JGit Reftable in Gerrit

past, present, and future adoption

Luca Milanesio

GerritForge CEO

Gerrit Maintainer & Release Manager

OpenInfra SUMMIT › EUROPE'25

GerritForge

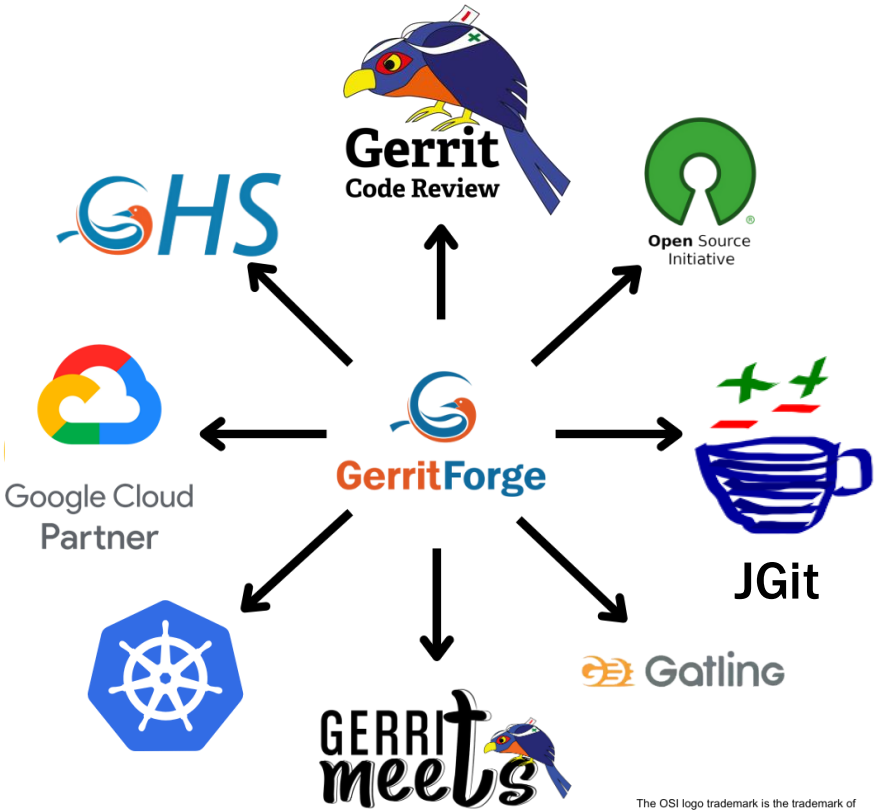
Engineering
Git and Gerrit
for Enterprise Velocity

Official Sponsors of

THE GERRIT USER SUMMIT 2025

PARIS, FRANCE

About GerritForge Inc.



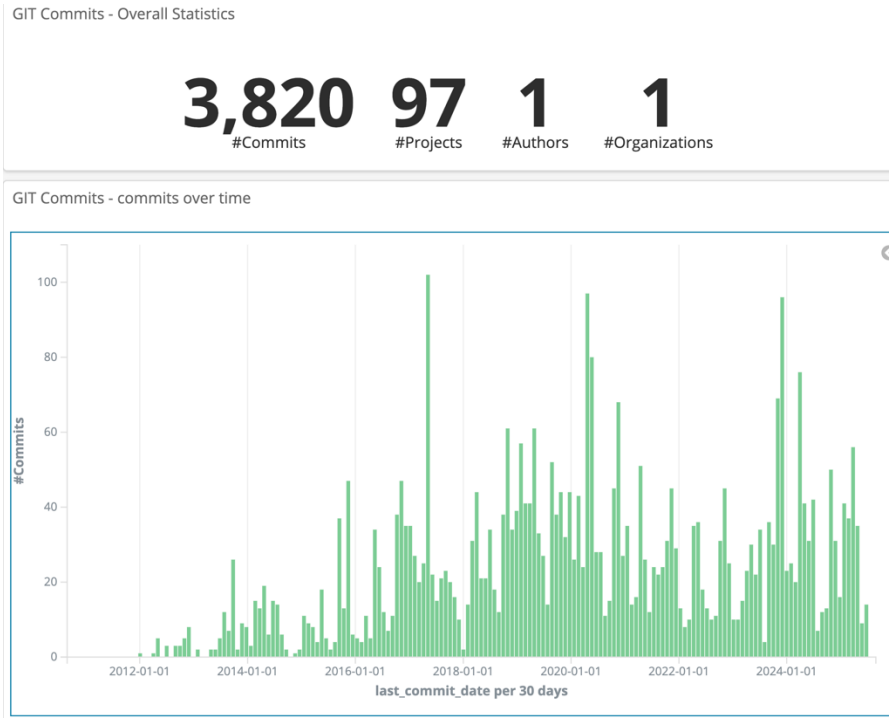
- HQ in Sunnyvale – California
- Distributed worldwide

  USA, UK, Italy, Poland, Germany, Israel
- JGit and Gerrit Code Review since 2012
- 16k merged contributions, 1M LOCs
- 5x Gerrit Maintainers | 17x Contributors
- 2x JGit Committers
- 2x Gerrit Release Managers
- 1x Gerrit Engineering Steering Committee
- 1x Gerrit Community Managers

Luca Milanesio
GerritForge, Inc. CEO

- JGit Committer
- Gerrit Code Review Maintainer
- Gerrit Code Review Release Manager
- Gerrit Code Review ESC
- Created the Gerrit Plugins ecosystem
- Maintainer of GerritHub.io since 2014

JGit & Gerrit Contributions since 2012



Source: https://analytics.gerrithub.io

THE GERRIT USER SUMMIT 2025
PARIS, FRANCE

JGit Reftable (FileReftableDatabase) agenda for today

- Short history of Reftable support in JGit
- Introduction of JGit Reftable in Gerrit
- RefDirectory vs. FileReftableDatabase
- JGit and CGit interoperability
- FileReftableDatabase next steps
- Q&A

A new high-performance ref database – July 2017

Shawn Pearce (Google) – Reftable format for Git@Google (no filesystem)

commit b9e818b556d15672ad589e1dbdf5f6f1ea557c78
Author: Shawn Pearce <spearce@spearce.org>
Date: Sat Jul 22 10:12:32 2017 -0700

reftable: file format documentation

Some repositories contain a lot of references (e.g. android at 866k, rails at 31k). The reftable format provides:

- Near constant time lookup for any single reference, even when the repository is cold and not in process or kernel cache.
- Near constant time verification a SHA-1 is referred to by at least one reference (for allow-tip-sha1-in-want).
- Efficient lookup of an entire namespace, such as `refs/tags/`.
- Support atomic push `O(size_of_update)` operations.
- Combine reflog storage with ref storage.

Change-Id: I29d0ff1eee475845660ac9173413e1407adcfbf2

THE GERRIT USER SUMMIT 2025 PARIS, FRANCE

FileReftableDatabase for file-based storage JGit – October 2021

Han-Wen Nienhuys (Google) – Open-Source implementation on filesystem

- Brand-new implementation for JGit
  (following Shawn's design on Git@Google)
- Works on local filesystem and NFS
- Initial implementation for Git, handed over to Patrick Steinhardt (GitLab)

FileReftableDatabase with auto-refresh – March 2025

Matthias Sohn (SAP) – Addressed initial feedback from real users

- Makes FileReftableDatabase compatible with JGit's Repository cache
- Detected changes on filesystem and reload reftable
- Further fixes on thread-safety

FileReftableDatabase with Gerrit v3.12 – July 2025

Antonio Barone, aka Tony (GerritForge) – E2E stress testing with Gerrit

- Test of Reftable as part of the GHS (Git-at-High-Speed) project
- Simulated load of 10 years of production load, compressed in 10 hours
- Identified and fixed further thread-safety on FileReftableDatabase

Successfully validate and release v3.12.2 with production-ready reftable
https://www.gerritcodereview.com/3.12.html#3122

THE GERRIT USER SUMMIT 2025 PARIS, FRANCE

RefDirectory

Issues with packed/loose refs

- Mutability and locking (hard to achieve on NFS)
- Text sorted file format, slow in read/write
- Adding refs is easy, removing is hard

  > rewrite the *whole* packed-refs file, stop-the-world lock
- No tombstone & reorg support

Until 1 month ago, JGit had **no way to delete a ref** reliably from packed-ref

Deleted refs come back to life issue

FileReftableDatabase

Advantages of reftable

- reftable(s) are immutable
- tables.list is still a mutable file ☹
- Layered tables offer constant time for add / delete refs
- Binary format: no more wasting CPU in UTF8 codecs
- Introduce high-performance read snapshot
- JGit can cache them in-memory with auto-refresh

Reftable is now **production-ready** In Gerrit v3.12.2

FileReftableDatabase

Problems with reftable

- Updating a single ref requires writing a table
- Concurrency issues on table auto-reorg
- FileReftableDatabase works with snapshots

  the JGit and Gerrit upper layers NOT YET
- Older orphan files post-reorg lingering around

Updating a single **loose ref** is **way faster than reftable**. The difference is higher if auto-compaction kicks in.

RefDirectory

Advantages of packed/loose refs

- Years of tests and fixes for using it on shared filesystems

  (from stale file handles to NFS caching)
- Simple to troubleshoot: all files are readable
- Simple to fix: all files are text and editable
- Works just fine with most smaller to medium-sized repos

Gerrit | FileReftableDatabase vs. RefDirectory

End-to-End scenario	Mean	P95
Clone over http	-7%	-19%
Push to create change over http	-38%	-37%
Add Hashtag	25%	2%
Add Patchset	-70%	-61%

Massive improvements on batch-refupdate use-cases:

- Create a change over Git push / online
- Add new patch-set

Gerrit | FileReftableDatabase vs. RefDirectory

End-to-End scenario	Mean	P95
Add Topic	60%	26%
Approve Change	65%	79%
create tag	90%	261%

Slowdown on single-ref update use-cases:

- Metadata-only updates (topic, reviews)
- Creating tags

JGit <-> Git interoperability

	JGit v7.4.0	Git v2.51.1
Read each other reftables	100%	100%
Auto-compaction	100%	100%
Concurrency	100%	Corruption experienced with git gc + JGit auto-compact Reftable fully locked when running Git GC


JGit reftable
What’s next

JGit <-> Git interoperability FIXES

	JGit v7.4.0	Git v2.51.1
Read each other reftables	100%	100%
Auto-compaction	100%	100%
Concurrency	100%	Corruption experienced with git gc + JGit auto-compact Reftable fully locked when running Git GC

JGit and Repository cache FIXES

JGit Repository cache keeps the repo open

- Reftable is never closed or refreshed
- Auto-refresh by Matthias partially fixes the problem: multiple threads share the same snapshot auto-refreshed
- In-memory snapshot state isn’t mutable
- **Ideas?**

THE GERRIT USER SUMMIT 2025
PARIS, FRANCE