MAYAM

Introduction to Mayam Tasks

The Mayam Tasks workflow engine brings user task management and media workflow processing capabilities.

The Mayam Tasks UI is a key enabler for media centric task list driven workflows. System jobs, user task lists and approval screens etc are presented to users where they make sense – standalone use, inside a MAM, as a panel in the video editing application or embedded into a user portal.

With the newly released Mayam Tasks 4.0, we are introducing Marionette, a new embedded BPMN process editor bringing major usability improvements. In addition, several other services are part of the package, most notably a job execution engine, a rules engine and a metadata transformer.

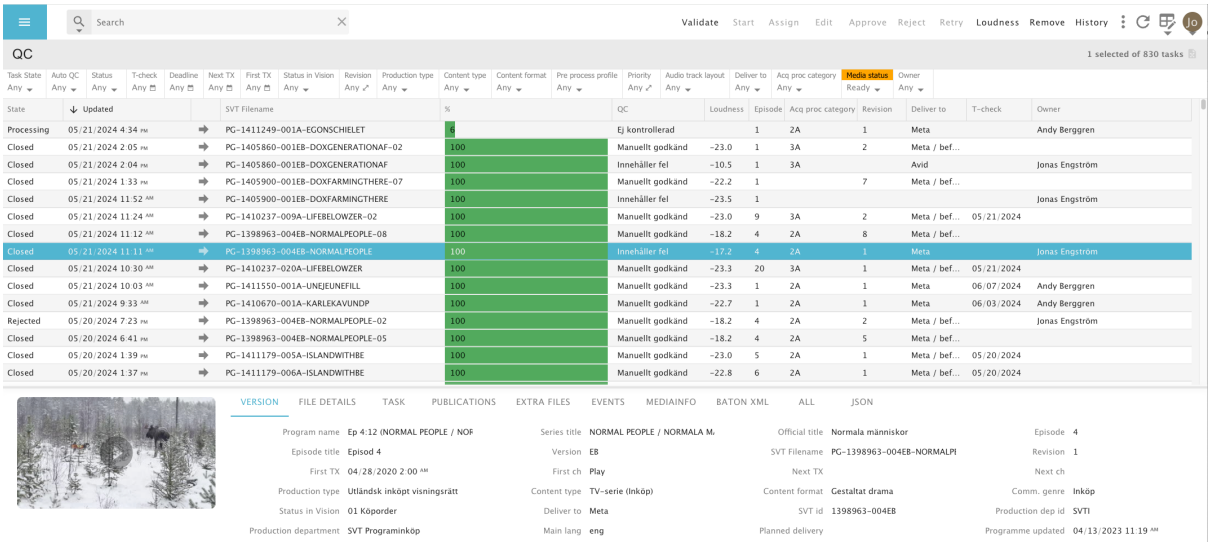


Figure. The Mayam Tasks GUI.

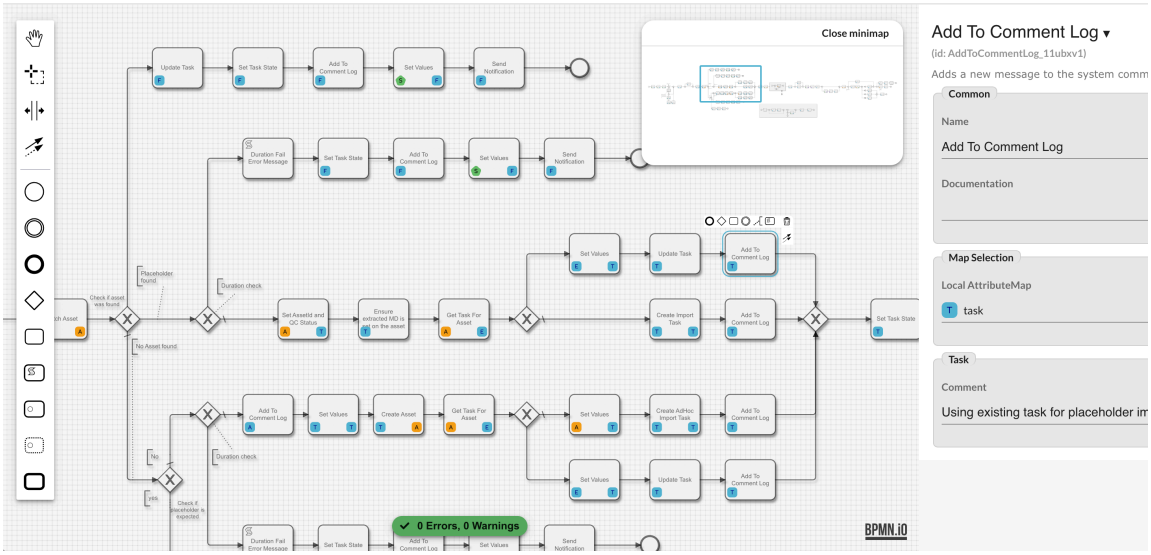


Figure. The Marionette process manager GUI, part of Mayam Tasks.

Mayam Tasks can operate in cloud and on-prem deployments. The whole solution operates as a set of independent micro services provided as containers.

Task UI

A key component of the workflow solution is the UI for human tasks, automated jobs and a mix thereof. Typically, the data shown to users and the associated actions and permissions are configured to match specific workflow requirements. For example, a program delivery workflow can use different views (task lists) for content reception, editing and approvals.

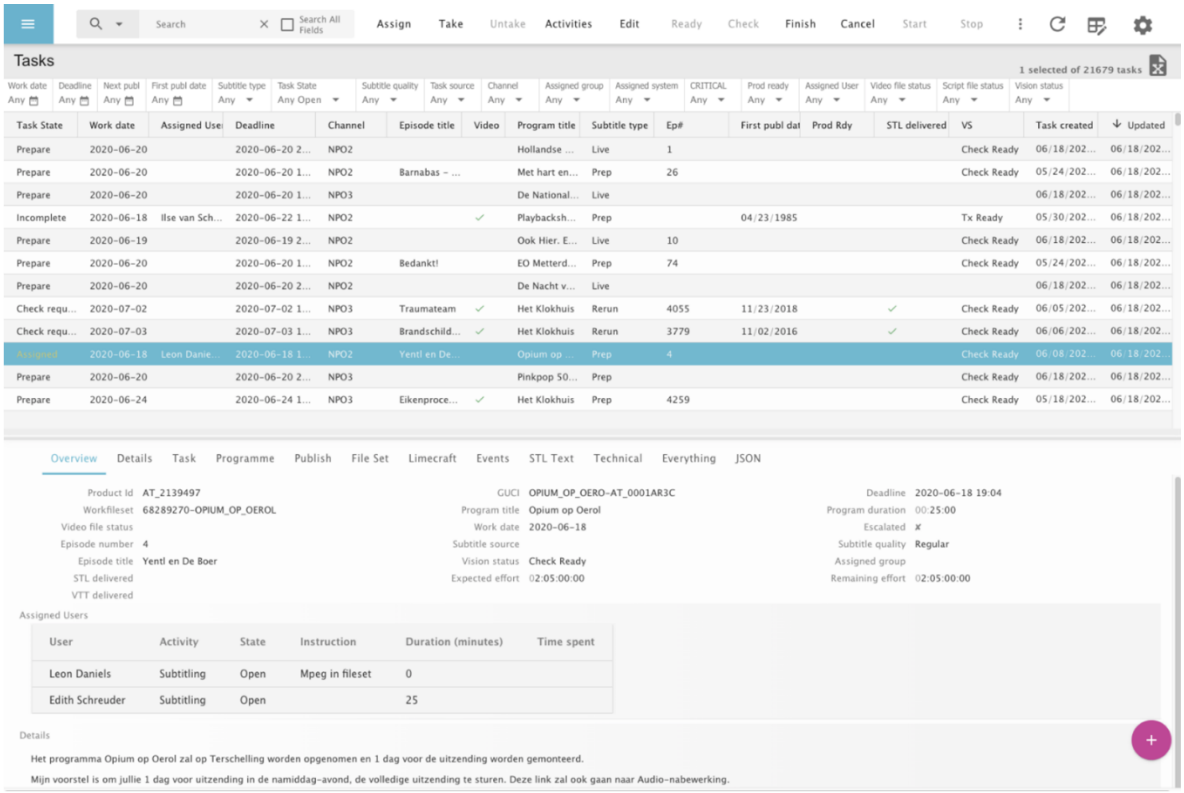


Figure. Sample task list, in this case used for manual and automated subtitling.

Functionality:

- A split view with a task list view and details for the selected task
- The ability to select multiple tasks and perform an action on that selection
- Task list filters, search (right truncation or left plus right truncation) and sort options
    - Free text search is also available
- A memory function that remembers the last used filter, search and sort settings on a per user and task list level (stored as user settings in the database)
- State dependent task list actions including: Assign to user, Pickup, Edit, Reject, Revert, Approve, Finish, etc. For example, the ‘Finish’ operation is only available on tasks in assigned/active states
- MAM assets, as well as task specific metadata, can be edited with the task list ‘Edit’ function. A page wizard view is also available for large forms
- A string of immediate actions such as file transfers can be tied to task list actions
- MAM invoker buttons for operations like ‘go to asset page’, invoke logging tool etc.
- Access restrictions:
    - Task list – who can open the task list
    - Task list actions – group level access control to individual actions
    - Tasks – mapped to the access of the primary asset of the task. If a user doesn’t have access to a MAM asset, the task will not appear in the task list
- Files attached to tasks are managed in a folder view supporting file up- and download operations plus annotations
- Text field, drop downs, media status icons, check boxes, and comment logs
- Advanced table data and approval widgets

- Task list fields can be standalone or bi-directionally mapped to the corresponding MAM asset metadata fields
    - MAM dictionaries are supported as sources for drop down widgets (to eliminate double administration when making changes)
- Programmable task form validation rules, typically used to display error messages and prevent order submissions when illegal data is entered
- Full access to all historical tasks (available via a user setting)
- Admin users have access to task audit trails, where all task changes can be inspected
    - This mechanism is also used to store KPI data to be used in reports
- Task list export to Excel for further analysis
- Batch work order creation from Excel order form upload (via a site-specific module to parse excel sheet contents)
- Hierarchical tasks in two or three levels. For example, a high-level content work order can consist of child work orders, one per file, while the high-level task shows the total and open number of sub tasks
- Tasks can be assigned to one or more users
    - This is to support tasks with several operators working on the same task
- MAM Overview page can be shown in split view – Viz Studio and Limecraft
- Content validation tool can be shown in split view – Codemilll Validate
- Other services can be integrated and displayed as task actions (big modal dialog or opening a new browse tab)
- Built in basic video player and image viewer for quick content inspection
- The UI can be used as a web component or as a standalone UI
- ADFS/Azure AD/SAML supported for single sign on
- AD access for authentication and user group lookup
    - For example, a UI drop down showing members of a given group is configured as an “AD dropdown”
- Tasks can be styled using a site plugin, for example to highlight overdue tasks
- Advanced action activation criteria using expressions
    - For example, a task action should only be enabled for users of a group if the media status is ready and the first content check (a metadata field) is OK
- The concept of computed fields enables the presentation of scalar values sourced from raw JSON data. This way large data sets (like all details of an episode) can be presented in a task without having to copy all values into task fields.

Drop down field values can also be customized via the site plugin. The example below shows an “assign task” form. Here, the list of AD users available for task assignments has been augmented with person availability and role data (sourced from an HR system), it shows the people who work today at the top the list.

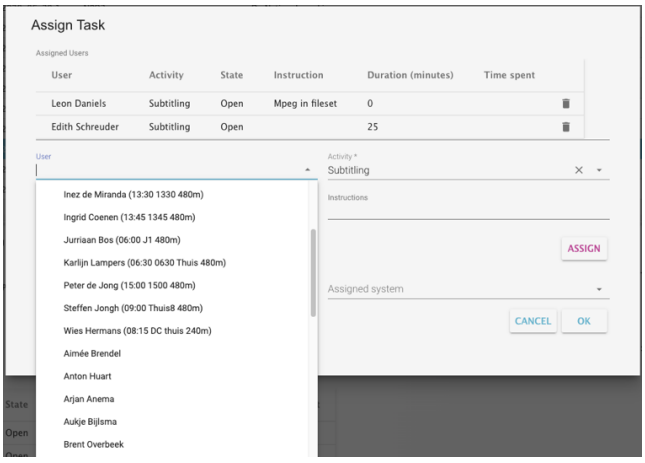


Figure. Sample task assign dialog with customized drop-down field contents.

The Mayam UI can operate standalone or as an embedded web component inside for example a Media Explorer UI, Premiere Pro or a customer facing portal.

Dashboard

Beyond access to the individual work order and task lists, a dashboard view is also available. Users can save mini views of a task lists (including filters, search and sort) as dashboard widgets. Users can also select which graphs to place on the dashboards.

Graphs are authored by super users in Kibana. By storing key KPI figures like timestamps, job status, media duration, etc., in the task and job records, these graphs can show work order status overviews as well as media processing volumetrics.

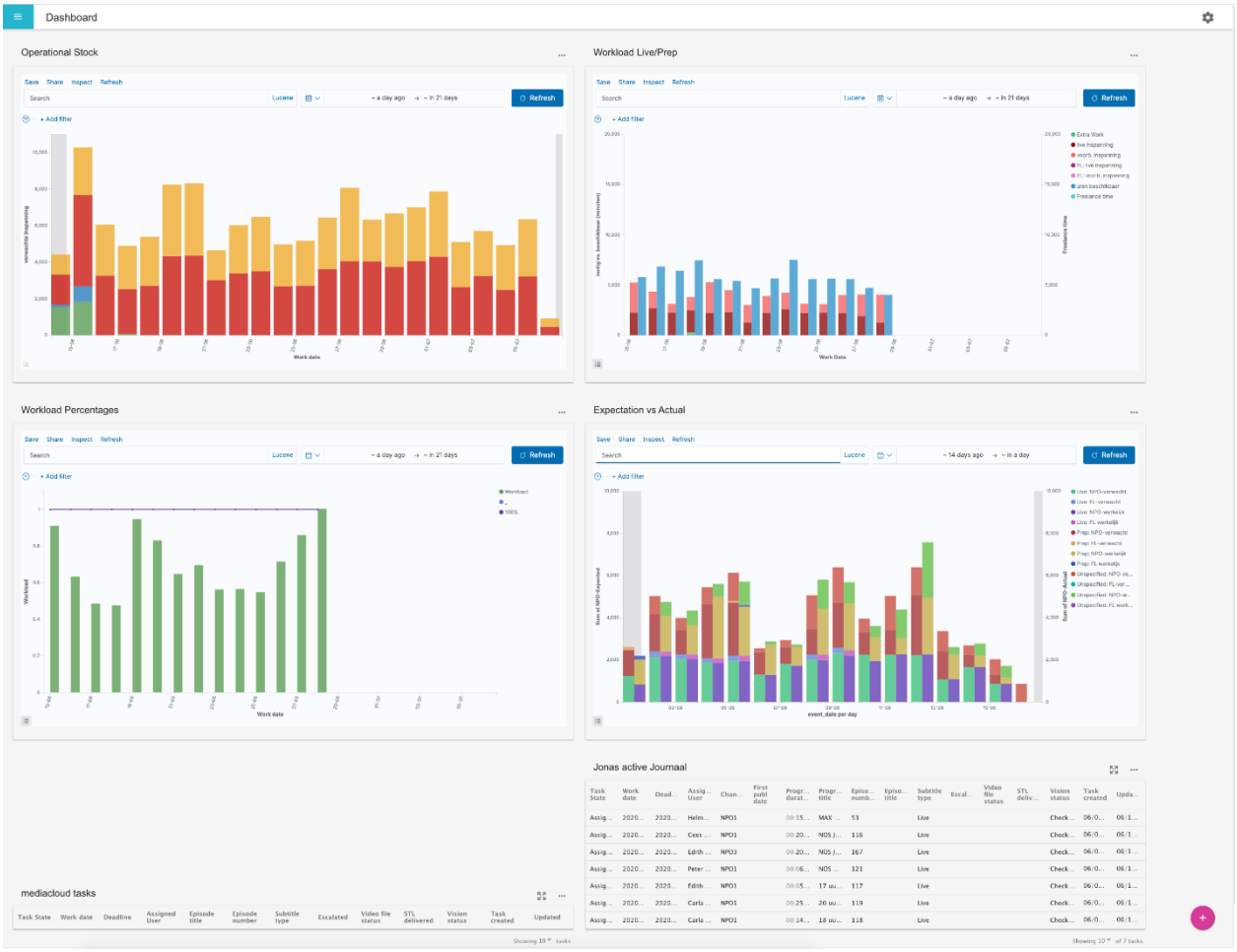


Figure. Dashboard showing workload graphs plus task overview widgets.

Workflow Process Modelling and Execution

Starting with Mayam Tasks 4, Mayam Marionette is used for process design and execution. Processed and modelled and deployed directly from the Marionette UI, which is part of the Mayam Tasks UI.

The new Marionette process modeller is BPMN-based with specific shapes for high level media operations. This is done via the concept of templates – pre-configured sub-workflows which can be dragged to process media via single shape actions.

An example of such a workflow is auto qc. In this case, the workflow steps to create a job record, a call for example ‘Baton’, update the asset with AQC result details and finally update the job record to signal completion of operator attention needed would be “hidden” under a workflow shape named AQC.

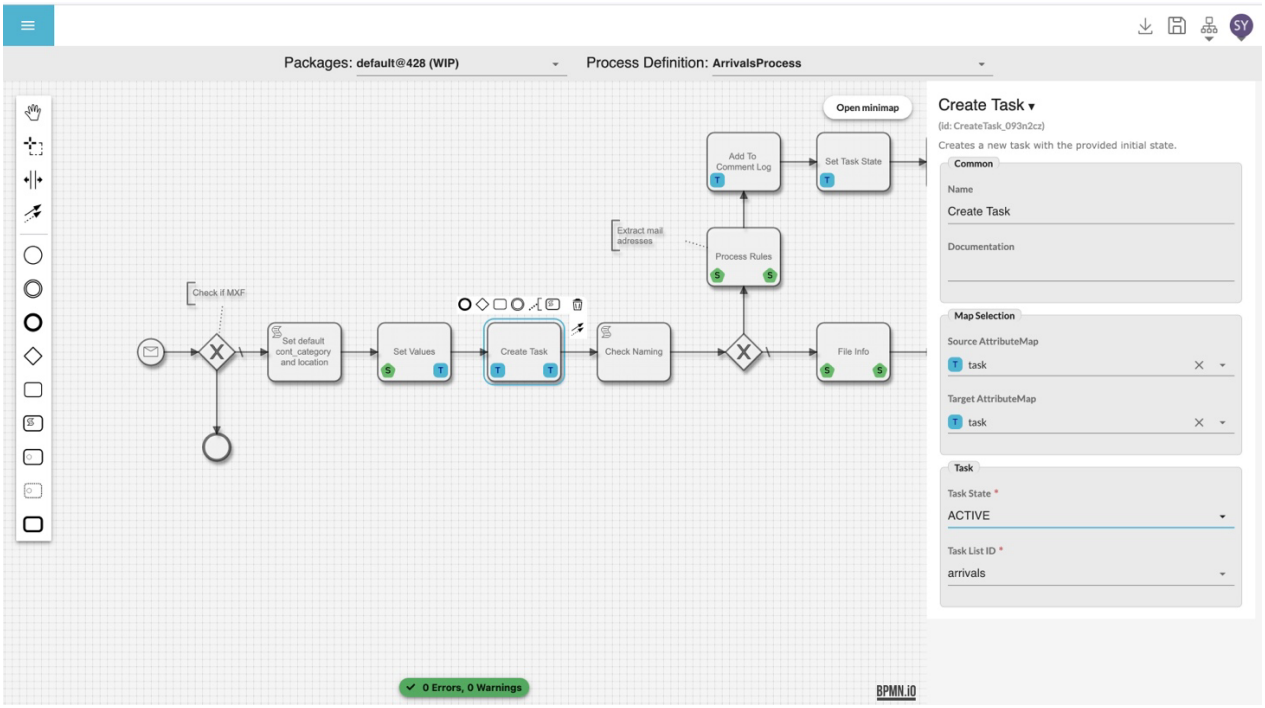


Figure. The new Marionette workflow process editor.

The workflows communicate with users via the Mayam task/job list. In this view, media details, processing results, and high-level logs are shown together with operator actions and navigation to media.

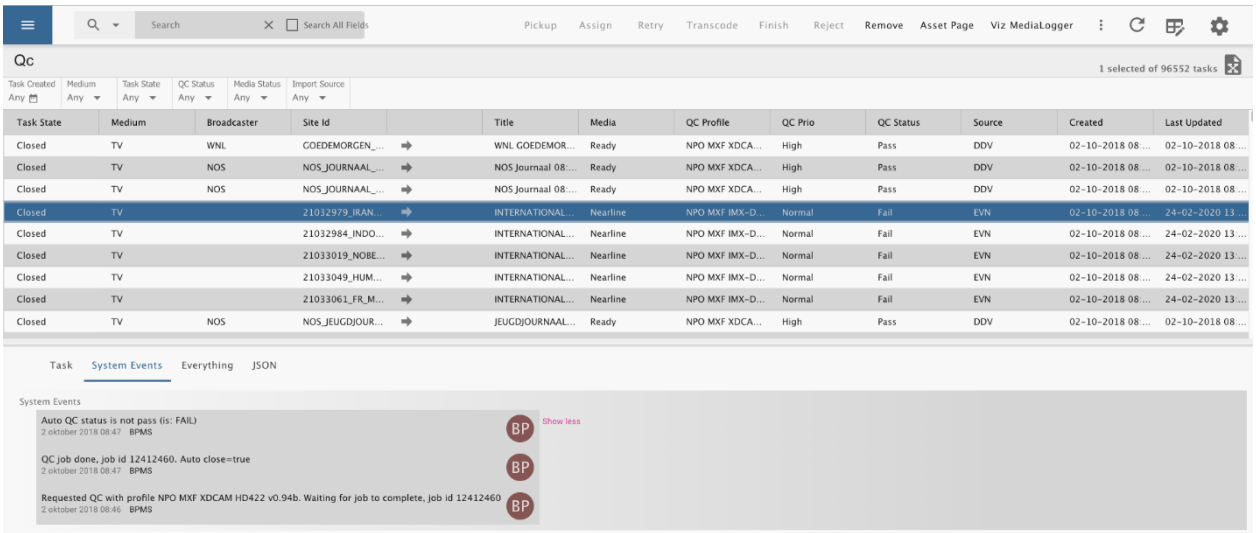


Figure. QC job list example.

Process execution can be inspected visually with Marionette highlights the workflow path taken.

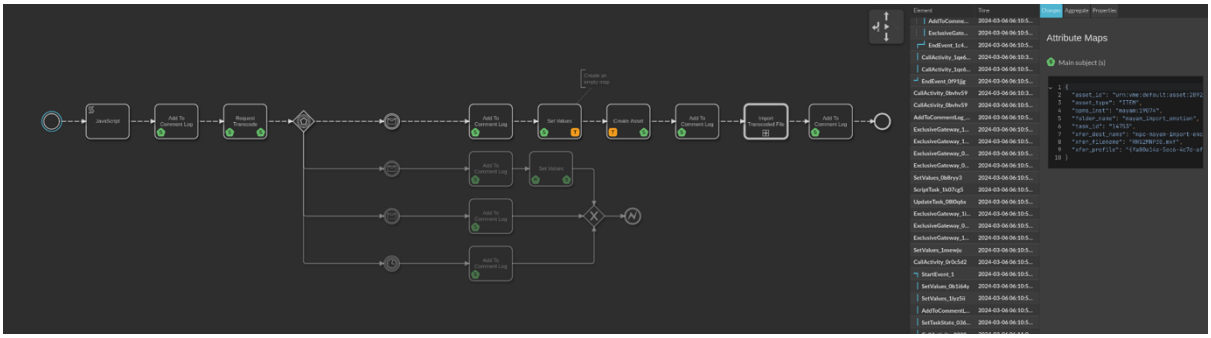


Figure. Visualizing the executed process path.

Editor Panel

The workflow UI can be accessed as a panel from inside Adobe Premiere.

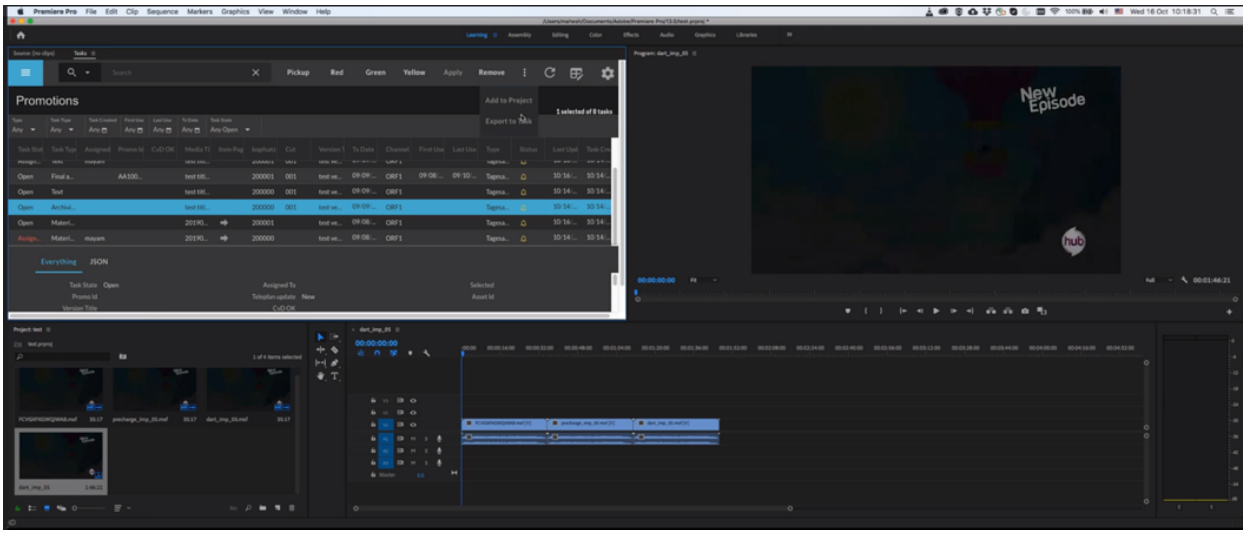
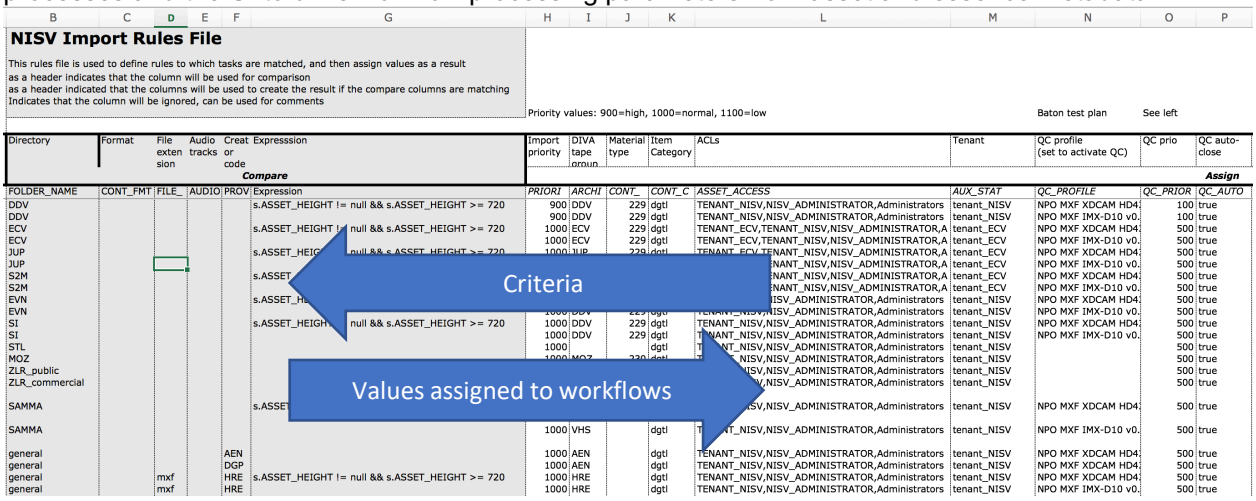


Figure. Mayam Tasks inside Adobe Premiere.

Editor-specific functionality exists to load source media from a task and to publish an edited version back to a task. This effectively offloads manual file handling and file names from the person editing the video. Edit tasks are typically arranged so that source material is available from the task assigned to the editor. The finished timeline is published back to the task using the *Publish* operation in the tasklist.

Table Rules Processor

A frequently used part of the solution is the table rules processor, which can be called from workflow processes and the UI to drive workflow processing parameters from asset and essence metadata.



NISV Import Rules File

This rules file is used to define rules to which tasks are matched, and then assign values as a result
as a header indicates that the column will be used for comparison
as a header indicated that the columns will be used to create the result if the compare columns are matching
Indicates that the column will be ignored, can be used for comments

Priority values: 900=high, 1000=normal, 1100=low

Baton test plan | See left

Directory	Format	File extension	Audio tracks	Creator code	Expresssion	Import priority	DIVA tape group	Material type	Item Category	ACLs	Tenant	QC profile (set to activate QC)	QC prio	QC auto-close
					Compare									Assign
FOLDER_NAME	CONT_FMT	FILE_	AUDIO	PROV	Expression	PRIORI	ARCHI	CONT_	CONT_C	ASSET_ACCESS	AUX_STAT	QC_PROFILE	QC_PRIOR	QC_AUTO
DDV					s.ASSET_HEIGHT != null && s.ASSET_HEIGHT >= 720	900	DDV	229	dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	100	true
DDV						900	DDV	229	dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF IMX-D10 v0.	100	true
ECV					s.ASSET_HEIGHT != null && s.ASSET_HEIGHT >= 720	1000	ECV	229	dgtl	TENANT_ECV,TENANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF XDCAM HD4	500	true
ECV						1000	ECV	229	dgtl	TENANT_ECV,TENANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF IMX-D10 v0.	500	true
JUP					s.ASSET_HEIG	1000	JUP	229	dgtl	TENANT_ECV,TENANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF XDCAM HD4	500	true
JUP										NANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF IMX-D10 v0.	500	true
S2M					s.ASSET					NANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF XDCAM HD4	500	true
S2M										NANT_NISV,NISV_ADMINISTRATOR,A	tenant_ECV	NPO MXF IMX-D10 v0.	500	true
EVN					s.ASSET_H					ISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
EVN										NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF IMX-D10 v0.	500	true
SI					s.ASSET_HEIGH null && s.ASSET_HEIGHT >= 720	1000	DDV	229	dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
SI						1000	DDV	229	dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF IMX-D10 v0.	500	true
STL						1000			dgtl	T NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV		500	true
MOZ						1000	MOZ	229	dgtl	T NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV		500	true
ZLR_public										NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV		500	true
ZLR_commercial										V,NISV_ADMINISTRATOR,Administrators	tenant_NISV		500	true
SAMMA					s.ASSET					NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
SAMMA						1000	VHS		dgtl	T NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF IMX-D10 v0.	500	true
general				AEN		1000	AEN		dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
general				DGP		1000	AEN		dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
general		mxf		HRE	s.ASSET_HEIGHT != null && s.ASSET_HEIGHT >= 720	1000	HRE		dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF XDCAM HD4	500	true
general		mxf		HRE		1000	HRE		dgtl	TENANT_NISV,NISV_ADMINISTRATOR,Administrators	tenant_NISV	NPO MXF IMX-D10 v0.	500	true

Figure. Sample table rules file.

In the example above, import priority, asset permissions, some metadata and the QC priority and profile is derived from import source, video format, file extension and a part of the filename.

Job Engine and File Processor

The Job Engine is used to drive job queuing and execution, including support for advanced priority rules and concurrency control. Key functions:

- Multiple job queues are configured to support per resource or operation type concurrency and priority rules
- The Tasks UI is used to manage jobs, for example list queues, edit priority, cancel jobs
- Live updates with progress bars
- Built-in operations:
    - Extract technical metadata using mediainfo
    - File transfer - S3, HTTPS, SFTP and plain FTP
    - File move, chown, delete etc
    - Transcode including proxy generation and keyframes (ffmpeg)
    - Transcode using Vantage (API integration)
    - QC with Baton (API integration)
- Site-specific operations are built using the core operations as templates

Jobs can be created from BPMN processes or via API integration. Manual job creation is of course also possible.

Jobs can be chained, for example a transfer following a transcode. Downstream jobs are held until upstream jobs are completed. A failure in an upstream job will cancel the downstream job.

Concurrency management is done on three levels:

1. Operation type level concurrency limits
2. Per resource concurrency limits, for example the number of transcode jobs per node
3. Metadata driven concurrency limits, for example specific rules for different tenants

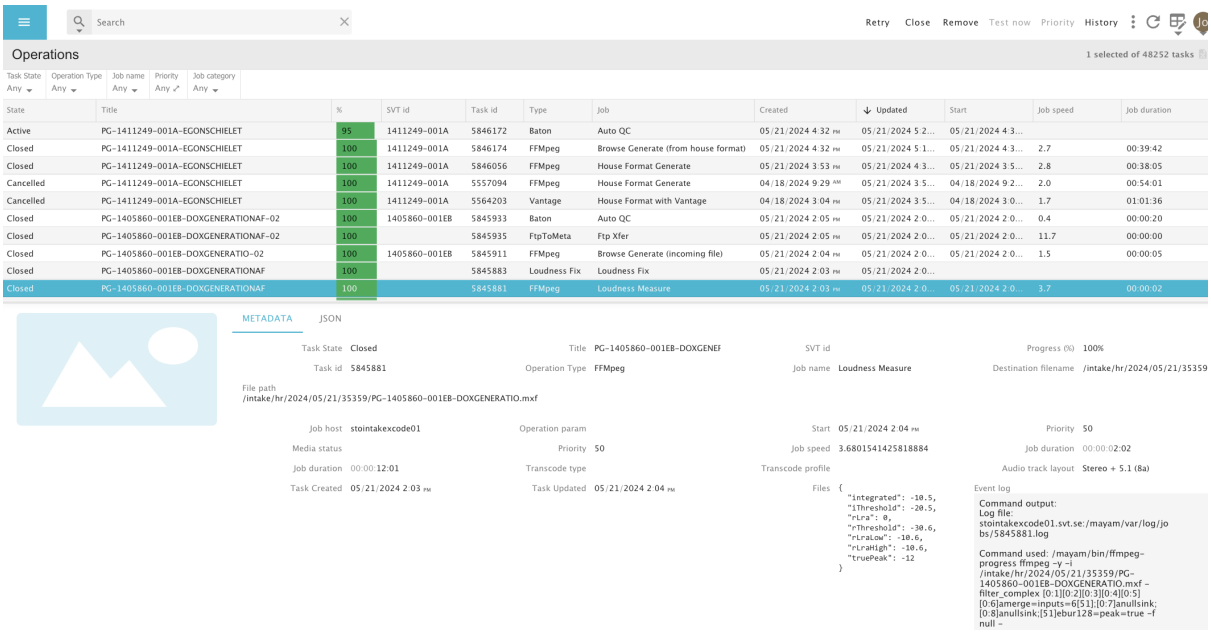


Figure. Job Queue.

In the screenshot above, examples of site operations are shown in the form of loudness measurement and calling a site specific inhouse cloud function for loudness correction.

MdMap: Built-In Data Mapper / Transformer

Mayam MdMap provides functionality to transform XML and JSON to common workflow format Data that can be saved directly to tasks via mapping micro service. MdMap can be used for simple sidecar file type reading applications. However, there are enough mapping functions and plugin capabilities to map for example their complete data set of a planning system.

In the example below, different titles from a planning system are placed in a metadata table for alternative title combined with title types. At the same time, the business rule “only copy title x to alternative titles if the series title is not the same + ignore case + strip white space”. The example below also performs thesaurus lookup and normalization for selected fields.

```
<ruleset from="originalTitle/value"><to mam="nisv.alttitle"/>
    <rule from="."><to mam="alttitle.name"/></rule>
    <rule static="111193"><to mam="alttitle.type"/></rule>
    <validate expr="!equalsic(s.SERIES_TITLE, f) &amp;&amp; !equalsic(s.ASSET_TITLE, f)" severity="SKIP"/>
    <!--  <validate expr="s.SERIES_TITLE != f &amp;&amp; s.ASSET_TITLE != f" severity="SKIP"/> -->
</ruleset>
<ruleset from="shortBroadcastTitle/value"><to mam="nisv.alttitle" append="true"/>
    <rule from="."><to mam="alttitle.name"/></rule>
    <rule static="67026"><to mam="alttitle.type"/></rule>
    <validate expr="!equalsic(s.SERIES_TITLE, f) &amp;&amp; !equalsic(s.ASSET_TITLE, f)" severity="SKIP"/>
</ruleset>
<ruleset from="subTitle/value"><to mam="nisv.alttitle" append="true"/>
    <rule from="."><to mam="alttitle.name"/></rule>
    <rule static="67026"><to mam="alttitle.type"/></rule>
    <validate expr="!equalsic(s.SERIES_TITLE, f) &amp;&amp; !equalsic(s.ASSET_TITLE, f)" severity="SKIP"/>
</ruleset>
<rule from="awards/value"><to mam="nisv.award"/></rule>
<ruleset from="director/person"><to mam="nisv.crew"/>
    <rule from="."><to mam="crew.name"/>
        <transform expr="wonPerson2name(f, 'THE22', context)"/>
        <validate expr="f != null" severity="SKIP"/>
    </rule>
    <rule static="69"><to mam="crew.role"/></rule>
    <rule from="code/value"><to mam="crew.annotation"/></rule>
</ruleset>
```

Figure. Advanced use of MdMap.

Micro Services Host

Mayam Tasks deployments normally interact with a set of related systems via integrations. Mayam Tasks also provides a framework for rapid development of micro services integrating with peer systems. In addition to hosting coded services, config only data transformation and task management services are supported using the MdMap framework described above.

APIs and SDK

The APIs for Tasks and Marionette are REST-based, supporting OpenAPI 3 for automatic client generation. Documentation is available as Swagger UIs for interactive testing and development. In addition, a Java SDK is available for Tasks to shorten client side development times further.

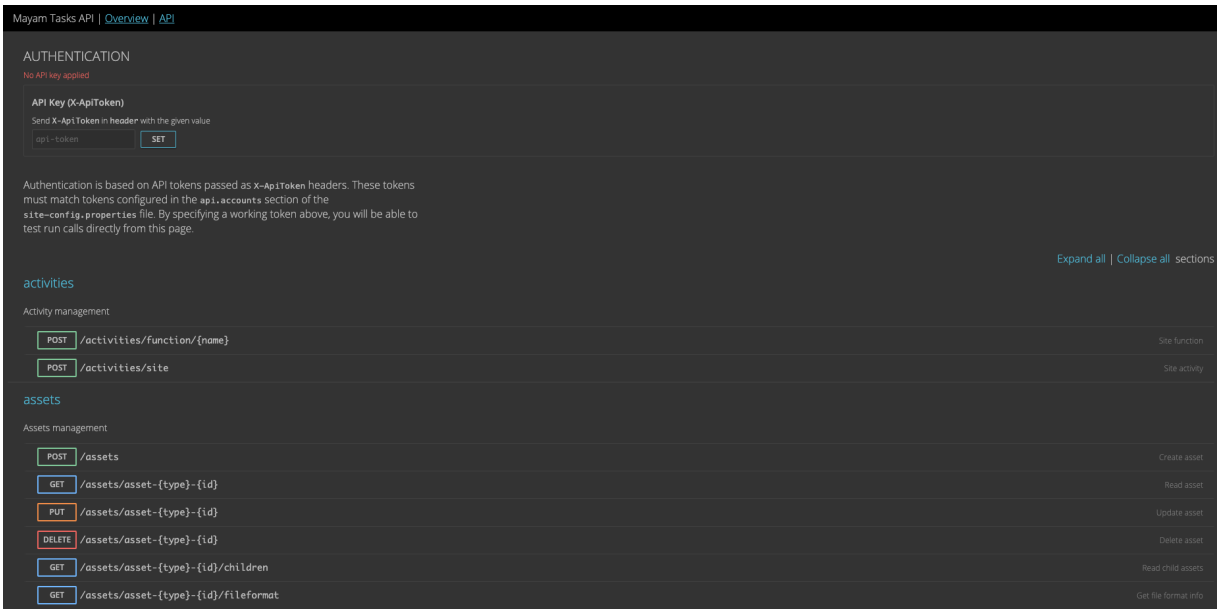


Figure. Tasks API documentation page.