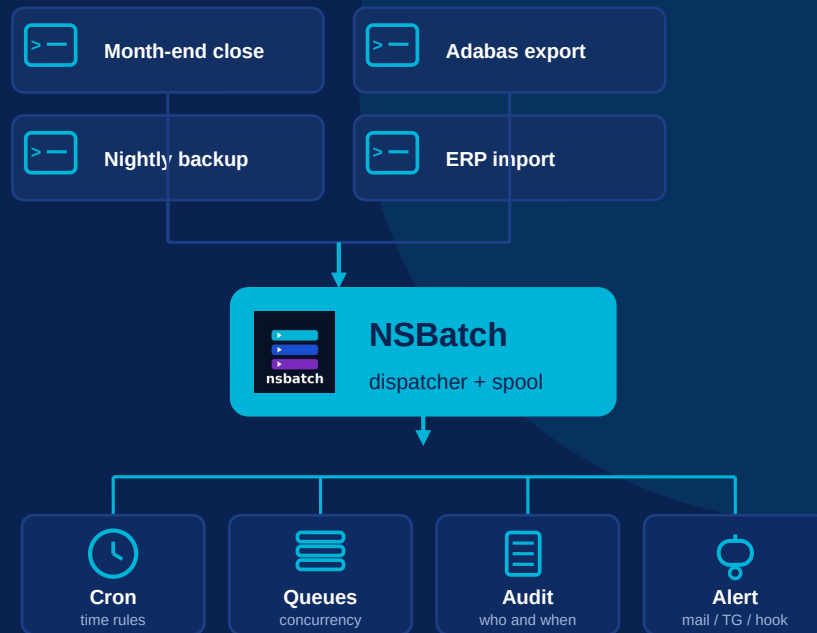


Your batch processes, queued, controlled and auditable

nsbatch queues, schedules, runs and audits every process on your server: cron-style control, queues with concurrency, live spool and alerts by email, Telegram or webhook.

[SEE FEATURES →](#)

Cron control · Queues · Auditing · Notifications



...and every execution is recorded

Queues · jobs · executions · spool · audit · notify

Batch jobs are launched blind

Loose cron scripts: no concurrency control, no central spool and no idea what is running right now.

01 No order

A month-end close and a backup fight for the CPU.
Whoever arrives first wins.

02 No visibility

Nobody knows what runs, since when, under which PID
or how long it took.

03 No trace

Logs rot in /tmp. The next day there is no way to
reconstruct the run.

04 No control

Anyone launches and anyone deletes. No roles, no
permissions, no record.

• Hours of delay

• Unseen errors

• Zero traceability

QUÉ ES

A process queue manager, mainframe style

nsbatch is Nacarsoft's JES: every job enters a queue, the dispatcher runs it under an isolated PID and its spool stays on disk. It is launched from the web, the CLI, the API or a cron rule; and who did it, and when, is always recorded.

- ✓ Queues with priority, concurrency and time windows
- ✓ Cron-style rules, importing the crontab you already had
- ✓ One isolated PID per run: if the process dies, the manager lives on
- ✓ Spool on disk with live tail, search and download

4

queues out of the
box

9

RBAC permissions

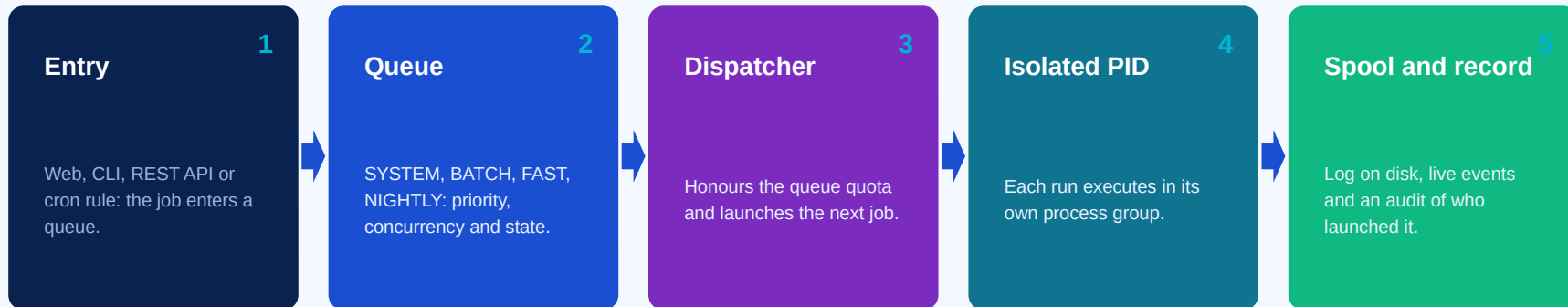
6

languages and
themes

1

single binary

How a process travels, step by step



⚠ If a job hangs, the timeout kills its process tree: the dispatcher never blocks.

No external cron

No loose scripts

All in one binary

Four ways to launch the very same job

1

By hand, web or CLI

`nsbatch submit CIERRE --params "--mode full":` it queues and returns the execution ID.

2

Through the API

`POST /api/jobs/{id}/submit:` your ERP or Natural queues the job without touching the console.

3

With a cron rule

Time and days per job, with the next run computed and the option to catch up the missed window.

4

Importing the crontab

Paste the crontab you already had: `nsbatch` parses it and leaves the job scheduled.

What lives in `/etc/crontab` today moves to the panel: with state, with spool and with alerts.

Scheduling, but with state and with alerts

- One rule per job: time, days of the week and days of the month
- Next runs computed and shown in the global view
- Catch-up of the missed window when the service starts
- Rule history: what ran, when and with which result
- Import of the existing crontab and a global view of every rule



0 2 * * 6 → BACKUP-NOCHE

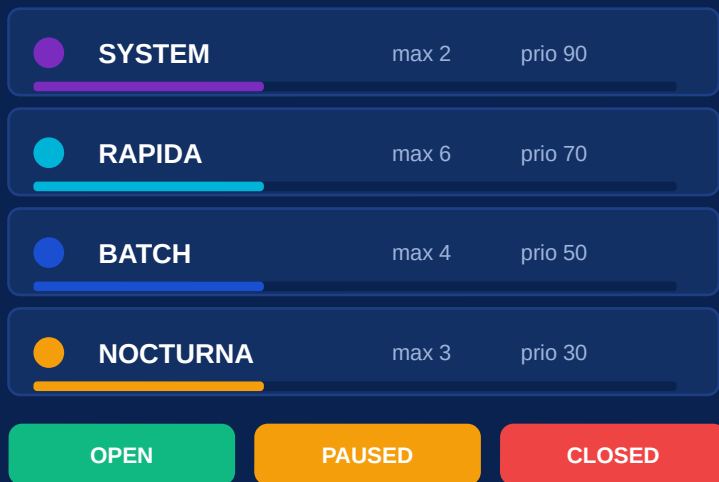
sábados y domingos a las 02:00, cola NOCTURNA

The screenshot shows the 'nav.sched' interface with a table of scheduled tasks. The table has columns for ID, JOB, SCHEDULE, QUEUE, USER, ON, STATUS, RUNS, LAST, NEXT, and ACTIONS. There are 5 rows of tasks listed.

ID	JOB	SCHEDULE	QUEUE	USER	ON	STATUS	RUNS	LAST	NEXT	ACTIONS
#3	BACKUP-NOCHE /tmp/openside/ada/bin/backup.sh --dest /backup	02:00 Mon-Sat	NOCTURNA	cl	Enabled	▶ 0 m0	—	2026-10-24 02:00		[icon] [icon] [icon]
#4	REORG-CUSTOMERS /tmp/openside/ada/bin/cierre.sh --file 306	02:15 Mon-S	SYSTEM	cl	Enabled	▶ 0 m0	—	2026-10-21 03:15		[icon] [icon] [icon]
#2	EXPORT-LEDGER /tmp/openside/ada/bin/export.sh --format csv	06:45 Mon 1,2,3,4,5	RAPIDA	cl	Enabled	▶ 0 m0	—	2026-10-20 06:45		[icon] [icon] [icon]
#5	IMPORT-ORDENES cp -R /data/ordenes /tmp	07:00 Mon 1,2,3,4,5	BATCH	cl	Enabled	▶ 0 m0	—	2026-10-20 07:00		[icon] [icon] [icon]
#1	CIERRE-MENSUAL month: full	23:30 Mon-S	BATCH	cl	Enabled	▶ 0 m0	—	2026-10-19 23:30		[icon] [icon] [icon]

Scheduled: every rule, its next run, its executions and its history.

Queues with concurrency, priority and colour



OPEN ejecuta · PAUSED retiene · CLOSED rechaza

- Four queues out of the box: SYSTEM, BATCH, FAST and NIGHTLY
- Concurrent-run quota and priority per queue
- OPEN, PAUSED and CLOSED: pause or close without losing jobs
- Time windows per queue and moving a queued job to another one

NAME	STATUS	CONC.	PRIORITY	SCHEDULES	ACCIONES	ACTIONS
SYSTEM Cada 60 minutos, máxima prioridad	CLOSED	0 / 2 - Q0	90	00:00	[Open] [Pause] [Close]	
RAPIDA Trabajos cortos interactivos	OPEN	0 / 6 - Q0	70	00:00	[Open] [Pause] [Close]	
BATCH Procesos batch general	OPEN	0 / 4 - Q0	50	00:00	[Open] [Pause] [Close]	
NOCTURNA Trabajos nocturnos, cierre contable	OPEN	0 / 3 - Q0	30	00:00	[Open] [Pause] [Close]	

Queues: quota in use, priority, colour and queued jobs for each one.

The catalogue: a job is a command and its rules

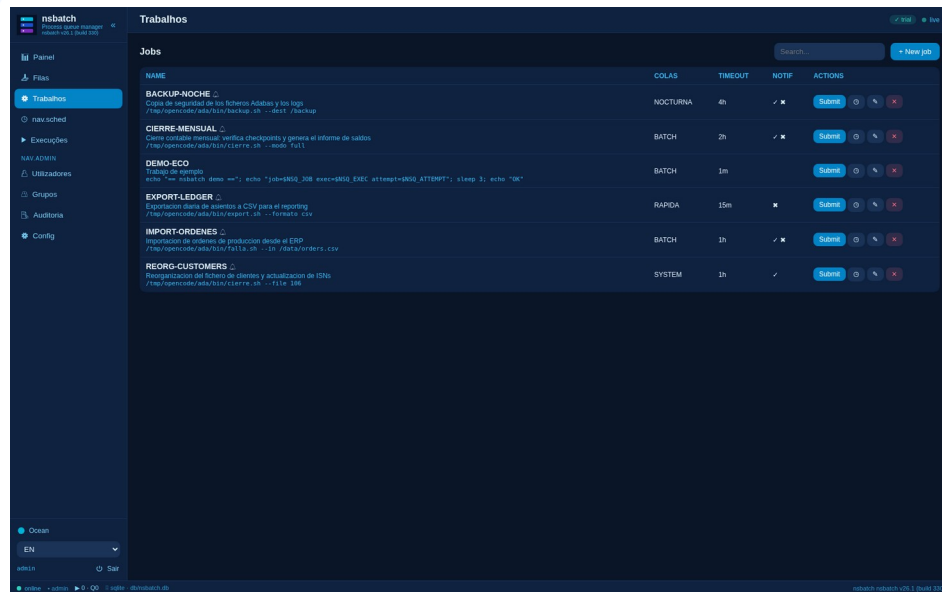
- Command and arguments: the job is a real shell line
- Target queue and the parameters of every run
- Timeout per job, with a process-tree kill
- Alerts on start, on end and on error, with recipients
- Retries and re-runs with the very same parameters

CIERRE-MENSUAL

```
/opt/ada/bin/cierre.sh --modo full
```

BATCH · timeout 7200s · aviso al terminar y al fallar

The same job can be assigned to several queues: the one the operator picks or the one of its cron rule.



Jobs: command, queue, timeout, alerts and retries for each one.

EJECUCIONES

Every execution leaves its spool

QUEUED

In the queue,
waiting its turn

HOLD

Held by hand

RUNNING

With its own PID

OK

Finished fine

ERROR

Finished with an
error

- States: QUEUED, HOLD, RUNNING, OK, ERROR and CANCELLED
- One PID per execution and an attempt number
- Exit code, wall time, CPU and peak memory
- Spool on disk with live tail, search and download
- Cancel, hold, move or re-run without leaving the panel

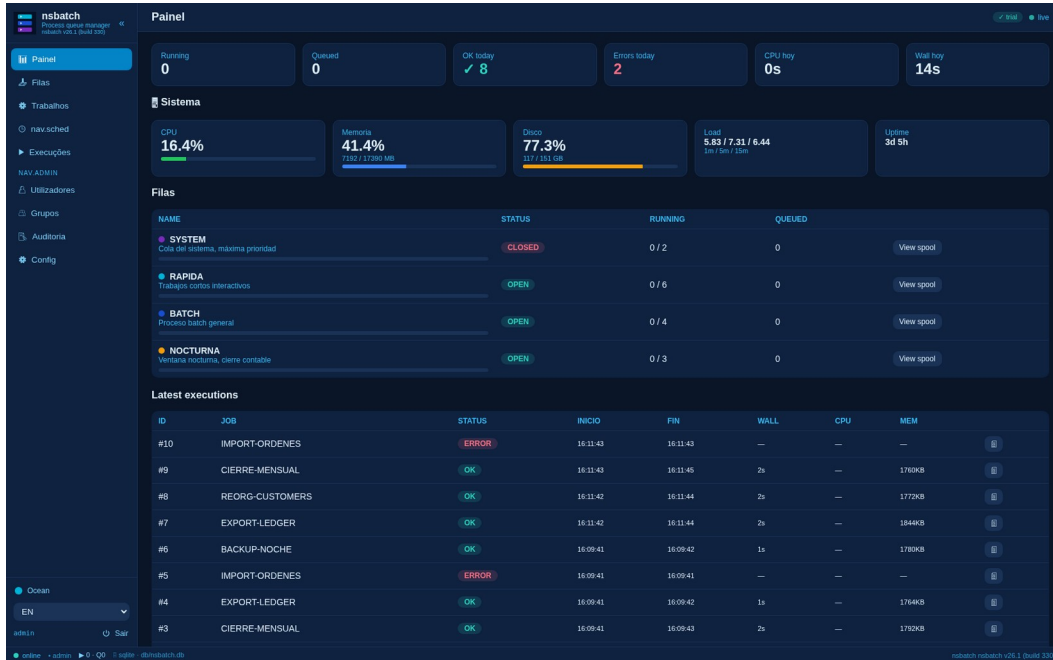
PID 2535367 · exit 0 · wall 2s · mem 1.7 MB

The screenshot shows the nsbatch web interface. On the left is a sidebar with navigation options like Files, Traducción, new batch, Ejecuciones (selected), Variables, Grupos, and Config. The main area is titled 'Ejecuciones' and contains a table of execution jobs. The table has columns for ID, JOB, QUEUE, STATUS, PID, and CREATED. A modal window is open over the table, showing details for 'spool #10' which is in an 'ERROR' state. The modal displays a log of the execution attempt, including the command used, the directory being processed, and the error message: 'ERROR: no such file or directory: /data/orders.csv'. At the bottom of the modal, there are buttons for 'Download log', 'Tail', and 'Close'.

ID	JOB	QUEUE	STATUS	PID	CREATED
#10	IMPORT-ORDENES	BATCH	ERROR	2530678	2026-10-19 16:11:42
#9	CIERRE-MENSUAL	BATCH	OK	2530679	2026-10-19 16:11:42
#8	REORG-CUSTOMERS	SYSTEM	OK	2530684	2026-10-19 16:11:42
#7	EXPORT-LEC				
#6	BACKUP-NO				
#5	IMPORT-ORD				
#4	EXPORT-LEC				
#3	CIERRE-MEN				
#2	CIERRE-MEN				
#1	CIERRE-MEN				

The spool: header with the job, the attempt, the dates and the exit code, plus the full log.

The whole state on a single screen



- Live KPIs: running, queued, OK and errors today
- CPU and wall time today, with the real cost of every job
- Per-queue semaphore with the concurrency quota in use
- Latest executions with their PID and spool in one click
- System state: CPU, memory, disk, load and uptime

Running

Queued

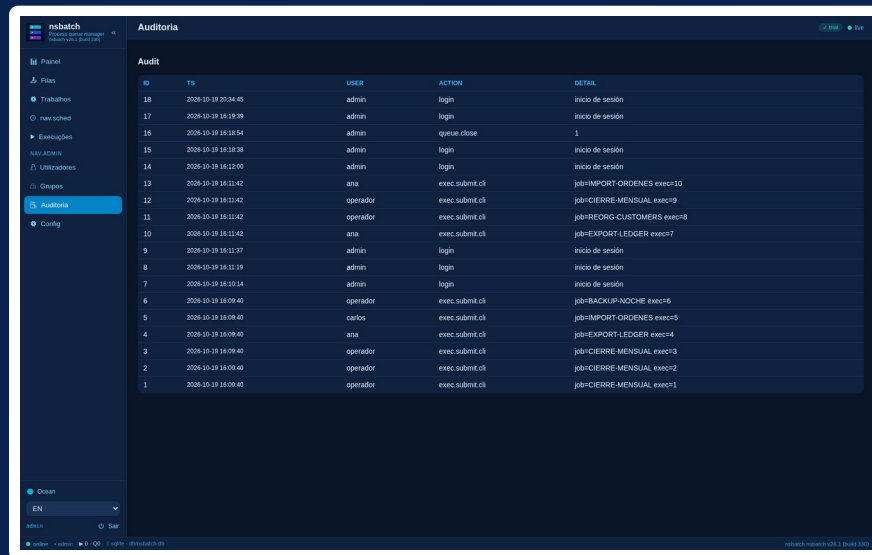
OK today

Errors today

AUDITING

Who launched what, and everything is recorded

- Every panel action goes into the audit log
- User, action, detail, timestamp and the queue involved
- Job and queue creations, state changes and permission changes
- Cancellations, re-runs and moves between queues
- Search and filter by user, action and date



The screenshot shows the 'Auditoria' (Audit) section of the nsbatch application. On the left is a sidebar with navigation options: Panel, File, Trabajos, new sched, Ejecuciones, NAV ADMIN, Utilizadores, Grupos, **Auditoria** (selected), and Config. The main area displays a table titled 'Audit' with columns: ID, TS, USER, ACTION, and DETAIL. The table contains 18 rows of log entries. At the bottom, there are filters for 'Onsai' and 'EN', and buttons for 'cancel', 'OK', and 'Bar'.

ID	TS	USER	ACTION	DETAIL
18	2026-10-19 20:04:40	admin	login	inicio de sesión
17	2026-10-19 18:19:39	admin	login	inicio de sesión
16	2026-10-19 18:18:54	admin	queue.close	1
15	2026-10-19 18:18:38	admin	login	inicio de sesión
14	2026-10-19 18:12:00	admin	login	inicio de sesión
13	2026-10-19 18:11:42	ana	exec.submit.cl	job=IMPORT-ORDENES exec=10
12	2026-10-19 18:11:42	operador	exec.submit.cl	job=CIERRE-MENSUAL exec=19
11	2026-10-19 18:11:42	operador	exec.submit.cl	job=REORG-CUSTOMERS exec=8
10	2026-10-19 18:11:42	ana	exec.submit.cl	job=EXPORT-LEDGER exec=7
9	2026-10-19 18:11:37	admin	login	inicio de sesión
8	2026-10-19 18:11:19	admin	login	inicio de sesión
7	2026-10-19 18:10:14	admin	login	inicio de sesión
6	2026-10-19 18:09:40	operador	exec.submit.cl	job=BACKUP-NOCHE exec=6
5	2026-10-19 18:09:40	carlos	exec.submit.cl	job=IMPORT-ORDENES exec=5
4	2026-10-19 18:09:40	ana	exec.submit.cl	job=EXPORT-LEDGER exec=4
3	2026-10-19 18:09:40	operador	exec.submit.cl	job=CIERRE-MENSUAL exec=3
2	2026-10-19 18:09:40	operador	exec.submit.cl	job=CIERRE-MENSUAL exec=2
1	2026-10-19 18:09:40	operador	exec.submit.cl	job=CIERRE-MENSUAL exec=1

Append-only, with user and IP, and exportable from the panel.

Audit log: what happened on the server and who did it.

NOTIFICATIONS

The process tells you when it is done

Every job decides whether to alert on start, on end or only on error. nsbatch sends the alert through the channels you choose and records it in the spool.

Email (SMTP)

STARTTLS or TLS, with a configurable user and sender. Several recipients per job.

Telegram

Bot + chat ID: the alert reaches the team phone without setting up any mail server.

Webhook

POST to the URL you define, with your own headers: it plugs into Slack, Teams or your ITSM.

The execution ends

OK, ERROR or CANCELLED



The alert is built

job, attempt, params, wall, CPU and exit



It goes out

email · Telegram · webhook



It lands in the spool

ok or ERROR per channel

Recipients per job, per user group or the creator of the execution.

To plug it into what you already have

CLI

```
nsbatch submit <job> --params "..."
```

```
nsbatch queues list | create | pause | open |  
close
```

```
nsbatch jobs list | create | update | cron
```

```
nsbatch execs [STATE] | cancel | rerun | move |  
log
```

```
nsbatch users | groups | roles | audit | dashboard
```

REST

```
POST /api/jobs/{id}/submit
```

```
GET /api/queues · /api/jobs · /api/executions
```

```
GET /api/cron · /api/cron/next · /api/cron/import
```

```
GET /api/executions/{id}/log
```

```
GET /api/events (live SSE)
```

The console is one of the ways in: the API and the CLI do the same without a browser.

Who sees what, and everything is recorded

Access control

- RBAC with 9 permissions
- Users, groups and roles
- Permission per operation
- No binary admin

Session and access

- HS256 JWT in a cookie
- Passwords with bcrypt
- Recovery by email
- Configurable expiry

Traceability

- Audit of every action
- User and client IP
- Log per queue and per job
- Configurable retention

config.yaml 0600 · database 0700 · bcrypt · JWT HS256 · 30-day trial · machine-signed license

FEATURES

Everything an operator needs

6 themes

Dark, Ocean, Light, Forest, Sunset and high contrast.

6 languages

ES, EN, FR, DE, IT and PT across the UI.

RBAC

Users → groups → roles → permissions.
No binary admin.

API + SSE

Script integration: REST submit and live events.

Timeouts

Timeout and process-tree kill per execution.

Retries

Re-run with the same parameters in one click.

Retention

Automatic purge of old spools and execution history.

License

30-day trial and a machine-signed license.

GETTING STARTED

From zero to queued jobs, in three steps

1

Create the environment

nsbatch init /opt/nsbatch: it creates config.yaml, the SQLite database, the spool and the admin user.

2

Tune and start

Fill in the JWT secret, the port and the alert channels. Start with nsbatch start.

3

Register the job

Create the queue, the job and its cron rule, or import the crontab you already had.

REQUIREMENTS

- Linux (RHEL 8 / 9)
- A single binary, no dependencies
- SQLite for state and spool
- Reverse proxy with HTTPS in production
- Web panel on port 11050

```
nsbatch init /opt/nsbatch → nsbatch start /opt/nsbatch → web panel :11050
```

FREQUENTLY ASKED

The questions we hear most

Do I have to drop the system cron?

Not necessarily, but you can: nsbatch imports the existing crontab and leaves it scheduled with its state, its spool and its alerts.

What if a job hangs?

The job timeout kills its process tree and the execution goes to ERROR. The dispatcher keeps going with the queue.

Can I run several environments?

Yes: each instance has its own directory, database, spool and port. They are registered by name for quick switching.

Who may launch a job?

Every permission is an operation: submit, cancel, re-run, move, view the spool or manage users. Granted by role and group.

Can it integrate with my ERP?

Yes: the REST API and the CLI expose the same job catalogue. POST /api/jobs/{id}/submit queues the execution.

What does the audit log show?

Every panel action: who queued, who cancelled, who created a queue or changed a permission, with user, time and IP.

Your processes, queued and fully accounted for

Try nsbatch with your own jobs: we set it up and watch it run on your data.

[Guided demo](#)

[Try it in your environment](#)

[Let's talk about your case](#)

nacarsoft.com · support@nacarsoft.com

