AVARATAK

CASE STUDY · ENTERPRISE SOFTWARE

Bringing a 1,000-User Jira Environment Back Under Control

A governance-first cleanup of workflows, screens, filters, and automation that preserved the legacy still doing real work.

ATLASSIAN Solution Partner

Enhancing Teams, Enabling Success

THE SITUATION

One reasonable local decision at a time, for years

Roughly 1,000 users run product, engineering, support, and program management work through a single Jira Cloud site. Jira is the system of record for delivery: boards drive standups, dashboards feed leadership reporting, and automation moves work between teams.

Administrative rights had been spread across many hands for years. Each new team received a cloned configuration, adjusted it, and moved on. Nobody could say what was safe to change, so administrators practiced a policy of add, never remove.

Change requests stalled while someone tried to work out what might break. Trust in Jira data eroded, and spreadsheet workarounds started to reappear.

Workflows

Hundreds of near-duplicates, differing by a renamed status or one extra transition.

Screens & fields

Cloned screens carrying fields nobody filled in. Three separate fields recorded when work was due.

Filters

Thousands saved. Many broken or shared with the entire site, silently feeding dashboards.

Automation

Overlapping triggers, rules reacting to other rules, and owners long gone from the company.

THE AVARATAK DIFFERENCE

Knowing what is safe to delete in a decade-old Jira environment is judgment, not documentation.

Senior-led delivery

Avaratak engagements are delivered by Senior-level or higher consultants with at least five years of hands-on Atlassian experience.

Governance before cleanup

Deletion without ownership and evidence is guesswork. We establish who owns each object, and what actually uses it, before anything is removed.

Advice, not agreement

Where legacy automations and fields still carried real dependencies, the recommendation was to keep them, deliberately and with owners.

OBJECTIVES

What the engagement set out to do

01

Establish ownership

A named owner for every configuration object in the site.

02

Consolidate on evidence

Workflows, screens, fields, and filters reduced against usage data, not opinion.

03

Make automation accountable

Predictable behavior, deliberate scope, and a person answerable for every rule.

04

Preserve what still works

Legacy automations and fields with genuine dependencies retained by documented exception.

05

Leave it maintainable

Standards and a change intake that keep the environment clean after handover.

Inventory everything. Classify everything.

Discovery

A complete configuration inventory: every workflow and scheme association, every screen, every custom field with its contexts and fill rates, every filter with its sharing and the boards and dashboards consuming it, and every automation rule with scope, triggers, execution history, and failure record.

Interviews with administrators and team leads added what no export captures: how work actually flows, and which oddities exist for a reason.

Assessment

Every object in the inventory was classified into one of four outcomes, with the evidence recorded alongside the decision:

Retain **Consolidate** **Retire** **Redesign**

Classification was an artifact of the engagement, not a spreadsheet in someone's head. The client keeps the decision record.

Two rules governed the exercise: nothing was retired without usage evidence, and nothing was retained without a named owner.

Design the future state, then move in waves

Design

Governed workflow library

A small standard set with defined variation points. Standardization without forcing identical workflows on teams that genuinely work differently.

Field catalog

Naming standards, context discipline, duplicates merged, and a request path for anything new.

Automation standards

Project scope by default, global by exception. A naming convention, standardized rule actors, and a recorded human owner for every rule.

Legacy exception register

Legacy automations and fields with active dependencies formally retained: dependency documented, owner assigned, review date set.

Implementation

1. Archive obsolete projects first, shrinking every downstream inventory.
2. Move projects onto governed workflow schemes in groups.
3. Merge duplicate fields, migrating values where history mattered.
4. Retire broken and orphaned filters, with advance notice to anyone still touching them.
5. Remove automation on a disable, observe, delete pattern: switched off, watched through a quiet period, deleted only when nothing surfaced.

Every wave was rehearsed in the Jira Cloud sandbox before it touched production.

ENGAGEMENT FLOW

From evidence to enablement

1	2	3	4	5	6
Discovery	**Assessment**	**Design & Standards**	**Cleanup Waves**	**Validation**	**Enablement**

Archive · Workflows · Fields & Filters · Automation

Waves were sized to team readiness. Each closed only after the teams affected had validated their boards, dashboards, and automated handoffs, with automation failure rates monitored throughout.

RESULTS

A different kind of environment, not just a smaller one

A governed workflow library

Consolidated workflows now cover every active project, replacing per-project cloning.

A field catalog worth trusting

Every remaining custom field has a stated purpose, a documented owner, and known screens.

Accountable automation

Overlapping rules consolidated. Every active rule is named to convention, scoped deliberately, and owned.

Intentional filters

Broken, orphaned, and over-shared filters retired or corrected, with sharing standards for new ones.

Legacy on the record

Automations and fields with real dependencies retained under documented exceptions, with owners and review dates.

Known blast radius

Administrators evaluate proposed changes against documented dependencies instead of discovering them.

The count of unowned configuration objects is effectively zero.

WHAT WE DELIVERED

Built to be operated after we leave

- Complete configuration inventory and dependency map
- Retain, consolidate, retire, redesign decision record
- Governed workflow library and consolidated screen schemes
- Field catalog with naming standards and a field request path
- Automation standards and ownership register
- Legacy exception register with review cadence
- Administration playbook and configuration change intake

The cadence that keeps it clean

A quarterly configuration review keeps the exception register honest and catches drift early. The change intake process means the environment now drifts toward the standards over time, instead of away from them.

WHAT MADE THE DIFFERENCE

Two decisions shaped the outcome

Sequencing

Governance came before cleanup. Establishing who owned each object, and what actually used it, is what made removal safe and automation reliable again. Automation behaves predictably now because the configuration underneath it is governed, not because the rules got cleverer.

Success, redefined

The target was never zero legacy. Giving legacy automations and fields a documented way to stay changed the politics of the engagement: teams volunteered candidates for retirement once they trusted that what mattered would survive, with their name on it.

Zero unowned configuration, not zero legacy.

KEY TAKEAWAYS

What travels to other environments

1 Configuration debt is an ownership problem before it is a volume problem.

2 Evidence, not opinion, decides what gets retired.

3 Legacy that works deserves an owner and a review date, not a deletion deadline.

4 Reliable automation is a product of governed configuration.

5 Standards only hold when a change intake keeps applying them to whatever comes next.

AVARATAK

If your Jira environment has grown faster than its governance model

Avaratak can help you determine what should be retired, what should be consolidated, and what genuinely deserves to stay, before cleanup decisions become outages.

www.avaratak.com

ATLASSIAN Solution Partner

Enhancing Teams, Enabling Success