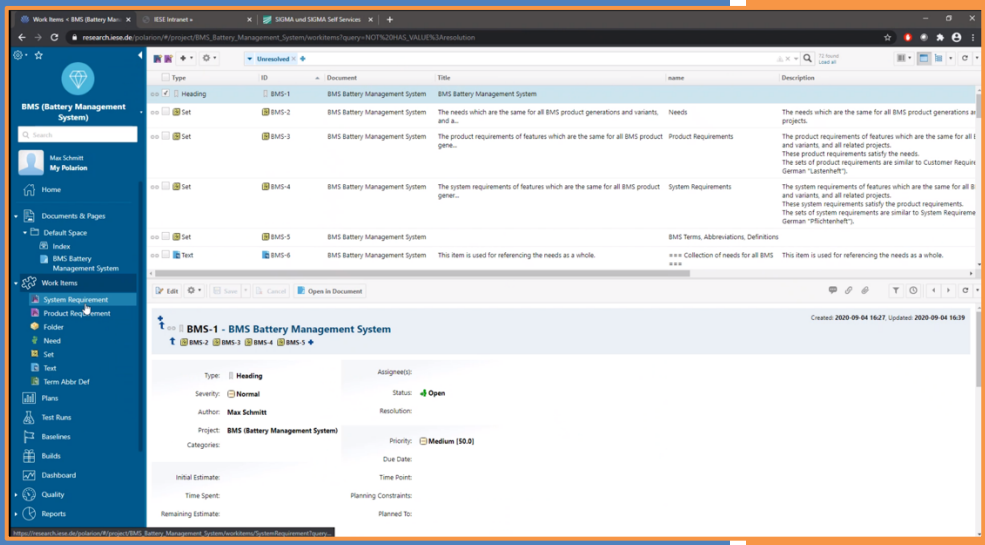
PaR

Processes as Requirements

for Polarion



Systematic

Software

Engineering

PaR – Processes as Requirements
for Polarion

Version: 31. May 2021

Publisher: Ralf Bürger
SSE – Systematic Software Engineering
Wilhelmstraße 24
45527 Hattingen
eMail@RalfBuerger.de

Print: PDF only

Community: https://ProcessesasRequirements.info

Acknowledgement: Thanks to Max Schmitt from Fraunhofer IESE
for providing most of the Polarion specific content.

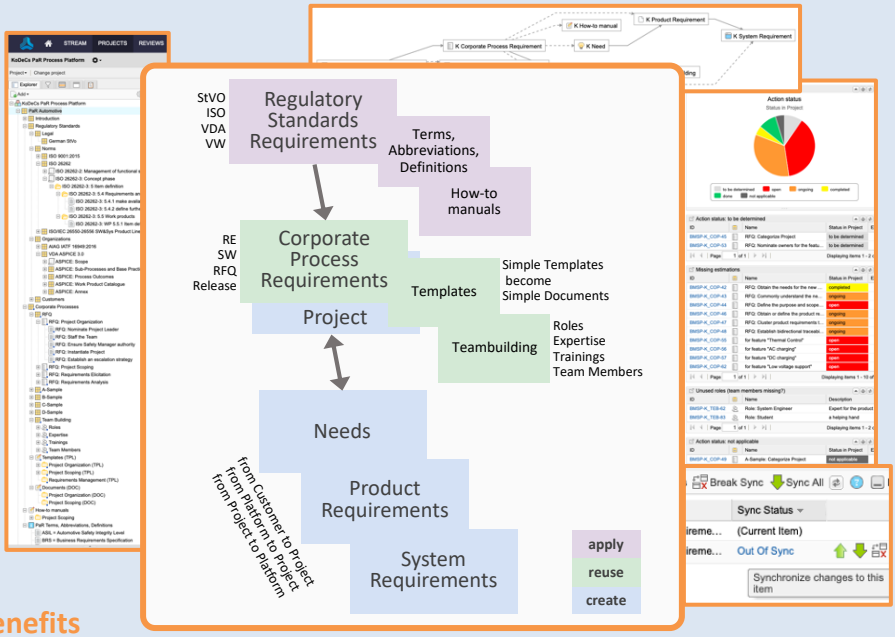
"First, don't be afraid. ...
Second, do what you think is right. ...
Finally, build a community.
No one does big things by themselves."
(President Obama, 18.May.2020)

PaR excellence

ProcessesasRequirements.info

your challenges / PaR solutions

1. / 1 ***help*** **all types of projects with flexible corporate development processes**
   PaR: define and reuse the processes as requirement sets in your RE tool
2. / 2 ***merge*** **regulatory standards with corporate development processes**
   PaR: define also the standards as requirements and add traceability
3. / 3 ***learn*** **with the project teams by established sustainable processes**
   PaR: unite process and product requirements, but improve both
4. / 4 ***comply*** **continuously with processes and standards in the projects**
   PaR: use the features of your RE tool for bi-directional traceability
5. / 5 ***monitor*** **actual project and product maturity progress**
   PaR: measure the status of all requirements, documents and reviews



benefits

① lightweight efficient processes are welcomed by developers
② uniting “what” and “how” in the teams’ tools is more agile
③ standards and processes are focusing on projects and learning culture
④ project teams are empowered to self-organize the compliance
⑤ step by step, teams apply flexible platform techniques also for processes
⑥ it’s a systematic holistic methodical framework that is easy to adopt and adapt
⑦ it makes true transparency for actual process and product maturity

Table of Contents

PARADE OF CHALLENGES SHOWING THE NEEDS 5

PARIS (PAR INFORMATION SYSTEM) 6

PARTOUT - TOOL FEATURES TO SATISFY THE NEEDS OF PAR 7

FEATURE 1: DEFINITION OF REQUIREMENT ITEM TYPES 7
FEATURE 2: IMPLEMENTATION OF THE PARIS MAP 9
FEATURE 3: EVALUATION OF PROJECT MATURITY 10
FEATURE 4: COMPLIANCE CHECKS BY STANDARDS COVERAGE 12
FEATURE 5: SUPPORT FOR PROCESS VERSIONS 13
FEATURE 6: REUSE OF REQUIREMENTS SETS 13
FEATURE 7: SYNCHRONIZATION OF REQUIREMENTS SETS 14
FEATURE 8: DEFINITION AND MANAGEMENT OF VARIABILITY 14

PARTIAL IMPORT, EXPORT, BACKUP 15

PaRade of challenges showing the needs

I have identified **5 challenges** from my coaching of large projects over the past few years. They all can be addressed by changing the way to deal with bulky standards and processes. Bringing these deeply into the projects makes them more intrinsic and natural.

From those challenges I derived needs for a methodical framework. These are outlined on **The Page** (see at the beginning of this document) that is also available as **The Slide**. The whole **PaR** framework is described in a nutshell in **The Booklet**. For more details **The Book** is available.

This methodical framework requires some tool features for realization in organizations. Most modern requirements management tools that are in use in those organizations have at least the basic features on board that are requested here. Nonetheless it is sometimes tricky to configure the tools correctly.

Implementing multiple regulatory standards and setting up a reusable corporate development project process is still a lot of work and needs to be configured correctly. Often it is good advice to get help, support, coaching and maybe also manpower for setup from the experts. **PaR** offers help from the community of experts as described on the website.

This document shows an exemplary PaR implementation for the tool Polarion©[1] created by Ralf Bürger (SSE).

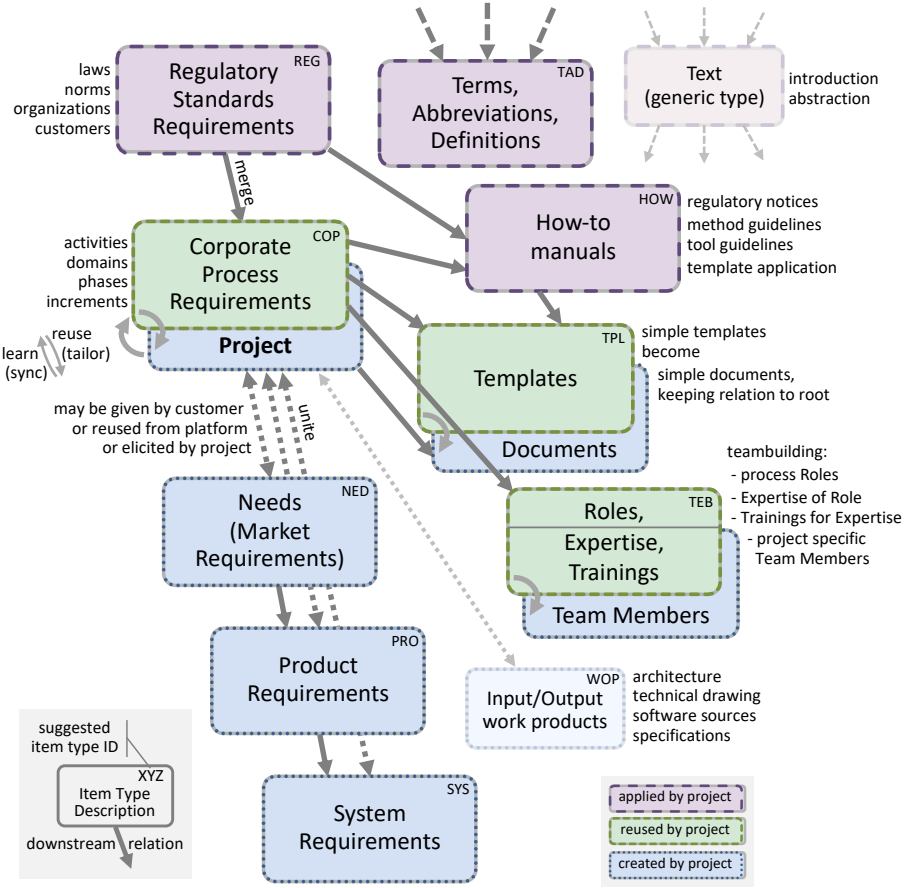
Thanks for the support from the **PaR** community members Max Schmitt and Martin Becker from Fraunhofer IESE.

[1] see https://polarion.plm.automation.siemens.com

PaRis (PaR information system)

This PaR information system satisfies the discovered needs. It can be implemented in tools by a corresponding set of requirements item types with an item type relationship model. The PaRis is explained in The Booklet and The Book. We show it here only for quick lookup.

- Some requirements are simply applied by projects without change, for detailed lookup or guiding help.
- Other requirements are rather inputs to be reused by projects, also to be modified or extended.
- Reused items, including certain work products, finally become items that are created by projects.



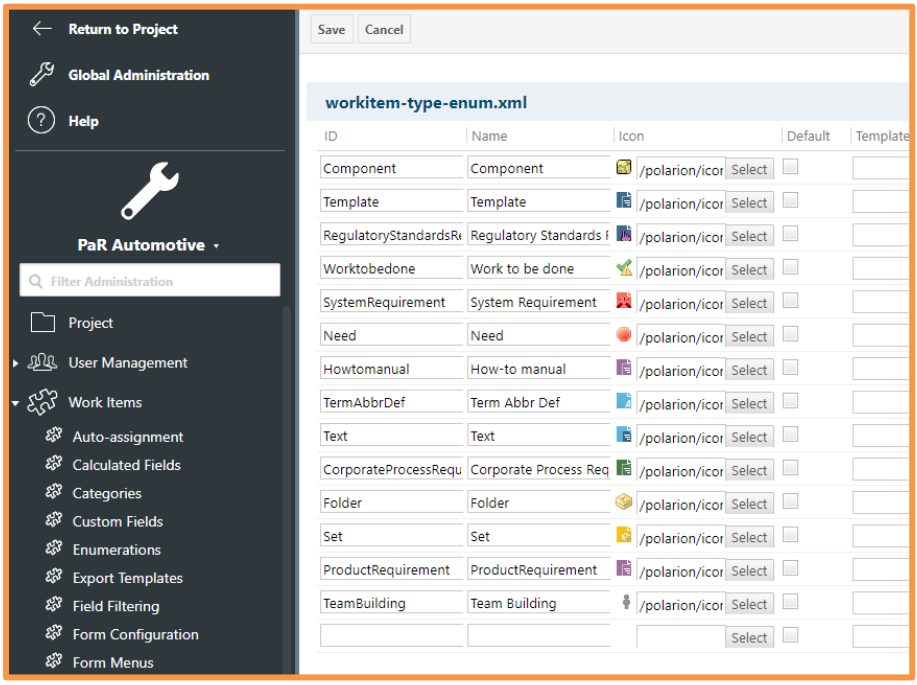
PaRtout - Tool Features to Satisfy the Needs of PaR

This set of features defines what a tool should bring in to be able to fully support the methodical framework with the described PaRis. These features are explained in The Booklet and in more detail in The Book.

Feature 1: Definition of requirement item types

Polarion differentiates between “Global Administration” and a project specific “Project Administration”.

In this Polarion implementation, for the process family “PaR Automotive”, we define the following item types in the project configuration according to the PaRis (shown in order of the time of their creation):



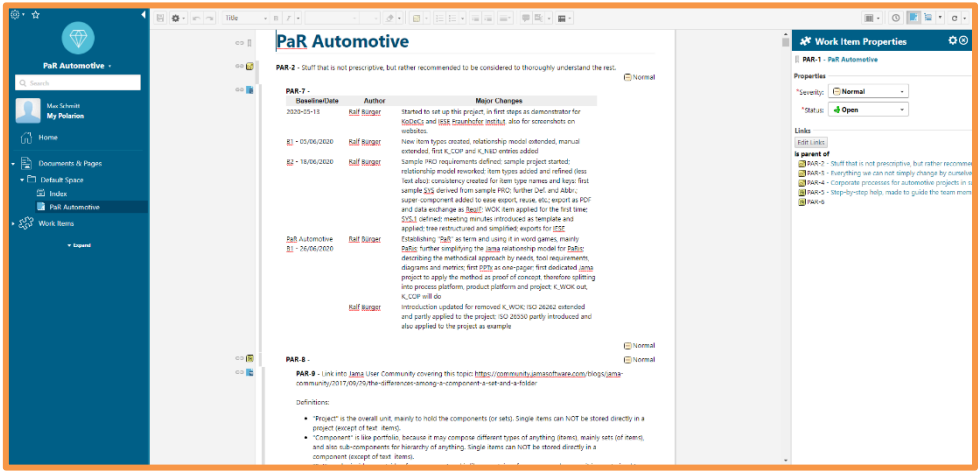
Each newly created item type has a set of default fields that are customized so that only the relevant attributes of each item type are available. When importing a project into Polarion, these custom fields are created semi-automatically during the import. The following screenshot

shows these custom field definitions from the item type “Regulatory Standards Requirement” as an example.

RegulatoryStandardsRequirement-custom-fields.xml

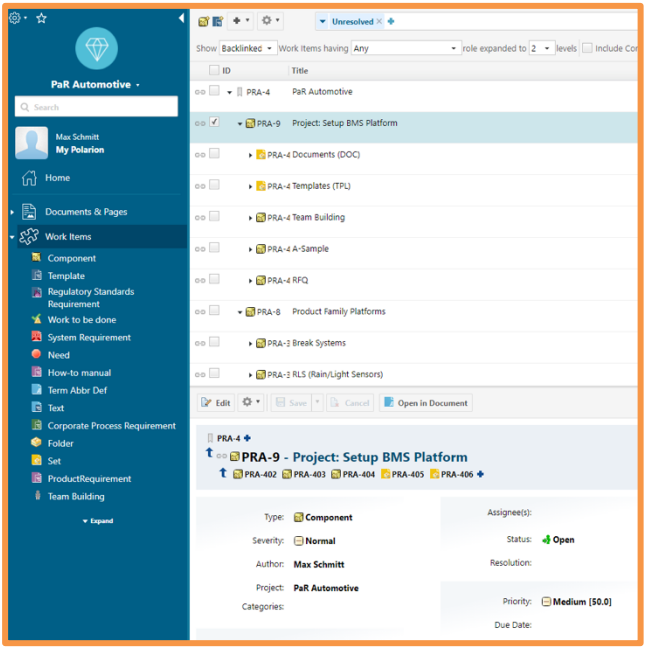
ID	Name	Type		Description	Mu...	Required	Default Value	Acti...
k_std	K_std	Enum	k_std					
documentKey	documentKey	String (sin						
globalId	globalId	String (sin						
external_ID	External_ID	Integer						
name	name	String (sin						
external_Project	External_Project	Integer						
external_Type	External_Type	Enum	external_Typ					
		String (sin						

Special for **Polarion** is that items are typically stored in what is called “Live Documents”. This allows to structure, view and edit items similar to regular text documents, therefore enabling a pure document centric work approach.



As can be seen in this screenshot, items can additionally be viewed in a table view with limited hierarchical structure.

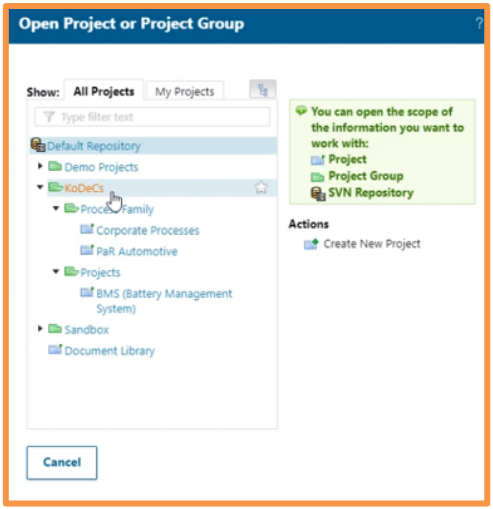
Changes to the items can be made in either the document or in the table and will reflect in the other view.



Feature 2: Implementation of the PaRis map

This Polarion project setup screenshot shows the configuration of three project groups for “Process Family” (for regulatory and corporate standards), “Product Families” and “Projects” in the project browser.

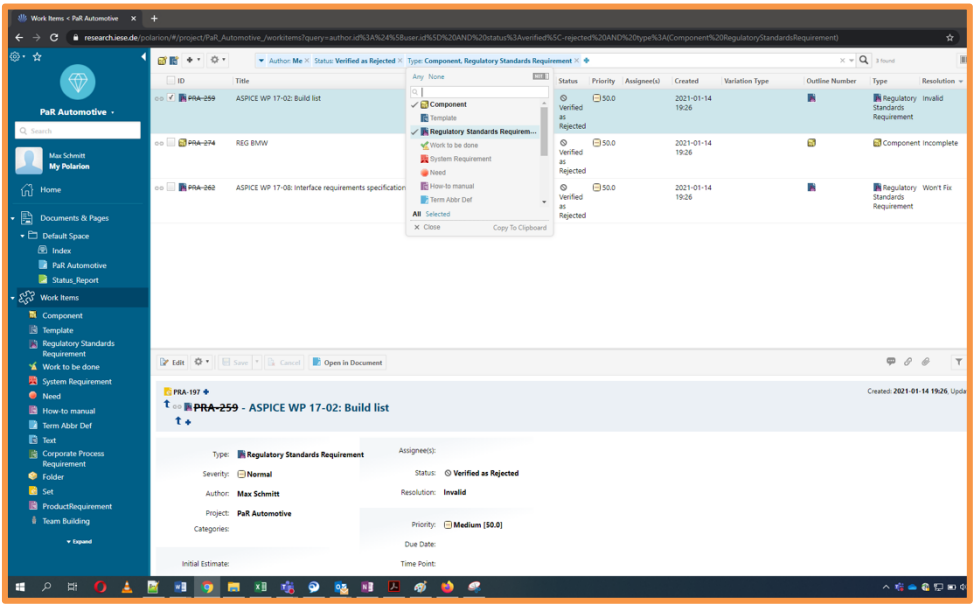
Projects can be created by administrators, but project groups cannot be created directly, only by moving existing projects. Their main purpose is to structure projects. In Polarion, relationships can only be defined between item types.



Feature 3: Evaluation of project maturity

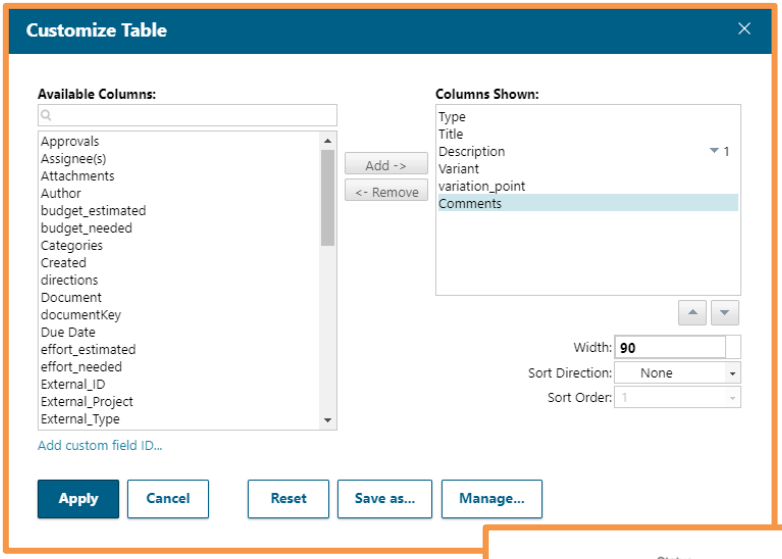
In Polarion, all users can set up ***filters***, name and save them, and also share them with other users within in the same project. As the following screenshot shows, filters can easily be created without knowing any syntax and can be combined to create complex queries.

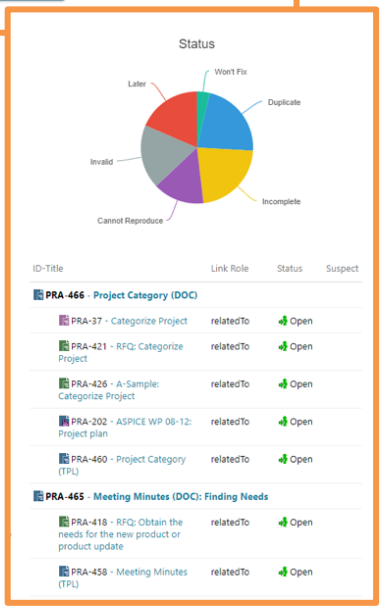
The example shows a filter that lists items of the types *Component* and *Regulatory Standards Requirements* that are assigned to the current user and have the status *Verified as Rejected*. The filters can be applied to the table view as well as to live documents.



The table view can be customized by all users to create multiple configurations of the attributes that are to be displayed as columns in the table.

Unlike filters, ***views*** can only be saved for the current user account and therefore cannot be shared between different users. Administrators can only set one default view for all users of the project.





In Live Documents, attributes can either be displayed inline in the document itself or next to the document in a separate tab. However, only one configuration can be saved per item type in one document.

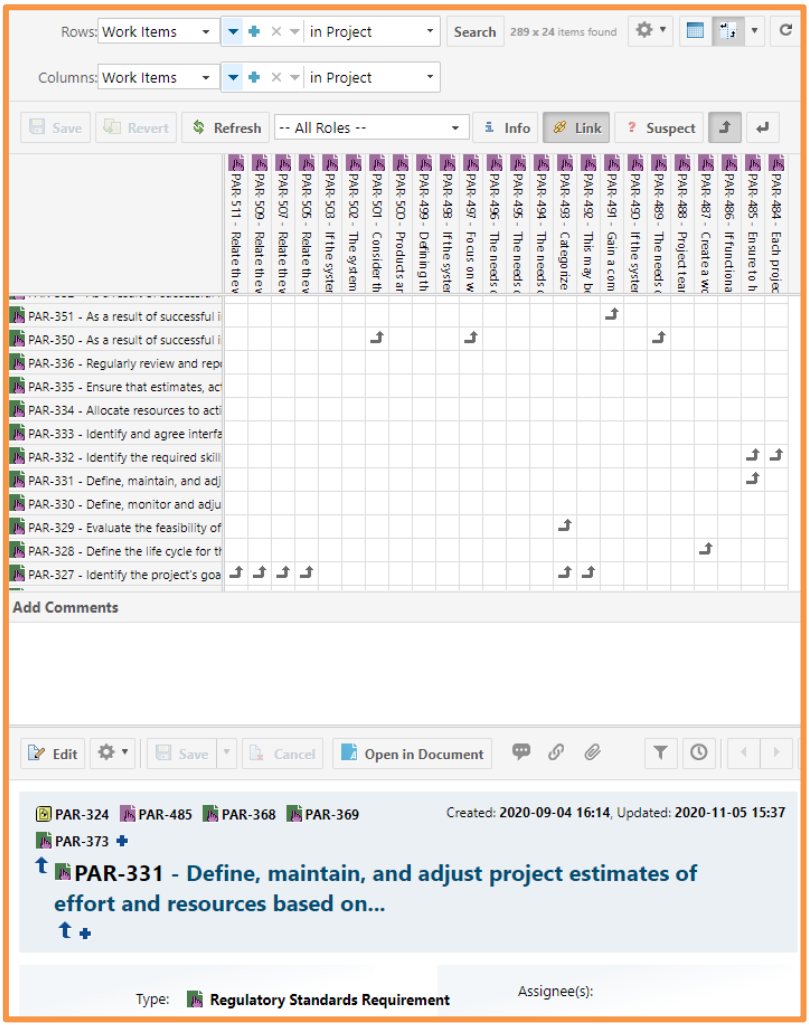
Polarion also supports ***dashboards*** that can be customized to show several different widgets.

In the screenshot aside, the dashboard to monitor project maturity is configured to display the attributes *Resolution* and *Status* of all items in the project as pie charts. Additionally, an activity feed to show recent changes and a tracability table are displayed on the dashboard.

Feature 4: Compliance checks by standards coverage

The previous chapter already showed how filters, lists and diagrams can be used in **Polarion** for compliance checks also.

Traceability and impact analysis are supported via a matrix view, that displays a grid of work items. The sets of work items on each axis are determined by a query. The direction of the links is indicated by icons in the matrix, while the detail view on the bottom shows all relations for the selected work item.



In the example above, we display the relations between Regulatory Standards Requirements and Corporate Process Requirements. In the grid, it becomes clear that there are only links going from the Regulatory Standards Requirements to the Corporate Process Requirements while some Regulatory Standards Requirements do not have any relation to a Corporate Process Requirement. The detail view for the selected element PAR-331 shows that it has more relations to other work item types such as a Set and three other Regulatory Standards Requirements.

Feature 5: Support for process versions

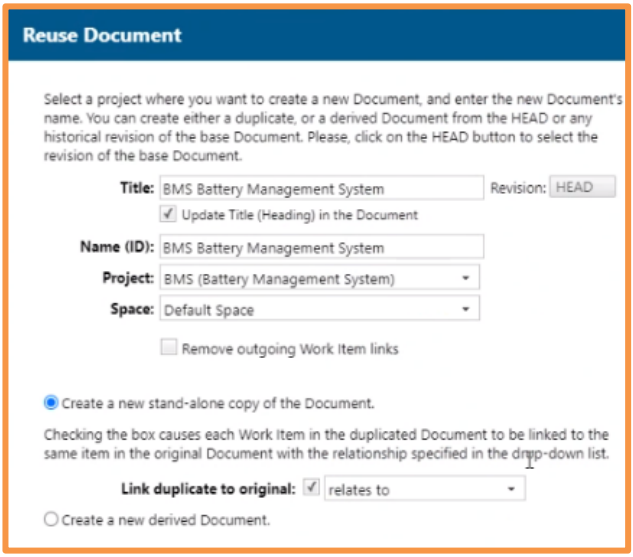
In Polarion, Baselines are only meant to be compared to one another and cannot be reused.

Feature 6: Reuse of requirements sets

As Polarion uses “Live Documents” as a central element, the reuse of documents is supported out of the box in two ways:

- The document can either be duplicated as a new stand-alone copy allowing changes to be made to the duplicated items.
- The other option is to reuse the document as a newly derived element where the items are duplicated, and the derived fields cannot be edited.

In both cases, the reused items will be linked to the current revision of the original items, while any modifications to the original items will be indicated to the user. Additionally, multiple documents can also be reused together.



On a document level, Polarion also supports branching which creates documents referencing items from the master document. These branched documents can then be enhanced with new items or items that are specific to that branch.

The previously described projects or project groups cannot be easily reused. However, for advanced administrators, it is possible to copy entire projects from the repository on the server.

Feature 7: Synchronization of requirements sets

Connectors and several extensions allow Polarion to synchronize work items with other tools such as DOORS. For example, one extension enables the synchronization of work items and diagrams with Enterprise Architect while another extension allows to synchronize work items with issues in Jira.

Since only documents can be reused a synchronization of reused requirements for improving the corporate standard from a project is not directly possible.

Feature 8: Definition and management of variability

Out-of-the-box, a very basic management of variants can be achieved by using attributes that contain information for which variant(s) each work item is relevant for. As previously described, Polarion also supports the reuse and branching of Live Documents.

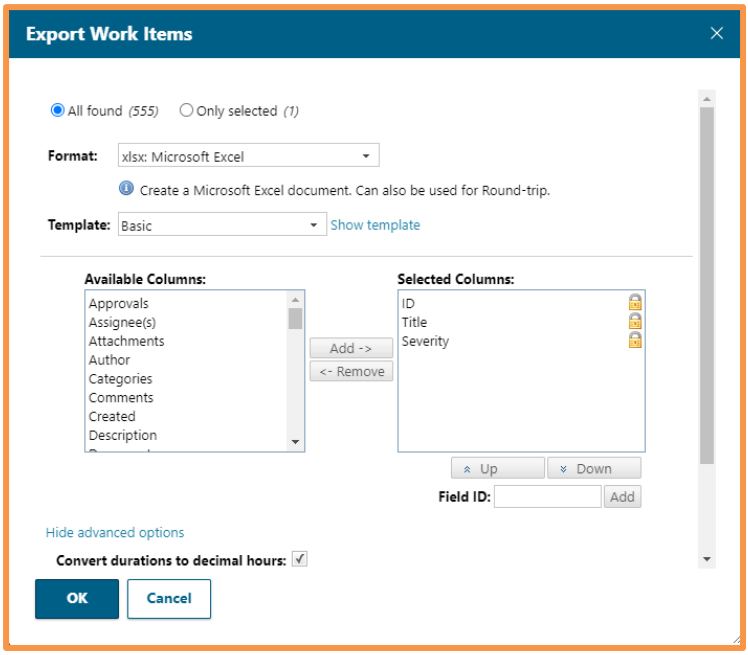
However, extensions can increase the existing reuse and variability functionality of Polarion. For example, “Polarion Variants” enables the proper use of a feature model and provides functionalities such as automated checks to further manage variability.

PaRtial Import, Export, Backup

For migrating processes from existing process design tools to the requirements engineering tool it is essential to have some import options. Then all processes and sub-processes can be migrated step by step, and finally maybe the expensive licenses for the process design tools can be saved (the requirements engineering tools are needed anyway).

Some architectural design tools can work quite good with requirements. Therefore, it makes sense to also transfer process requirements to those tools. This requires functionalities for partial exports.

At least it makes sense to perform partial backups from time to time. Of course, the complete databases and clouds are saved regularly by central tool administrators of the IT departments, but saving a process release now and then should be possible.

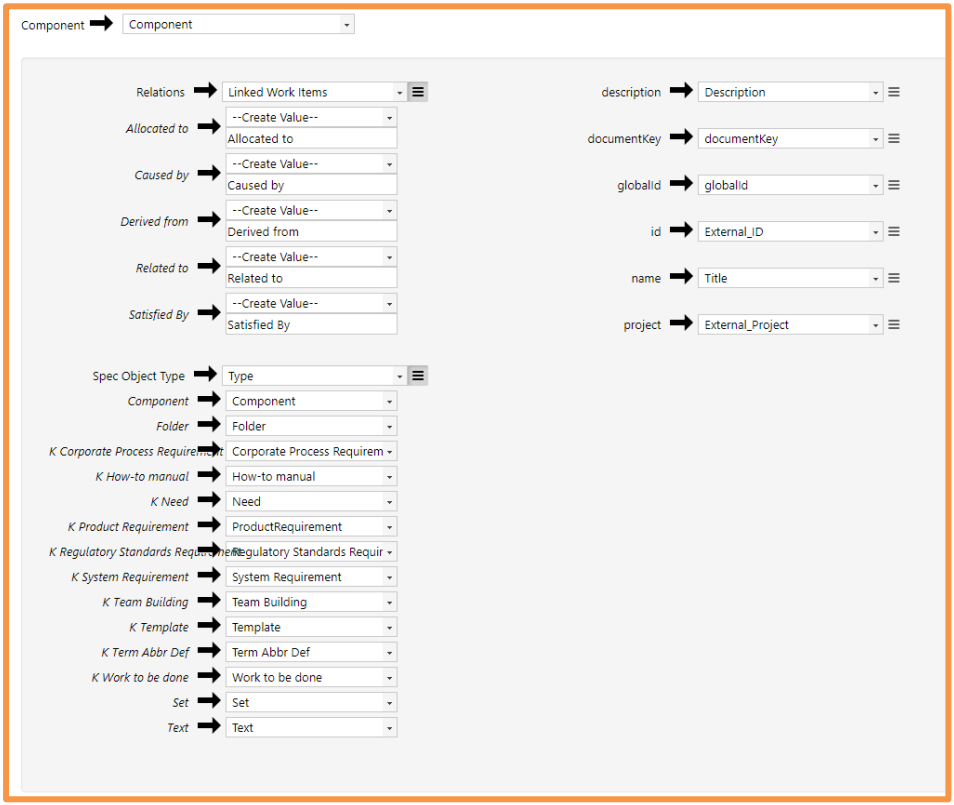


Besides providing Round-trip functionality for documents via ReqIF and Word, Polarion also supports exporting items to formats such as XLS or CSV. By applying filters to the work items beforehand, only a set of selected work items can be exported. The attributes to be exported can be

set individually and in case of an XLS-export, that is intended to be reimported, attributes can be locked so they cannot be edited in the exported file.

Items can be imported into **Polarion** via XLS, XML. Alternatively, ReqIF-files can be imported into a Live Document. Before the ReqIF import can be initiated, the data has to be mapped to their respective data types in **Polarion**.

The following screenshot shows the mapping process of the work item type *Component*. Each attribute is mapped manually to the respective attribute that is used in **Polarion**. If the attribute does not exist, it can be set to be created by the tool during import. Additionally, for the enumerations *Relations* and *Spec. Object Type*, each of the entries of the enumerations need to be mapped as well.



… From the graphical representation in 3 levels we can derive an organizational responsibility that is not that visible today in many of our departments. Setting up the regulatory standards – i.e. the boundary conditions – separately from the processes and then relating it to each other is something we don't do that clear today. But that would be much more efficient also for sure. …

Central process department, a German automotive supplier

… I'm convinced that PaR is the next step to be more efficient and agile in project even though you have to fulfil A-SPICE, ISO 26262 and ISO 21434. …

Sascha Kobus, CEO KoDeCs GmbH

… Very promising approach, which exploits the reuse potential for product and process aspects in a unified manner. …

Dr. Martin Becker, Department Head Embedded Systems Engineering at Fraunhofer Institute for Experimental Software Engineering (IESE)

… The variant development process for architecture is among the best we have seen. We consider this approach to be state of the art and benchmark. Especially the strong link between platform and project architecture …

Feedback from an expert discussion of a process for platform-based product development, that I created over some years for a German automotive supplier. My basic platform ideas of that process also made their way into the **PaR** approach for process platforms.

Did your processes become a heavyweight backpack to be carried by the projects, rather than a lightweight intrinsic approach that really helps the teams to navigate through the storms of the projects?

It gets better when you design regulatory standards and Processes as Requirements that are reused in and improved by the projects.

Ralf.Buerger@ ProcessesAsRequirements.info
https://ProcessesAsRequirements.info