Mayam Tasks v3

Technical Reference

Table of Contents

Document Information

Revision History

Revision	Date	Author	Comment
1.0	Sep. 2018	André Cruz	
1.1	Mar. 2023	Ganesh Srinivasson	

Contact Information

Mayam AB,

Vikingavägen 2,

182 63 Djursholm

Sweden

Phone: +46 (0)730-808012

E-mail: info@mayam.com[1]

[1] mailto:info@mayam.com

1. Introduction

The Mayam Tasks application extends the MAM environment with advanced task management, business process and integration capabilities. Through the Mayam approach to workflow implementation, workflows in a MAM environment are provided through the use of a dedicated task and process modelling and execution subsystem, relying on Mayam Tasks for GUI functions and MAM-independent APIs.

This document provides technical reference information used for installation and configuration, system operation and reports. For an overview of the system, please refer to the *Mayam Tasks Overview* document.

1.1. What You Should Know Before Reading this Manual

The reader will benefit strongly from a general overview of how the Mayam Tasks and MAM environments are consituted. The information in this document assumes general knowledge of how Mayam Tasks. In addition, system administration and Linux skills are required for the sections covering system installation and configuration.

1.2. Related Documents

Information about Mayam Tasks is available in the following documents:

- Mayam Tasks Overview
- Mayam Tasks TaskAdmin Guide
- Mayam Tasks Programming Guide
- Mayam Tasks BPM Process Guide
- Mayam Tasks Technical Reference (this document)

These documents can be downloaded from the installed online resource:

```
http://<the-mayam-server>:8084/tasks-ws
```

2. System Architecture

The main component in the system is a purpose-built task management application with its web UI. In addition, it bis also possible to attach a third-party BPM solution such as Acitivi or Intalio to let background business processes drive the workflows.

The above software, which runs as Web/REST services, is supported by an IT infrastructure consisting of an application server (Jetty), a message queue (ActiveMQ) and a database (PostgresSQL or Oracle, support for DB2 also exists).

Mayam Tasks and Taskadmin's UI plus the Tasks API. Serving task lists, forms, and API access to tasks and MAM assets		
PostgreSQL (option Oracle)	App Server Jetty	ActiveMQ (option: JBossM)
Linux (RHEL or SLES)		

Table 2.1. Mayam Tasks software stack

2.1. Workflow Management Solution

The Mayam workflow management solution consists of three major applications:

1. Task management using Mayam Tasks
2. A BPM platform, typically Activiti
3. A reporting engine, typically Eclipse BIRT

2.1.1. Task Management Application

The most important aspect of the management application is to allow human users to monitor and to take actions on tasks including view and editing of task related information. The task management application is deeply integrated with both the BPM server processes and the MAM system. For example, when some metadata fields and edited in the task UI, the value change is propagated both to the MAM and to the running business process instance.

The task management application is also used for job monitoring. In this case, the same UI is used to manage automated jobs and where relevant, support interaction like prioritization, job cancel and error handling.

2.1.2. BPM Platform and Processes

A BPM platform may or may not be supplied as part of a Mayam workflow solution. The information below does not imply that a BPM platform is part of Mayam Tasks.

Workflow logic, typically customized for each major workflow installation, executes as externalized high-level BPM processes. The general principle is that customer specific business logic should reside in processes and rules tables while common functionality

such as message transformation for access to a MAM system should reside in lower level release managed code. In addition, a number of message routing adapters are required to facilitate the flow of messages between the task management part, the MAM and the processes. Mayam Tasks is integrated with two BPM solutions: Acitiviti and Intalio. In the case of Activiti, the BPM server is embedded into the Mayam Tasks software stack and runs in the same application server. In the case of Intalio, the BPM server runs as a standalone service, using a separate application server and database.

In addition, Mayam Tasks can be driven from any BPM solution capable of using a REST API or Java SDK for system-to-system interfaction. The following key parts constitute the business process part of a typical workflow solution:

Component	Name	Description
BPM Server / management console	Activiti REST + Explorer	The BPM server runs two services, a REST interface to BPM operations and a management console used to monitor and manage processes and instances.
Process modeler	Activiti Designer	The graphical business process modeling tool used to model and test processes.
Processes	Site specific	The actual processes. Refer to the the site- specific document “Mayam Workflow – Process Reference” for information regarding the processes used in this installation

Table 2.2. Business process related part of the solution

2.1.3. Reporting Engine

A reporting engine may or may not be supplied as part of a Mayam workflow solution. The information below does not imply that a reporting engine is part of Mayam Tasks.

The Eclipse BIRT reporting engine can be downloaded and installed directly into the Mayam tasks application server environment. Through an integration plugin, BIRT can access Mayam tasks data using the Mayam tasks-ws REST API. Through this API, BIRT can request task recording by supplying query criteria such as a date range, task list id and so etc.

2.1.4. Solution Elements and Services

The following key parts constitute the workflow management solution. Note that all services are not installed in all deployments.

Component	Service or Library Name	Description
Web UI	`/tasks`	A task UI designed for embedding into the MAM UI. Implemented in GWT (Google Web Toolkit) and supports most browsers from IE8 and upwards.
Configuration and administration UI	`/taskadmin`	The task management configuration UI.
Mayam Tasks Web Services	`/tasks-ws`	A web services API that provides access to task and MAM asset data. Used by the BPMS processes and other systems to create and manipulate tasks and task data. This API also acts like a common simplified front to a number of MAM systems.
Activiti BPM	`/activiti-explorer`	The Activiti BPM management console
Activiti REST	`/activiti-rest`	The Activiti BPM REST API
Eclipse BIRT	`/birt`	The Eclipse BIRT reporting service

Component	Service or Library Name	Description
MAM event forwarder	`ardome4-daemon`, `vme-daemon`, `vidispine-daemon`	Messages from the task UI to the integration platforms / business processes are sent asynchronously using JMS and ActiveMQ Camel rules for routing.
Bpms event daemon	`Bpms-event-daemon`	Forwards task and MAM events to BPM process instances
Notification daemon	`Notification-daemon`	Generates notifications based on a configured event rules and properties and delivery via email
Report generator daemon	*`Jasper-daemon`*	*Deprecated:* Renders reports and delivers the output to a drop folder.
Web Services client / Event handler client	`tasks-ws-client`	A client library providing Java and object level access to task and MAM asset entities. The library is used both for synchronous WS calls as well as parsing of JMS events.

Table 2.3. Task management application components.

2.1.5. IT Infrastructure

The workflow solution relies on a stack of IT infrastructure functions. The solution is designed to function with IT infrastructure stacks ranging from open source community editions through to fully clustered enterprise variants. For cost reasons, the default IT stack is open source, but enterprise options (in terms of products and support offerings) are available. The options are listed in Table 2.4, "IT infrastructure options" below:

Component	Default	Description / Options
Application server	Jetty (Tasks) Tomcat (Intalio)	Mayam Tasks and Activiti runs in the Jetty app server. Enterprise support is available as Intalio Jetty. If Intalio is used, it runs in a dedicated Tomcat application server.
ESB	MULE community edition	Enterprise options: MULE enterprise
Message queue	Apache ActiveMQ	Mayam workflow requires a JMS capable persistent message engine. If required, there are commercial support offerings for ActiveMQ.
Database	Postgres, Oracle Express, DB2 Express-C	Enterprise options: • Oracle 11g + RAC or higher, • DB2. *Special rules apply for Intalio: the community edition supports MySQL only, while Intalio Enterprise also supports Oracle.*
Operating system	Linux OpenSUSE, RedHat Fedora, Ubuntu, CentOS	Enterprise options: • RedHat RHEL • Suse SLES

Table 2.4. IT infrastructure options

3. Data Model

Mayam Tasks uses a traditional SQL database to store information about current and historic tasks together with related configuration and audit trail data. In addition, full task data is fed to Elastissearch (option) to facilitate full text search.

3.1. Overview

A visual overview of the Mayam Tasks data model is shown in the Figure below.

FIXME: Mayam Tasks data model

Active tasks are stored in the table set task, task detail and task_aux_id. Task list level task data is mapped to the table task while task_detail stores additional potentially large field sets. The information in task_detail is access on a task by task basis. It is also possible to store multiple external source identifiers against one task in a many to one relationship. This is done in the table task_aux_id. All user and API actions operating on task data are logged to the task_audit table set (as the operation plus the associated task data). In addition, a number of supporting entities exist to support the system:

config
: attribute, action and task list configurations

settings
: user settings

3.2. Interfaces

3.2.1. Internal Interfaces

Key system interfaces are specified in Table 3.1, "System interfaces" below.

Component	Description	Interface type
Task UI	Web app	HTTP on port 8084
Task UI RPC	Web app RPC	GWT RPC on port 8084
Activiti REST	BPM web service	HTTP on port 8084
Tasks-ws	Web service	HTTP / Jax WS on port 8084 HTTP / SOAP on port 8084 (WS API)
Tasks UI and Tasks-ws API	JMS messages, both in- and outbound	Typically mapped to local activemq on localhost:61616

Table 3.1. System interfaces

4. System Installation

This chapter describes how to install and configure the Mayam workflow software on Linux servers.

5. Running the System

The standard procedures for operating the system are described in this section. Common tasks include starting and stopping the system or individual components, checking system health and reading log files as part of troubleshooting work.

5.1. Host and User

Two simple rules apply to all workflow system operations:

- Commands are performed on the host where the action is supposed to take place. A server start command, for example, should be issued on the machine where the server is to be started
- All work should be performed as user `mayam`. Specifically avoid work as user `root`. Even though the `mayctl` control command does not permit any operations if started with any user aother than `mayam`, it is easy to create files by mistake that the ordinary user, `mayam`, is not allowed to modify.

5.1.1. The Workflow Daemon Control Command

Most system activities can be performed using a single control command, **`mayctl`**. All daemon start, stop, status check and reading of the primary log and trace files can be performed using this command. The syntax is as follows:

```
mayctl <command> [<service>|all[/host]]
```

Command	Description
`env`	Show environment variables
`log`	Monitor service log file output
`restart`	Restart a service
`start`	Start a service
`status`	Show service status
`stop`	Stop a service
`trace`	Monitor service trace file output

Table 5.1. mayctl commands

5.2. Check System Status

The **`status`** command shows which applications that are configured to run on this machine and their running status.

```
[mayam@wf01 ~]$ mayctl status
Operation on all: status
activemq    [UP: 8077]
bpms        [UP: 11862]
mule        [UP: 21128]
```

```
tasks       [UP: 15872]
```

Application status is shown as either `[UP:<pid>]` or `[DOWN]`

5.3. Startup

The simplest way to start all workflow daemons is to use the **`start all`** command:

```
[mayam@wf01 ~]$ mayctl start all
Operation on Apache ActiveMQ: start
[STARTED]
Operation on BPM server (Intalio BPMS in Apache Tomcat): start
[STARTED]
Operation on MuleSoft MULE: start
[STARTED]
Operation on Mayam Tasks app server: start
[STARTED]
```

5.3.1. Starting an Individual Application

To start an individual application, the syntax is **`mayctl start <application name>`** as show in the example below. Use **`mayctl status`** to learn the list of applications configured to run on this host.

```
[mayam@wf01 ~]$ mayctl start activemq
Operation on Apache ActiveMQ: start
[STARTED]
```

5.4. Shutdown

Shutdown is performed using the **`mayctl stop`** command. To shut down all running applications, use the **`stop all`** command:

```
[mayam@wf01 ~]$ mayctl stop all
Operation on Apache ActiveMQ: stop
[STOPPED]
Operation on BPM server (Intalio BPMS in Apache Tomcat): stop
[STOPPED]
Operation on MuleSoft MULE: start
Sent TERM signal to 21128,
Operation on Mayam Tasks app server: stop
[STOPPED]
```

If a controlled shutdown is possible using stop commands, this is attempted first. As a secondary mechanism, `mayctl` sends the **`TERM`** signal and waits for the process to disappear.

5.4.1. Stopping an Individual Application

To stop an individual application, the syntax is **`mayctl stop <application name>`** as show in the example below. Use `mayctl status` to learn the list of applications configured to run on this host.

```
[mayam@wf01 ~]$ mayctl stop mule
Operation on MuleSoft MULE: stop
```

```
Sent TERM signal to 21128
```

5.5. Log and Trace Files

Top level application log output is typically written to two different files:

- a log file where errors and important events are written
- a trace file which provides more verbose information related to normal system activites

To read the log or trace file for an application, use **`mayctl log <application name>`** or **`mayctl trace <application name>`** as shown in the example below. Use `mayctl status` to learn the list of applications configured to run on this host.

```
mayam@wf01 ~]$ mayctl trace activemq
Showing trace file for Apache ActiveMQ: /mayam/var/trace/activemq.trace
 INFO | Initializing Spring root WebApplicationContext
 INFO | Connector vm://localhost Started
 INFO | Camel Console at http://0.0.0.0:8161/camel
```

5.5.1. Tasks and API Logs

Log files for the various mayam compoents are written individual log files:

Mayam Service	Description	Log File
Web UI	Mayam Tasks, the the task list UI	`/mayam/var/log/tasks.log`
Configuration and administration UI	The Mayam task data configuration UI	`/mayam/var/log/taskadmin.log`
Mayam Tasks Web Services	API service used by BPMS and external systems	`/mayam/var/log/tasks-ws.log`
Notification daemon	Email and iNEWS notifications	`/mayam/var/log/notification-daemon.log`
Jasper daemon	Report generation	`/mayam/var/log/jasper-daemon.log`
Ardome4 daemon log	Listens to Ardome4 events. Not used in all installations	`/mayam/var/log/ardome4-daemon.log`
VME Daemon log	Listens to VME events. Not used at all installations	`/mayam/var/log/vme-daemon.log`
Site daemons	Many installations include a daemon performing site-specific functions. These daemons follow the above conventions for logging.	`/mayam/var/log/<name of site-daemon>-daemon.log`

5.6. Further Status Monitoring

In addition to monitoring status by using `mayctl status` and reading log and trace files for individual applications, the workflow applications can also update a **Nagios** monitoring system with application status and version information. If Nagios is used to monitor the MAM system, this is the recommended way to perform day-to-day system monitoring.

6. Reporting with BIRT

BIRT is used within Mayam Tasks to produce reports. From the initial HTML preview launched directly from our webapp, it is possible to export CSV, Excel, PDF, and more. Tables and charts support grouping and filtering of task information and also our statistics snapshot feature.

6.1. Components

The following components make up Jasper Reports and its support within Mayam Tasks:

- The **`INVOKE_BIRT`** UI activity uses the `REPORT_*` attributes, and optionally a few others for filtering, to prepare a report preview URL.
- The com.mayam.wf.oda.runtime plugin acts as an adapter between BIRT and our REST client, providing access to tasks and snapshots.
- The com.mayam.wf.oda.ui plugin provides an Eclipse wizard which matches the runtime plugin.
- The tasks-oda-update-site artifact is used for installation in the Eclipse based designer, and includes all of the plugins.
- A few other supporting artifacts found under tasks-oda-parent indirectly affect the development and installation but not the runtime itself.
- The birt.war holds the BIRT preview/export web application. The stock distribution is modified to include our ODA runtime plugin.

6.2. Creating a Report

Setting up Environment

- Make sure a minimum of tasks-ws is up and running since REST calls will be made.
- If an API account is not available, add one and restart the web apps.

Example site-config.properties snippet api.accounts=test:test

6.3. Setting up BIRT Designer

1. Downoad and install latest All-in-one release o from http://download.eclipse.org/birt/downloads/
2. Download our latest tasks-oda-update-site and unpack it in a location of your choice.
3. Open the designer, which is an Eclipse installation with BIRT plugins preinstalled.
4. Under Help ➡ Install New Software click "Add" to add a new repository. Give it a name. Click Local and find the unpacked update site.
5. After it has been added, select the update site. "Tasks ODA" should now be an install candidate. Check it and click forward through the wizard until installed. 6.4 New Report in BIRT Designer
6. Do a File ➡ New Project and follow up with a right click on the project in the navigator and do a New ➡ Report.

7. Open the empty report by double clicking it in the navigator.
8. Find the Data Explorer. Right click Data Sources and click New Data Source.
9. Select Mayam Tasks Data Source. Optionally rename your instance and click next.
10. Enter the base URL for your tasks-ws. The default should work if you have the webapp running on port 8084 on the same machine as you are running the designer.
11. Enter the API account token (test:test in the example - which would be terrible in production).
12. You should now be able to successfully test the connection. Click finish when done.
13. Right click Data Sets and click New Data Set. You will be using the previously defined Data Source. Give it a name since you may end up with more than one per report. Next.
14. Enter a query. This can be one of two things.
15. Javascript expression where "s" is our subject and "p" is our parameters. Examples can be found on a separate page dedicated to expressions.
16. @Snapshot(xyz) where xyz is the name of a snapshot.
17. Continue and you will be presented with a list of columns. For tasks, this will be all configured fields. For snapshots, only timestamp and value are available. Do not edit.
18. Go to the Parameters section. Edit each parameter, copying the name and then clicking {} to the right of Linked To Report Parameter. Paste the name.
19. Set a default value that will be useful during report development or keep empty (should be empty for production reports). Click Ok.
20. Drag your Data Set onto the report and you should be able to create a good starting point.
21. Click Run ➡ View Report ➡ As HTML and you should see your new report. Make a change, save and reload the page. Repeat.

Appendix A. Actions

Action	Description
`ACKNOWLEDGE`	
`APPROVE`	Approve task. This sets the task state to approved.
`ARCHIVE`	Archive asset in MAM
`ASSIGN`	Assign task to a user
`CHECKIN`	
`CLONE`	
`CLOSE`	Close an active task.
`COMMENT`	
`COMMIT`	
`CONFIRM`	
`CREATE`	Create a new task manually.
`EDIT`	Edit a task's data without executing business rules to move the task to next state.
`ESCALATE`	
`ESCALATE_A`	
`ESCALATE_B`	
`EXTEND`	
`FAIL_A`	
`FAIL_B`	
`FINISH`	Finish task executing the business rules to move it to next state.
`KEEP`	
`LINK`	
`MATCH`	Match a task with asset in the unmatched media.
`MOVE_TO_ARCHIVE`	
`NOTIFY`	
`OPEN`	Move task to state open.
`PASS`	
`PICKUP`	Same as assign task. But is done to self.
`REJECT`	Set task state to reject and close task.
`REMOVE`	Remove task from tasklist without processing business rules.
`RESUME`	
`RETRY`	Retry last failed operation on a task.
`RETURN`	
`REVERT`	Revert task to the start or any selected state.
`SPECIAL`	
`STAGE`	
`SUSPEND`	
`TRANSFER`	Perform MAM xfer
`UNASSIGN`	Free an assigned task.

Action	Description
`UPLOAD`	Upload attachment to a task.
`TRANSCODE`	Invoke MAM transcode operation.
`INVOKE_APPLE_FCP`	Invoke FCP
`INVOKE_ASSET_WEB_PAGE`	Open MAM asset page
`INVOKE_VIZ_CAPTURE`	Open Viz Capture
`INVOKE_VIZ_CAPTURE_OUTGEST`	
`INVOKE_VIZ_CAPTURE_PLAYOUT`	
`INVOKE_VIZ_EASYCUT`	Open EasyCut.
`INVOKE_VIZ_MEDIALOGGER`	Open Media Logger.
`PEER_INVOKE_VIZ_MEDIALOGGER`	
`INVOKE_VIZ_PRECUT`	Open PreCut.
`INVOKE_VIZ_PRESEG`	
`PEER_INVOKE_VIZ_EASYCUT`	
`PEER_INVOKE_ASSET_WEB_PAGE`	
`REPORT`	Generate report (configured through jasperreports)
`TASK_HISTORY`	Show task history.

Appendix B. Activities

UI Activity	Description
ASSIGN_VALUES	Assign values to task variables/fields
UPLOAD	
DIALOG	Invoke configured dialog.
FORM	Invoke a configured form.
HIGHLIGHT_TASK	Highlight a task.
INVOKE	
PEER_INVOKE	
REFRESH_TASKLIST	
TASK_HISTORY	Show Task History

Server Activity	Description
ADD_ASSET_ACL	
ADD_TASK_STYLES	
ASSIGN_TO_GROUP	Assign task to a specified group.
ASSIGN_TO_USER	Assign task to a specified user.
CHANGE_PARENT_TASK	
CLEAR_TASK_STYLES	
CLOSE_TASK	Close a selected task.
COMMIT	
CONDITIONAL_DELETE_ASSET	
CREATE_ASSET	Create Asset in MAM.
CREATE_BPMS_PROCESS_INSTANCE	Create a task in BPMS (intalio).
CREATE_SUBTASK	Create a sub task. Eg. For sub task – Task – Promo, SubTasks – Material Selection, GFX Import, Audio Mix, Tailoring, Rough cut, Tx Approval etc.
CREATE_TASK	Create mayam task
DELETE_ASSET	Delete MAM Asset.
EDIT	Edit mayam task without processing the business rules to move it next state
ESCALATE	Escalate the task.
EXTEND	
FETCH_FROM_MAM	
FIND_UNIQUE_TASK_FOR_ASSET	Find mayam task for the specified asset/item.
KEEP	
LINK	
MATCH	Match a task with asset in the unmatched media.
MOVE_TO_ARCHIVE	
PROCESS_ATTACHMENTS	
REJECT	
REJECT_HIERARCHY	

Server Activity	Description
REMOVE	
REMOVE_ASSET_ACL	
REQUEST_REPORT	
RETIRE_ASSET	
RETURN	
REVERT	
SAVE	
SET_TASK_STATE	
SIMPLE_FILE_XFER	
SITE_ACTIVITY	
UNASSIGN	
UNINGEST_ASSET	
UPLINK	
UPDATE_BPMS_PROCESS_INSTANCE	