Digital Personnel File

The **digital personnel file** is the most important use case of the DM module: every employee gets an **identical, predefined folder structure** per client in a document repository (usually Alfresco). The documents are visible in HR-Expert under the person's **"Documents"** tab and can additionally be exposed to end users through configurable **partial views** (action dm_folderExplorer) – for example a read-only view of one's own payslips.

Architecture overview

Four building blocks work together:

- **Repository** (DmRepository): the physical storage location. Two types: **DB** (storage in the Webdesk database) or **CMIS_ALFRESCO** (Alfresco via CMIS). A repository belongs to a client (or is client-less on single-client systems) and has an immutable *root path* – changing it later would make existing documents invisible.
- **Metadata structure** (DmMetaStructure): the *template* of the folder structure. It is bound to an **entity class** (for the personnel file: HrPerson) and consists of a tree of **meta folders**.
- **Meta folder** (DmMetaFolder): one node of the template with name, description, **folder name macro** (Velocity) and permissions. The macro of the root meta folder produces the person-specific folder name, e.g. ${HrPerson.getPerson().getLastName()}_${HrPerson.getPerson().getFirstName()}. Child meta folders usually have fixed names ("Contracts", "Payroll", "Miscellaneous", ...).
- **Mapping** (DmRepositoryMetaStructure): connects, per client, exactly one metadata structure with one repository and defines the **parent folder path** inside the repository.

The full path of an employee folder is therefore:
repository root path + mapping parent folder path + macro result of the root meta folder + fixed subfolders

The person # folder association is not stored as a path but as a persistent **entity link**. Every generated folder also remembers its originating meta folder – this is how the system recognizes **system folders** (generated from the template, neither renamable nor deletable by users) and distinguishes them from freely created **user folders** below them.

Setup step by step

1. Create a repository

Under **Repositories** (dm_showRepositories / dm_editRepository): choose name, client and type. For **CMIS_ALFRESCO**:

- **Use default connection**: uses the centrally stored Alfresco connection (URL, user, password from the DM options); alternatively per-repository connection data (the password is stored encrypted).
- **Site**: select an existing Alfresco site or create a new one directly from the form (short name, title, visibility).
- **Additional root path**: optional subpath within the site.
- **Test connection** verifies reachability before saving.

2. Define the metadata structure

Under **Edit metadata structure** (dm_editMetaDataStructure):

- Choose the **entity class** – for the personnel file HrPerson. This binding cannot be changed afterwards.
- Build the **folder tree**; set the **folder name macro** per meta folder (empty = fixed name). Inside the macro the entity is available under its simple class name. The system rejects macros that would produce the same folder name for two persons.

- Define **permissions** per meta folder (see below).
- **Map** the structure to a repository (specifying the parent folder path) and optionally **propagate** directly from the editor (asynchronously, with progress indicator).

For new installations, the **client setup wizard** can create the Alfresco site, the repository, the mapping and the initial propagation in one pass.

3. Generate folder structures (propagation)

Employee folders come into existence through three complementary mechanisms:

1. **Automatically on creation** of a new employee in HR-Expert. If the repository is unreachable at that moment, saving the person does *not* fail – only a warning is logged; the structure is created later.
2. **Lazily on first access**: when the Documents tab is opened in HR-Expert – and since version 4.64 also when a dm_folderExplorer configuration is opened – the structure is created on demand, and an incomplete structure is repaired automatically.
3. **In bulk via the job** propagateMetaStructure (or via the button in the structure editor): enforces the template for all entities of the mapped clients. Propagation creates missing folders, renames folders when the macro result has changed (e.g. after a person's name change), moves folders when the template hierarchy changed, and removes folders of deleted meta folders – the latter only if they no longer contain documents.

The Documents tab in HR-Expert

The **Documents** tab on the person (available with licensed DM and WF modules) shows the folder tree on the left and the document table on the right:

- **Folder tree**: system folders are fixed; below them, user folders can be created, renamed (F2) and deleted.
- **Documents**: upload via button or drag & drop (maximum size configurable), inline rename, move to other folders via dialog, delete, download by clicking the name. Uploading a document with an existing name updates the existing file after confirmation.
- **Document from template**: creates a new document from a document template (mail-merge fields are filled from the person data).
- **Digital signature**: for PDF/DOCX a signing workflow can be started (when the option is enabled); DOCX is converted to PDF for this.
- **Documents on sub-sections**: sub-sections of the person (salary, education, trainings, certificates, termination, ...) have their own "Documents" buttons; these documents are associated with the respective record via entity links.

When a person is deleted or historicized, the personnel file folder is handled as well: soft-delete (rename with a deletion suffix) on historicization, hard-delete on final deletion – unless prevented by the DM option "keep DM data when hard-deleting a person".

Partial views for end users (dm_folderExplorer)

The configurable action dm_folderExplorer makes portions of the personnel file accessible in the Webdesk portal. There are two configuration types:

- **Repository**: shows a fixed folder path of a repository (with hidable subfolders). Suitable for shared storage areas, not for person-specific views.
- **Metadata structure**: the personnel file case. You configure:
  - The **metadata structure** and, as root, one of its **meta folders** – e.g. the "Payroll" meta folder for a payslips-only view.

- The **entity creation script**: a JavaScript that maps the logged-in user to the personnel file entity. For the standard HrPerson case: appCtx.getBean('HrPersonService').getPerson(entity) (the context provides entity = logged-in PoPerson, currentUser and appCtx).
- **Hidden meta folders**: subfolders of the chosen root that should not appear in this view.

In addition there are the general options **Read only** (recommended for self-service views), **Display folder tree by default** and **Default page size**. The action offers breadcrumbs, upload (unless read-only), version management, display of linked entities and ZIP download of entire folders (folder permissions are respected).

For every configuration: the user sees their *own* portion – at runtime the root is resolved via the script and the metadata structure to the concrete folder of the user's own person, and created on demand if necessary.

Payslip distribution (PS integration)

How do payslips get into the personnel file? Through the PS module (action ps_editSalaryAccounting) combined with the DM module's **split-PDF configuration**:

1. Payroll delivers a collective PDF with all statements of a month. It is uploaded in the salary accounting form; a meta folder is chosen as target (default: **"Lohn- und Gehaltsverrechnung"** / payroll).
2. A **split-PDF configuration** (dm_editSplitPdfConfiguration) describes how the collective PDF is split: start/end detection texts, a **regular expression** extracting the employee ID per page, and a Velocity expression for the file name of the individual documents (with documentIdentifier and referenceDate).
3. During the **split**, each extracted employee ID (including client prefix and leading-zero handling) is resolved to a person; parts that cannot be assigned are logged as warnings.
4. A workflow step then copies the individual documents into the respective employee folders. Optionally a subfolder per year is created (pattern configurable, default: Lohnzettel $year).

Combined with a read-only dm_folderExplorer partial view on the payroll meta folder, the circle closes: upload the statement # split # distribute # each employee sees only their own payslips.

Permissions

Two layers work together:

Folder permissions (meta folder level)

- Per meta folder: a **default permission** plus optional **role permissions** (role # permission). Values: **FULL** (edit), **VIEW** (read only), **NONE** (invisible).
- Resolution walks from the folder up to the root meta folder; when several rules apply, the **most restrictive** one wins (NONE over VIEW over FULL).
- The **owner** of the personnel file (the person themselves) always has FULL on their folders; folders without a meta folder origin (user folders) are generally FULL.
- This covers use cases such as "the role Training only sees the categories Personnel Development and Training": set NONE as role permission for that role on the other meta folders, or restrictive default permissions.
- Permissions apply at **folder level**, not per document.

Document URL authorization

Every document download additionally passes a central check: the user's action permission, the folder permission (see above), and – for documents associated via entity links – module-specific **authorization resolvers** (e.g. for HR person documents, salary statements, travel expenses, workflow attachments). Without a matching rule, access is **denied**. This check is controlled by the DM option "document authorization enabled".

Operations and troubleshooting

- **Missing folders for individual persons** (e.g. "folder Miscellaneous missing"): run the propagateMetaStructure job for the affected mapping – it adds missing folders without disturbing the structure. Since version 4.64 the dm_folderExplorer also repairs incomplete structures on access.
- **Integrating pre-existing Alfresco folders**: the bindFoldersToMetaStructure job links existing folder trees to a metadata structure (initially or repeatedly).
- **Legacy data migration**: the convertAlfrescoObjectsToDmObjects job converts generic Alfresco folders/documents into DM-managed objects (with test and statistics functions).
- **Retention periods**: the deleteDocumentsFromMetaFolders job deletes documents of a meta folder older than X years/months, logging to the deletion log.
- **Name changes** of employees only reach the folder names at the next propagation.
- **Repository outage**: creating persons keeps working (warning in the log); the Documents tab and the explorer report the error on access.

Related pages: Document Management Module[1] (overview).

1. /daisy/personalwolke-admin/11522-dsy.html?language=4