Agent Tool Exit Plan

A five-day plan for moving your AI agent workflow when a tool is sold, shut down or replaced.

Who it's for: engineering leads and platform teams whose developers run AI coding agents through a hosted tool.
Time needed: two hours for the inventory and test; five working days for a full move.
Use it: as a drill this quarter, or on the day a shutdown notice arrives. Companion to the Delphilens insight "Two Weeks to Pack".

01 How to use this pack

Agent tools are now part of how software gets built, but their most valuable contents are rarely code. When a tool is bought, closed or replaced, the risk is losing the way your team works: the instructions, templates, history and approvals that live in the vendor's cloud.

Who should be in the room

- The engineering lead who owns the agent workflow
- One engineer who runs agents daily
- Whoever owns the vendor contract
- Security, for the keys and access review

Order of work

- Section 2: the two-week test (15 minutes)
- Sections 3–4: inventory and triage (90 minutes)
- Section 5: the five-day plan
- Sections 6–10: templates, next tool, contract terms, drill, sign-off

02 The two-week test

Answer yes or no for your team today. Every "no" points to the section that fixes it.

#	QUESTION	YES / NO	FIX IN SECTION
1	Are your agent instructions stored as files in your own repository?		05, Day 3
2	Could you export every run log and approval record today, without contacting support?		05, Day 2
3	Are the model accounts and keys in your company's name?		05, Day 2
4	Do you know the vendor's notice period for shutdown or change of control?		08
5	Could one recurring job run tomorrow without the tool's cloud?		05, Day 4
6	Has anyone on the team done this before, as a drill?		09

Three or more "no" answers: run the five-day plan as a drill this quarter, before you need it.

03 Inventory

One row per artefact. Fill in where it lives in your set-up and who owns it. The same table comes as a spreadsheet with this pack.

ARTEFACT	TYPICALLY LIVES	WHY IT MATTERS	HOW TO GET IT OUT	WHERE (YOURS)	OWNER
Agent instructions and system prompts	Tool dashboard or workspace settings	They encode what “good” looks like for your code base, tuned over months.	Copy into plain files in the repository (for example AGENTS.md); review like code.		
Task templates and recipes	Tool library or saved prompts	Recurring jobs (tests, dependency bumps, triage) depend on them.	Export or copy each template into a /agents/tasks folder, one file per task.		
Schedules and triggers	Tool scheduler	Overnight and event-driven runs stop silently when the tool goes.	List every schedule with its time, scope and owner; recreate in CI or the new runner.		
Run history and logs	Vendor cloud	The record of what each agent tried, and why, is how you learn and how you explain decisions.	Use the tool’s export or API; otherwise script a nightly copy to your own storage.		
Approval and review records	Vendor cloud, sometimes PR comments	Auditors and incident reviews ask who approved what, and when.	Export with timestamps and reviewer identity; link each record to its commit.		
Workspace and model settings	Tool settings	Model choice, limits and sandbox rules change agent behaviour.	Write them down as configuration in the repository, not screenshots.		
Secrets and API keys	Tool secret store	Keys held by the vendor die with the account.	Rotate every key the tool held; store new ones in your own secret manager.		
Model provider accounts and billing	Your provider, or the vendor’s	If the vendor resold model access, your usage and limits vanish with it.	Move to accounts in your company’s name; record current spend and limits.		
Integrations	Tool integrations page	GitHub apps, chat, ticketing and CI connections break without notice.	List each integration, its permissions and owner; revoke the old ones after cut-over.		

Custom tools and MCP server configs	Tool settings or a vendor-hosted runtime	Agents lose capabilities they relied on.	Keep definitions in the repository; confirm each server runs without the vendor.		
Evaluation sets and test prompts	Tool or a team member's laptop	Without them you can't tell whether the new set-up is as good.	Move into the repository with expected outcomes; run them on Day 4.		
Cost and usage data	Vendor dashboard	Needed to compare the new tool fairly and to budget.	Export the last 90 days by team, task type and model.		
User accounts and permissions	Tool admin console	Shows who could run what; needed for the security review of the replacement.	Export the user list and roles; remove access once migrated.		
Unmerged work in vendor workspaces	Vendor cloud branches or sandboxes	Half-finished agent work disappears with the cloud.	Push every live branch to your repository, then decide: merge, park or discard.		

04 Triage: what to move first

Score each inventory row twice, from 1 (low) to 3 (high). **Impact:** how much the team would suffer without it. **Portability risk:** how hard it is to get out and rebuild. Multiply the two.

	Portability risk 1	Portability risk 2	Portability risk 3
Impact 3	3 · Day 3–4	6 · Day 1–2	9 · Day 1
Impact 2	2 · Archive	4 · Day 3–4	6 · Day 1–2
Impact 1	1 · Archive	2 · Archive	3 · Day 3–4

Patterns we see often

Agent instructions and run history usually score 9: high impact, and stuck in the vendor's cloud.

Secrets score high on risk but are quick to fix: rotate them on Day 2 whatever their score.

Cost data is low impact but cheap to export. Take it anyway; you'll want it when comparing tools.

05 The five-day plan

Five working days, one goal per day. Keep shipping code throughout: only agent configuration is frozen.

Day 1 Freeze and take stock

Owner: Engineering lead

- [ ] Announce a change freeze on agent configuration (not on code).
- [ ] Fill in the inventory with one row per artefact, even if the answer is "don't know".
- [ ] Score each row for impact and portability risk (section 4).
- [ ] Push every unmerged branch from vendor workspaces to your repository.
- [ ] Confirm the vendor's key dates in writing: read-only, shutdown, deletion.

Done when: A complete, scored inventory and a list of key dates.

Day 2 Get everything out

Owner: Named engineer

- [ ] Export run history, approval records and usage data for the period you need to keep.
- [ ] Copy agent instructions and task templates into the repository, unchanged for now.
- [ ] Rotate every secret and key the tool held.
- [ ] Store exports somewhere you control, with the export date in the file name.
- [ ] Check each export opens and is complete. A file that won't open isn't a backup.

Done when: Verified exports in your own storage; rotated keys.

Day 3 Move the way of working into the repository

Owner: Engineering lead and one reviewer

- [ ] Turn the copied instructions into a reviewed AGENTS.md (template in section 6).
- [ ] Move model choice, limits and task definitions into configuration files.
- [ ] Recreate schedules in CI or the replacement runner, disabled for now.
- [ ] Write down anything the team did by habit that no file captures.
- [ ] Open one pull request with all of the above and get it reviewed.

Done when: One reviewed pull request that holds your agent set-up as code.

Day 4 Run both, compare

Owner: Named engineer

- [ ] Run your evaluation set and three real tasks on the replacement.
- [ ] Compare results with the old tool’s recent runs: quality, time, cost.
- [ ] Fix gaps in instructions rather than working around them by hand.
- [ ] Enable the recreated schedules for one low-risk job.

Done when: Evidence that the replacement does the work at least as well.

Day 5 Cut over and sign off

Owner: Engineering lead

- [ ] Switch remaining schedules to the replacement.
- [ ] Revoke the old tool’s integrations and user access.
- [ ] Archive the exports with a retention date.
- [ ] Complete the sign-off sheet (section 10).
- [ ] Book the first quarterly fire drill (section 9).

Done when: A signed-off move, with nothing left in the old tool that you need.

06 Templates

AGENTS.md starter

A plain file at the root of the repository that most coding agents read. Keep it short, specific and reviewed like code.

```
# AGENTS.md

## Purpose
What this repository does, in two sentences.

## Map
- /src: application code
- /tests: how we test, and which suites agents must run
- /agents/tasks: one file per recurring agent task

## Running and testing
Exact commands for build, lint and tests. Agents run all of them before proposing a change.

## Standards
Naming, style, error handling and logging rules that reviewers check.

## Agents may
- Write tests, fix failing tests, update dependencies within minor versions
- Propose changes up to N files or M lines

## Agents must not
- Change authentication, payments or data deletion code without a named reviewer
- Add new dependencies or external calls without approval

## Review
Who reviews what. Changes above the limits need two people.

## When stuck
Stop, summarise what was tried, and open a draft pull request with questions.
```

Run log export (one row per agent run)

`run_id`, `started_at`, `finished_at`, `agent`, `model`, `task_template`, `repository`, `branch`, `commit_sha`, `status`, `tokens_in`, `tokens_out`, `cost`, `reviewer`, `decision`, `decision_at`, `notes`

Approval record (one row per approved change)

`change_id`, `commit_sha`, `pull_request`, `risk_level`, `approver`, `approver_role`, `approved_at`, `checks_passed`, `reason`

07 Before you choose the next tool

Ask every candidate these questions, in writing. Vague answers to the first two groups are a warning sign.

Data ownership and export

- [ ] Do we own prompts, configurations, logs and outputs, in writing?
- [ ] Can we export all of them at any time, in a documented open format, without asking support?
- [ ] Is there an API for run history and approval records?
- [ ] Is our data used to train models? Can we opt out by default?

Shutdown and change of control

- [ ] How much notice do we get before a shutdown, acquisition-related change or price change?
- [ ] What happens to our data after we leave: how long is it kept, and do we get a deletion confirmation?
- [ ] Will the service stay available, unchanged, during the notice period?
- [ ] Has the vendor shut down or materially changed a product before? How was it handled?

Security and access

- [ ] Where is our data processed and stored?
- [ ] Can we use our own model accounts and keys?
- [ ] Is single sign-on supported, and are actions logged per user?
- [ ] Which sub-processors are involved, and are we told about changes?

Lock-in signals

- [ ] Does the tool read instructions from files in our repository, or only from its own dashboard?
- [ ] Can the same workflow run from the command line or CI without the vendor’s cloud?
- [ ] How much of our set-up would we have to rebuild by hand to leave?
- [ ] Is there an independent alternative we could switch to within a week?

08 Contract terms worth asking for

Terms that turn a two-week scramble into a planned move. Not every vendor will agree to all of them; the conversation itself tells you a lot.

TERM	WHAT TO ASK FOR
Notice of shutdown or change of control	At least 90 days' written notice before service ends, or before material changes following an acquisition.
Data return	Full export of your data, configuration and logs in a documented, machine-readable format, at no extra cost, during the notice period.
Deletion confirmation	Written confirmation that your data has been deleted after export, within an agreed period.
Ownership	You own your prompts, configurations, logs and outputs. The vendor gets only the rights needed to run the service.
No training without opt-in	Your data is not used to train or improve models unless you opt in.
Sub-processor changes	Advance notice of new sub-processors, with the right to object or leave.
Continuity	Service levels and APIs remain in place, unchanged, through the notice period.

This is practical guidance, not legal advice. Agree final wording with your legal team.

09 Quarterly fire drill (30 minutes)

- [ ] Pick one recurring agent job at random.
- [ ] Run it from the repository alone, on the fallback runner, without opening the main tool.
- [ ] Time it. Note anything you had to look up in the tool's dashboard; move it into the repository.
- [ ] Check the latest exports exist and open.
- [ ] Confirm the list of keys the tool holds is current.
- [ ] Record the result and the date in the inventory.

10 Sign-off

Complete at the end of Day 5, or after each drill.

ITEM	SIGNED BY	DATE
Inventory complete and scored		
Exports verified and stored		
Keys rotated		
Agent set-up merged into the repository		
Replacement passed evaluation set		
Old integrations and access revoked		
Exports archived with retention date		
Fire drill booked		