SOLUTION GUIDE · RELEASE 2.0

Turnkey Report and Extension Library

Every PRO.FILE PLM extension and SSRS report we deliver — what each one is for, how your people use it, and what an administrator configures.

- **Ready to deploy.** Every item ships as one self-contained package with a single installer, its own documentation and its own changelog.
- **Configured to your site.** Numbering conventions, status names, field locations and user groups are settings, not code — and they survive upgrades.
- **Version-tracked.** Each item carries its own version and can be upgraded on its own, with no library-wide migration to schedule.

FOR PRO.FILE 10.4 AND 10.5

16	8	2	1
EXTENSIONS	REPORTS	PLATFORM LINES	INSTALLER PER ITEM

Cre8tive Technology and Design · Turnkey Library for PRO.FILE PLM

August 2026

About this library

The Turnkey Library is a set of ready-made additions to PRO.FILE PLM. Each one closes a specific gap between what the product does out of the box and what an engineering organisation actually needs day to day — and each one is delivered as a single, self-installing package.

THE TWO HALVES OF THE LIBRARY

Extensions change how PRO.FILE behaves. They run inside your system and act automatically as people work — allocating a part number the moment a part is created, moving a superseded drawing out of the way when its replacement is released, keeping a unit of measure in step across a bill of materials.

Almost all of them are invisible in daily use. There is no button to press and no window to close; the right thing simply happens as somebody saves their work.

Reports present what is already in PRO.FILE. They are SQL Server Reporting Services reports published to your report server, run on demand, and printed or exported like any other report.

They cover the two questions engineering asks most often: *what changed?* — the structure comparisons — and *what change work is in flight?* — the change-management formats and dashboard.

WHERE AN EXTENSION RUNS

Every extension in this guide is labelled with where its work happens. It matters because it determines which machine the package is installed on.

LABEL	RUNS ON	WHAT THAT MEANS FOR YOU
SERVER	The PRO.FILE AppServer	Acts for everyone, through every client, including work arriving from an integration. Installed once per config point.
DESKTOP CLIENT	Each user's PRO.FILE Desktop Client	Acts on the form in front of the user — validating as they type, or offering a dialog. Installed on the client machines that need it.

TWO PLATFORM LINES, ONE FEATURE SET

Every extension is maintained as two independent builds — one for **PRO.FILE 10.4** and one for **PRO.FILE 10.5**. The version numbers are kept aligned, so a given version means the same behaviour on both platforms, and each package states its platform in its own filename:

`Cre8tivePartNumberGenerator-2.0.5_PLM10.4_RELEASE.zip`

Take the package that matches the platform your server runs. Reports are not split this way — a report definition has no platform dependency, so there is one package for every installation.

> **CONFIGURED, NOT CUSTOMISED**
>
> Nothing in this library is hard-coded to one customer. Field names, numbering conventions, status names, document types and user groups are all configuration — held in a plain text file per extension, editable by your PRO.FILE administrator, and preserved across upgrades. Several extensions ship deliberately inert, doing nothing at all until somebody writes the rules for your site. That is the safe default, not a fault.

What is in the library

Twenty-four items in total. Every one is version-tracked, change-logged and ships with its own user, administrator and installation guides.

EXTENSIONS — NUMBERING AND IDENTITY

Cre8tiveDocNumberGenerator	Automatic document numbers, with a different convention per document type
Cre8tivePartNumberGenerator	Automatic part numbers, with a different convention per classification
Cre8tiveProcessNumberGenerator	Automatic process numbers
Cre8tiveProjectNumberGenerator	Automatic project numbers
Cre8tivePartNumberRegex	Checks a hand-typed part number on the form, before it can be saved
Cre8tiveDuplicateDefaults	Clears identity fields on a copy, so no two records claim one number

EXTENSIONS — REVISIONS AND STATUS

Cre8tiveCustomRev	Allocates the next revision code on a new version of a released item
Cre8tivePrevRevStatusChange	Supersedes the earlier revision when a new one is released
Cre8tiveAutoDocStatusChange	A part's status change carries its documents with it
Cre8tiveDynamicStartStatusByGroup	New documents start in a status that reflects who created them
Cre8tiveSetRestrictedStatus	A document's status follows its restriction flag, and back again

EXTENSIONS — STRUCTURE AND DATA QUALITY

Cre8tiveBoOandMaterialLinking	Turns two fields on a new part into a BOM line and a routing
Cre8tiveEUOM	Keeps BOM units of measure in step with the component part

EXTENSIONS — PROCESS AND WORKFLOW

Cre8tiveProcessInitiator	Whoever starts a process is cast into it and given its first task
Cre8tiveAssignProcessRoles	Staff a whole process, and its sub-processes, from one dialog
Cre8tiveAssignProcessRolesExt	Server companion that supplies that dialog with its data

REPORTS — STRUCTURE COMPARISON

BOMCompareLatest2Revisions	What changed between a part's last two revisions
ProfileBoMCompare	Structure comparison of any two items you choose

REPORTS — CHANGE MANAGEMENT

ChangeManagementDoc	Prints any change document, working out the right format itself
ProblemReportDocFormat	Printable Problem Report
ChangeRequestDocFormat	Printable Change Request, with the problems behind it
ChangeNoticeDocFormat	Printable Change Notice, with the requests behind it
EngineeringChangeDashboard	All change work in flight, on one page

REPORTS — SYSTEM

ProfileAnalytics	System size, growth, storage and licence consumption

01

Extension Library

Sixteen additions to how PRO.FILE behaves. Each one is described here in the same shape: what it is for, what your people do differently because of it, and what an administrator sets up to make it fit your site.

ONE CONVENTION WORTH KNOWING FIRST

Four of these extensions allocate numbers — for documents, parts, processes and projects. All four follow the same rule, so learning it once covers all of them: **a number is generated only when the field is empty, or still holds an AUTO GENERATED placeholder** — `{AUTO GENERATED}`, `[AUTO GENERATED]` or `(AUTO GENERATED)`. Case and spacing do not matter. Anything else in the field is treated as a number a person deliberately entered, and is never overwritten — which is what makes it safe to carry legacy numbers across from an old system.

Numbering and identity

Making sure everything that enters PRO.FILE arrives with a correct, unique identity — without anyone having to look up what the last number was.

Cre8tiveDocNumberGenerator

SERVER 2.0.5

Fills in the document number when a document is created or duplicated, following your site's convention — and can rename the stored file to match.

IN USE

Create a document as normal and leave the number field empty. The number appears on save, filled in and unique. Duplicating a document gives the copy its *own* new number rather than the original's, because a duplicate is a new document and needs its own identity.

NOTABLE

This is the only numbering extension that can apply a **different convention to different documents** — specifications can be numbered one way and everything else another.

CONFIGURED

`Cre8tiveDocNumberGenerator.cfg – [DOCUMENT_NUMBERING]`

`DOC_NUM`	Which convention applies to which document — by field value, by document type, or to all. Rows are tried in order and the first match wins, so the catch-all goes last.
`DOC_NUM1…n`	How each convention builds its number. Every row contributes one piece, joined in order: a literal string, a field from the document, or a counter. Formatting options include zero-padding, dates and initials.
`RENAME_FILE`	Rename the stored file to match the number. Ships `0` — it is a visible change for users, so turn it on deliberately.
`ORIGINAL_NAME_FIELD`	Where the file's original name is preserved when renaming is on.

Ships working out of the box, producing numbers like `DOC-000123`.

Cre8tivePartNumberGenerator

SERVER 2.0.5

Fills in the part number when a part is created or duplicated, following your site's convention.

IN USE Create a part as normal and leave the number field empty. The number appears on save. As with documents, a duplicate is given its own new number.

NOTABLE Conventions can be selected by the part's **classification** or by a field on the part — so purchased parts can be numbered differently from manufactured ones.

CONFIGURED `Cre8tivePartNumberGenerator.cfg – [PART_NUMBERING]`

`PART_NUM`	Which convention applies to which part — by classification or by field value. First match wins.
`PART_NUM1…n`	How each convention builds its number: literal text, a field from the part, or a counter, each optionally formatted and joined in order.

Ships with one convention applied to every part, so it works on a clean install.

Cre8tiveProcessNumberGenerator

SERVER 2.0.6

Fills in the process number the moment a process is created.

IN USE Start a process as normal, leaving the number field empty. The number is allocated on creation.

CONFIGURED `Cre8tiveProcessNumberGenerator.cfg – [PROCESS_NUMBERING]`

`PROCESS_NUM`	How the number is built — literal text, a field from the process, or a counter, each row formatted and joined in order.

Ships producing numbers like `PR-00001009`.

Cre8tiveProjectNumberGenerator

SERVER 2.0.5

Fills in the project number the moment a project is created.

IN USE Create a project as normal, leaving the number field empty.

CONFIGURED `Cre8tiveProjectNumberGenerator.cfg – [PROJECT_NUMBERING]`

`PROJECT_NUM`	How the number is built, in the same row-by-row way as the other generators.

Ships producing numbers like `PJ-00001001`.

Cre8tivePartNumberRegex

DESKTOP CLIENT 2.0.4

Checks a hand-typed part number while the user is still on the form, where a mistake is cheap to fix.

IN USE

Type a part number and it is checked as you leave the field, and again as you leave the form. Four things are caught: disallowed characters, a number that is too long, a number that does not start with an approved prefix, and a number another part already has. Leave the field empty instead and it fills itself with `{AUTO GENERATED}`, handing the job to the server.

CONFIGURED

`Cre8tivePartNumberRegex.cfg — [PART_REGEX]`

`REGEX`	The character rules, each with the message the user sees when it fails. Ships allowing uppercase letters, digits and `.` `-` `/` `\`.
`MAX_LENGTH`	Longest acceptable part number. Ships at `39`.
`PREFIX_LIST`	Approved prefixes a number must start with. Ships at `0` — the prefix check is off until you list some.

Cre8tiveDuplicateDefaults

DESKTOP CLIENT 2.0.2

Clears the fields that must not carry over when a document or part is copied — above all, its number.

IN USE

Duplicate as normal; the copy opens with the nominated fields already reset. Despite the name it also applies to new items added through the **check-in wizard**, on the same principle: a new item should not inherit an identity it has not earned.

WHY IT MATTERS

Without it a duplicate arrives carrying the original's part number. Two records claiming one number is cheap to prevent and expensive to unpick later.

CONFIGURED

`Cre8tiveDuplicateDefaults.cfg — [DUPLICATE_DEFAULTS]`

`DOCUMENT_DEFAULT`	Which document fields are reset, and to what.
`PART_DEFAULT`	The same for parts.

Ships setting the number to `{AUTO GENERATED}` and the revision back to `A`.

Revisions and status

Keeping lifecycle state honest: the right revision code, the previous one moved out of the way, and statuses that stay consistent across related records.

Cre8tiveCustomRev

SERVER 2.0.2

Works out the next revision code when a new version of a **released** document or part is created, and clears the fields that belong to the old revision.

IN USE

Create a new version as normal. The revision field arrives already filled in — `A` becomes `B`, `H` becomes `J`, `Z` becomes `AA` — and last revision's change note or ECO reference has been cleared rather than silently carried over.

NOTABLE

It acts **only on released items**. Revise something still in work and nothing happens: an item that has not been released has not earned a revision, so re-versioning it should not consume the next code. The ladder itself is held in the database and skips the letters that are easy to misread — `I`, `O`, `Q` and `V`.

CONFIGURED

`Cre8tiveCustomRev.cfg – [CUSTOM_REV]`

`DOC_REV_BLANK`	Document fields cleared on the new revision.
`PART_REV_BLANK`	Part fields cleared on the new revision.

Cre8tivePrevRevStatusChange

SERVER 2.0.3

When a new revision is released, moves the earlier revision on — to superseded, obsolete, or whatever your site calls it.

IN USE

Nothing to click. It acts in two situations: when a new version of a document is created, and when a document's status changes. Matching earlier revisions are moved as part of the same operation.

WHY IT MATTERS

Without it, releasing rev B leaves rev A still marked released. Two revisions both look current, and eventually somebody builds to the wrong one.

CONFIGURED

`Cre8tivePrevRevStatusChange.cfg – [PREV_REV_STATUS_CHANGE]`

`DOC_PREV`	One row per rule, and all three conditions must line up: the document *type*, the status the *new* revision has reached, and the status the *earlier* revision is currently in — plus the status to move it to.

Ships with no rules, and therefore does nothing until they are written. That third condition is the one that surprises people: a rule that supersedes a *Released* rev A will not touch a rev A that was still *In Work*.

Cre8tiveAutoDocStatusChange

SERVER 2.0.2

Cascades a part's status change down to the documents hanging off it, so the paperwork follows the part.

IN USE

Change a part's status the way you normally do — say from *In Work* to *Released* — and save. Matching child documents move at the same time, instead of being opened and released one by one.

NOTABLE

It runs **one way only**: a part can move its documents, a document never moves its part. And it acts on **direct children** only — documents hanging off a different part further down the structure are left alone.

CONFIGURED

`Cre8tiveAutoDocStatusChange.cfg – [AUTO_DOC_STATUS_CHANGE]`

`DOC_STATUS_CHANGE_TABLE`	One row per rule: the status the *part* moved to, the document *type*, the status the document must currently be in, and the status to move it to. All three conditions must match.

Ships with no rules and is inert until they are written.

Cre8tiveDynamicStartStatusByGroup

SERVER 2.0.2

Gives a new document a starting status that reflects who created it and what kind of document it is.

IN USE

Add or duplicate a document as normal. Its status is set for you. An engineer adding a drawing and a contractor adding the same kind of drawing can land in different starting statuses, without either having to remember to set it.

NOTABLE

New documents only — existing ones are never re-statused, and changing the configuration does not reach back. If nothing matches, the document simply keeps the system default; that is normal, not an error.

CONFIGURED

`Cre8tiveDynamicStartStatusByGroup.cfg – [DYNAMIC_START_STATUS_GROUP]`

`START_STATUS_GROUP`	The user groups that carry a rule, **in priority order**. The first listed group the creator belongs to is the one that applies; the rest are not considered.
`START_STATUS_GROUP1…n`	For that group, the starting status for each document type.

Cre8tiveSetRestrictedStatus

SERVER 2.0.2

Keeps a document's status in step with its restriction flag, swapping between paired ordinary and restricted statuses.

IN USE

Set the restriction field on a document — export-controlled, commercially sensitive, however your organisation frames it — and save. The status switches to the restricted twin. Clear the field and it switches back. This applies both to new documents and to existing ones being edited.

NOTABLE

The document moves **sideways**, not forwards: *In Review* becomes *In Review (Restricted)*, holding its place in the workflow while its restriction changes.

CONFIGURED

`Cre8tiveSetRestrictedStatus.cfg – [RESTRICTED_STATUS]`

`RESTRICTED_FIELD`	The field on the document that carries the restriction.
`RESTRICTED_VALUE`	The value in that field that means “restricted”. Ships as a placeholder that must be set before the extension does anything.
`RES_STATUS`	The status pairs — each ordinary status alongside its restricted twin.

Structure and data quality

Building structure from what people already type, and keeping related records from drifting apart.

Cre8tiveBoOandMaterialLinking

SERVER 2.0.2

Turns two fields filled in on a new part into real structure: a bill of materials line for the material, and a Bill of Operations for the routing.

IN USE

When creating a part, fill in the material field with a material part number and the quantity used, and list the operations in the operations field. On save the BOM line is linked and a Bill of Operations document is built, with one Operation document beneath it for each step.

WRITING THE LIST

Each operation is the operation code followed by its time in brackets. Separate them with commas or put each on its own line — both work:

```
OP1[2], OP2[5], OP3[1.5]
```

CONFIGURED

`Cre8tiveBoOandMaterialLinking.cfg – [MATERIAL_LINKING] / [BILL_OF_OPERATIONS]`

`MATERIAL_FIELD` `MATERIAL_QTY_FIELD`	Which fields on the part hold the material number and its quantity.
`OPERATION_FIELD`	Which field holds the operations list.
`OPERATION_DELIMITER` `OPERATION_END_CHAR`	The characters that wrap the time. Ship as `[` and `]`; your site may use others.
`OPERATION_CODE_FIELD` `OPERATION_TIME_FIELD_OP`	Where the code and the time are written on each Operation document.
`OPERATION_DOC_TYPE` `BOO_DOC_TYPE`	The document types used for an operation and for the Bill of Operations that holds them.

Cre8tiveEUOM

SERVER 2.0.2

Keeps a BOM line's engineering unit of measure in step with the inventory unit of the component part it points at.

IN USE

Nothing to click. Add a part to a bill of materials and the line's unit of measure is filled in from the part. Change that unit on the part later and every BOM line already using it is updated in the same operation.

WHY IT MATTERS

Without it a part stocked in metres can sit on a line measured in each, and the quantity means something different depending on which record you read.

NOTABLE

It acts only on parts classified as **PLM Component**; other kinds of part are deliberately left alone for you to set. If a unit is not being filled in, the part's classification is the first thing to check. It also reads the part and writes the line, never the reverse — editing a unit directly on a BOM line will be overwritten next time the part changes. Changing the unit on a widely used component can update a great many lines at once, so check before saving.

CONFIGURED

No configuration file. Field locations come from the shipped field-path configuration, which needs adjusting only if your site keeps these values somewhere non-standard.

Process and workflow

Reducing the administrative steps between starting a piece of work and the right people having it in their list.

Cre8tiveProcessInitiator

SERVER 2.0.4

Casts whoever starts a process into the Process Initiator role, and hands them the tasks that role is responsible for.

IN USE

Start a process from its template. That is all — you are cast into the role and its first task is already waiting for you, instead of your having to work through the wizard casting yourself before anything can happen.

NOTABLE

It happens **once, at start**; tasks created later in the process are not re-delegated, and processes already running are unaffected. Any existing casting of that one role is replaced — the role means "whoever started this run". Every other role is left exactly as configured, and a template without a Process Initiator role is simply unaffected.

CONFIGURED

`Cre8tiveProcessInitiator.cfg – [PROCESS_INITIATOR]`

`PROCESS_INITIATOR_ROLE`	The name of the role to cast. Ships as `Process Initiator`.
`PROCESS_INITIATOR_USER_ID`	Optional stand-in user account. Where a template assigns its first task to a placeholder "Process Initiator" user, that task is handed to the real starter instead — which lets the wizard skip asking them to cast the role at all.

Cre8tiveAssignProcessRoles

DESKTOP CLIENT 2.0.2

A single dialog for casting every process role at once — including the roles belonging to sub-processes that have not started yet.

IN USE

Select a started process, run the **Assign Process Roles** command, and staff the whole thing in one sitting: roles already cast can be reassigned, and roles not yet cast are gathered from the template and its sub-processes. Cancelling changes nothing. Normally each role is cast as the process reaches it, one wizard at a time; this replaces that with one pass.

WHO SEES IT

The command appears only when the user is looking at a process that has **started and not finished**, and is a member of the nominated user group. Not seeing it is a condition not being met, not an error.

NOTABLE

Where user groups named `PR_<role>` exist, their members are offered as suggestions for that role. It is a convenience, not a restriction — any enabled user can be assigned.

CONFIGURED

`Cre8tiveAssignProcessRoles.cfg – [ASSIGN_PROCESS_ROLES]`

`ASSIGN_GROUP_ID`	The user group allowed to cast roles. Ships at `0`, which is not a real group — so the command stays hidden for everyone until it is set. That is the safe default.
`COMMAND_ID`	The command id used to wire the dialog into the client's navigation. Ships at `0`, meaning “use the built-in default”.
`TASKS_ASSIGN_PR`	Task names that open the dialog automatically when reached.

REQUIRES

Cre8tiveAssignProcessRolesExt on the server. The two are a pair.

Cre8tiveAssignProcessRolesExt

SERVER 2.0.2

The server half of the pair above. It answers the dialog's two questions — which roles this process has, and which sub-processes hang off it.

IN USE

Nothing visible, by design. Users interact with the dialog in **Cre8tiveAssignProcessRoles**; this component supplies its content.

NOTABLE

The pair must be deployed together and kept on matching versions. With the client half alone the dialog opens but cannot fill itself in — typically an empty list of roles or sub-processes. With this half alone, nothing calls it and nothing happens.

CONFIGURED

No configuration. Its behaviour is entirely governed by the client half's settings.

02

Report Library

Eight SQL Server Reporting Services reports, published to your report server and run on demand like any other. Each entry gives what the report answers, where it is published, and the parameters you supply.

EVERY REPORT IS DRIVEN BY A PLMID

Except the dashboard and the analytics page, these reports identify their subject by **PLMID** — PRO.FILE's internal object id — and not by the document or part number a person would recognise. `CN-002144` is a document number and will not work. The PLMID comes from the PLM client, or from the Engineering Change Dashboard, which lists it alongside the work it describes.

Structure comparison

Two answers to *what changed?* — one that picks the two sides for you, one that lets you choose them.

BOMCompareLatest2Revisions

REPORT 2.0.2

Answers the most common change-review question — what changed between this part's last two revisions? — without making you identify either revision.

PUBLISHED Report server → **PARTS**

IN USE Supply one part. The report resolves its revision family itself and always compares the two most recent revisions, then heads the output with both so a printed copy always records which two it describes. You can pass *any* revision in the family, not only the latest.

REQUIRES The part must have **at least two revisions**. With only one there is nothing to compare and the report reports an error rather than an empty page.

PARAMETERS

PARAMETER	DEFAULT	MEANING
PARAM_PLMID	—	Any part in the revision family to compare.

ProfileBoMCompare

REPORT 2.0.2

The general-purpose structure diff: compares any two PLM items you choose and lists what was added, removed and changed.

PUBLISHED Report server → **COMMON** and **PARTS**. The same report in two places for convenience; either gives identical results.

IN USE Supply both sides. Differences are described as changes needed *to reach the left-hand side*, so putting the newer revision on the left makes “added” mean “added in the newer one”.

WATCH FOR Depth defaults to **1** — only the immediate children are compared, so a change three levels down does not appear until you raise it. This is the single most common surprise. Deep comparisons of large structures take noticeably longer, so raise it deliberately.

PARAMETERS

PARAMETER	DEFAULT	MEANING
PARAM_LHS_PDMID	—	Left-hand item — conventionally the newer.
PARAM_RHS_PDMID	—	Right-hand item — conventionally the older.
PARAM_MAX_DEPTH	1	How many structure levels to descend.
PARAM_DELTA_ONLY	0	`1` lists only the lines that differ.
PARAM_PARTS_ONLY	0	`1` ignores documents and compares parts only.

Change management

The engineering change chain — Problem Report (what is wrong) → Change Request (what we propose) → Change Notice (what we are doing) — in printable form, plus a dashboard over all of it.

ChangeManagementDoc

REPORT 2.0.2

One report that prints **any** change document. It works out whether the id you gave it is a Problem Report, a Change Request or a Change Notice, and renders the matching format.

PUBLISHED Report server → **DOCUMENTS**

IN USE This is the one to bookmark. Without it you would have to know which of the three specific reports to open before you could print anything. The output is identical to opening the right one directly — this report adds no content of its own, it only chooses.

REQUIRES The three format reports below must be published, since it renders through them.

PARAMETERS

PARAMETER	DEFAULT	MEANING
DOCID	—	PLMID of any change document.

ProblemReportDocFormat

REPORT 2.0.2

The printable Problem Report — a problem found against one or more parts, recorded before anyone has decided what to do about it.

PUBLISHED Report server → **DOCUMENTS**

ON THE PAGE **Header** — number, title, status, workflow state, originator, dates and the descriptive free-text fields, with multi-line descriptions laid out as readable blocks rather than truncated to one line. **Affected parts** — every part linked to the report, with number, revision, description and its own current status.

READING IT A part's status is its own lifecycle state, independent of the Problem Report's — a released part under an open PR is the normal case, not a contradiction.

PARAMETERS

PARAMETER	DEFAULT	MEANING
DOCPLMID	—	PLMID of the Problem Report.

ChangeRequestDocFormat

REPORT 2.0.2

The printable Change Request — the document that proposes a change and asks for approval, shown together with the problems that led to it.

PUBLISHED Report server → **DOCUMENTS**

ON THE PAGE **Header**, **affected parts**, and **associated Problem Reports**. That third section is what makes this more than a form print: it answers "why are we being asked to approve this?" without anyone having to go and look. The Problem Reports are found through the *parts* the request affects, not through a direct link.

PARAMETERS

PARAMETER	DEFAULT	MEANING
`DOCPLMID`	—	PLMID of the Change Request.

ChangeNoticeDocFormat

REPORT 2.0.2

The printable Change Notice — the document that implements an approved change, and the end of the change chain.

PUBLISHED Report server → **DOCUMENTS**

ON THE PAGE **Header**, **affected parts**, and the **Change Requests raised against each affected part's previous revision**.

WHY THE PREVIOUS REVISION Deliberate, and the one genuinely subtle thing here. A Change Notice creates a *new* revision of a part; the Change Request that asked for it was raised earlier, against the revision that existed at the time — the previous one. Looking there is what finds the request that caused the notice.

PARAMETERS

PARAMETER	DEFAULT	MEANING
`DOCPLMID`	—	PLMID of the Change Notice.

EngineeringChangeDashboard

REPORT 2.0.2

One screen answering “what change work is in flight right now, and which parts does it touch?” Intended as the change board's landing page.

PUBLISHED

Report server → **COMMON**

ON THE PAGE

Problem Reports, Change Notices and Change Requests each broken down by status — including transfer-failure states such as *PLM transfer failed*, usually the first place an ERP hand-off problem becomes visible — then the parts pulled into open Change Requests and the in-process parts from Change Notices, with number, revision, description and status.

WATCH FOR

The one parameter changes **which question the report answers**, not just how much comes back. `0` shows everything currently open, unfiltered by date. A value of *N* instead shows change activity whose status moved in the last *N* days — whether or not it is still open. So `30` is not “the last 30 days of the default view”; it is a different population, and work you expect to see may be absent simply because its status has not moved inside the window.

PARAMETERS

PARAMETER	DEFAULT	MEANING
daysback	0	`0` = all open work. *N* = activity in the last *N* days, open or closed.

System

A report about the PRO.FILE installation itself, rather than the products managed inside it.

ProfileAnalytics

REPORT 2.0.2

A one-page health and growth summary of a PRO.FILE installation — the questions that come up in a system review or a licence renewal conversation.

PUBLISHED

Report server → **COMMON**

ON THE PAGE

- **Object counts** — documents, parts, projects, processes, tasks.
- **Total storage** — the combined size of all managed files, in GB.
- **Storage by document type** — which types account for the space.
- **Documents and disc space by month** — the last twelve months, so the growth trend is visible.
- **Parts by month** — the same trend for parts.
- **Licence versus users** — full and limited licences held, against enabled user accounts.

PARAMETERS

PARAMETER	DEFAULT	MEANING
NUMBEROF	1	A noise filter on the storage-by-document-type chart *only*: a type appears only if it has more than this many documents. Raise it to suppress the long tail on a busy system. Every other panel ignores it.

03

How the library is delivered

Every item in this guide arrives the same way: one zip file, one thing to run, and its own documentation inside the package. This section is for whoever will install it.

INSTALL TO EVERY CONFIG POINT

An installation commonly has more than one config point. An extension present in only one of them **silently does nothing** for the users served by the other — no error, no warning, just an extension that appears not to work for half the business. Both install routes offer to cover them all; take that offer unless you have a specific reason not to.

What is in a package

Each release zip is self-contained and customer-deployable on its own. Nothing has to be fetched from anywhere else.

AN EXTENSION PACKAGE

- **The built components** and the descriptor that registers them.
- **A default configuration file**, ready to tune to your site.
- **Three guides** — User, Administrator and Installation.
- **The changelog** for that item.
- `DeploySolution.ps1` — the installer.
- `<Name>-Setup.exe` — the graphical alternative.

A REPORT PACKAGE

- **The report definition** and any images it uses.
- **A create script** for the SQL objects the report needs.
- **A check script** that reports which of those already exist on a target system.
- **Three guides** — User, Administrator and Installation.
- **The changelog** for that report.
- `DeploySolution.ps1` and a matching **Setup.exe**.

Two ways to install

Both routes produce an identical result, because the graphical installer does not deploy anything itself — it collects the parameters and then runs the same script.

THE INSTALLER WINDOW

1. Extract the zip on the target machine.
2. Double-click `<Name>-Setup.exe` and approve the elevation prompt.
3. It finds the config points on that server and shows you exactly what it is about to run.
4. Results stream into a log pane you can save.

The recommended route for a normal install. It asks for elevation up front rather than failing a check after the form has been filled in.

THE SCRIPT

1. Extract the zip to a local path — not a network share, and not inside the zip.
2. Unblock the files if the zip came from another machine.
3. Open PowerShell **as Administrator** in that folder.
4. Run `DeploySolution.ps1`.

The route for unattended and scripted installs, and the only one offering a dry run. It is copy-only, validated, idempotent, and safe to re-run.

Upgrading, and your configuration

Version numbers advance together across the library, and your tuning survives.

Installing a newer version over an older one is the supported upgrade path; the installer is safe to re-run. **Configuration files are never overwritten once they exist** — a file tuned to your site's numbering convention, status names and field locations stays tuned across upgrades, and new settings introduced by a release are documented in that item's changelog and Administrator Guide.

Every item carries its own version, its own changelog and its own release zip, so items can be upgraded independently — there is no library-wide upgrade to schedule, and taking a fix for one extension does not oblige you to move any other.

WHERE TO GO NEXT

Every package contains its own **User Guide**, **Administrator Guide** and **Installation Guide**. This document is the overview; those three are the detail — the full configuration reference, the processing sequence, target paths, verification steps and rollback. Your Cre8tive contact can supply any package in the library for the PRO.FILE platform your installation runs.

Turnkey Report and Extension Library

16 extensions · 8 reports · PRO.FILE 10.4 and 10.5 · Release 2.0 · August 2026