tuleap joins CATIA

E-BOOK

Jira Software to Tuleap Migration Guide

tuleap

E-BOOK

Table of Contents

Before you start

Following requests from our clients, we've spent the past several months building tools to make migrating data from Jira Software[1] to Tuleap as smooth as possible. What started as a simple approach has grown into a full-featured migration path that handles even complex scenarios.

That said, Jira Software and Tuleap are two different products. They solve similar problems, but they are built on different models. A one-to-one migration isn't really possible, and in most cases it wouldn't make sense anyway. You will need to make a few decisions about how to adapt things based on how your teams actually work.

Our tools and methods keep evolving, so this guide reflects where things stand today. We'd recommend making sure you're reading the latest version.

To set your project up for success, we strongly recommend getting in touch with our team early on. We'll help you define the scope and choose the right approach for your situation.

We've got your back

Getting the tooling right is only part of the equation, **the real key to a successful migration is support.**

From day one, **our team helps you scope your project, answers your questions, and figures out the best migration strategy** for your specific context.

During onboarding, **we guide you through setting up Tuleap and configuring your projects,** including complex or org-specific setups. And we stay with you beyond go-live: up-to-date docs, responsive support, and resources you can access anytime, **so your team builds real autonomy with Tuleap over time.**

[1] In this guide, we use the terms *Jira Software* or *Jira* interchangeably to refer to the same ticket-tracking software published by *Atlassian*.

Key concepts in Tuleap

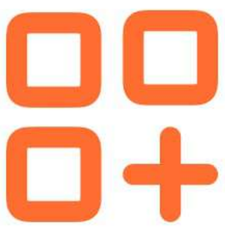
Before diving in, here's a quick recap of the core Tuleap concepts that differ from what you're used to in Jira Software.

There are **4 key concepts to keep in mind:**

Artifacts

1 issue = 1 ticket

Trackers



1 Tracker = a collection of same-type Artifacts[1]

Projects

1 Project = several Trackers

An Instance

Multiple Projects with different tools

Tuleap Artifact = Jira Ticket

In Tuleap, Artifacts are the equivalent of **Jira tickets or issues.** They represent units of work: a task, a bug, a user story, anything you want to track and manage.

Artifacts and tickets have a lot in common: they're form-based, they carry information and attachments, they have a change history, and they get created, assigned, and followed by team members.

Where things differ is in the underlying data model. Tuleap Trackers and Jira Software are built on different logic, which is why some things need to be adapted during migration rather than simply copied over.

[1] Each Tuleap tracker is independent:
- Its own fields
- Its own workflow and triggers
- Its own permissions
- Etc...

Jira Issue Type → Tuleap Trackers

The Tracker is one of Tuleap's most powerful modules: highly flexible and fully customizable. You can create one or more Trackers per project, with configuration at two levels: project-wide and per Artifact type.

A Tuleap Tracker is a collection of Artifacts of the same kind. Most Tuleap projects have several Trackers: one for Epics, one for User Stories, one for Tasks, and so on.

Each Artifact type lives in its own Tracker, which means you can set up custom workflows, permissions, and automation rules that make sense for that specific type.

Jira works differently: issue types share a common structure, with less room to differentiate behavior per type.

During migration, we map each Jira Issue Type to a corresponding Tuleap Tracker.

One thing worth noting: in Tuleap, planning milestones (Sprints, Releases, Months, Products) are also managed in Trackers, which makes them fully customizable. That's not the case in Jira.

Moving your Jira data into separate Tuleap Trackers per issue type puts you back in full control of your workflow. An "Epic" can have a completely different workflow from a "Bug", tailored exactly to how each type actually moves through your process.

WORKFLOW & TRIGGERS IN TULEAP

A **Workflow** in Tuleap defines how an Artifact moves through its lifecycle with transition rules, field dependencies, the whole thing. It keeps things consistent and reduces manual errors.

On top of that, **Triggers** let you automate actions based on events. For example: when all tasks are marked "DONE", the parent Story automatically moves to "CLOSED". Or: when a sub-task moves to "ready to test", a CI job kicks off.

For more examples and details:

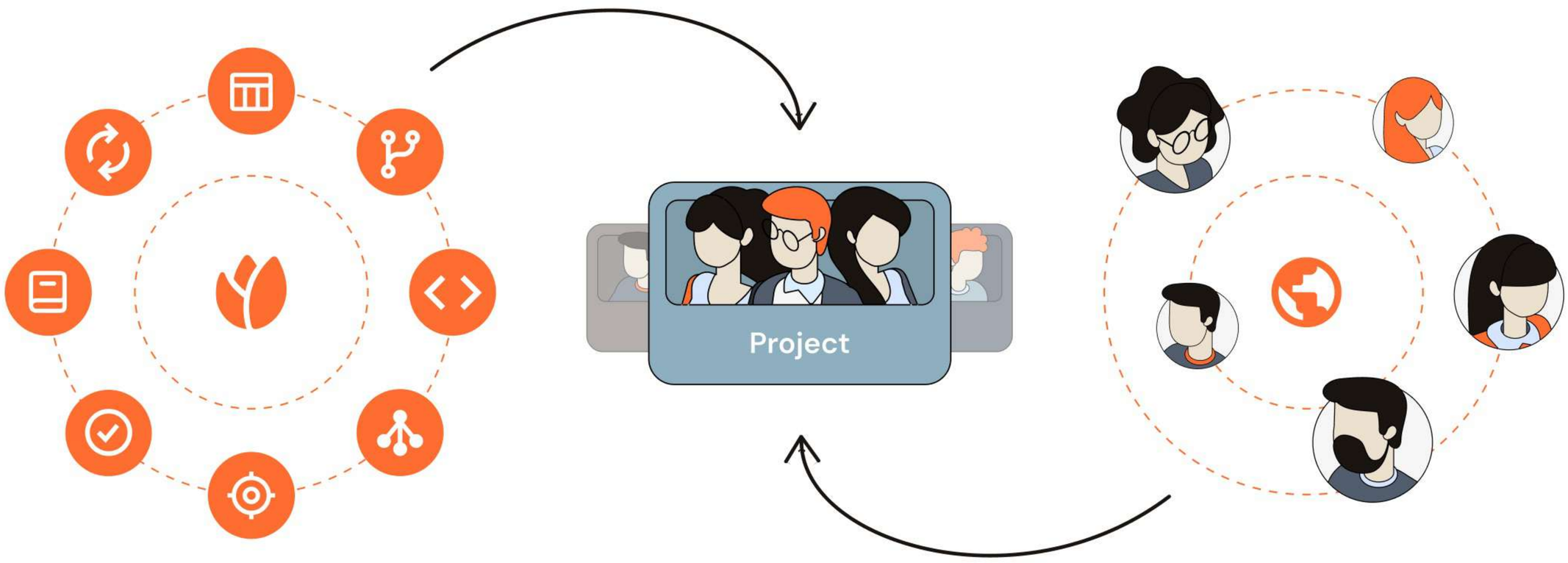
https://www.tuleap.com/issue-tracking/
https://docs.tuleap.com/user-guide/trackers/administration/configuration/workflow.html

Tuleap Projects

A Tuleap Project (also called a Workspace) is a shared environment where your team works together. Each Project is self-contained and comes with one or more of Tuleap's modules activated (see the Appendix "Tuleap EE Feature Overview").

This makes it easy to support cross-functional teams (ex: Business Analysts, Product Owners, Developers, QA, Clients, you name it) all working in the same space, each with the tools that fit their role.

Improve collaboration with personalized team workspaces



E-BOOK

2 ways to migrate your data

When moving from Jira Software to Tuleap, you've got a couple of options. The right one depends on how much data you're moving and how complex your setup is.

Here's the quick version:

Simple Migration

Tuleap Interface

Import tickets by ticket type

+ Simplicity
+ Team Autonomy

Advanced Migration

Launching a script

Ticket migration into a mono tracker

Project migration with planning (multi-tracker)

+ More complete
+ High volume

Simple migration:

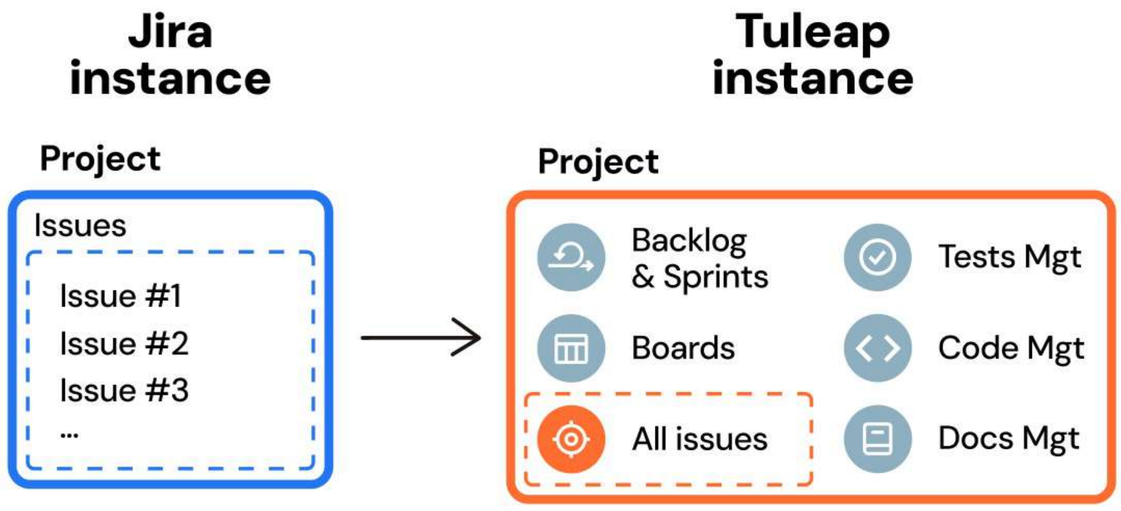
Simple migration means moving your tickets from a Jira instance directly into a Tuleap Tracker straight from the Tracker admin interface, no scripts needed.

The big plus here is simplicity. Teams can run this themselves. You create a new Tuleap Tracker, point it at your Jira instance, pick a ticket type, and import.

If you have 3 ticket types in Jira, you'll end up with 3 Trackers in Tuleap.

The **only prerequisites** are having admin access to both your Jira and Tuleap projects. You enter the Jira URL, your credentials, select the ticket type, and you're done. **The import runs automatically in the background.**

Simple migration via the Tuleap interface



Advanced Migration:

Advanced migration is the right choice for large-scale moves, involving multiple instances, numerous projects, large data volumes, and many users.

It goes well beyond ticket migration. You can bring over complete agile environments, including:

- Backlog
- Sprints and their contents
- Links between Artifacts

This keeps the structure and logic of your existing projects intact.

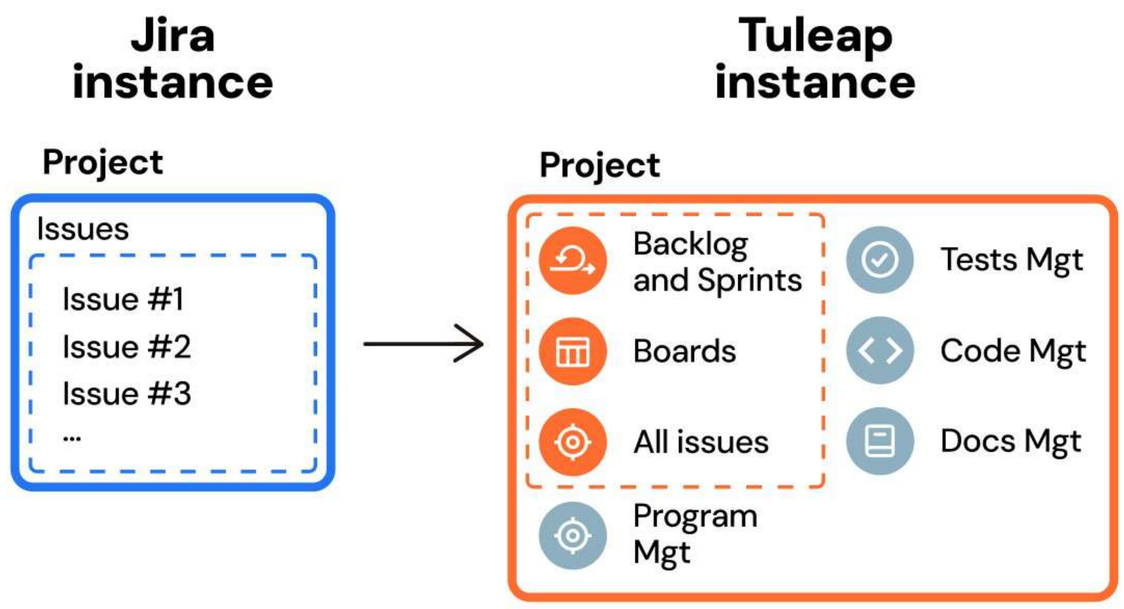
You also have the option to consolidate all Jira tickets into a single Tuleap Tracker, regardless of type, useful for Kanban-focused teams or anyone who wants to simplify their setup.

For large scopes, the script-based approach makes it easy to scale.

Our team can support you through the whole thing if needed.

See de Migration process → p.16

Advanced migration via script



1/What gets migrated

Simple Interface Migration

During a simple migration, tickets, custom fields, and change histories are replicated. In Tuleap, links between Artifacts of the same type are reproduced (e.g., a Bug linked to another Bug) but not links between Artifacts of different types (e.g., an Epic will not be linked to a User Story).

You'll need one Tuleap Tracker for each Jira ticket type you want to bring over.

For example: if you're tracking Requirements, Tasks, and Bugs in Jira, you'll create three Trackers in Tuleap, one for each type.

If you need a more complete migration, the advanced migration path (detailed below) is the way to go.

Good to know

Simple migration is a great way to kick the tires on Tuleap without committing to a full move. Import a handful of tickets, see how they look, and iterate before going all in.

In just a few steps, you get a real feel for how Tuleap works with your data. Which helps you make better decisions, spot any gaps early, and move forward with confidence.

For details on imported fields, please refer to the documentation:

https://www.tuleap.com/importing-jira-projects-to-tuleap/

https://www.tuleap.com/moving-issues-from-jira-to-tuleap/

https://help.tuleap.com/docs/en/create-a-tracker

2/What gets migrated

Advanced Script-Based Migration

The advanced migration script handles more than just tickets.
It can bring over your full Scrum environment and user groups too.

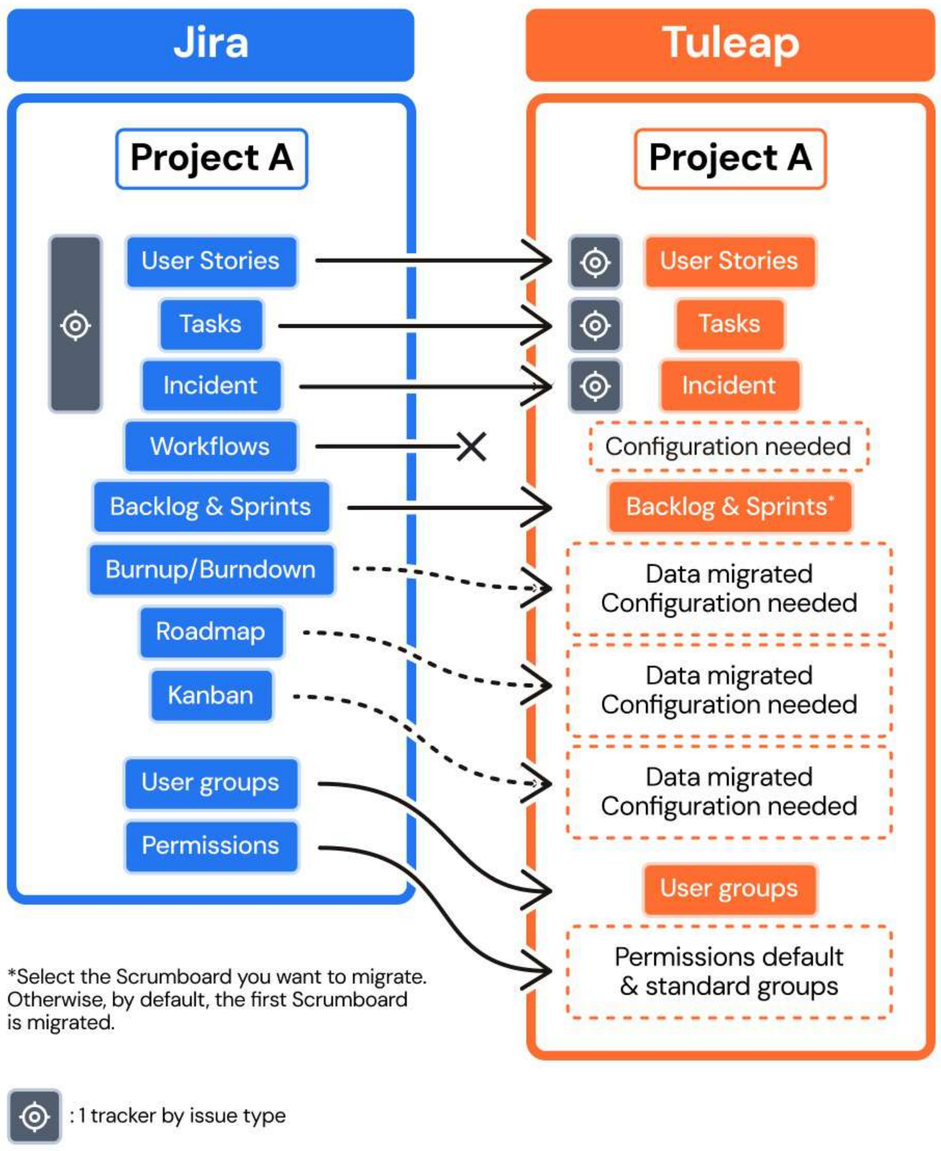
For each project, you pick your import mode:

- **Multiple Trackers:** one Tuleap Tracker per Jira issue type.
The right choice if you're running Scrum with Planning.

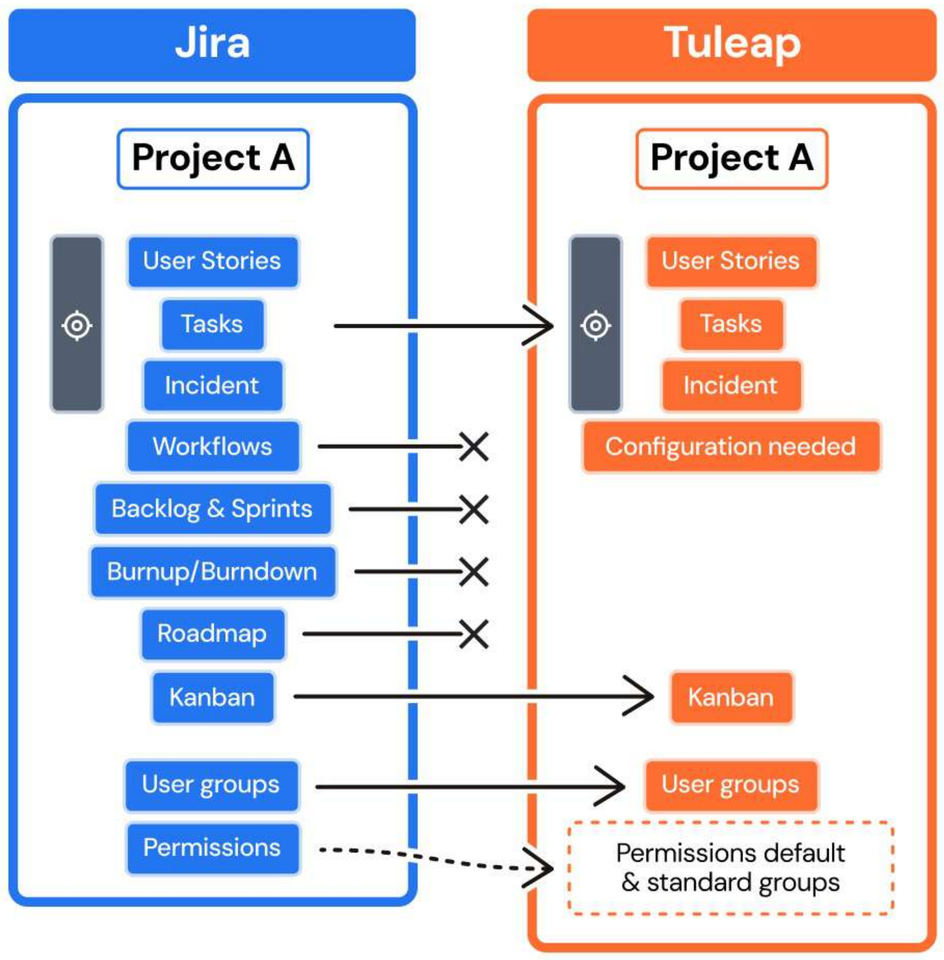
- **Single Tracker:** everything goes into one Tracker. Works well for Kanban teams or anyone who wants a simpler setup with a shared workflow.

How your Artifacts are organized in Tuleap will depend on which mode you choose.

Multi-Tracker import
Advanced script migration



Mono-tracker import
Advanced script migration



Issues / Artifacts

Every Jira issue type can be imported into Tuleap, along with **custom fields, comments, change history, and time tracking.** With the advanced migration, **links between Artifacts** within the same project come along too.

Full details on which fields are imported are in the docs:

https://docs.tuleap.com/user-guide/trackers/administration/tracker-creation.html

In Tuleap, **Artifacts are sorted into separate Trackers by type.** The script automatically creates one Tracker per Jira issue type. So you'd end up with a User Stories Tracker, a Tasks Tracker, an Incidents Tracker, and so on.

Jira workflows don't get migrated, you'll need to reconfigure them in Tuleap. This is actually an advantage: it gives you a clean slate to set up workflows, triggers, and permissions that fit how each Artifact type works for your team.

Links between Tickets / Artifacts

With advanced migration, **links between Artifacts within the same project are preserved:** "related to", "covered by", "child of", and so on.

Your existing hierarchies come with you:

- Task – sub-task
- Epic – User Story – task – sub-task
- Requirement – test case – test execution

Links between Artifacts from different projects are not reproduced.

Backlog and Sprints

Your **Scrum backlog and Planning** data comes over too. **Backlog content, sprints, and sprint statuses all get migrated.**

When you run the script, you'll pick which Jira Scrum Board to import.
If you don't specify one, it defaults to the first Scrum board it finds.

Once you're in Tuleap, you have a lot more flexibility with planning. You can customize your milestone hierarchy and naming however you like. For example, one project might run "Release > Sprint > Week" while another uses "Product > Version".

It's a genuinely more flexible setup than what Jira offers out of the box.

Kanban boards

If you go with a mono-Tracker import, the migration automatically creates a Kanban board showing all your issue types.

With a multi-Tracker import, you can create one or more Kanban boards, one per ticket type if you want.

User Groups and Permissions

User groups are migrated from Jira Software to Tuleap.

So if you had “Developers” and “Clients” groups in Jira, you’ll find them in your Tuleap project too, with users matched up where possible (see the “User Mapping” section).

On the permissions side, Tuleap pulls over:

- create/edit rights by user group
- mandatory fields

Note that **Tuleap applies its own default permissions,** it doesn’t carry over Jira roles directly. You’ll need to review and set those up in Tuleap after migration.

Check the **Tuleap docs** for the full details.

What is not migrated automatically

Heads up

We keep updating this guide as we get new requests and tackle new use cases. Some things that aren't supported today may well be supported by the time you read this. Reach out and we'll check.

When something is listed as “not migrated,” it’s usually for one of two reasons:

- The concept simply doesn’t map cleanly between Jira and Tuleap. There’s no direct equivalent, so a workaround is needed.
- We haven’t built it because it hasn't been required yet.

Either way, we’re open to a conversation. Get in touch and we’ll figure out what’s possible.

Kanban boards

With a mono-Tracker import, the migration creates a Kanban board that shows all your issue types together.

In a Scrum setup, you can create separate Kanban boards per ticket type: one for User Stories, one for Tasks, etc. Multi-Tracker Kanban boards aren’t available in Tuleap yet.

The upside of per-type Kanban boards is that each one respects the specific workflow and permissions you’ve configured for that Tracker. So they’re more precise.

Workflow

Because of how Trackers work in Tuleap and how much you can customize them, Jira workflows don’t carry over. You’ll need to set them up fresh in Tuleap after migration, defining the right transitions for each issue type.

Some fields

Most fields come over just fine. A few specific ones may not be supported yet. Get in touch and we’ll check for your case.

From Jira to Tuleap: An advanced migration

The rest of this guide focuses on the advanced migration, specifically the project-by-project import approach.

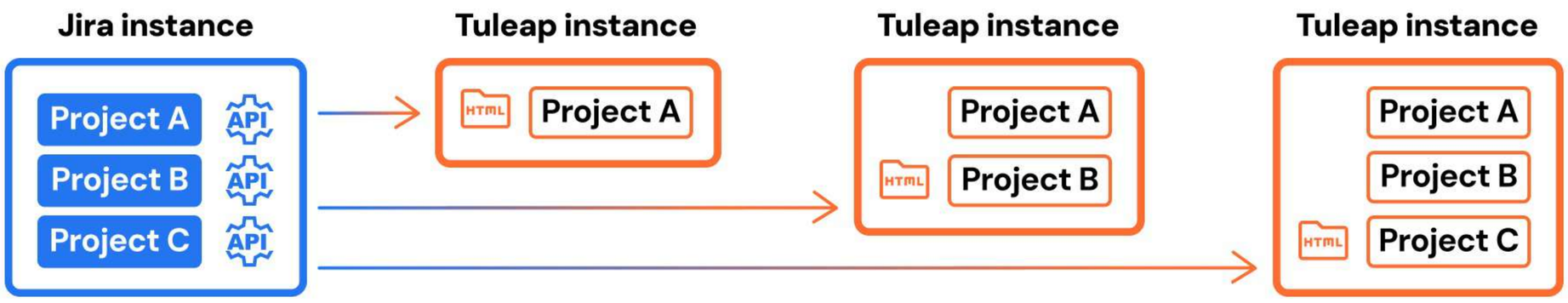
How the project-by-project migration works

The steps:

1. The script queries the Jira API
2. An XML file is generated for each project
3. The project gets imported into Tuleap
4. You review the migrated data
5. You kick off the next project import

You move through your projects one at a time, at your own pace.

Project-by-project migration



You stay in control

- Review each project before moving to the next.
- Choose which projects to migrate and which to skip.

In the example above, the script imports Project A, then B, then C. You could have skipped Project B entirely if you didn't need it.

Technical prerequisites

Here's what you'll need in place before running the advanced migration script:

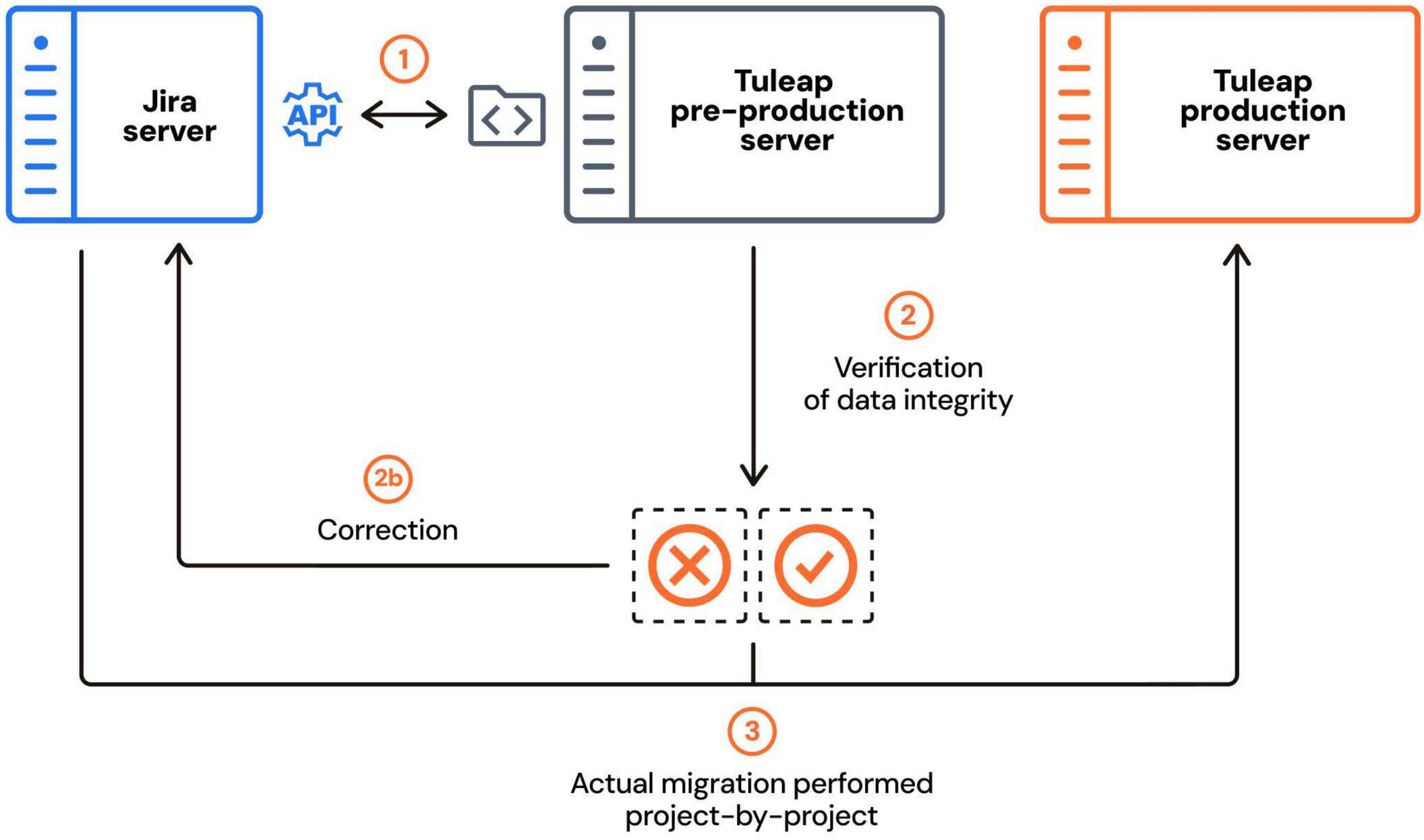
On the Jira side

- Admin access to a Jira Server, Jira DataCenter, or Jira Software Cloud instance.
- Make sure the Jira instance can talk to the Tuleap instance over the network.
- Enable email address exposure via the Jira API.

On the Tuleap Enterprise Edition (TEE) side

- **On-premise:** SSH access to two TEE servers:
  - → One in pre-production
  - → One production
- **Cloud:** If you're on Tuleap Cloud Premium, you're already on the latest TEE. Our team handles the server access and runs the migration for you.
- **In both cases:** whoever runs the project-by-project migration needs to be a Tuleap project admin.

Migration process



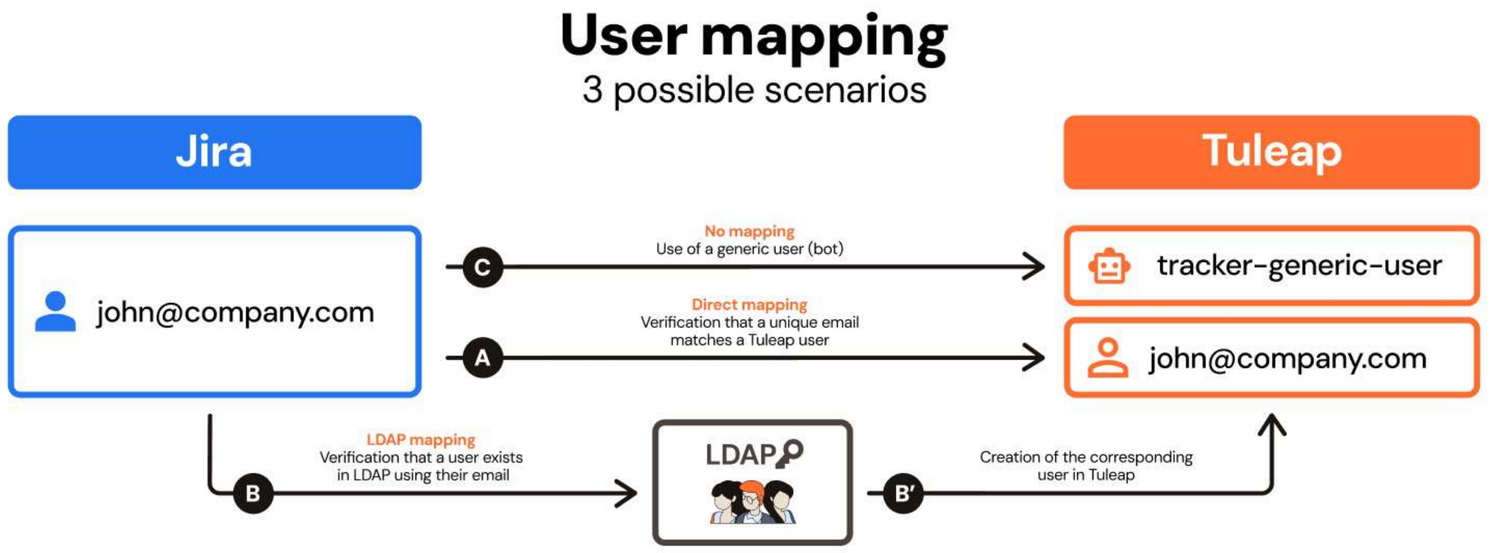
1. Start by **putting your Jira instance in read-only mode,** that way nothing changes while you're migrating. From the Tuleap **pre-production server,** the admin runs the migration script, which calls the Jira API, generates an XML file for the project, and imports it into Tuleap.

2. Review the imported project on your **pre-production server.** If something's off, make corrections, run tests again, and repeat until everything looks right.

3. Happy with the result? The admin **then runs the script against the production Tuleap server,** migrating projects one by one until you're done.

User Mapping

There are three ways users can be matched between your Jira and Tuleap instances. The goal is simple: **keep as much traceability as possible** knowing who did what, and when.



Direct mapping

This works when **the same user exists in both Jira and Tuleap,** identified by their email address. If John is already in Tuleap, anything he created in Jira will show up in Tuleap as created by John. Simple.

LDAP mapping

If your users aren't in Tuleap yet, they can be created automatically through your corporate LDAP directory. It's a clean way to onboard dozens or hundreds of users in one shot.
So if John isn't in Tuleap but is in your LDAP (with the same email), he gets created automatically during migration. The result is the same as a direct match.
Full traceability preserved.

Not mapping

If the script can't find a user, either directly or via LDAP, it falls back to a generic bot account (Tracker-generic-user). Anything that user created in Jira will be attributed to that bot in Tuleap.

Let’s Test the Migration Together!

To make the switch as painless as possible, we built a methodology where we work side by side with you from the start: testing, tweaking, and validating the migration together before you commit.

We’ve run enough migrations to know what works. You benefit from that experience directly: less trial and error, less time, less budget.

“Pilot migration” Package

INSTALLATION

Installation of a dedicated Tuleap Cloud Premium instance for your organization

MIGRATION TESTS

Migration of up to 3 pilot projects

REVIEW

Migration validation

Fine-tuning (on estimate)*

CONTACT US!

Our Guarantees

- Listening
- Responsiveness
- Solutions

* ***Fine-tuning :***
Identification of specific needs,
Consulting/Development,
New test datasets,
Review and assessment.

Time & resources to plan for

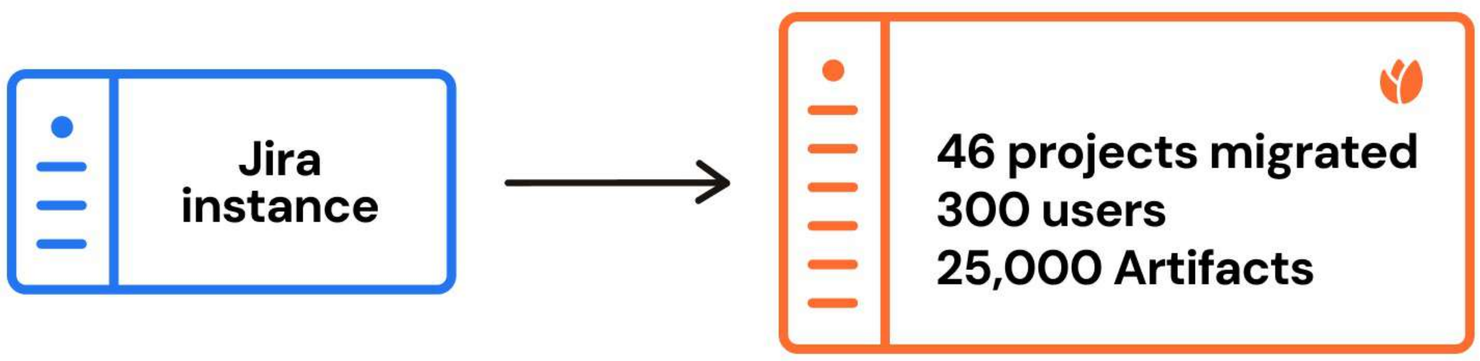
How long a migration takes depends on a few things: your technical setup, the number of projects, and the volume of data. In some cases, it can be completed quickly.

We've tested against multiple Jira versions, so even if your setup has specific requirements, that's not a blocker: we adapt and work iteratively.

Large-scale migration Customer Story: Defense Industry

One of our customers, a strategic player in the French defense industry, wanted to move away from Jira Software to regain full control over its data while providing its teams with a more capable and autonomous working environment.

They chose to migrate their Jira Software projects to Tuleap.



That's **25,000 Jira tickets** across **46 projects** moved to a Tuleap on-premise instance. The migration also automatically created **300 user accounts** in Tuleap, so everyone could log in and find their work exactly where they expected it.

People & resources

- The team that owned Jira became the Tuleap team.
- They were trained by Tuleap before the migration.
- Only one person was needed to run the actual migration.
- After go-live, the team became self-sufficient and handled user onboarding through training sessions, FAQs, and webinars.

Timeline

- **Before migration day,** several working sessions and test runs with our team to prepare.
- **On migration day,** the client ran it themselves. Total time: 1.5 days. The biggest project took 45 minutes; most took just a few minutes.
- **Post-migration,** the team spent about 10 person-days reconfiguring elements in their Tuleap projects.

We've stayed in close contact with them since, as they continue migrating from other tools.

Other migrations we've run

We've helped teams migrate in all kinds of contexts: MedTech, research, education, IT services, and more.

The complexity has ranged from targeted issue imports (preserving traceability) to full migrations bringing over projects, structures, and workflows.

A few things hold true across all of them:

- Large data volumes migrate reliably
- Project structures and links between data are preserved
- The gaps between tools need to be thought through, not just copied over

Every migration we run also makes our tools and methods better.
We bring that accumulated knowledge to every new project.

A migration is a real project. It takes planning and some judgment calls.
But when it's scoped well, tested properly, and run methodically,
it's very much manageable.

Ready to make the move? See what Tuleap can do for your team.

Migrating from Jira to Tuleap

WE CAN HELP YOU!

Let's discuss your project!

Appendices

What's Available in Tuleap Enterprise Edition™

Trademarks

PDF v2
Published by Enalean SAS, 87 Avenue de Chambéry, 73230 Saint-Alban-Leysse, France.
Tel : +33 (0)9 81 41 96 16 — Email : contact@enalean.com
Business identification (Siret) number : 53179662100047

Key questions to ask before you start

A migration involving multiple instances, hundreds or even thousands of projects, large data volumes, and many users isn't purely a technical exercise. It requires scoping, trade-off decisions, and clear organization to keep every step on track.

Before the migration

Goal: scope the project and align stakeholders

Define the scope and objectives

- What's the goal: a test run, a phased migration, or a full cutover?
- How many instances, projects, and users are involved?

→ Risk if skipped: underestimating the project and schedule drift

Map out exactly what needs to move

- Data: tickets, history, attachments
- Structures: projects, trackers, fields, workflows
- Users, groups, and permissions
- Integrations and dependencies

→ Risk if skipped: a gap between what you expected and what you get

Decide what data is actually worth migrating

- Migrating everything isn't always the right call
- Figure out what should be migrated, transformed, archived, or dropped

→ Risk if skipped: unnecessary clutter and added complexity

Define your target model in Tuleap

- Reproduce your existing setup, simplify it, or redesign it?

→ Risk if skipped: a technically correct migration that doesn't fit how people work

Formalize the project framework

- Scope, transfer rules, roles, timeline, success criteria

→ Risk if skipped: lack of governance and decisions made by default

Set up clear governance

- Who owns it, who signs off, who makes the calls?

→ Risk if skipped: blockers and slowdown

Plan a phased migration

- Break it down by entity, scope, or complexity level

→ Risk if skipped: a project too big to control

During the Migration

Goal: keep execution on track and validate quality

Validate your mapping rules

- Statuses, fields, users, permissions, structures

→ Risk if skipped: inconsistencies in migrated data

Test on a representative scope

- Include complex cases, not just the easy projects

→ Risk if skipped: a false sense of success

Check data quality

- Completeness, readability, attachments, change history

→ Risk if skipped: users lose trust in the tool

Bring in business users

- Check real-world usage, not just technical compliance

→ Risk if skipped: adoption issues down the line

Handle edge cases and exceptions

- Identify them, document them, resolve them

→ Risk if skipped: a pile-up of blockers at the end of the project

Stay flexible on the timeline

- Adjust based on what you learn as you go

→ Risk if skipped: uncontrolled drift

Manage external dependencies

- SSO, directories, connected tools

→ Risk if skipped: service disruptions after go-live

After the Migration

Goal: secure go-live and stabilize

Validate the results

- Compare against your pre-defined success criteria

→ Risk if skipped: an uncontrolled go-live

Identify remaining gaps

- Data, usage, configuration

→ Risk if skipped: deferred technical debt

Set up post-migration support

- Channels, priorities, ownership

→ Risk if skipped: rapid loss of user confidence

Onboard your users

- Training, documentation, internal champions

→ Risk if skipped: partial adoption

Handle old tool decommissioning

- Archiving, access management, compliance

→ Risk if skipped: confusion over the source of truth

Build on each wave

- Apply lessons to improve the next phases

→ Risk if skipped: repeating the same mistakes

Recommendation

For large-scale migrations, setting up a proper project framework is strongly recommended. It keeps teams aligned, decisions documented, and the migration manageable at every stage.

A phased approach that includes a representative pilot, business validation, and a staged rollout is consistently the most reliable strategy.

tuleap

E-BOOK