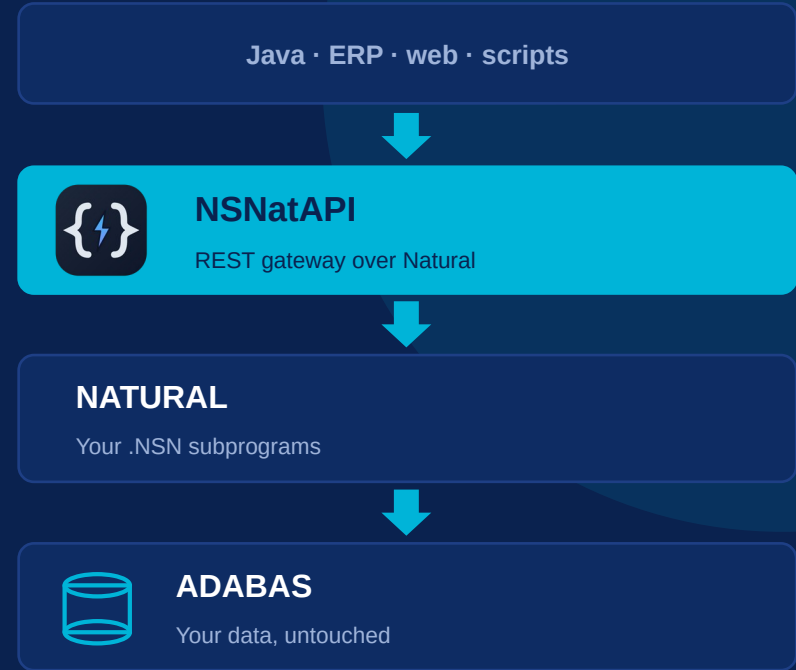


Your Adabas data, exposed as a REST API

NSNatAPI turns your Natural subprograms into secure REST endpoints: Bearer tokens, RBAC, auditing and multi-environment. No EntireX, no RPC broker, not a single line of business logic rewritten.

[SEE HOW IT WORKS →](#)

26.1 build 185 · 2 execution modes · ~1–5 ms per call



One Natural subprogram, two execution paths

THE PROBLEM

Adabas is a great data store, but it is not an API

01

The silo

Your data lives in Adabas and is reached through Natural subprograms. For a web app, an ERP or a phone, that is not a contract: it is an INTEGRATION.

02

The toll

The usual way is EntireX or an RPC broker: licences, a broker to run and maintain, and a deployment that breaks the day something changes.

03

The risk

Exposing Natural to the network without control: no tokens, no scopes, no auditing, no concurrency or time limits.

- Without an API, every app rebuilds the connection

- With EntireX, the infrastructure is the project

- Without control, security is a problem

WHAT IT IS

A gateway, not a proxy

NSNatAPI sits between your applications and Natural. Your business programs stay exactly the same. NSN: they are not rewritten, not compiled by nsnatapi, and their contract never changes. What changes is how callers reach them.

- ✓ REST endpoints defined in the console: method, path, environment, program and fields.
- ✓ Two execution modes per endpoint: batch (ephemeral process) or session (persistent pool).
- ✓ Bearer tokens scoped per endpoint, with expiry, revocation and IP allowlist.
- ✓ Admin console with RBAC (roles and groups), TOTP 2FA and hot configuration.
- ✓ Every call audited with metadata only: never the request or response bodies.

0

EntireX, brokers or
daemons in Natural

1–5 ms

per call in session
mode (measured)

2

execution modes
per endpoint

195k

calls audited in
the certification

ARCHITECTURE

From application to data: two paths, one program

Consuming services

Java / Backend

ERP

Scripts & cron

Microservices

Web portal

Mobile app

iPaaS / SOAP



NSNatAPI — REST gateway

REST /api/ext · Bearer + RBAC · rate limit · auditing · interactive OpenAPI

REST /api/ext

HTTP 200 · rc · output · duration_ms

BATCH MODE

One ephemeral Natural process per call

LOGON → CALLNAT → FIN. Total isolation. 30–90 ms.

SESSION MODE

Pool of persistent sessions that open the connection

LOGON to the library and the session STAYS OPEN, reused. 1–5 ms.

NATURAL

Natural runtime · NSNAT libraries · NATBP buffer pool · FUSER per environment

Open connections: 1 per pooled session (one buffer-pool user each)



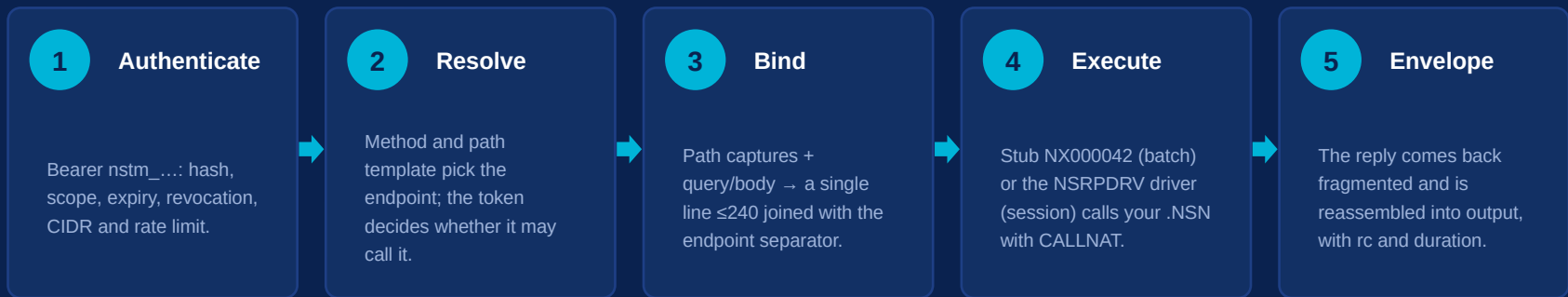
ADABAS

Business files: READ / WRITE / ST

Same .NSN subprogram, same contract, same data. Only the caller of the connection — and how long it is kept — changes.

HOW IT WORKS

From the HTTP request to your subprogram



```
curl -H "Authorization: Bearer nstm_..." \
      http://host:11032/api/ext/v1/saldo/001234
```

```
HTTP/1.1 200 OK
{ "ok": true, "rc": 0,
  "output": "SALDO 1234.56",
  "duration_ms": 22 }
```

HTTP 200 = it ran. The business error travels in rc: the client always tells the two apart.

output · rc · data · duration_ms

EXECUTION MODES

Two modes, one program contract

BATCH (default)

One Natural process per call

Start-up	~30–90 ms on every call
----------	-------------------------

Isolation	Total: a new process per call
-----------	-------------------------------

CPU / start-up	High under heavy load
----------------	-----------------------

Risk	None: ephemeral process
------	-------------------------

Best for	Low/medium traffic, strict isolation
----------	--------------------------------------

SESSION (opt-in)

Pool of persistent Natural sessions

Start-up	~1–5 ms with a warm pool
----------	--------------------------

Isolation	Session shared across calls
-----------	-----------------------------

CPU / start-up	Low: the pool is reused
----------------	-------------------------

Risk	A fatal error recycles the session (automatic)
------	--

Best for	Polling and high-frequency lookups
----------	------------------------------------

Pick the mode per endpoint, in the console (Endpoints → Execution mode) or via API with `exec_mode`.

Your programs do not change. Only the input and the output.

```
0010* CPOLIZA - policy lookup
0020DEFINE DATA PARAMETER
00301 #PARM-IN (A) DYNAMIC
00401 #PARM-OUT (A) DYNAMIC
00501 #RESP (I4)
0060END-DEFINE
0070RESET #PARM-OUT #RESP
0080* ... your business logic ...
0090MOVE 'BAL 1234.56' TO #PARM-OUT
0100MOVE 0 TO #RESP
0110END TRANSACTION
0120END
```



Three parameters, always

#PARM-IN (A) DYNAMIC, #PARM-OUT (A) DYNAMIC and #RESP (I4).
The house convention.



End with END

Never STOP or LOGON: in session mode they would kill the driver.



No state between calls

No globals or AIVs; always END TRANSACTION or BACKOUT.



Short name

A-Z0-9-_, max 8 characters, uppercase.

nsnatapi never recompiles your programs: it only generates and compiles the stub or driver that calls them.

RESPONSES

The business decides the outcome; HTTP only carries it

HTTP 200 · the envelope

ok	true	The call ran
rc	0	Business #RESP: 0 = ok, >0 = business error
output	"BAL 1234.56"	Raw #PARM-OUT text (verbatim)
data	{ "balance": 1234.56 }	Only in the fields and json modes
duration_ms	22	Call duration

rc ≠ 0 is still HTTP 200. The client tells a business error from a technical failure.

Machine API error codes

400	401	403	404	413	429	502	503	504
-----	-----	-----	-----	-----	-----	-----	-----	-----

Response modes

text	output only. rc is what matters.
fields	The program returns separated positions and the endpoint types them into data.
json	The program returns valid JSON and nsnatapi validates and publishes it in data.

Every request goes through a filter

Bearer tokens

192 bits, SHA-256 hash with pepper, per-endpoint scope, expiry, revocation, rotation and CIDR allowlist.

Console RBAC

Roles → groups → users. Permissions are the union. No `is_admin`. Anti-lockout for the last admin.

TOTP 2FA

Optional second factor per account, with password confirmation to disable it.

Strict validation

Everything reaching Natural or the FUSER is validated: no shell with user data, no path traversal.

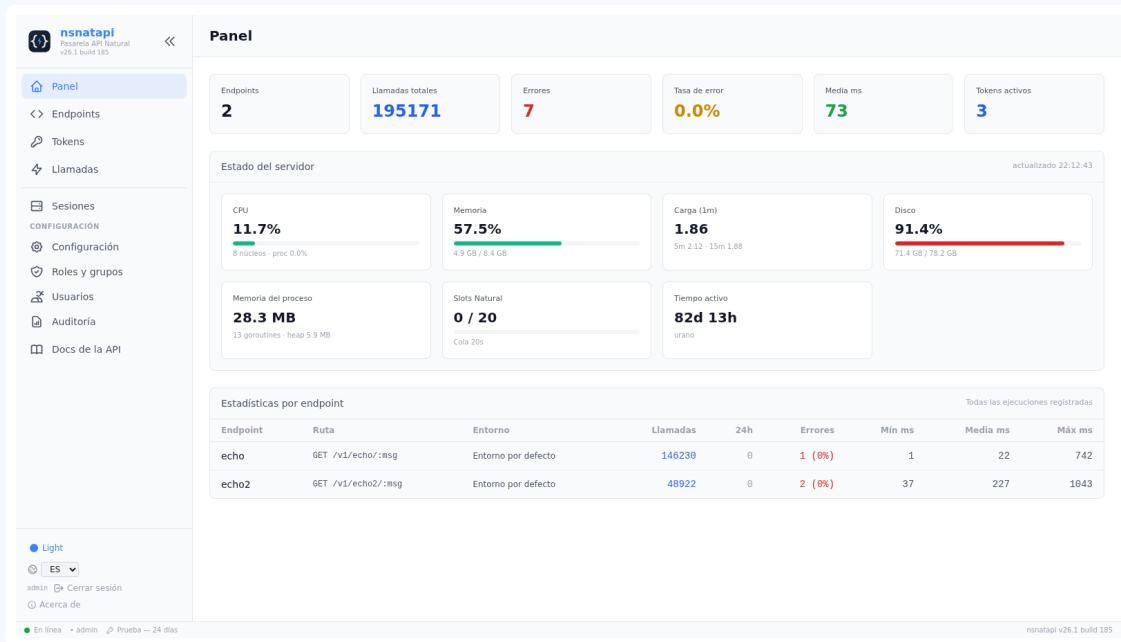
Auditing without bodies

Every call leaves metadata: environment, library, mode, PID, duration, error. Never the body.

Licence and network

Without a valid licence, login and the machine API answer 403. Only loopback is a trusted proxy by default.

What happens in Natural, in plain sight



- Live server telemetry: CPU, memory, load, disk, RSS, heap and goroutines, refreshed every 5 s.
- Session panel: PID, state, uptime, last call, calls served, errors, latency, CPU and memory per session.
- Call log with environment, library, execution mode and session PID: every call is correlated with its session.
- Hot actions: warm up, restart, kill a given session or resize the pool.

All of it from the console, without SSH-ing into the box.

CAPACITY

Measured capacity, not promises

SESSION MODE · DEV pool of 4 sessions

Clients	rps	p50	p95	Errors
1	106	7.8 ms	19.5 ms	0
4	251	13.9 ms	31.3 ms	0
8	312	23.1 ms	47.7 ms	0
16	321	46.9 ms	79.9 ms	0

Session certificate: 16 concurrent clients (~320 rps, p95 80 ms, 0 errors)

BATCH MODE · one process per call

Clients	rps	p50	p95	Errors
1	8.4	111 ms	191 ms	0
4	25.7	148 ms	230 ms	0
8	40.0	192 ms	285 ms	0
12	42.2	258 ms	483 ms	0

Batch certificate: 8 concurrent clients (~40 rps, p95 285 ms, 0 errors)

Closed-loop load with no think time and trivial programs: this is the platform ceiling, not the business one. 15-minute soak: 61,225 calls, 0 errors.

MULTI-ENVIRONMENT

One endpoint, one environment: DEV, TST and PRD never mix

DEV

6 sessions

NATBP · fuser=22,20 · FUSER
/data/ns/fusers/dev

Own FUSER + buffer pool

TST

2 sessions

NATTST · fuser=22,30 · FUSER
/data/ns/fusers/tst

Own FUSER + buffer pool

PRD

to be defined

Own buffer pool and FUSER

Own FUSER + buffer pool

- ✓ Each environment declares its Natural binary, sagenv, buffer pool, NATPARM and FUSER.
- ✓ The endpoint picks the environment; the default environment covers those that do not.
- ✓ The session pool is sized per environment: 3 sessions in DEV and 8 in TST coexist without mixing.
- ✓ Calls are recorded with their environment, and the Calls page filters by it.

The same nsnatapi binary serves all three: the destination changes, not the code.

DEV · TST · PRD

USE CASES

Where it pays off from day one

Back-office lookups

Balances, policies, catalogues: a query from the web without opening the terminal.

Capture and entry

Entry forms writing into Adabas with the business validation inside the program itself.

Payment gateways

A POST endpoint with its business rc: the app tells 'rejected' from 'down'.

Mobile radar

High-frequency lookups in session mode, at millisecond cost.

ERP / iPaaS integration

The ERP calls over HTTP; if it speaks JSON, the endpoint publishes typed data.

Progressive modernisation

Migrate endpoint by endpoint, without rewriting the Natural business logic or stopping it.

GETTING STARTED

Up and running in four steps

1 Install and start

```
nsnatapi init ./data  
nsnatapi start ./data
```

2 Create the endpoint

```
POST /api/endpoints  
{  
  method, path_template,  
  library, program,  
  environment, fields_json  
}
```

3 Deploy the stub

```
POST /api/endpoints/1/deploy  
  
(ftouch + STOW in the  
environment FUSER)
```

4 Issue a token and call

```
nsnatapi token ./data \  
  --name java-prd \  
  --endpoint saldo  
  
curl -H "Authorization: \  
  Bearer nstm_..." \  
  http://host:11032/api/ext/  
  v1/saldo/123
```

The web console (Configuration, Endpoints, Tokens, Sessions) covers the rest: RBAC, environments, auditing and hot tuning.

REQUIREMENTS

What your environment needs

Requirements

- Linux RHEL 8 or 9 (x86-64).
- Software AG Natural and Adabas client (sagenv), with the buffer pool running.
- FUSER with the business subprograms, readable and writable for deployment.
- A reverse proxy with HTTPS in production (nsnatapi does not serve TLS itself).
- One Natural per environment: DEV, TST and PRD may point at different installs.

Licence

- Per-installation licence, with machine fingerprint and validity.
- Without a valid licence, login and the machine API answer 403.
- 30-day trial on install, no key needed.
- List price: €7,900 / €6,320 annual renewal.

The licence is checked at login and on every machine call: the inside of Adabas stays protected.

NEXT STEP

Your data is already in Adabas. Only the door is missing.

NSNatAPI opens it: with tokens, with scope, with auditing and with whichever execution mode you need. Without rewriting your Natural logic and without EntireX.

1. 30-minute demo with your own subprograms
2. Trial in DEV with one real endpoint
3. Rollout to TST and then to PRD

REQUEST INFO · nacarsoft.com

Nacarsoft Soluciones Informáticas S.L. · Adabas / Natural Suite · nsnatapi 26.1 build 185

