Rust port of the iRODS Server

Mher Kazandjian

Ekogo Computing S.R.L

2026-07-01

Port timeline

- Started as an idea around early April
- Plan with claude extensively for two to three weeks
- Point Claude 4.6 to the iRODS code base and asked it to study it
  - Produce a abstract tree + summary of every file
  - Learn about the design of iRODS
  - Check feasibility: 100% API compliance with 5.0.2 (deviations not negotiable – zero tolerance)
  - All unit tests must pass
- Refine and iterate and prepare psychologically
- All the work was done with Claude 4.6, 4.7, 4.8 (xHigh)
- Dive in!

Compatibility

- Legacy and python rule engine
- C++ shim runtime
- Audit pipeline
- Resources and federation
- PRC, jargon, go
- Genquery2
- Federation and delay server
- Full API
- TLS
- Extension: Blake3 checksum

Agent work guidelines: 11 step procedure

1. Discuss scope with user first. Confirm against MASTER-PLAN.md.
2. ~/bin/gh issue create → assign to mherkazandjian → get number X
3. Diagram in the issue body before code. Mermaid for non-trivial flows;
4. Implement on the branch
5. Mirror relevant C++ tests into Rust.
6. Lint + test locally in docker compose before pushing
7. Commit, push, PR. ~/bin/gh pr create → target main → body has Closes #X
8. Max-effort code-review subagent on the PR diff. Opus 4.8 max effort. Prompt includes engineering-quality criteria and test-coverage criteria; review recommends concrete new test names + expected I/O.
9. 9. Implement review feedback at xHigh effort on the same branch. Per-fix commits include a before/after snippet. Push to the same branch; no new PR per round.
10. File a follow-up issue for everything deferred. Title: <crate-or-area>: nice-to-have polish from the #<PR>/<review-issue> review. Body is a markdown checklist with before/after per item + an Acceptance section.
11. Cross-link from the PR body. After merge: update rustyrods#17. Tick the merged component's box. Add the follow-up issue from step 10 under the phase's follow-ups sub-list. Don't edit phase-section body text — update MASTER-PLAN.md first if scope changes.

My guidelines / duties

- Babysit claude, don’t let it decide on its own and guide it
  - auto-mode only for polish issues
- Review and eyeball all the merge requests
- Ensure that what is implemented and referenced in the tracking issues is actually updated
  - Claude tends to not obey instructions and skips ticking boxes as it goes
    - Things improved in 4.8
- Challenge Claud’s statements and ask it to re-work, refine
- Ensure that there is no compromise on performance
- Develop skills for benchmarking and profiling (lots of fun)
  - Amd uprof, valgrind

Outcome

- iRODS 5.0.2 compatible server in Rust
- ~4500 unit tests
- ~800 slow tests (run nightly only)
- Bench suite

Outcome

```
github.com/AlDanial/cloc v 1.96
-------------------------------------------------------------------------------
Language                     files          blank        comment           code
-------------------------------------------------------------------------------
Rust                           553          27853         140963         284993
Bourne Shell                    53           1759           8525          13886
Markdown                        30           1423              2           5928
JSON                            20              0              0           4486
Python                           9            746           1949           3438
TOML                            63            283           1345           1435
SQL                             56            150           2209            973
YAML                            11             95            707            778
C++                              1             55            364            488
Java                             1             46            173            439
Go                               1             49             81            351
make                             1             70            440            246
Dockerfile                       8             64            314            209
CSV                              1              0              0            119
Bourne Again Shell               8             11             58             95
ReasonML                         2              3              0             87
Pest                             1             17             96             53
XML                              2              3             24             51
Maven                            1              9             34             45
C/C++ Header                     1             18            202             43
Text                             6              0              0              6
-------------------------------------------------------------------------------
SUM:                           829          32654         157486         318149
-------------------------------------------------------------------------------
```

Benchmarks

Microbenchmarks (single-op, no I/O)

Operation	Rust	C++	Ratio
`DataObjInp_PI` encode	115 ns	1,949 ns	**Rust 17×**
`DataObjInp_PI` decode	419 ns	2,386 ns	**Rust 5.7×**
GenQuery codec round-trip	3,987 ns	15,987 ns	**Rust 4.0×**
Error-stack `from_legacy_code`	72 ns	395 ns	**Rust 5.5×**
Error-stack context-chain (depth 10)	222 ns	3,060 ns	**Rust 13.8×**
BLAKE3 1 MiB	8.14 GiB/s	8.43 GiB/s	parity (within ~4%)
SHA-256 1 MiB	2.30 GiB/s	2.34 GiB/s	parity
GenQuery2 parse (small / medium / large)	794 ns / 4.4 µs / 14.6 µs	1,752 ns / 3.4 µs / 9.4 µs	crossover — Rust 2.2× on small; C++ 1.3–1.6× on medium/large

Benchmarks

End-to-end (stock C++ icommands against each server, same Postgres)

Operation	Rust median	C++ median	Ratio
`iinit` handshake	14.8 ms	42.7 ms	**2.89×**
`ils` empty collection	14.5 ms	46.3 ms	3.20×
`iput` 1 KB	15.5 ms	58.6 ms	**3.79×**
`iput` 64 KB	15.8 ms	57.8 ms	3.65×
`iput` 1 MiB	20.1 ms	62.7 ms	3.12×
`iget` 1 KB	14.6 ms	49.5 ms	3.39×
`iget` 64 KB	14.7 ms	49.8 ms	3.38×
`iget` 1 MiB	15.9 ms	52.2 ms	3.29×
`iquest` simple SELECT	14.0 ms	42.4 ms	3.03×
`iquest` SELECT WHERE (RLS)	13.9 ms	42.6 ms	3.06×
`imkdir` + `irm -r` cycle	27.6 ms	101.0 ms	**3.66×**

Benchmarks

Connection concurrency (N concurrent clients, 50 sequential `ils` each)

Measures connection-handling under sustained concurrent load. Server pinned to CCD0 (`0–7,16–23` — the 96 MiB V-cache CCD), clients pinned to CCD1 (`8–15,24–31`). 5 timed iters + 1 warm-up per (stack, N).

N	Rust ops/s	C++ ops/s	Rust gain	Rust success	C++ success
1	72.3	21.9	**+230%**	100%	100%
10	592.8	172.7	**+243%**	100%	100%
100	722.0	237.5	**+204%**	100%	**44%**
1000	706.2	201.7	**+250%**	100%	**57%**

Benchmarks

Connection concurrency with GenQuery body (Tier-3 B3.18)

GenQuery-driven sibling of B3.16. Same N ∈ {1,10,100,1000} concurrent shell-client structure, but each call is `iquest "%s" "SELECT COLL_NAME WHERE COLL_NAME = '/tempZone/home/rods'"` instead of `ils`. The query returns one row and exercises GenQuery1 wire codec + catalog query layer + (Rust-side) RLS predicate.

Expected matrix (architectural prediction extrapolated from B3.16 measured + Tier-2 B2.11/B2.12 single-stream measured — see `expected_results_rationale` in `perf/e2e/b318_baseline.json`; maintainer-side run populate the per-cell numbers):

N	Rust ops/s (pred)	C++ ops/s (pred, ok-only)	Rust success (pred)	C++ success (pred)
1	~72	~24	100%	100%
10	~600	~190	100%	100%
100	~720	~250	100%	**~40-60%**
1000	~700	~200	100%	**~50-60%**

Cost

- 500M Claude tokens (in/out)
  - Claude Fable @ Max effort did review the whole codebase (before it was gone!)
- 23,000 github actions minutes (~400 min / day)
- Claude Max x20 sub (hit the limit at the end of every week - barely enough)
- Copilot Max
  - Mainly used for reviews
- OpenAI Max x5
  - Mainly used for reviews
- Development server on Hetzner: Ryzen 7950x3D , 128GB ram
  - Always runs at 4.5Ghz+, hits 5.2GHz continuous for single threads
  - excellent for dev work compared to Cloud VMs on dense servers: never exceed 2GHz
- My time: 2.5 months
- Value: ~50-70k$

Demo

```
└─ ~/.../examples/00-rust-server-c++-icommands $ docker compose up -d
[+] up 5/5
 ✓ Network rustyrods-example-00_default        Created
 ✓ Container rustyrods-example-00-postgres-1   Healthy
 ✓ Container rustyrods-example-00-setup-1      Exited
 ✓ Container rustyrods-example-00-rustyrods-1  Healthy
 ✓ Container rustyrods-example-00-icommands-1  Started
```

```
└─ ~/.../examples/00-rust-server-c++-icommands $ docker exec -it rustyrods-example-00-icommands-1 bash
rods@66d272acf3ca:~$ ils
/tempZone/home/rods:
rods@66d272acf3ca:~$ imiscsvrinfo
RCAT_ENABLED
relVersion=rods4.3.4
apiVersion=d
rodsZone=tempZone
up 0 days, 0:1
SSL/TLS Info:
    enabled: false
    server_implementation: rustyrods 0.1.0
```

Demo

```
└─ ~/.../examples/00-rust-server-c++-icommands $ docker compose logs -f rustyrods
rustyrods-1  | 2026-06-30T22:21:19.261724Z  WARN irods::server::stacktrace: could not create stacktrace direct
rustyrods-1  | 2026-06-30T22:21:19.262816Z  INFO irods::server::boot: binding listener bind_addr=0.0.0.0:1247 
rustyrods-1  | 2026-06-30T22:21:19.262827Z  WARN irods::server::boot: WARNING: bound to non-loopback 0.0.0.0:1
rustyrods-1  | 2026-06-30T22:21:19.262830Z  INFO irods::server::boot: IRODS_CATALOG=postgres — building Postgr
rustyrods-1  | 2026-06-30T22:21:19.262834Z  INFO irods::server::stacktrace: stacktrace-file processor started 
rustyrods-1  | 2026-06-30T22:21:19.266091Z  INFO sqlx::postgres::notice: relation "_sqlx_migrations" already e
rustyrods-1  | 2026-06-30T22:21:19.274441Z  INFO irods::server::boot: Postgres catalog ready (migrations appli
rustyrods-1  | 2026-06-30T22:21:19.276697Z  INFO irods::server::boot: atime resolution sourced from R_GRID_CON
rustyrods-1  | 2026-06-30T22:21:19.277249Z  INFO irods::server::boot: delay() submitter wired (irule -> delay(
rustyrods-1  | 2026-06-30T22:21:19.277274Z  INFO irods::server::boot: resource configured name=unixfs0 kind=uni
rustyrods-1  | 2026-06-30T22:21:19.277277Z  INFO irods::server::boot: data-I/O enabled resources=1 default=Some
rustyrods-1  | 2026-06-30T22:21:19.277319Z  INFO irods::server::boot: legacy rule engine loaded (dynamic PEPs 
rustyrods-1  | 2026-06-30T22:21:19.277323Z  INFO irods::server::boot: legacy core.re static PEPs registered (s
rustyrods-1  | 2026-06-30T22:21:19.277709Z  INFO irods::server::boot: listening on 0.0.0.0:1247 addr=0.0.0.0:1
rustyrods-1  | 2026-06-30T22:21:24.878445Z  INFO irods::server::listener: accepted connection from 10.128.23.4
rustyrods-1  | 2026-06-30T22:22:41.492464Z  INFO irods::server::listener: accepted connection from 10.128.23.4
rustyrods-1  | gen_query: WARN KeyValPair_PI dropped non-ticket keys (slice 2c3 propagates TICKET_KW only — ZO
rustyrods-1  | 2026-06-30T22:22:44.074117Z  INFO irods::server::listener: accepted connection from 10.128.23.4
```

Demo

```
rods@66d272acf3ca:~$ truncate -s 1024 1KB.bin
rods@66d272acf3ca:~$ iput 1KB.bin
rods@66d272acf3ca:~$ ils
/tempZone/home/rods:
  1KB.bin
rods@66d272acf3ca:~$ iget 1KB.bin 1KB.bin.out
rods@66d272acf3ca:~$ md5sum *
0f343b0931126a20f133d67c2b018a3b  1KB.bin
0f343b0931126a20f133d67c2b018a3b  1KB.bin.out
```