. For previews that belong to a characteristic or a characteristic value, the name of the attached images is changed.

`char_preview-<characteristic value>` indicates the previews for the characteristic values.

The name of the characteristic is used to name the images that describe the characteristic itself.

In the SAP ECTR options, the checkbox to always display preview windows for characteristic values must be selected on the Classification tab.

For previews of characteristics, the document must be named as follows:
`<preview name>.<file extension>`.

For previews of characteristic values, the document must be named as follows:
`<char_preview-characteristic value name>.<file extension>`.

The workstation application must be named as defined in the relevant preference variable
`plm.charact.preview.wsappls`.

5.3 Classification information

Setting the Display of the Default Class

You can ensure that the default class of a document is displayed on the `Classification` tab as well as on the Basic Data tab in the Object Browser. To do this, set the following preference variable:

Preference Variable	Description
`obr.classification.edit.additional = false`	The manipulation of the characteristic values of this default class is only possible on the classification tab.

Configuring Additional Properties in the Desktop and in the Object Browser

You can use preference variable `plm.om.doc.layout.extraProps` to configure additional properties for documents in the `default.txt` file:

```
# -----------------------------------------------------------------------
# 3.5.2 Document display / Containers

# Define which classification values are presented in the list of data
# objects that are selectable for the document tree and property display.
# This makes sense for standard class data only, that is available for all
# documents. Multiple definitions may be separated by ;

plm.om.doc.layout.extraProps = CHAR%MODELTYPE
```

Proceed as follows:

1. Enter the desired characteristics in the preference variable either as `CHAR%<characteristic name>` or simply as `<characteristic name>` in semicolon-separated form.
2. Ensure that the function for reading the classification of folder contents is activated.
3. To display a user-defined name in the Object Browser and layout manager, generate a corresponding dictionary key in customer dictionary `customer.txt`. Here, the spelling of the dictionary key depends on the spelling of the entry in the preference variable.

   Dictionary key for additional characteristics using the notation `CHAR%<characteristic name>`:

   ```
   CHAR%MODELTYPE = Model type
   ```

   Dictionary key for additional characteristics using the notation `<characteristic name>`:

   ```
   @MODELTYPE = Model type
   ```

   Note

   The dictionary key must be spelled using upper case.
4. Make analogous settings for materials as well. The preference variable used for this is: `plm.om.mat.layout.extraProps`

   The variable is defined as follows in the `default.txt` file:

   ```
   # Define which classification values are presented in the list of data
   # objects that are selectable for the desktop tree and property display.
   # This makes sense for standard class data only, that is available for all
   # documents. Multiple definitions may be separated by ;
   ```

   ```
   plm.om.mat.layout.extraProps = <CHARACTERISTIC_NAME>
   (prefix CHAR% may be omitted)
   ```

Filtering User-Specific Classification Information

There are often classes of materials and documents that are not relevant to all users.

Using preference variables, you can control which class types, classes and class groups are read and displayed.

List of Preference Variables for the Display Mode of the Object Browser

Preference Variable	Description
`obr.classification.clstypes.hide`	List of class types that are not read: `<Class type>;<Class type>;…`
`obr.classification.cls.hide`	List of pairs of class names and class types that are not read: `<Class name>:<Class type>;…`
`obr.classification.clsgroups.hide`	List of class groups that are not read: `<Class group>;<Class group>;…`

Example

```
obr.classification.clstypes.hide  = Z01;Z02
obr.classification.cls.hide       = ZUS_DATEN:017;HIST_DATEN:017
obr.classification.clsgroups.hide = CLS_GRP1;CLS_GRP2
```

These preference variables apply to the classification data in the display mode of the Object Browser. When you switch to change mode in the Object Browser, the classification data of the document is read and displayed for characteristic value assignment.

List of Preference Variables for the Change Mode of the Object Browser

If certain classification data is not intended to be read and displayed in change mode in the Object Browser, you can apply the DType-dependent preference variables listed.

⚠ **Note**

You cannot assign characteristic values for classes which are filtered out, nor their classifying attributes.

Preference Variable	Description
`obr.classification.clstypes.hide.<dtype-id>`	List of class types that are not read for documents with the DType `<dtype-id>`.
`obr.classification.cls.hide.<dtype-id>`	List of class names and class type pairs that are not read for documents with the DType `<dtype-id>`.
`obr.classification.clsgroups.hide.<dtype-id>`	List of class groups which are not read for documents with the DType `<dtype-id>`.

ℹ **Note**

In this method, the default class of a document cannot be filtered out, because the default class represents an enhancement of the master data, which has a special status.

5.4 Filtering Objects Displayed in Windows

If too many objects are displayed in certain windows of SAP ECTR, you can easily lose the overview.

Object filters allow you to apply user-specific settings to objects in windows.

You can filter objects based on their properties or status, whereby only homogeneous groups of objects can be filtered. In the window in which you want to filter, the structure must be fully loaded. If the structure is loaded level by level, filtering does not work.

⚠**Warning**

Do not try to filter objects in the desktop window, in the active list, or in the object browser. Filtering is not possible in these windows because different objects are displayed here. Filtering in these windows can lead to program errors.

5.4.1 Defining Filters for Windows

Define filters for windows in the `default.txt` file.

Activating Filters for a Window

You first activate filters for a certain window by setting the following variable to `TRUE`:

```
obr.tree.filter.active.<ctx-key>
```

`<ctx-key>` is the context key of the relevant window. You can currently include the following windows:

<context>	Description
ctx_ass	Assembly Window
ctx_asv	Application Structure
ctx_pmb	Asset Browser

Example

```
obr.tree.filter.active.ctx_asv = TRUE
```

Creating a Namespace

To enable the program to find filter definitions, you define a namespace for the filters of the relevant window. To do this, use the following preference variable:

```
obr.tree.<ctx_key>.filter.namespace = <NameNamespace>.
```

Make sure that the expression <ctx_key> matches your previous entry.

You name the namespace in <NameNamespace>.

Example

```
obr.tree.ctx_asv.filter.namespace = MyNamespace
```

Naming Filters

Name the filter that you want to define.

```
<NameNamespace>.tool.<Mode>.<Indexnumber>=<Filtername>
```

With <NameNamespace>, specify the previously defined namespace.

The expression <Mode> defines the mode in which the filter operates. You can display those objects addressed by the filter as the only objects in a list. In this case, specify the parameter filter as the <Mode>.

If the objects that you address with the filter are to be hidden from a list, specify the parameter skip.

You number your filter with <Indexnumber>. Here you enter a number greater than zero and that has not already been used to number a filter in this namespace.

You name your filter with <Filtername>.

Example

```
MyNamespace.tool.skip.1 = MyFilter
```

Creating a Filter Definition

Now you define the filter itself.

A filter compares an attribute or a status of a specific object and returns the following:

Attribute	Description
`true`	The object attribute or the status fulfills the filter operation.
`false`	The object attribute or the status does not fulfill the filter operation.
`null`	The object does not support the named attribute or the comparison returned an error.

Example

```
obr.filter.MyFilter = DOC;PRP;$OBJECT_TYPE;STRING;MAT;==
```

In this example, the expression `MyFilter` returns `true` for an object of type "MAT".

The filter definition consists of the following components:

```
obr.filter.<Filtername> = <OTYPE>;[PRP | FLG];<PRP | FLG_NAME>;<DATA_TYPE>;
<VALUE>;<OPERATION>;[INVERT]
```

Attribute	Description	Values
`<Filtername>`		Name of the filter that is being defined.
`<OTYPE>`	Defines the object type of the object to be filtered. This entry is mandatory.	ID of the relevant object type, such as `DOC` for documents, `MAT` for material, `ECN` for change number, `ECR` for change request, `WORK_ITEM` for work item, `FLOC` for functional location, `EQUI` for equipment, and `PROJ` for project. `DOCPOS` for documents in the assembly window `MAT_BOM_COMPONENT` for materials in the BOM container or BOM window
`[PRP	FLG]`	Defines whether objects are filtered by attribute or status flag. This entry is optional.	Either `PRP` for attribute or `FLG` for status flag
`<PRP	FLG_NAME>`	Defines the name of the attribute or status flag. If a specific object was not previously defined, you can filter here by certain object types. This entry is mandatory.	Name of the attribute or status. Alternative: `$OBJECT-TYPE`

`<DATA_TYPE>`	Defines the data type of the attribute or status flag. This entry is mandatory.	For example:
		`STRING`
		`DATE`
		`INTEGER`

Attribute	Description	Values
`<VALUE>`	Defines the value with which the attributes or status flag of the object can be compared. This entry is mandatory.	Value of an attribute or a status flag. `Date` for comparing with the current date.

`<OPERATION>`	Defines the regular expression for comparing the values. This entry is mandatory.	For example:
		`>`
		`>=`
		`!=`

`[INVERT]`	Defines whether the results are inverted. This entry is optional.	`INVERT`

Example

```
#Attribute "VERSION" is "4"
```

```
obr.filter.MyFilter2 = DOC;PRP;4;STRING;02;>=
```

If the object is a document and its version number is larger or equal to 2, `MyFilter2` returns `true`.

```
obr.filter.MyFilter3  = DOC;FLG;ICON2;STRING;.*green;RE
```

If the object is a document and its status flag `ICON2` ends with `green`, `MyFilter3` returns `true`. This means that the document has a green status.

The material object and document object change their object type. In the `Bill of Material` container or `Bill of Material` window, the material object type `MAT` becomes `MAT_BOM_COMPONENT`.

In the `Assembly` window, `DOC` becomes `DOCPOS`.

The filters must always catch both object types, for example:

Example

```
obr.filter.Filter_1 = DOC;[PRP | FLG];<PRP |
FLG_NAME>;<DATA_TYPE>;<VALUE>;<OPERATION>;[INVERT]

obr.filter.Filter_2 = DOCPOS;[PRP | FLG];<PRP |
FLG_NAME>;<DATA_TYPE>;<VALUE>;<OPERATION>;[INVERT]

obr.filter.OMF= (Filter_1;Filter_2)

obr.filter.OMF.combination.mode = ANY
```

Example

```
obr.filter.Filter_1 = MAT;[PRP | FLG];<PRP |
FLG_NAME>;<DATA_TYPE>;<VALUE>;<OPERATION>;[INVERT]

obr.filter.Filter_2 = MAT_BOM_COMPONENT;[PRP | FLG];<PRP |
FLG_NAME>;<DATA_TYPE>;<VALUE>;<OPERATION>;[INVERT]

obr.filter.OMF= (Filter_1;Filter_2)

obr.filter.OMF.combination.mode = ANY
```

Linking Filters

You can link together multiple filters using set operations (OR and AND).

First define two filters.

The filters can be linked using the following lines:

```
obr.filter.<FilternameX> = (<Filtername1>;<Filtername2>;...)
obr.filter.<FilternameX>.mode = [ANY|ALL];
```

Mode `ANY` stands for the non-exclusive disjunction [OR]; `ALL` stands for the conjunction [AND].

Example

```
#Attribute "USER" is "7"
obr.filter.MyFilter4 = DOC;PRP;7;STRING;GROSS;==
obr.filter.MyFilter5 = DOC;PRP;7;STRING;B.*;RE;INVERT
obr.filter.MyFilter_Link = (MyFilter4;MyFilter5)
obr.filter. MyFilter_Link.mode = ANY
```

If `MyFilter4` (`true` if user name is "GROSS") or `MyFilter3` (`true` if the user name begins with "B") returns `true`, then `MyFilter_Link` returns `true`.

5.4.2 *Defining Filters for SmartContainers*

Object filters enable SmartContainers to be displayed or not depending on the status of an object.

There are two ways to use SmartContainers:

Include-Filter

```
plm.smart.container.<CONTAINER_NAME>.includeFilter = <FILTER_NAME>
```

When an include-filter is used, SmartContainers are visible only if the filter returns `true`.

Exclude-Filter

```
plm.smart.container.<CONTAINER_NAME>.excludeFilter = <FILTER_NAME>
```

When an exclude-filter is used, SmartContainers are visible only if the filter returns `not true` (`false` or `null`).

Linking an Exclude-Filter and Include-Filter

If both an include-filter and exclude-filter are defined, both filter results are evaluated. The SmartContainer is displayed only if both filters allow this.

Procedure

- First create in the `default.txt` file a SmartContainer with reference to a macro. The exact procedure is explained in the `Operations Guide`.

  The following examples demonstrate how a SmartContainer is defined for objects of type DOC (documents). This SmartContainer contains the assigned change number of a document. If a document does not have a change number, the SmartContainer is not displayed.

Example

```
plm.om.DOC.containers =
OLINKS;VERSIONS;ORIGINALS;WHEREUSED;CLASSES;SMART_CONT(ASGD_ECN)
...
plm.smart.container.ASGD_ECN.macro = assigned_ecn.txt
plm.smart.container.ASGD_ECN.icon = {0}/sap/change_number
```

In the example above, you are adding a SmartContainer to the document object type. At the same time, you are referencing an icon for the SmartContainer and assigning the SmartContainer the macro `assigned_ecn.txt`, which is stored in the installation directory.

```
keylist = KEYLIST_FROM_REFOBJ()
ecn_prp = GET_PROPERTY("12", keylist)
if(!!ecn_prp && "" != ecn_prp)
{
  var e = [ecn_prp[0] + "@AENR"]
  WRITE_RESULTLIST(e)
}
```

- To name the SmartContainer, create a key for the SmartContainer in the dictionary `customer.txt`.

  ```
  plm.smart.container.dic.<Name_Smartcontainer> = <Text>
  ```

  Example

  ```
  plm.smart.container.dic.ASGD_ECN = Assigned Change Number
  ```

- In the `default.txt` file, add the filter that hides the SmartContainer if documents do not have a change number.

  ```
  plm.smart.container.<CONTAINER_NAME>.includeFilter = <FILTER_NAME>
  ```

  Example

  ```
  plm.smart.container.ASGD_ECN.excludeFilter = ecn_asgd_filter
  ```

- Define your filter in the default.txt file. For details, see section 5.4.1 Defining Filters for Windows.

  ```
  obr.filter.<Filtername>= <OTYPE>;[PRP | FLG];<PRP | FLG_NAME>; <DATA_TYPE>;
  <VALUE>;<OPERATION>;[INVERT]
  ```

  Example

  ```
  obr.filter.ecn_asgd_filter = DOC;PRP;12;STRING;;EQ
  ```

5.5 Long Texts in the Object Browser

longtext_config.xml

Path: `%PLM_INSTDIR%\customize\config\`

Add this file as needed.

You can divide the long text in the Object Browser into various groups and according to language.

Attribute	Description	Values

`title`	Defines the title of the tab in the Object Browser	Example:
		`${SYSTEM.T.DOC_DESCR}`
		`Change texts`

Attribute	Description	Values
`language`	Optional: Specifies for which logon language the tabs are displayed in the Object Browser.	Language ID, Example: `DE, EN`
`type`	Type of input field	Example: `TEXT, AREA`
`length`	Defines the permitted length of the field	Example: `40`
`rows`	Defines the number of rows of the field	Example: `1, 4`
`editable`	Specifies whether the field can be edited	Example: `yes, no`
`backgroundColor`	Defines the background color of the field	RGB color values Example: `255;114;135`

Example

```
<sections>
  <section title="${SYSTEM.T.DOC_DESCR}">
    <components>
      <component type="TEXT"
                 length="40"
                 rows = "1"
                 editable = "yes"
                 backgroundColor = "255;114;135"/>
    </components>
  </section>
  <section title="Change texts" language="EN">
    <components>
      <component type="AREA"
                 rows = "4"
                 editable = "yes"
                 backgroundColor = "20;220;255"/>
    </components>
  </section>
</sections>
```

In the file `%PLM_INSTDIR%\customize\config\default.txt`, set the following preference variables:

Preference Variable	Description
`plm.ob.doc.useCustomizableLongtext`	Display configurable long text dialog (default: `false`).
`plm.ob.doc.customizeableLongtext.useScrollbars = true/false`	Show/hide scroll bars for area with long text entry fields (default: `false`).

Preference Variable	Description
`plm.ob.doc.customizeableLongtext.config.file.<DTypeId> = <file name>`	File name of the XML long text configuration file without path.

5.6 Configuring Tabs in the Object Browser

Customer-specific and add-on-specific tabs can be configured for the object browser in a specific sequence. The following preference variable is used for this purpose:

```
plm.ob.preferred.tab.order.<tableType>[.<functionGroup>]
```

Example

Preference variable:

```
plm.ob.preferred.tab.order.DRAW = BAS;CUS;CLS;REF;ORI;TXT
```

For each object, you can specify which tabs are displayed in the object browser. You can use inclusion or exclusion filters to show or hide the tabs.

```
plm.ob.tab.<TableType>.[.<FunctionGroup>.]<TabKey>.includeFilter
plm.ob.tab.<TableType>.[.<FunctionGroup>.]<TabKey>.excludeFilter
```

When the inclusion filter is used, the tab is included in the objects captured by the filter.

When the exclusion filter is used, the tab is not included in the objects captured by the filter.

If both filters are defined, the inclusion filter is evaluated first.

If both filters are defined, the inclusion filter is evaluated first, followed by the exclusion filter.

Example

The tab `myTab` is displayed only for documents with the function group `NXM`:

```
plm.ob.tab.DRAW.myTab.excludeFilter = myDrawTabFilter
plm.ob.tab.DRAW.NXM.myTab.excludeFilter = myNXMTabFilter
```

Remove classification tab for documents with the function group `MACRO`:

```
plm.ob.tab.DRAW.MACRO.CLS.includeFilter = false
```

5.7 Object Lists and Active List

5.7.1 Saving Contents of Object Lists

When SAP ECTR is closed, all folders are stored with their content in a user-specific XML file. The XML file is located in the following directory:

```
$(system.start.datadir)\ user\$(user)\ tfolder\ (Environment.getUserDataDir("tfolder"))
```

Whether an object list remains available after SAP ECTR has been closed can be controlled by choosing `Enable/Disable list persistence` from the context menu. If this context menu function is activated for an object list, this object list receives a folder flag. The object list remains available after SAP ECTR has been closed and reopened.

Any objects that no longer exist in the SAP system and objects for which the user has lost the authorization are ignored during loading. The objects are then output in trace level 1.

The action cannot be applied to the default list. The following message is displayed in the status bar: `The default list cannot be persistent.`

5.7.2 Configuring the Process "Add to Active List"

You can configure the process `Add to Active List` from SAP transaction `CV04N`. To do so, proceed as follows:

1. Call transaction `SPRO` (SAP Reference IMG).
2. Choose the SAP Reference IMG.
3. Navigate to `Cross-Application Components - Document Management - General Data - Enter Editing Process for Documents`.
4. Click the clock icon.
5. The dialog for changing DMS processes is displayed.
6. Choose `New Entries` and define the new entry as follows:
   - `Process:` ADDTOALL
   - `Process Type:` Call from CV04
   - `Description:` Add to the active list of SAP ECTR
   - `Sequence:` 1
   - `Function Module:` /DSCSAG/ADDTOOM
7. Choose `Save,` to save your new DMS process and display it in the overview.
8. In transaction `CV04N`, check whether the new DMS process is available.

Troubleshooting

If individual users do not see a `Processes` button or if the process list does not include the desired process, no profile has been defined for these users. Proceed as follows:

1. First, check which profile is affected.
2. Add the process to the profile.
3. Select the profile and click the `Processes` folder.
4. Create a new entry.
5. Save the changes.

5.7.3 Adding Additional Objects to the Active List

When you insert certain objects into the active list, you can directly add additional assigned objects. This is controlled using the following preference variables:

Preference Variable	Description
`plm.activelist.material.addDocuments  = true`	For materials, add the assigned documents to the active list.
`plm.activelist.document.addNonMasters = true`	For documents, add the corresponding dependent documents to the active list.
`plm.activelist.orderBomItem.addMaterial = true`	For sales order items, add the assigned material to the active list.
`plm.ecm.relobj.hideNoPermission.doc = true`	Display only documents with authorizations in the `Assigned Objects` container in ECM.

5.8 Defining an Assignable Description for User Folder <username>_NS

If you maintain the following dictionary key in the `customer.txt` file, the value of the key is used as the description when the user folder is created.

```
plm.fld.create.descr.<folder name> = <description>
```

Here, `<folder name>` is inserted in the notation that was defined in preference variable `plm.om.desktop.rootFolders`.

Example

Preference variable:

```
plm.om.desktop.rootFolders  = <USERNAME>_NS;PROJECTS
```

Dictionary key:

```
plm.fld.create.descr.<USERNAME>_NS = storage
```

This dictionary key has no effect on existing user folders.

5.9 Using Object Filters for SmartContainers

In some cases, it makes sense to hide SmartContainers under objects or to deactivate functions of objects.

5.9.1 Defining Object Filters

An object filter compares a characteristic or the status of a transferred object with the existing value and returns the following values:

Value	Description
`true`	The object value fulfills the filter operation.
`false`	The object value does not fulfill the filter operation.
`null`	The object does not support the defined values.

A filter consists of the following parts:

```
obr.filter.<FILTER_NAME> = <OTYPE>;[PRP | FLG];<PRP | FLG_NAME>;<DATA_TYPE>;
<VALUE>;<OPERATION>;[INVERT]
```

Attribute	Description	Values
`<FILTER_NAME>`	The integration of the filter in SAP ECTR is based on the filter name.	Name of the filter.
`<OTYPE>`	Defines the object type of the filtered object.	`*` for every object
		`DOC` for documents
		`MAT` for materials
		`FLOC` for functional location
`[PRP	FLG]`	Defines whether the filter contains a status or a characteristic	`PRP` for property

Attribute	Description	Values
`<PRP	FLG_NAME>`	The name of the status or the property. If no specific object type is defined, the `$OBJECT_TYPE` can be used to filter specific object types.	Name of the property or the status. `$OBJECT_TYPE`

`<DATA_TYPE>`	Data type of the property or the status.	`STRING`
		`DATE`
		`INTEGER`

`<VALUE>`	The value with which the property or the status is compared. It is possible to use predefined keywords.	`value` of the property or the status. DATE for comparison with the current date. ASK

`<OPERATION>`	Either a logical comparator or a comparison using the generic expression.	`">", "G"`
		Greater than
		`>=", "=>", "GE"`
		Greater than or equal to
		`"<", "L"`
		Less than
		`"<=", "=<", "LE"`
		Less than or equal to
		`"<>", "!=", "NE"`
		Not equal to
		`"~", "RE"`
		Roughly equal to

`[INVERT]`	Optional parameter that inverts the results	`INVERT`

Example

```
obr.filter.my_filter_1 = *;PRP;$OBJECT_TYPE;STRING;MAT;==
```

`my_filter_1` returns true if the object type of the property of an object is "`MAT`".

5.9.2 Object Filters for SmartContainers and Functions

Configuring an object filter enables you to

- Hide or display a SmartContainer under a referenced object, depending on the properties of the referenced object.
- Deactivate or activate a function for an object, depending on the properties of the referenced object.

Object Filters for SmartContainers

There are two options for using object filters with SmartContainers.

- Include filter

```
plm.smart.container.<CONTAINER_NAME>.includeFilter = <FILTER_NAME>
```

Here, the SmartContainer is visible only if the filter conditions are fulfilled.

- Exclude filter

```
plm.smart.container.<CONTAINER_NAME>.excludeFilter = <FILTER_NAME>
```

Here, the SmartContainer is visible only if the filter conditions are not fulfilled.

Example

```
# Filter to hide SmartContainer at documents that have no ECN assigned
plm.smart.container.ASGND_ECN.excludeFilter = ecn_asgnd_filter
obr.filter.ecn_asgnd_filter = DOC;PRP;CHANGE_NUMBER;STRING;;EQ
```

The SmartContainer for assigned change numbers (`ASGND_ECN`) is hidden if there is no value for the `CHANGE_NUMBER` property.

Object Filters for Functions

Object filters can also be used in direct combination with functions:

```
obr.filter.<FUNCTION_NAME> = <FILTER_VALUES>
```

Example

```
# Filter to deactivate the "Open Original for Edit" function when a document has status "Final Release" (FR)
obr.filter.fnc.doc.open.fast.edit = *;PRP;STATUS;STRING;FR;==
```

The `Open for Edit` function is deactivated if the value of the `Status` property is “`FR`”.

6 USABILITY

6.1 Setting Keyboard Shortcuts

Keyboard shortcuts are defined in the `menu.guidef` file.

Example

Copy:	`fnc.clpbrd.copy.keystroke`	`= control C`
Cut:	`fnc.clpbrd.cut.keystroke`	`= control X`
Insert:	`fnc.clpbrd.paste.keystroke`	`= control V`
Delete:	`fnc.clpbrd.delete.keystroke`	`= DELETE`

6.2 Output of Tooltips Dependent on the Object and the Position

If you want to change the global display time for tool tip texts, set the following preference variable to the desired value:

```
system.toolTipDuration = 10
```

If you do not enter a specific duration, the display remains visible for 60 seconds. If you move the mouse before then, the display stops sooner.

6.3 Functions when Double-Clicking an Object

An object type has an assigned default function that can be executed by double-clicking it. This default function is defined in the file `menu.guidef` according to the following schema:

```
om.popup.menu.<object_type>.function = <Identifier of the function to be executed when
object of the specified type is double-clicked>
```

Example

Using documents as the example, the assignment of these functions to an object is as follows:

```
om.popup.menu.DOC.function = fnc.doc.open.fast.view
```

This default function opens a document for viewing when you double-click it.

In SAP ECTR, when you select a folder from the `Desktop`, the objects in this folder are listed in the `Object Browser`. Double-clicking one of the listed objects opens the detailed view of the relevant object in the `Object Browser`. Alternatively, you can specify that double-clicking executes the default function assigned to the object type, e.g. opening in the associated application. To do this, use the following preference variable in `default.txt`:

```
plm.om.tree.navigation.on.double.click = false (default value: true)
```

Note

If the preference variable is set to `true`, open the relevant object in the detailed view in the Object Browser using `shift`-click.

Note

If you define a menu, you can assign shortcuts or double-clicks to the functions from the menu. After the shortcut or double-click, the relevant function is searched for within the menu.

6.4 Making Phone Numbers Clickable in the User Display

If the preference variable has been defined, the phone number displayed in the user data turns into a clickable link. This link calls up the executable file from the preference variable and transfers the displayed phone number as a parameter. This calls a script that starts phone software with this number.

Enter the corresponding preference variable in `default.txt`:

`plm.control.phoneDataScript = <path>` - script for calling

6.5 Marking SAP Mails as Read

SAP mails can be marked as `read` either manually via the context menu or automatically after a certain time.

In the `default.txt` file, set the following preference variable:

```
plm.sapoffice.msg.read.delay = <time in milliseconds>
```

If the preference variable is set, the message is marked as read after the specified time.

If the variable is not set, you can only manually mark the message as read.

6.6 Setting a Statistics Filter

You use the following preference variables to specify what is logged and what is not logged. SAP ECTR measures how long individual users work with SAP ECTR.

To do this, use the following preference variables:

```
plm.collect.stats.filter.name.include=regexp1;regexp2
plm.collect.stats.filter.name.exclude=regexp3
plm.collect.stats.filter.group.include=regexp4
plm.collect.stats.filter.group.exclude=regexp5;regexp6
```

7 DOCUMENTS

7.1 DTypes

DType settings control the response of SAP ECTR in the following functions:

- Creating new documents
- Properties of the documents

DType settings are defined in `Dtype.xml` files. As well as a central `DType.xml` in `%PLM_INSTDIR%\customize\config` , each application has its own `DType.xml` in %`PLM_INSTDIR%\applications\<<Name of application>>\customize\config`. Settings in an application-specific `DType.xml` overwrite settings in the central `DType.xml`.

The structure of `DType.xml` files is defined in the definition file `dtype_4110.dtd`. The file is located in directory `%PLM_INSTDIR%\basis\aux-files`.

DType settings are administered centrally for all SAP ECTR users through the SAP system. To ensure that changes to `DType.xml` files can be applied globally, DType settings must be loaded into the SAP system. As an SAP Engineering Control Center administrator you have access to the function `Transfer SAP Engineering Control Center Settings to SAP System`. You can find the function in the SAP Engineering Control Center, when you are in active administration mode, in the menu tab `Administration` under `SAP Engineering Control Center System Setup`. On start, the local DType settings are compared to the global settings in the SAP system. If the settings differ, the user receives a message when the SAP ECTR is started.

The function transfers those parts of the DType definitions which the ABAP part requires to the SAP system. If a DType definition differs, the function is called up. Documents affected by this change are updated by clicking on the context menu `Update`.

When DTypes are uploaded to the SAP system, the file is time-stamped at the point of uploading. When SAP ECTR is started, the function determines which is the most recent of the imported DType files and checks them against the time stamp. If the SAP system time stamp is older, this means a DType file has been altered in the meantime but not uploaded yet. In this case an error message appears on start.

7.2 Changing the DTypes of Existing Documents

SAP ECTR offers a function for changing the document type (DType) of existing documents and their dependent documents.

To change the document type, proceed as follows:

- In the desktop of SAP Engineering Control Center, select the corresponding documents.
- Then select the `Change document type` function.

  If administration mode is activated, you will find the function in the `Administration` tab under `Utilities`

7.2.1 Changing DTypes of existing documents (standard users)

Standard users can change DTypes of documents into other specified DTypes. This requires original and target DTypes to be specified in the `default.txt` as follows:

Preference Variable	**Description**
`plm.document.changeDtype.possibleDtypes.for.<Original-DType> = <Target-DType>;<Target-DType>;...`	Entry of the `<Target DTypes>` to which you want to be permitted to change from the `<Original DType>`

Preference Variable	Description
plm.document.changeDtype.allowed.status = AC	Entry of the status in which the DType change is permitted
plm.document.changeDtype.setDirty <DTypID>	true: A dirty flag is set for the newly generated document that corresponds to the DType-ID false: A dirty flag is not set for the newly generated document that corresponds to the DType-ID
plm.document.changeDtype.setDirty.default	true: A dirty flag is set for the newly generated document false: A dirty flag is not set for the newly generated document

7.2.2 *Changing DTypes of existing documents (administrator)*

As an administrator, you can change the DType of all existing documents.

The following preference variables of default.txt configure the Change document type function:

- plm.document.changeDtype.setDirty.<DTypID>
- plm.document.changeDtype.setDirty.default

Both preference variables are used together. The variables define whether a dirty flag is set. The first preference variable is more specific and therefore has precedence.

7.2.3 *Changing parts to standard parts*

With SAP ECTR, you can change existing parts to standard parts by changing the DType.

To execute the function, proceed as follows:

- Select the document you want to change into a standard part.
- Choose Change Part to Standard Part in the Administration menu.
- In the dialog, select the new DType and click OK.
- To count as a standard part, a DType must be assigned the application_role="N" in DType.xml. In addition, the change from an original DType to the target DType must be permitted. The setting is made using the following preference variables in default.txt:

Preference Variable	Description
plm.document.changeDtype.possibleDtypes.for.<Original DType> = <Target DType>;<Target DType>;...	Entry of the <Target DTypes> to which you want to be permitted to change from the <Original DType>
plm.document.changeDtype.allowed.status = AC	Entry of the status in which the DType change is permitted

Note

If the selected source and the target are both standard parts, or if both are non-standard parts, there is no change.

If the selected source is a standard part, and the target is non-standard part, the source becomes a non-standard part.

7.3 Configuring Additional Input Fields for the Document Creation Dialog

SAP ECTR allows the configuration of input fields which are available for data entry in the document creation dialog.

Creating a Pseudo Class

Entry fields in the SAP ECTR document creation dialog are configured using the SAP classification system. Execute transaction `CL01` in the SAP system and establish a pseudo class for class type `017` with an arbitrary name and description. Characteristics are later assigned to this pseudo-class. Each characteristic stands for an additional entry field in the document creation dialog.

Creating Classification Characteristics in this Class

These characteristics define which data fields are displayed in the document creation dialog.

The characteristic names are defined according to the following schema:

- `<any string>_<Document data field name>`

  or

- `<any string>_<Document data field name>_<language code (LANGUAGE_ISO)>`

For each of the characteristics, an input field is displayed in the dialog and the entered value is assigned to the corresponding data field of the document. The properties and value tables of the characteristics determine the input options and input helps in the dialog. These input options and input helps are then also available in the Object Browser when you change the basic data.

Example

`Class: DOC_ADDIT_FIELDS`

Characteristics:

DOC_DESCRIPTION_DE	German description
DOC_DESCRIPTION_EN	English description
DOC_AUTHGRP	Authorization group
DOC_LABORATORY	Laboratory/office
DOC_CHANGE_NUMBER	Change number
DOC_CM_RELEVANCE	CM relevant

Assign the class to a document type using the DType definition.

This is done using the XML tag `<program_options>` in the DType definition, e.g.:

```
<program_options  show_for_create="yes"
                  has_primary_application="yes"
                  dirty_flag="yes"
                  attempt_status_change_on_checkin="yes"
                  function_group="STM"
                  customer_data_section=""
                  document_data_class="DOC_ADDIT_FIELDS"
  <new_version    status_list="AC;FR"
                  ignore_wsappl_list="ZIP"
                  ignore_nm_dtype_list=""/>
  <copy_as        ignore_wsappl_list="ZIP"
                  use_initial_version="yes"/>
</program_options>
```

In SAP ECTR, you can make different input fields available in the dialog for documents with different DTypes.

Prefilling Input Fields in the Document Creation Dialog

You can prefill the input fields in the document creation dialog via SAP user parameters. Here, you can use the value of the SAP user parameter directly or reference an environment variable whose value is used for prefilling the input field.

Example of the reference to an environment variable

Definition of environment variable:

```
set PLM_LABOR = XYZ
```

SAP user parameter:

```
/DSCSAG/PLM_LABOR    @PLM_LABOR@
```

To prefill the input field, SAP Engineering Control Center uses the value of environment variable `PLM_LABOR` to which user parameter `/DSCSAG/LABOR` refers.

7.4 Applying Values from the document

The value of the document takes priority over the value from the user parameter. If a user parameter has been defined for a characteristic and the document has the same characteristic at the same time, the value of the document takes priority. If no value is available, the default values are set.

Preference Variables

```
sap.user.parameterMapping.<sap_user_parameter_name> = <parameter_name_to_use_in_ectr>
```

General mapping of SAP user parameters for SAP Engineering Control Center. This mapping can be used to give the user parameter a new name, which is then used in SAP ECTR for checking whether names match. This avoids the renaming of existing SAP user parameters. The parameter must be maintained in the `saplogon.txt` or `R3_<System_ID>.txt`.

Example

User parameter: `Z_DEFAULT_CAD_PRODTYPE`

Characteristic: `CAD_PRODTYPE`

```
sap.user.parameterMapping.Z_DEFAULT_CAD_PRODTYPE = /DSC_CHAR/CAD_PRODTYPE
```

```
plm.doc.createGui.userParameterCharacteristicPrefix = <prefix>
```

This parameter defines a prefix for determining SAP user parameters for setting default characteristics when documents are created. Maintain the parameter in the `default.txt`.

Example

```
plm.doc.createGui.userParameterCharacteristicPrefix = /DSC_CHAR/
```

7.5 Status

7.5.1 Hiding a Status

You can exclude certain statuses from the user interaction. To do this, set the following preference variables:

Preference Variable	Description
`plm.document.statuschange.<document type>.notAllowed.statusList = <status>;<status>;...`	Use semicolons to separate the list of statuses to be hidden.

Preference Variable	Description
`plm.document.statuschange.<document type>.notAllowed.statusList.<current status> = <status>;<status>;...`	Use semicolons to separate the list of successor statuses to be hidden.

7.5.2 *Setting a Status Without a Predecessor*

In the SAP system, you can switch from anywhere to a non-primary status without a predecessor. In the GUI, this status is not available for selection. SAP GUI offers the status as the target status for any status. In SAP ECTR, you can use the following preference variable to activate this response:

```
plm.document.statuschange.allowStatusWithoutPredecessors = true
```

7.5.3 *Filtering Status Transitions Defined in the SAP System*

All status transitions are defined in the SAP system. The display of the possible successor statuses in the SAP system can be filtered using a BAdI. In the SAP GUI, only allowed status changes are displayed.

In the Object Browser, you can directly change the status in the table. To do this, go to the `Basic Data` tab page and use the pencil icon to switch to change mode. Execute the status change in the table. The disadvantage of this approach is that the filter for the allowed status changes is not processed.

7.5.4 *Use Digital Signature for Status Change*

If the configuration in the back end includes a digital signature for status change, the following preference variable must be set. The dialog for the digital signature then appears:

Preference Variable	Description
`sap.bapi.useGuiTransaction`	/DSCSAG/DOC_CHANGE_MULTI3

7.5.5 *Filtering Target Status*

When the status is changed, only certain target statuses can be displayed and certain target statuses hidden.

If you wish to hide certain statuses when the status is changed, enter the value `Blacklist` in the `dtype.xml`. Once you have done this, the relevant original status is listed followed by the target status to be hidden.

```
<program_options>
  <status_transition>
     <status_transition_restriction type="blacklist" from="SW" to="IA"/>
  </status_transition>
  </program_options>
```

If you wish to display certain statuses when the status is changed, enter the value `Blacklist` in the `dtype.xml`. Once you have done this, the relevant original status is listed followed by the target status to be displayed.

```
<program_options>
  <status_transition>
     <status_transition_restriction type="whitelist" from="IW" to="AC;IA;FR"/>
  </status_transition>
  </program_options>
```

7.6 Displaying Content Versions in the Object Browser

SAP ECTR offers you the option to list all content versions of originals in the Object Browser. You can display the content versions in the Object Browser in the "Originals" tab by "expanding" the originals. The content versions are listed as additional originals underneath the original.

Here, the following preference variable specifies for each workstation application whether the content versions are displayed underneath an SAP original (`true`) or not (`false`):

```
plm.showContentVersions.wsAppl.<wsappl> = true
```

The default value is `true`.

7.7 Linked Materials

7.7.1 Reading a Follow-up Part

If you activate the `follow-up` function and the copying of materials in the DType, a drop-down menu with the option `Follow-up Part` appears in the copy as dialog.

If you choose this option, the successor material is entered in the classification of the predecessor.

For Documents

For the document method, the following preference variables are set:

Preference Variable	Description
`plm.doc.successor.classname = <DOCUMENT CLASS>`	Specifies the document class name.
`plm.doc.successor.charname = <CHARACTERISTIC_WITH_SUCCESSOR_NUMBER>`	Specifies the characteristic name.

For Materials

For the materials, the following preference variables are set:

Preference Variable	Description
`plm.mat.successor.classname = <MATERIAL CLASS>`	Specifies the material class name.
`plm.mat.successor.charname = <CHARACTERISTIC_WITH_FOLLOW-UP_NUMBER>`	Specifies the characteristic name. The material format needs to numeric. In the `dtype.xml` set `copy_as` to `yes`.

7.7.2 Setting the Object Link Filter

Defining Filters

You can define a filter for the preference variable described below. When you open documents via a material, the filter no longer displays all object links. The filter only shows the document types that are in the list. Set the following preference variables in the file `default.txt`:

Preference Variable	Description
`plm.document.open.allowed.bymaterial = <document type>`	The value `<document type>` is a semicolon-separated list of document types.
`plm.document.open.allowed.bymaterial.<parameter> = <document type>`	The value `<parameter>` is an identification that enables the assignment between the menu function and the preference variable.

Example

`plm.document.open.allowed.bymaterial = UGD;UGM`

Corresponding menu function: `fnc.mat.open.doc`

`plm.document.open.allowed.bymaterial.NX = UGD;UGM`

Corresponding menu function: `fnc.mat.open.doc(NX)`

Procedure in SAP ECTR

Right-click on the material to open the selection menu on the function `Open/install assigned material`.

SAP ECTR opens a filtered document list.

Ignoring Document Types

In preference variable `plm.mat.olinks.docTypeFilter` you can store a list of all document types that are not displayed in the `Linked Objects` container. The document types must be specified explicitly, wildcards with `"*"` are not supported.

Ignoring Material Types

You can use this setting to specify material types that are ignored for the evaluation of `'has_material'` flags and of the CAD material. The material types are displayed in the `Linked Objects` container. You can transfer several material types separated by `';'`. This setting is made in the `default.txt` file.

Preference variable: `SAP.PLM.MATLINK.MATTYPE.IGNORE`

7.8 Checkin through saphttp or URL

7.8.1 No Check on saphttp

You can save documents without using `saphttp`. If you want to deactivate the existence check, set this preference variable:

Preference Variable	Description
`plm.doc.checkin.useCadServices = true`	If the preference variable is set to `false`, the document is saved as `saphttp` as before.

7.8.2 Converting Characteristic Values into URLs and Displaying them in the Browser

You use a classification characteristic to define a basic material for a material. You select the possible characteristic values, e.g. wood or iron, from a list. You can store a specification in the form of a URL, a document in SAP or a material for each individual characteristic value. You can view and edit these specifications by clicking the characteristic value.

Customizing in the SAP system

1. Create a classification with its characteristics.
2. Create the characteristic groups and assign the groups to the individual characteristics. Choose transaction `SPRO → SAP Reference IMG → Cross-Application Components → Class System→ Characteristics → Define Characteristic Groups` .
3. You can use the name of the characteristic group to specify what type of information is displayed:

Characteristic group	Description
`<Characteristic group prefix>_I`	A URL is called.
`<Characteristic group prefix>_D`	The system searches for an SAP document where the information about the classification or the display of attached files is shown.
`<Characteristic group prefix>_M`	The system searches for an SAP material where classification values can be displayed.
`<Characteristic group prefix>_U`	SAP user information is displayed. For characteristics with several values, all characteristic values are displayed underlined. When you click on the user data, this data is displayed on one tab page per user in the info dialog.

4. Set the prefix of the characteristic group using the preference variable `plm.specifications.chargrp.basename`.

   Example

   `plm.specifications.chargrp.basename = DSCSPC`

   Resulting characteristic group name:

   - `DSCSPC_I` for URL
   - `DSCSPC_D` for document
   - `DSCSPC_M` for material
   - `DSCSPC_U` for SAP user information

   After the characteristic part of the characteristic group name, there can be additional characters that are ignored, such as:

   - `DSCSPC_I_URL` is treated like `DSCSPC_I`; `"_URL"` is ignored.
   - `DSCSPC_D_DOK` is treated like `DSCSPC_D`; `"_DOK"` is ignored.
   - `DSCSPC_M_MAT` is treated like `DSCSPC_M`; `"_MAT"` is ignored.
   - `DSCSPC_U_USER` is treated like `DSCSPC_U`; `"_USER"` is ignored.

5. Assign a characteristic a previously defined characteristic group:

If specifications are available in the form of documents in the SAP system, you also have to create the document type for the specification documents, e.g. SPC:

6. You define the response of SAP Engineering Control Center in the `default.txt` file.

Calling a URL as a Specification Object

Preference variable `plm.specifications.baseurl.<characteristic name>` contains the name of the base URL as well as a placeholder for the characteristic value.

The following placeholders are available

- `$NW$` = neutral characteristic value
- `$SW$` = language-specific characteristic value

Example of the base URL of the characteristic BASIC MATERIAL

```
plm.specifications.baseurl.WERKSTOFF = http://de.wikipedia.org/wiki/$SW$
```

For individual characteristics, characteristic-specific URLs are composed.

With this setting it is not checked whether this characteristic-specific web page exists.

Calling a Document Info Record as a Specification Object

Preference variable `plm.specifications.chargrp.doctype` defines the document types that are used for the specification documents:

```
plm.specifications.doctype = <document type>
```

When you click on the link, the system searches for a document with the document type according to the preference variable (e.g. `SPC`) and a document number according to the neutral characteristic value. The system displays the data if it finds it.

Example

```
plm.specifications.chargrp.doctype = SPC
```

Neutral characteristic value for iron = "`FE`"

```
FE/SPC/*/*
```

Resulting search terms

- Document type: SPC
- Document number: FE
- Document part (fixed): 000
- Document version (fixed): 00

If no document is found with these search terms, you get an error message when you click on the link: `Unable to determine specification object`.

Calling a Material as a Specification Object

When you click on the link, the system searches for a material number according to the neutral characteristic and displays it in the `Object Browser`. If no material is found with these search terms, you get an error message when you click on the link: `Unable to determine specification object`.

7.9 Applying an Authorization Group

The authorization group can be transferred from the master to the dependent document. The following preference variables are used for this purpose:

- `plm.document.authgrp.readFromMaster`
- `plm.document.labor.readFromMaster`

If the preference variable is set to `true`, the new dependent document receives the same authorization group as the document. The default value is `false`.

7.10 Configuring the Originals Container

It this file is e.g. accessed using copy&paste or the file is stored in the file system, the original of a document can be displayed in the `Originals` container. You cannot edit the original in this context. The editing of the original is still only possible using the functions on the document.

Read-only originals are additional originals for which the option `<view_files> ... </view_files>` has been set under `<additional_files>`in the DTypes. In this context, the originals are always read-only, irrespective of whether or not they were mentioned under `<view>`.

7.11 Hiding Object Types

The following preference variable makes it possible to hide certain object types in the `Linked Objects` container under the document:

```
plm.om.doc.olinks.hide = <object type>;<object type>
```

Example

Hiding technical locations and document structure.

```
plm.om.doc.olinks.hide = IFLOT;STPO_DOC
```

7.12 Transferring Attributes

To ensure the field is filled with a component attribute, SAP ECTR must be set as follows:

1. Define the component attributes that are read and transferred to SAP. If you use the component attributes for bills of material, this definition is required.

   Directory: `\applications\ugs\customize\config\default.txt.`

   ```
   UGII_D_COMPATTR_TO_SAP =
   IGNORE;MENGE;EINHEIT;PTYP;RML;RMB;RME;REPLACE_COMP;USE_MATERIAL;ZDSC
   ```

2. Sort the document structure according to component attributes.

   Directory: `\customize\config\default.txt`

   ```
   plm.docstructure.mapping = ZDSC@ZZDSC

   plm.docstructure.itemkey.UGM = <DIRKEY>;ZZDSC

   plm.docstructure.showZfields = ZZDSC
   ```

3. Specify the field label in the `Assemblies` window.

   Directory: `\customize\dictionary\de\customer.txt`

   ```
   plm.docstructure.mapping.ZZDSC = ZZ DSC (KIT)
   ```

7.13 Applying Characteristic Values

The last evaluation of the document classification in the document creation user interface is saved. If you make the setting using preference variables, the evaluation is pre-filled when the new DType is selected. SAP ECTR saves the evaluations either in a creation transaction of for the duration of an SAP ECTR session.

Preference Variable	Description
`plm.document.create.gui.classification.storeLastValue = true/false (standard)`	Save the last entered evaluations in the creation window.
`plm.document.create.gui.classification.storeLastValue.session = true/false (standard)`	Save the last entered evaluations in the document classification in the creation window for the entire SAP ECTR session.

7.14 Excluding Folders

You can exclude certain folders from the "Display the latest document version in folders" option.

To do this, the following preference variable is used:

Preference Variable	Description
`plm.folder.loadLatest.ignore.nameRegexp = <regular expression>`	The regular expression defines the names of the folders that are excluded. If the regular expression is blank or the preference variable is not defined, no folders are excluded.

7.15 Outputting Errors for Multiple Selections

If you insert a component using your CAD application, you can select more than one document in SAP ECTR. Depending on the application, only one component is transferred at once.

To deactivate the selection of multiple components in SAP ECTR, you have to set the following preference variable:

```
plm.document.addComponent.singleSelectionOnly.UGS = false (Default)
```

If you set this preference variable to `true`, the following error message appears when you select multiple components in SAP ECTR: `Please select only one document`.

7.16 Displaying the Top Nodes During Import on Demand

If you only want to display only the top nodes in the active folder for the function `ImportOnDemand,` set the following preference variable:

`plm.folder.add.only.selected.parts = true` (Default: `false`)

This setting evaluates the selected status of the parts. Only these parts are added to the active folder.

7.17 Displaying Individual DTypes Only for Certain Actions

In the `dtype.xml` file, you can specify that a DType is displayed only for a listed action. This is done using the `<allowed_actions>` tag.

This tag is optional.

The `show_for_create` variable must be set to `yes`, otherwise the DType is never displayed.

If, for example, a DType is displayed only for Import on Demand, this could be as follows in the `dtype.xml`:

```
<DType ...> ...
  <program_options ... show_for_create="yes" ...>...</program_options ...> ...
  <document_create ...> ...
```

```
    <allowed_actions>
        <action name="application_doc_create_multi"/>
    </allowed_actions>
  </document_create> ...
</DType>
```

With `show_for_create="no"`, the DType is no longer displayed in actions.

For more detailed information about possible actions, see the Operations Guide section relating to the tag `<allowed_actions>`.

7.18 Support of Status Type "L"

The status type "L" is treated as an editable status with limitations. Documents with status type "L" can be edited in the mass change dialog and in the object browser with limitations. The configuration for editable fields is read from the SAP system.

Mass Change Dialog

For status type "L", the editability of the `LABOR, AUTHORIZATION_GROUP, DESCRIPTION` fields (in all languages) and the entire classification are checked. ExtensionServices (OSGI/F4) is disabled for non-editable fields.

Object Browser

For status type "L", the editability is checked for the following tabs/fields:

- Basic data fields:
  - LABOR
  - AUTHORIZATION_GROUP
  - USER
  - ECM
  - DESCRIPTION
- Classification
  - Classification tab
  - Classification on the basic data tab
- Long text tab

All other fields are not editable in "L" status. If the classification is not editable, the editing function is disabled for the entire tab. It is also not possible to edit the classification of linked objects.

7.19 Suppress BOM items

Whether a BOM item is to be suppressed or not is included as information in the sort string (SORTF) of the SAP document BOM item.

The information in the sort string is specified in the form 'CAT:C'. The application type is specified before the colon, and an SAP-defined letter describing the relationship of the BOM item to the higher-level document is specified after the colon. In the following this letter is called relationship type.

At each check-in or save, a BOM from CAD data is completely rewritten by default.

If, however, information in the sort string of the BOM items is subsequently entered manually with SAP, it is ignored and not transferred when the BOM is saved and rewritten from pure CAD data. However, there is a preference variable that allows you to specify the relationship types after ":" that are to be transferred when the BOMs are updated. The letters are transferred from the sort string of the old BOM item existing in SAP to the sort string of the new BOM item.

There is an application-dependent preference variable and an application-independent preference variable.

The application-independent preference variable serves as a fallback, i.e., it is only used if no application-dependent preference variable is defined.

```
plm.docstructure.sortfield.transferChar.<applType> = <relationship types to be
transferred, semicolon separated>.
```

In the application dependent preference variable, specify the application type as below. The characters to be transferred are listed without spaces and separated by semicolons, e.g:

```
plm.docstructure.sortfield.transferChar.UGS = W;S
```

The fallback preference variable is set up in the same way as the application-dependent preference variable, only without application specification.

```
plm.docstructure.sortfield.transferChar = <relationship types to be transferred,
semicolon separated>.
```

e.g.

```
plm.docstructure.sortfield.transferChar = W;S
```

If either variable is defined, a check is made to see if an item in the new document structure that has not yet been written already occurs in the old document structure. If so, the system checks whether the sort string of the old BOM item contains a relationship type defined in the

preference variable. If the sort string contains the relationship type, it is copied to the new item.

8 MATERIALS

8.1 Marking Material

The following applies to the setting SAP.PLM.MATLINK.HAS.SAMENR.LIKE.DOC:

If this setting = 'X', then:

If no link is marked as exclusive, SAP ECTR searches for a material with the same number as the document. If no material with the same number is available, the first material link is used. Usually, the smallest material number that is linked to the document is used. This setting can negatively affect performance.

8.2 Configuring the Input Help

The material data fields that are displayed in SAP ECTR can be configured in two different ways:

- Using the Administrator menu function SAP Engineering Control Center System Setup -> Selection of Object Data for SAP Engineering Control Center
- Using transaction /DSCSAG/SEL_FIELDS

You use this to specify which material data fields are available and can be changed on the Material tab of the document creation dialog and in the Basic Data tab in the Object Browser. You can use a pseudo class to define the input options and input helps for material data fields that can be changed.

Creating a Material Class (Class Type 001) as a Pseudo Class

The material class describes the classification characteristics that correspond to the material data fields. The class is not assigned to a material.

Creating Classification Characteristics in this Class

The characteristics define which input options and input helps are available for the respective material data field in the dialogs.

The following naming conventions apply to the materials:

- <Customer-specific preamble>__<table name>-<field name>
- <Table name>-<field name>

Preference Variable	Description
<Customer-specific preamble>	Any character string
<Table name>	The name of the table in the SAP system (e.g. from the material data selection dialog in the SAP system)
<Field name>	The name of the field in the SAP system (e.g. from the material data selection dialog in the SAP system).

For each of the characteristics, an input field is displayed in the SAP ECTR dialogs and the entered value is assigned to the corresponding material data field. The properties and value tables of the characteristics determine the input options and input helps in the SAP ECTR dialog.

Note

A characteristic can have a max. length of 30 characters, which is why the input help for the characteristic is also limited to 30 characters. Hence, fields such as `Basic Material` or `Measurements of the Basic Material Data` can only be filled to a limited extent using the input help.

For characteristics of the material pseudo-class, no units must be defined because the function module for changing the material data cannot be avoided with this.

Example

Class: `MATEDIT`

Characteristics:

```
MARA-BRGEW          Gross weight g
MARA-ZEIFO          DIN format
MARA-GROES          Size /dimensions
MARA-MSTDE          Valid from
MARA-NTGEW          Net weight
MARA-PRDHA          Product hierarchy
MARA-SPART          Division
MARA-MATKL          Goods groups
MARA-MSTAE          Cross-plant material status
```

Assigning the Class to a Material Type

You can use a preference variable in the `default.txt` file to assign a pseudo class either to all material types or to a specific material type:

```
plm.mat.edit                   = <pseudo class>
plm.mat.edit.<material type> = <pseudo class>
```

Example

```
plm.mat.edit = MATEDIT
```

For different material types, you can make different input fields available in the dialogs.

8.3 Displaying Object Management Records

If the following preference variable is set, object management records without revision are also displayed in the revision container:

```
plm.om.mat.container.saprevision.show.withoutrevision = true
```

8.4 Creating Several Material Master Records

To make it possible to create and assign additional materials to an existing component or assembly with an assigned material, the following preference variable must be set:

```
plm.mat.create.skipFirstScreen = false
```

8.5 Configuring the Material Data for the User Interface

The material properties that are available in the SAP ECTR user interface can be configured dynamically. These material properties can be defined in report `/DSCSAG/SEL_FIELDS`. The material properties to be changed are specified using the characteristics of a pseudo class.

In the Object Browser, the selected characteristics are then displayed on the Basic Data tab. If a change is made, the system reads the characteristics from the pseudo classification and then makes the characteristics that can be changed available for changing in the Object Browser. The system links the characteristics of the material and the characteristics of the pseudo class. The default values for material properties that can be changed come from the characteristic default values of the pseudo class.

Selecting Material Data Using Transaction /DSCSAG/SEL_FIELDS

1. In SAP ECTR, choose Administration → ECTR System Setup → Select Object Data. Alternatively, you can call transaction /DSCSAG/SEL_FIELDS in the SAP system.
2. Select the object properties that you want to display in SAP ECTR.

   The first check box shows that the object property cannot be changed.

   The second check box applies to change rights.

   The third check box shows that the data of this characteristic is applied to the copied material when this characteristic is copied in the background.
3. You can display the selected object properties on the Basic Data tab in the Object Browser. You can use the Properties layout to configure the object properties according to your requirements.

Note

Even if you have not made any changes in the /DSCSAG/SEL_FIELDS transaction, save the transaction once when performing the configuration. This helps ensure that all standard values are transferred correctly to other tables in the SAP system.

Recommendation

When you configure the material data, keep the following in mind:

SAP ECTR does not support all material data fields in the SAP system. When you configure the material data, proceed in a restrictive manner and configure only the material data that you need. This procedure also improves performance because less data has to be transferred from the SAP system to SAP ECTR.

The user interface does not respond dynamically to the configuration of the material data. For performance reasons, SAP ECTR reads the configuration information via material data to be displayed when it is started. That is why you have to restart SAP Engineering Control Center, after you have configured the material data.

If you want the material data fields to respond dynamically, you can configure the dialog for changing material data in the SAP GUI.

Note

If problems arise when displaying plant (MARC) and storage location data (MARD), the values for the parameters "WRK" and "LAG" must be maintained in transaction SU3.

Making Material Data Fields Editable Using Table /DSCSAG/TBAPIFL

This table is used to map the fields of a material to the BAPI structure. In the function Select Object Data for SAP Engineering Control Center, you can select fields for displaying and changing. This mapping is visible in the displayed selection dialog. Data can only be changed from SAP ECTR if a mapping exists for a field. Otherwise, it can only be displayed.

Note the special procedure for customer-specific fields.

Creating Customer-Specific Attributes

First, you have to enhance the existing structures (`BAPI_TE_MARA and BAPI_TE_MARAX`) with customer-specific fields. In the following example, these are the fields:

- `ZZCADSYSTEM`

  of type `/DSCSAG/CAD_SYSTEM`

  in table `BAPI_TE_MARA`
- `ZZCADSYSTEM`

  of type `BAPIUPDATE`

  in table `BAPI_TE_MARAX`

For enhanced fields, the following applies:

- They must have the type `CHAR` or another type that can be converted to `CHAR`.
- You must enhance the `MARA` structure with the `ZZCADSYSTEM` fields of type `/DSCSAG/CAD_SYSTEM`.

You can assign these fields to a field group. If the fields are not assigned, they are not updated using BAPIs and function module /`DSCSAG/ADV_MAT_SETALL` returns a warning.

Defining the Pseudo Class for the Material Data Selection

To ensure you can change the material data in the Object Browser, preference variable `plm.mat.edit.<material type>` must be used to specify a material pseudo class in `default.txt`. This preference variable is also used for the document creation with material.

- Preference variable in `default.txt`:

```
plm.mat.edit.<material type> = <name of the material pseudo class>
```

- Fall-back item for all material types:

```
plm.mat.edit = <name of the material pseudo class>
```

Rules for Naming the Class/Characteristics

The material pseudo class is any SAP material class. You can freely select the names for the material class. The characteristics are used to define which material data can be changed. If materials receive default values, they are displayed when changing.

The following naming conventions apply to the materials:

- `<Customer-specific preamble>__<table name>-<field name>`
- `<Table name>-<field name>`

They mean the following:

Preference Variable	Description
`<Customer-specific preamble>`	Any character string
`<Table name>`	The name of the table in the SAP system (e.g. from the material data selection dialog in the SAP system)
`<Field name>`	The name of the field in the SAP system (e.g. from the material data selection dialog in the SAP system).

8.6 Setting up Z Fields in the Material Master Record

You can set up Z fields (which belong to MARA). To do this, go to report /DSCSAG/SEL_FIELD using the SAP Engineering Control Center Object Browser. Then proceed as follows:

1. Use the name of the Z field in the BAPI structure (BAPI_TE_MARA).
2. Change the setting in transaction /DSCSAG/SE16N or using SE16N with &sap_edit.
3. For a selected material in the tab pages of the Object Browser, you can configure whether it can be edited. To do this, use the pencil icon and the following preference variable:

Preference Variable	Description
plm.om.mara.bas.editable	Basic data is editable (default: true)

8.7 Linking Material Master Records to Documents

SAP ECTR can link material master records to master documents and dependent documents.

The dependent document is thereby assigned the same material reference as the master document. If you set the following preference variable to true, material masters are linked to master documents and dependent documents:

```
plm.document.markMatLink.addNonmaster = true
```

9 PRODUCT STRUCTURE MANAGEMENT (PSM)

9.1 General Information

SAP ECTR supports the objects from SAP Product Structure Management (PSM). All PSM objects that are based on node and variant types are supported. Product assemblies are not supported.

Objects, containers, and functions are available on the desktop.

The functionality is a part of the `ECTRWUI` add-on, which is available as an installation option in the SAP ECTR installer.

9.2 Configuration

The following types are delivered as standard:

Node types

- S_HD (product family)
- S_ST (product item)
- S_VW (product view)

Variant types

- S_CMPHD (product variant)
- S_CMPST (product item variant)

9.2.1 Icons

Icons can be adjusted to customer-specific node and variant types.

To do this, the content of the file `\addons\ectrwui\basis\aux-files\ standard_icons.txt` must be copied, pasted into the file `\customize\aux-files\ customer_icons.txt`, and adjusted.

The icon definitions are structured according to the following schema:

- Node types: `plm.psm_node.$(NODE_TYPE)`
- Variant types: `plm.psm_variant.$(VARIANT_TYPE)`

The following icon definitions are delivered as standard:

- Node types
  - `plm.psm_node.S_HD = {0}/sap/product_family`
  - `plm.psm_node.S_ST = {0}/sap/product_item`
  - `plm.psm_node.S_VW = {0}/sap/product_view`
- Variant types
  - `plm.psm_variant.S_CMPST = {0}/sap/product_item_variant`
  - `plm.psm_variant.S_CMPHD = {0}/sap/product_variant`

10 SEARCH MASKS

10.1 Fully User-Definable Search Masks

User-definable search masks can be used to add objects to folders. The adjustments are made in `default.txt` or in SAP Engineering Control Center itself.

10.1.1 SAP Easy Document Management System: Configuring Compatibility Mode

SAP Easy Document Management System and SAP ECTR behave differently as regards authorization checks when documents are added to a folder. SAP ECTR checks the change authorizations for the folder. SAP Easy Document Management System checks only the authorization for the change to the document structure. To achieve compatibility with the SAP Easy Document Management System authorization check, you can deactivate this additional authorization check for changes to the folder. To do this, set the following preference variable: `SAP.PLM.EASYDMS.COMPATIBLE = true`

The standard value is `false`.

10.1.2 Configuring User-Defined Search Masks

The SAP ECTR user interface enables you to freely configure search masks for the search categories `DOC`, `MAT`, and `ECM`. The search masks created can be saved in the form of a text file. When the interface is started, the directory in which the text files are located checks whether the masks are in the SAP system. If they are not, documents are created for this purpose, and the text files are saved in the SAP system. After this, the document must be opened in read-only mode. With an after-checkout script (which can be defined using DType), the mask layout file can be copied from the session directory to the mask directory intended for this purpose.

When SAP ECTR is started, it reads the selected search query using the following text file:

```
<SearchmaskName><SearchmaskType>.txt
```

The file is in the definable directory:

```
plm.search.generic.mask.directory = @PLM_MASKDIR@
```

Define the document type for search masks (MSK) in `dtype.xml`.

If the directories are not defined otherwise, masks are searched for in the following directories when called:

```
%PLM_INSTDIR_CONF%\addons\<addon>\templates\masks
%PLM_INSTDIR_CONF%\applications\<appl>\templates\masks
```

Other Preference Variables

Preference Variable	Description
`plm.search.useProcessLevel3 = true`	Search dialog editor
`plm.generic.search.masks.dtype = SMASK`	DType for editing search masks
`plm.searchGui.searchmasks = STANDARD_DOC;STANDARD_ECM;STANDARD_MAT`	Search mask list imported when system is started

SAP ECTR delivers preconfigured search masks in the templates folder for `DOC`, `ECM`, and `MAT`. You can edit the labels of the individual search masks depending on language:

File: `customer.txt`

Directory: `%PLM_INSTDIR%\customize\dictionary`

Integrating TREX Search into SAP ECTR

Preference Variable	Description
`plm.doc.search.enterprise.active`	Enterprise Search active (standard value: `false`)
`plm.generic.search.menu.item.doc`	List of the document search field types available when the search masks are put together. The prefilled values are: `PROP;CHAR;LATEST;LABEL;PRV;LTXT;FTXT;ETP;SLOG;DYN`
`plm.generic.search.menu.item.mat`	Extend material search to include Enterprise Search field The prefilled values are: `PROP;CHAR;LABEL;PRV;DYN;ET`

You can remove the SAP Search button from search masks:

`plm.search.showSapSearchButton = false` (standard value: `true`)

Abbreviations

Abbreviation	Explanation
PROP	Property
CHAR	Class characteristic
LATEST	Version selection
PRV	Preview
LTXT	Long text
FTXT	Full text
ETP	Enterprise
SLOG	Status log characteristic
DYN	Dynamic classification area

Long Text Search for Document Search

You can search for long texts.

1. Activate the configurable search mask.
2. You can specify a list of long text languages to be searched for (standard DE; EN).
3. Enter the following preference variable:

   `plm.search.longtext.languages` = <value>

Configuring the Search Mask for the Material Search

The selection of search mask fields for material searches is predefined in the SAP system. Proceed as follows:

1. In SAP ECTR, select `Administration → System Setup → Select Object Data`.
2. Select `Material → General Material Data`. The system shows the selection list (material data report) of the selectable search characteristics.
3. Next to the binoculars icon, select the search characteristic you require.
4. Under `System → Maintain Search Masks → Add a Field`, add the characteristic selected in the SAP system.

10.1.3 Configuring Search Fields

In SAP ECTR, you can select the available search fields in two ways:

- Using the administrator menu, with the `Select Object Data` function
- Using transaction `/DSCSAG/SEL_FIELDS`

Transaction `/DSCSAG/SEL_FIELDS` has an additional column in which the available search fields can be selected.

The configurable material data fields are available for defining a material search dialog.

10.2 Searching from the Clipboard

You can place object numbers and document keys on the clipboard. You can search for the content of the clipboard. The result is displayed in a search result table, which you can use to fill a folder. You can use a preference variable to adjust the form of the object numbers and document keys.

10.2.1 Functions for searching from the clipboard

The following entries have been added to the menu file for this purpose:

```
fnc.object.search.from.clipboard(document)
```

Searches for document numbers that have been placed on the clipboard.

```
fnc.object.search.from.clipboard(document,qualified)
```

Searches only for documents with the document keys that have been placed on the clipboard and previously adjusted using a preference variable. This is described in the following section.

```
fnc.object.search.from.clipboard(material)
```

Searches for material numbers that have been placed on the clipboard.

```
fnc.object.search.from.clipboard(ecn)
```

Searches for change numbers that have been placed on the clipboard.

10.2.2 Preference variables for configuring the search from the clipboard

Regular expressions are used in these preference variables.

You can flexibly divide the content of the clipboard. For this purpose, the separators for dividing the search terms are configured using the following preference variable:

```
plm.om.search.clipboard.separators = " \t\n\r\f,;"
```

For example, the following are specified here:

```
\t = Tab

\n = New line

\r = Return

\f = Form feed
```

In addition, the pattern of the specified object numbers and keys must be defined for a search. Here, you must take into account the setting of the `plm.om.search.clipboard.separators`. The following preference variable is used to define the pattern.

```
plm.search.from.clipboard.qualified.pattern.<index> = <pattern>
```

The `<index>` is the sequence number of the search pattern, beginning with 1. Note that this number must also be defined beginning with 1.

<pattern> contains the relevant search pattern.

The following expressions are used as the group name:

```
N (=DOKNR),
```

```
T (=DOKAR),
```

```
P (=DOKTL) and
```

```
V (=DOKVR)
```

```
plm.search.from.clipboard.qualified.pattern.1=
```

```
(?<N>\\S{1,25})#(?<T>\\S{3})#(?<P>\\S{3})#(?<V>\\S{2})
```

This pattern allows a material number of the following form, for example:

```
4711#ugm#000#00.prt
```

If you create your own forms of document keys, you may not use forbidden characters.

Non-existent / empty groups are defined in the pattern as follows:

```
(?<X>\\S{0})
```

`X` can stand for `N, T, P` or `V`.

The following forms of object numbers and document keys are automatically recognized and do not require special configuration:

```
4711ugm000_00.prt
```

```
4711ugm000~00.prt
```

```
0000000000001000001123700000@3dprv
```

```
UGM00000000000000100000112370000@13
```

```
UGM0000000000000010000009991000000
```

`fnc.object.search.from.clipboard(document)` works only for document keys that contain a type, part, version, and document number and that exist in one of these formats:

```
4711ugm000_00.prt, 4711ugm000_00,prt, 4711ugm000_00, 4711ugm000~00.prt,
4711ugm000~00,prt, 4711ugm000~00
```

```
10000003709-DOK-ENU-00, <FolderName>-FOL-000-00-<Description>
```

```
UGM0000000000000010000011237000000@3dprv, UGM00000000000000100000112370000@13,
UGM000000000000001000000999100000,
```

```
UGM 4711 00 000, UGM4711A 00000
```

```
UGM/4711/000/00
```

```
UGM 4711 000 00
```

10.3 Searching for Documents in the Folder Structure

The `Search for documents in folder structure` function (`fnc.find.obj.in.subtree(search_docs_in_subtree.txt)`) can be used to search all documents in a selected folder.

By default, the `search_docs_in_subtree.txt` macro is defined in `<ECTR_INST>/customize/scripts/macros/` and controls the search query:

```
p = GET_MASK_INPUT( "STANDARD_DOC", "doc.mask.title", "DOC")
setId = CREATE_SET(KEYLIST_FROM_REFOBJ())
resolveParameter = PARAMETER_MAP( )
resolveParameter.LOADOPTION="LV"
resolveParameter.IV_MAX_DEPTH="5"
setId1 = TRANSFORM_SET_WITH_PARAMETERS( setId, "FM:/DSCSAG/FOL4_OPEN",resolveParameter)
setId2 = FILTER_BY_TYPE( setId1, "DOC" )
setId3 = FILTER( setId2, "DOC", p )
WRITE_RESULTLIST( KEYLIST_FROM_SET( setId3 ) )
```

You can also define your own macro. In this case, however, you not only have to create it in the directory mentioned above, but you also have to enter the macro name in parentheses after the OMF when configuring the function in `menu.guidef`.

10.4 Searching with SAP Search Helps

You can search using SAP search helps via a quick search, macro, or function.

Currently, the search works only with elementary search helps, which you can configure using transaction SE11.

The search helps must return the ECTR key fields of the object.

SAP Preferences must be set and uploaded for the search helps:

```
SAP.PLM.searchhelp.<SEARCH_HELP>.tabletype
```

```
SAP.PLM.searchhelp.<SEARCH_HELP>.keyfields
```

For the search help `ZVKO_CHANGE_REC_ISSUE_ID`, this would be:

```
SAP.PLM.searchhelp.ZVKO_CHANGE_REC_ISSUE_ID.tabletype=CHANGE_REC
```

```
SAP.PLM.searchhelp.ZVKO_CHANGE_REC_ISSUE_ID.keyfields=CHANGERECORDUUID
```

You upload the setting to SAP using the admin function with the same name.

10.4.1 Search Help via Function

To embed a search function with the SAP search help, use
`fnc.add.with.search.help(<SearchHelp>)`

For the search help `ZVKO_CHANGE_REC_ISSUE_ID`, this would be:
`fnc.add.with.search.help(<ZVKO_CHANGE_REC_ISSUE_ID>)`

To embed a search function with the SAP search help, use
`fnc.add.with.search.help(<SearchHelp>)`

For the search help `ZVKO_CHANGE_REC_ISSUE_ID`, this would be:
`fnc.add.with.search.help(<ZVKO_CHANGE_REC_ISSUE_ID>)`

10.4.2 Search Help via Macro

The command GET_SEARCHHELP_INPUT is used to display the search help GUI. For a more detailed description and an example, see the macro documentation in the Operations Guide.

10.4.3 Search Help and Quick Search

Many search helps search directly in the table of the object and are therefore fast enough to be integrated into the quick search. To this end, a new search help provider was implemented, to which the name of the search help and the search field are transferred.

You can configure the solution to search for an equipment or the equipment name, for example. A prototypical design is shown below, for which the search help `EQUIT` has been created and now needs to be implemented:

```
# Add new type (in this case, EQUIP):
```

```
plm.search.provider.types = @plm.search.provider.types@;DOC;MAT;ECN;EQUIP;FLOC
```

```
# Add search help providers
# EQUIT = elementary search help
# EQKTU = field in the search help
plm.search.provider.EQUIP = SH_PROVIDER(EQUIT,EQKTU);SH_PROVIDER(EQUIT,EQUNR)

# Add SAP Preferences for search help EQUIT
SAP.PLM.searchhelp.EQUIT.tabletype = EQUI
SAP.PLM.searchhelp.EQUIT.keyfields = EQUNR
```

10.4.4 Macro and Quick Search

The quick search can also be extended to include a macro provider.

A macro is transferred as a parameter to the provider `MACRO_SEARCH_PROVIDER`. The macro has a query object as the first parameter and MaxRows as the second. The macro writes the result with the command `WRITE_RESULTLIST`.

Prerequisite

The prerequisite is that the macro is fast enough for the quick search.

The macro is integrated as follows. Replace macroname.txt with the name of your macro:

```
# Add new type
plm.search.provider.types = @plm.search.provider.types@;DOC;MAT;ECN;EQUIP;FLOC;MACRO
#  Add provider definition
plm.search.provider.MACRO =
MACRO_SEARCH_PROVIDER(macroname.txt);MACRO_SEARCH_PROVIDER(macroname.txt)
```

11 FUNCTION ENHANCEMENT

11.1 Generic Call of Transactions and RFC-Enabled SAP Function Modules

Function List

- New, simplified syntax of the definition file
- Support for tables
- Support for structures
- Evaluation of the standard return table (error output)
- Update of objects
- Improved implementation (Java source code)
- Improved call (OMF parameters)

This functionality enables transactions and RFC-capable SAP function modules to be called directly from SAP ECTR using a generic OMF (OmfApiGeneric2) and supplied with object keys.

Defining the XML File

Customer definitions and standard definitions are handled separately.

DSC standard definitions: `%PLM_INSTDIR%\basis\aux-files\api_definitions2.xml`

Customer definitions: `%PLM_INSTDIR%\customize\aux-files\api_definitions2.xml`

These files are used to configure the generic OMF, which are used to call the transactions and the SAP function modules.

Customer definitions have precedence. If the standard definitions file and the customer definition file contain definitions with identical names, the customer definition is used.

Possible Object Types (type="...") and the Possible Inputs

Object	type=”…”	Parameter
Change number	`ecm`	`CHANGENUMBER_INT` `CHANGENUMBER_EXT`
Material	`material`	`MATERIALNUMBER_INT` `MATERIALNUMBER_EXT`

Document	`document`	`DOCUMENTNUMBER_INT`
		`DOCUMENTNUMBER_EXT`
		`DOCUMENTTYPE`
		`DOCUMENTPART`
		`DOCUMENTVERSION`
		`DOCUMENTDESCRIPTION`

Folder	`folder`	`SAPKEY`
Classification	`class`	`CLASSNAME`

Object	type="…"	Parameter
		CLASSTYPE
Sales order	sales_order	SD_DOCUMENT_NUMBER
Sales order item	order_bom_item	SD_ITEM_NUMBER SD_DOCUMENT_NUMBER SD_ITEM_POSITION
Equipment	equipment	EQUIPMENT EQUIPMENT_EXT
Functional location	funcloc	FUNCLOC
Work item	workitem	WORKITEM - WI_ID
WBS element (work breakdown structure)	wbs	WBS_EXT WBS_INT
Project (project definition)	project_definition	PROJ_NUMBER
Material BOM	mat_bom	BOMUSAGE PLANT ALTERNBOM C D MAT_NUMBER
Material BOM Component	mat_bom_component	ITEM_NUMBER QUANTITY COMP_UNIT COMP_MATERIAL + all values defined by Material BOM
PSM objects (ProductFamily, ProductView, ProductItem)	psm_productfamily psm_productview psm_productitem	GUID
CAD variant (for example, SolidWorks configurations)	cadvar	VARIANT_GUID VARIANT_EXTERNAL_ID GENERIC_DOC_TYPE GENERIC_DOC_NUMBER GENERIC_DOC_VERSION GENERIC_DOC_PART VARIANT_DOC_TYPE VARIANT_DOC_NUMBER VARIANT_DOC_VERSION VARIANT_DOC_PART GENERIC_DOC_KEY VARIANT_DOC_KEY MATERIALNUMBER_EXT MATERIALNUMBER_INT
Generic SAP ECTR object	generic	KEY TYPE DESCRIPTION

XML File with Module Call Definitions

Elements

XML module	Element	Description
Generic_API_Calls		Root element
function		Definition of the individual calls
	name	Name of the function (required for the menu options)
	type	Possible object types
	apiname	Name of the function module to be called
	transaction	"bright" or "dark" transaction
	background	"true" or "false" true means that the system does not wait for the transaction to be executed.
	refresh	"true" or "false" true means that the selected objects are updated.
	return	Name of the table or structure for the check for module errors
	multiselection	"true" or "false" true means that a multiselection is permitted.
	getecm	"none", "active", or "document" Change number determination.
	parentRefresh	"true" or "false" "true" means that when an object is created or deleted, the folder above it is also changed. Does not work together with background parameter.
	askConfirmation	"true" or "false" "true" means that the user has to actively confirm the action. The text of the window is determined via confirmationText.
	confirmationText	References entry in customer dictionary. Example: confirmationText="plm.omf.2900"
import		Definition of the import parameters Can contain <parameter>, <structure>, and <table>.
parameter		Definition of field contents for both input parameters and for tables
	name	Name of a field of the module to be called

XML module	Element	Description
	`string`	This value, as it appears in the definition file, is transferred the module.
	`value`	Attribute of the input object
`structure`		Definition of a structure
`tables`		Definition of the table, can contain `<table>`.
`table`		Definition of a table. If multiple objects were selected, multiple rows are generated in the table.
	`name`	Table name" Name of a table of the module to be called
	`foreach_object`	"`true`" or "`false`" false means only one row is inserted, irrespective of the object selected.

Additional Functions

Additional Object Determination

Normally the functions may be executed only on the objects of the same type (attribute `type`) as the functions. For some object types, there is the option of executing the function on another object type.

- Change number: `type="ecm"`

  Can also be executed on a document. A change number is taken from the document.

- Material: `type="material"`

  Can also be executed on a document. The CAD material is taken from the document.

- Document: `type="document"`

  Can also be executed on a material. All the linked documents are displayed in a GUI. A document can be selected.

Additional Change Number Determination

In addition to the change number determination as a dedicated object type, there is the option of determining the change number of the document and material functions. You can use either the active change number (`getecm="active"`) or the number of the document (`getecm="document"`). If you call the material function for a document, the change number of the selected document is used.

Multiselection of Objects

If the attribute `multiselection="true"` is set, multiselection is activated. Multiselection means that the tables (or import tables) are filled with characteristics of the selected objects. This creates a table with material numbers of selected materials.

If multiselection is deactivated, an error message is displayed in the SAP ECTR status bar.

Table Functions

In addition to the option of adding a row in a table for each object selected, there is the option of adding table rows irrespective of selected objects.

You can set the attribute `foreach_object="false"` for the `<table>` element. This row is then added only once, irrespective of the number of the objects selected. If you want to add multiple rows, create multiple `<table>` elements for the same table. In this case, possible object attributes always go to the first object selected.

Access to environment and preference variables

Within the call definition you have the possibility to use variables for the value definition of the xml attributes, which access the values of environment variables or SAP ECTR preference variables.

The notation of these variables is as follows:

```
@<Variable name>@
```

in which `<Variable name>` stands for the name of the respective environment variable or SAP ECTR preference variable.

For example, the variable `@sap.user.parameter.WRK@` accesses the content of the preference variable `sap.user.parameter.WRK`.

Defining XML File

You can find the definition files under the following paths:

Standard definitions: `%PLM_INSTDIR%\basis\aux-files\api_definitions2.xml`

Customer definitions: `%PLM_INSTDIR%\customize\aux-files\api_definitions2.xml`

You can call the module defined in `api_definitions2.xml` using the following function in `menu.guidef`:

```
fnc.api.generic2(<Generic_API_Call>)
```

Example

```
<Generic_API_Calls>
  <function name="cl02_cls"
            type="class"
            apiname="/DSCSAG/TRANSACTION_CALL2"
            transaction="bright">
     <import>
        <parameter name="TCODE" string="CL02"/>
        <parameter name="SKIP_FIRST_SCREEN" string="X"/>
        <parameter name="CLASSNUM" value="CLASSNAME"/>
        <parameter name="CLASSTYPE" value="CLASSTYPE"/>
     </import>
     <tables/>
  </function>


  <function name="zeia_mat"
            type="material"
            apiname="/DSCSAG/TRANSACTION_CALL2"
            transaction="bright"
```

```
          refresh="true">
   <import>
      <parameter name="TCODE" string="ZEIA"/>
      <parameter name="MATERIAL" value="MATNUM_INT"/>
      <parameter name="MATERIAL" value="MATNUM_EXT"/>
   </import>
   <tables/>
</function>

<function name="cdesk_doc"
          type="document"
          apiname="/DSCSAG/DOC_CDESK_SHOW"
          transaction="bright"
          background="true">
   <import>
      <parameter name="INITIAL_VIEW" string="SAPSTRUCT"/>
      <parameter name="SAP_VIEW_ONLY" string="X"/>
      <parameter name="CALLBACK" string=" "/>
      <parameter name="DOCID" value="DOCUMENTTYPE"/>
   </import>
   <tables>
     <table name="DOCUMENTKEY">
       <parameter name="DOCUMENTNUMBER" value="DOCNUM"/>
       <parameter name="DOCUMENTTYPE" value="DOCTYP"/>
       <parameter name="DOCUMENTPART" value="DOCPRT"/>
       <parameter name="DOCUMENTVERSION" value="DOCVERS"/>
     </table>
   </tables>
</function>

<function name="sm12"
          type="none"
          apiname="/DSCSAG/TRANSACTION_CALL2"
          transaction="bright"
          background="true">
   <import>
      <parameter name="TCODE" string="SM12"/>
   </import>
   <tables/>
</function>

<function name="mat_bom_create"
          type="material"
          apiname="/DSCSAG/MAT_BOM_RFC_CREATE"
          transaction="bright">
   <import>
     <structure name = "MATERIAL_HEADER_DATA">
        <parameter name="MATNR" value="MATERIALNUMBER_INT"/>
        <parameter name="CLASSTYPE" value="CLASSTYPE"/>
     </structure>
   </import>
   <tables/>
</function>
```

```
    <function name="mat_test"
              type="material"
              apiname="/DSCSAG/ADV_MAT_GETALL"
              transaction="bright"
              return="T_RETURNS">
      <import>
        <table name="IT_MATERIAL">
           <parameter name="MATERIAL" value="MATNUM_INT"/>
        </table>
      </import>
    </function>
  </Generic_API_Calls>
```

11.2 Configuring Customer-Specific Instant Conversions

Options for instant conversion:

- Using the menu function `fnc.doc.regenerate(<identifier>):`

  Here, the menu function starts the instant conversion defined using the `<identifier>`.
- The DType setting `after_create_trigger="<identifier>"` in document creation:

  This DType setting is located in the `<document_create>` section and executes the instant conversion defined using the `<identifier>` immediately after the creation of the document.

  ```
  <document_create ... after_create_trigger="<identifier>" >
     ...
  </document_create>
  ```

In both cases `<identifier>` is part of a series name of preference variables that define the customer-specific instant conversion in more detail:

Preference Variable	**Description**
`plm.control.afterDocCreate.<identifier>.class = <Java-Class>`	Defines the name of the Java class that is called for the instant conversion, which receives additional parameters and starts the conversion script.
`plm.control.afterDocCreate.<identifier>.script = <script>`	Defines the name of the conversion script that is transferred to the Java class for startup. The default value for `<script>` is `dsc_createDerivedOriginal.exe`.
`plm.control.afterDocCreate.<identifier>.attribs = true/false`	Specifies whether an attribute file is also made available. The name of the attribute file is identical to that of the original with `".att" appended, e.g. 4711ugm000_00.prt.att`.
`plm.control.afterDocCreate.<identifier>.loadOption = <load option>`	Defines the load option to be used when providing the assembly components.
`plm.control.afterDocCreate.<identifier>.extraParams =`	Defines the transfer parameters that are transferred to the conversion script.

The following settings are supported in `<load option>`:

Load option	Description
`loadLatest`	The respective latest version of each component is provided.
`asSaved`	The components are provided in the versions "as saved."
`released`	The released versions of the components are provided.
`relToday`	The released versions of the components that are valid today are provided.

Functions

The actual document and the reference documents are made available in a temporary directory. The script to be called processes the documents. After (correct) processing, the script stores the reference document in the SAP system.

> **Note**
>
> Actual document: Document on which the instant conversion is executed.
>
> Reference document: If the actual document is a dependent document, the assigned master document.

Process flow

Make the actual document available and do not check it out.

If available, make the reference document available and do not check it out.

Parameter	Description
`<Script name>`	Name of the conversion script from `plm.control.afterDocCreate.REGEN.script`
`<ActDoc>`	File name of the actual document
`<ApplType ActDoc>`	Application type of the actual document (value from RES4)
`<RefDoc>`	File name of the reference document
`<ApplType RefDoc>`	Application type of the reference document (value from RES4)
`<ExtraParams>`	Additional script parameters from `plm.control.afterDocCreate.REGEN.extraParams`

11.3 Displaying Characteristics with Check Module as Characteristics with Check Table

In SAP ECTR, you can specify both a value list (check table in SAP) and a function module (check module) for a characteristic to determine the values of the characteristic.

You can handle characteristics with a check module as if they were characteristics with a check table.

This is because the SAP system does not permit multi-column check tables. This means that you have to specify the characteristic names for which the defined check module is handled like a check table:

```
plm.characteristics.useAsChecktable = <list of characteristic names, separated by ;>
```

You can define whether for value selection you want table contents to appear in a DSC dialog or SAP GUI window when a check table is used. To do this, you need to define which columns of the table are to be displayed. If the variable is not set, only the key fields are displayed in the SAP GUI.

```
Plm.clh.checktable.addfields.<characteristic name> = <list of table columns separated by
;>
```

The `<list of table columns separated by ;>` specifies the SAP table columns that are displayed in the value help dialog. The characteristic `<characteristic name>` defines the language-neutral characteristic name. Do not specify the column of the key value in the preference variable.

Note

Restriction: This option is available only if the check module returns values from ONE table. The table must only have one key value. Field and column names, as well as their description, must be unique. A data element must be defined for every table column.

11.4 Handling of Non-Visible Objects in Rel. to their Structures

The following preference variable changes the authorization check from the NextGeneration SAP logic to the previous logic:

```
SAP.PLM.AUTH.CHECK = DSC / SAP
```

11.5 Displaying Download Authorization for Original File

In SAP ECTR, you can set a preference variable so that the user sees the download authorization in the level-1 data. The following indicates whether download authorization for the original file exists:

```
sap.plm.acm.actvt_download = true / false (Standard)
```

12.1 Advanced Error GUI

The `Advanced Error GUI` displays the errors that have occurred as a list. If it is not possible to display all errors at once, the list can be scrolled. If an SAP error occurs, the long text from the SAP system is also displayed. Error messages of the type `An error has occurred in module x` are not output; the module is appended to the trace instead.

12.2 Setting up Speech Bubbles for the Process Display

You make the configuration for displaying the running processes in the application in speech bubbles in the task bar in the `default.txt` file using the following preference variable:

```
plm.sess.comm.writeBusyFile = true
```

To deactivate the speech bubbles, set the variable to `false`.

13 ATTRIBUTE TRANSFER

An attribute transfer is an exchange of data between SAP PLM, SAP ECTR and applications.

All data available in SAP PLM can be transferred to SAP Engineering Control Center or an application to which it is linked. This data – hereafter referred to as attributes – can be manipulated and written back to SAP PLM.

Using either the central or the application-specific `DType.xml` at the DType level you can apply the following settings using the attribute `<primary_application>`:

Settings	Description
`update_attributes_for_ro=<yes>` or `<no>`	If the document is opened for viewing, an attribute update is carried out. `yes`: The attribute is updated `no`: The attribute is not updated (default)
`update_attributes_for_rw=<yes>` or `<no>`	If the document is opened for editing, an attribute update is carried out. `yes`: The attribute is updated (default) `no`: The attribute is not updated
`update_attributes_for_dirty=<yes>` or `<no>`	If a document has a dirty flag, an attribute update is carried out. `yes`: The attribute is updated (default) `no`: The attribute is not updated

Example

```
<primary_application  primary_workstation_application="UGI"
                      attribute_section="STDMOD"
                      update_attribute_for_dirty="yes"
                      update_attributes_for_rw="yes"
                      update_attributes_for_ro="no"
</primary_application>
```

Caution

The `update_attributes_for_ro` setting works only with CAD integrations that allow read-only attributes to be updated. For more information, refer to the documentation of the relevant integration. This setting may result in performance problems or long loading times.

Customizing the Attribute Mapping

Attribute mapping is used to automatically supply applications with SAP attributes. The files `attributes-from-sap.xml` and `attributes-to-sap.xml` define how values are transferred to applications from the document, SAP classification and material. Attributes can also be transferred directly from applications to SAP PLM.

A detailed description of setting options for attribute mapping is available in the Operations Guide for SAP ECTR.

Calling Function Module CAD_GET_TITLE_BLOCK_DATA Using from_sap

Alternatively, attributes can also be transferred directly from SAP PLM to applications. For this, the function module `CAD_GET_TITLE_BLOCK_DATA` must be called using `from_sap`. See section 13.2 ABAP User Exit.

You can set a default response and define document-dependent exceptions.

The settings are established in a two-step procedure with the following preference variables:

- `sap.plm.attributesmapping.CAD_GET_TITLE_BLOCK_DATA = true/false` (default)
- `sap.plm.attributesmapping.CAD_GET_TITLE_BLOCK_DATA.<DOKAR> = true/false` (default)

13.1 Configuration Files

Defining the Settings File

For each document type (DType), the attributes which are transferred are defined.

The definition is stored in the settings file `dtype.xml` in the `<primary_application attribute_section="MASTERMODEL"...>` section. The definition references a section in the attribute mapping file according to whether the values are being transferred from the SAP system (`from-SAP`) or to the SAP system (`to-SAP`):

- `%PLM_INSTDIR%\customize\config\attributes-from-sap.xml`

  or
- `%PLM_INSTDIR%\customize\config\attributes-to-sap.xml`

Example

Each individual block in the file defines a value to be transferred. A block has the following structure:

```
<APPL_ATTRIBUTE name="SAP_DOCID" hidden="false">
  <CUTLEADING character="0">
    <DIR field="Documentnumber"/>
  </CUTLEADING>
</APPL_ATTRIBUTE>
```

The program processes the block starting from the inside.

In the example above, the program first reads the field `document number` from the document. The value from this field is processed further by the `CUTLEADING` function. The value which results is then stored in the application attribute (part attribute) with the name `SAP_DOCID`.

Configuring Customer Data in the Object Browser

When defining a DType, you can enter an attribute section from the file `attributes_from_sap.xml`.

The Object Browser uses this section and shows the attribute defined there in the `Customer Data` tab.

Define the names of the attributes displayed in the Object Browser via the dictionary keys:

```
plm.attr.<ATTRIBUTE> = <Text>
```

13.2 ABAP User Exit

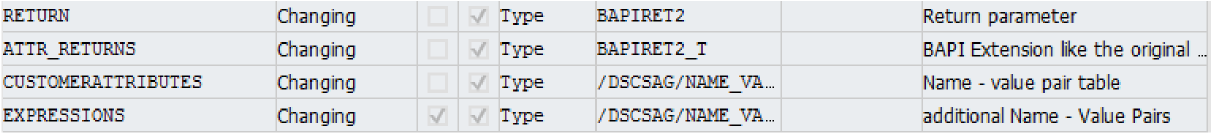
You can also populate the attributes directly from the SAP system using a user exit (Business Add-In, abbr.: BAdI). Attributes populated using the user exit can no longer be manipulated by the relevant application.

Procedure

1. Call up transaction SE18 in the SAP system.
2. In the entry field BAdI Name enter interface /DSCSAG/ENH_DOC.
3. Search for active implementations or create new ones.
4. Choose method DOC_GET_ATTRIBS_AFTER.

Example

```
DATA: ls_attribute TYPE /dscsag/doc_customer_attribs.
CLEAR ls_attribute.
ls_attribute-name = 'SAP_DOCID'.
MOVE documentdata-description TO ls_attribute-value.
APPEND ls_attribute TO customerattributes.
```



RETURN	Changing	□	✓	Type	BAPIRET2		Return parameter
ATTR_RETURNS	Changing	□	✓	Type	BAPIRET2_T		BAPI Extension like the original ...
CUSTOMERATTRIBUTES	Changing	□	✓	Type	/DSCSAG/NAME_VA...		Name - value pair table
EXPRESSIONS	Changing	✓	✓	Type	/DSCSAG/NAME_VA...		additional Name - Value Pairs

Description of a BAdI

The description is written into application attribute SAP_DOCID. All ABAP options are available.

13.3 Creating Customer-Specific Attributes

You can map the fields of a material to the BAPI structure in a table. In the Select Object Data function, you can select fields for displaying and changing. Data can only be changed from SAP ECTR if a mapping exists for a field. Otherwise, it can only be displayed.

Note the special procedure for customer-specific fields.

Creating Customer-Specific Attributes

This describes how customer-specific fields are created and can be updated with BAPI modules.

First, you have to enhance the existing structures (BAPI_TE_MARA and BAPI_TE_MARAX) with customer-specific fields. In the following example, these are the fields:

- ZZCADSYSTEM of type /DSCSAG/CAD_SYSTEM in table BAPI_TE_MARA,
- ZZCADSYSTEM of type BAPIUPDATE in table BAPI_TE_MARAX.

They must have the type CHAR or a type that can be converted to CHAR. The fields (ZZCADSYSTEM of type /DSCSAG/CAD_SYSTEM) must enhance the MARA structure. Following that, these fields must be assigned to a field group. If the fields are not assigned, they are not updated using BAPIs and function module /DSCSAG/ADV_MAT_SETALL returns a warning.

13.3.1 Enhancing Structures

BAPI_TE_MARA

Create ZZCADSYSTEM of type /DSCSAG/CAD_SYSTEM.

- Transaction: SE11

BAPI_TE_MARAX:

Create `ZZCADSYSTEM` of type `BAPIUPDATE`.

- Transaction: `SE11`

13.3.2 Assigning Fields to a Group of Fields

You can assign fields to a group of fields in transaction `OMSR`. Field group numbers 211 - 240 are reserved for customers.

1. Check whether fields have been assigned to the field group.
    - Transaction: `OMS9`
2. Create a new field data record.
3. Choose transaction `OMSR` and then choose `New Entries`.
4. Enter the required data.
5. Maintain the name and the status.
6. Add the field to a field group.

13.3.3 Creating an Entry in Table /DSCSAG/TBAPIFL

Create an entry in table `/DSCSAG/TBAPIFL`.

The data is then selected for changing in report `/DSCSAG/SEL_FIELDS` - transaction `/DSCSAG/SEL_FIELDS`.

13.3.4 Setting Fields in the "Pseudo Material Class"

Once you have assigned the new field to the field group, the field must be created as a characteristic in the material pseudo class. You can then change the field.

13.4 Function for Transferring Values from Document to Material

To reduce the maintenance effort required between a document and its linked material, a function exists that can be used to transfer attributes from a document to its linked material.

To do this, it is only necessary to create a source section and a target section in the `attribute-from-sap.xml` file and the `attribute-to-sap.xml` file.

The function `fnc.doc.metadata.transfer(FROM-Section,TO-Section)` with the names of the source and target sections must be entered in `menu.guidef`.

The function imports values from the selected document according to the configured source section.

If configured attributes from the source and target sections match, the values are overwritten in the linked material.

The following example shows how the function is used to transfer the description of a document to the linked material.

Example

The entries in **attributes-from-sap.xml** would be:

```
<section name=“TransferSource“ description=“Metadata transfer“>
       <APPL_ATTRIBUTE name=”SAP_DESCRIPTION” hidden=”false”>
             <DIR field=”Description”/>
       </APPL_ATTRIBUTE>
 </section>
```

The entries in **attributes-to-sap.xml** would be:

```
<section name=“TransferTarget“ description=“Metadata transfer“>
```

```
        <MA field=”MAKTI-MAKTX”>
                <APPL_ATTRIBUTE name=”SAP_DESCRIPTION” hidden=”false”/>
        </MA>
</section>
```

The corresponding configuration in `menu.guidef` would be:

```
? DOC_GENERAL = fnc.doc.metadata.transfer(TransferSource, TransferTarget)
```

(Please do not copy the last entry directly into `menu.guidef`, since tabs are not allowed here).

14 VE CLIENT CONVERTER

VE Client Converter is a program used by SAP ECTR for performing conversions from CAD file formats into SAP 3D Visual Enterprise (VE) visualization file formats. It also supports the tagging of viewable elements with additional data, using the VE metadata and identifier features.

When using conversion, the VE Client Converter produces a single monolithic VE visualization file as the output.

Note

https://wiki.scn.sap.com/wiki/display/PLM/How+to+replace+VE+Client+Converter+with+latest+version

Usage Restriction

Please note the SAP Note: 2386992.

Usage of VE Client Converter is only permitted through SAP ECTR; separate usage outside of SAP ECTR is not permitted.

15 GLOSSARY

Term	Definition
BAdI	Business add-in A type of user exit in the SAP ABAP source code. This can be used to apply specific rules and thus influence how SAP PLM integration for NX responds in the process.
BAPI	Business Application Programming Interface ABAP function module in SAP, which can be called by external programs (Java / C / C++).
document type	The document type groups documents according to typical characteristics as well as the resulting organizational flows.
document	A document refers to a material in the SAP system, which saves all business information for a document. While the document contains the metadata for a document (e.g. storage location), the original (e.g. design drawing) contains the actual document information.
DType	In SAP Engineering Control Center, you can define document types (DType), which categorize the documents to be created in various types. For example, you can define a document type for which you may create a material and one for which you may not do so. You can also specify different seed parts for each Dtype.

www.sap.com/contactsap

THE BEST RUN SAP®