

DOWNSTREAM / PROCESS ENGINEERING

The AspenTech Token Optimization Playbook

How to see inside the token pool, run chargeback that reconciles with the vendor report, and size the pool to real demand.

What you will take away

- Why a shared token pool hides both overspend and accountability
- How log-based metering reveals token cost per feature and per user
- How to run defensible chargeback that matches the vendor report
- A readiness checklist to bring your token model under control

The challenge

You cannot manage a pool you cannot see into

AspenTech bills from a shared pool of tokens. Different features draw different amounts, one instance of HYSYS Process alone draws 14 tokens, and the same log mixes token-based and non-token features. The pool total is not even in the log file. Without proper metering you size the pool by guesswork, top it up when engineers hit a wall, and have no defensible way to charge cost back to the teams that consumed it.

Open iT collects the raw Sentinel log files and reports maximum tokens in use and elapsed time by feature and by user, monthly, in a form that reconciles with the vendor report. It even accounts for the contract detail that usage under five minutes is not billable on some agreements.

This playbook turns an opaque token bill into a per-feature, per-user, per-team picture you can manage and charge back.

The framework

Three levels, applied to downstream and process engineering

For token models the emphasis shifts from seats to consumption and chargeback.

Level 1: Runtime usage	Collect the raw Sentinel logs and report token consumption and elapsed time per feature and per user, so the SLM_Pool stops being a black box.
Level 2: True active usage	Separate genuine consumption from incidental checkouts, including the sub-five-minute usage that some contracts do not bill.
Level 3: Managed usage	Where applicable, reclaim idle seats and consolidate license managers across plants into one console with CLIMS.

Six steps to a managed token model

- 1 Switch on log-based collection.** Use log-based Sentinel collection so your numbers reconcile with the AspenTech vendor report.
- 2 Map token cost per feature.** Build a mapping of token cost per feature (for example HYSYS Process at 14 tokens); costs update on renewals.
- 3 Report per user and per feature.** Produce monthly tokens and elapsed time by user and feature, net of the sub-five-minute rule.
- 4 Run chargeback.** Allocate cost to teams by token-hours, turning a shared line item into an accountable one.
- 5 Size the pool.** Set the token pool and seat counts to the demand you can now see, and stop over-topping-up for a rare peak.
- 6 Consolidate the plants.** Bring process-engineering managers across sites into one CLIMS console for one view of denials and failures.

Qualify yourself

Readiness checklist

The more of these that are true, the more you stand to recover. Tick what applies today.

- ☐ We buy AspenTech or similar tools on a token or credit model
- ☐ We cannot see token cost per feature or per user today
- ☐ Finance wants to charge software cost back to specific teams
- ☐ We top up the token pool reactively when engineers hit a wall
- ☐ We run several process-engineering license servers across plants

Your next step

Get a free token utilization snapshot. Open iT reads your AspenTech Sentinel log and reports real per-feature and per-user consumption, reconciled to the vendor report.

Pair this with the Token Savings and Chargeback Calculator to size over-provisioning and model team chargeback.

Sources: Open iT Sentinel (AspenTech) collection documentation (draft). Note current caveats: automatic token-cost collection and License Monitor support are limited; a mapping file may be required. Confirm release status before publishing.