Generic Load Driver

Development Manual

Table of Contents

TODO 3

Release Procedure 6

Release 6

# TODO

* **gld. Failure Counter Support.** Partially implemented, not tested end-to-end. Implement it when I need it and return to related tests that currently fail.
* **gld. Need an option to see exceptions right away.**
* **gld: --verbose should propagate to log4j.** Write down recipe in “Projects”.
* **gld Need configuration files**
* **gld.** Introduce total heap and free heap among the system metrics. Introduce tests.
* **gld.** Cumulative message counter in stats.
* **gld.** Fix the –background help text, explaining clearly when the process exits. Now both send and receive keep spinning until stopped externally.
* **gld.** ActiveMQ: how can I read the queue depth via JMX (or other command line means). Use that info to figure out where the messages are stuck – embedded broker or main broker (org.apache.activemq:type=Broker,brokerName=localhost,destinationType=Queue,destinationName=TEST)
* **gld.** Add configurable TTL per message.
* **gld.** Investigate the --endpoint-policy implications on Spring.
* **em** better statistics collection. Currently I am using a script (collect-stats), but this should be generalized and made better.
* **series.** I can aggregate statistics by just comparing the numbers in different files, a better solution will involve “series”.
* **gld.** Why does ‘q’ does not work on consumer.
* **gld**. Get rid of the Statistics interface and all associated classes.
* **gld** – bring gld on the “help for individual commands” convention where I type gld –h|--help <command> and I get help for that command only. Get rid of the long HELP file. Also, -h should apply to the commands handled by the bash wrapper (such as ‘stop’).
* **gld**. Implement a better final stat dump, the original was useful: System.out.println(((CollectorBasedCsvStatistics)statistics).aggregatesToString())
* **gld.** Clarify the SLF4 thing.
* **gld.** Collect gc logs on producer/consumers/brokers.
* **Sampler** Keep in mind that I want to zip and ship results remotely.
* **System-wide Metrics.** Aggregate base classes for metrics that have the same source MXBean.
* **From activeMQ load testing:** Count messages based on their ID.
* **Switch to cliff.** 05/03/15.
* **The Hierarchical Repository Key Reading is Slow.** Large variation in key loading time, locally. Try to load it in parallel and also look into how I am reading the keys from the file – the way I am doing it might be inefficient.
* **Compression.** A generic compression service? Apply it to the cache only?
* **Multiple independent agents.** This way I can increase the amount of load.
* **log4j**. Externalize log4j.xml so I can modify it directly, don't have to unjar for it. Ideally, the debugging should be turned on by --debug.
* Make it easy to dump the runtime configuration – implement it as a “dry-run” where I only have to specify “dry-run” somehow and not modify anything else.
* Introduce a CacheLoadStrategy and make all cache load strategies subclasses of that, the same Send and Receive are subclasses of JmsLoadStrategy.
* Implement support for --provider Currently we assume Infinispan for “keyvalue” and ActiveMQ for “message”.
* Move the Content’s StorageStrategy to Configuration.
* Make sure cld load embedded hierarchical fills up the embedded hash with the content of the hierarchical repo (on one and multiple threads).
* Isolate HierarchicalStorageStrategy. toHex(…) into its own class (or Util), test the heck out of it and update https://home.feodorov.com:9443/wiki/Wiki.jsp?page=SHA1
  + 10 different values – calculate hex sha1 with git and write 10 unit tests for them. Use / \* and other characters that are not FS-friendly
* Unify KeyStore and StorageStrategy.
  + Rename KeyStore to LocalStore – it stores keys and values.
  + Verify which of the top-level ConfigurationImpl attributes still make sense.
  + Keep all strategies in the Configuration and not in commands
* Exact distributions of max-operations between runner threads.
* If CLI starts in background, it cannot be controlled with System.in.read – find an alternative solution.
* All “strategies” should end in Strategy.
* Understand the commented out ReadTest tests, and either uncomment or write equivalent ones.
* The ReadThenWriteOnMiss.synthethicValue is created with the default value size of 1024. Make it more user friendly, hint somehow that the values in the cache are larger (or of different size).
* Move Util.getInstance() and associated test to Nova Ordis Utilities. \* Dynamic Logging in interactive mode until I find a way to get the information I need from infinispan and not rely on runtime.
* Reconcile the new style of load manager that preconfigures the operation inside, with the old style SingleThreadedRunner op.initialize(ThreadLocalRandom.current(), lastWrittenKey, keyStore); Old logic in "OldLoad" and "OldSingleThreadedRunner"

saving keys into the file bug: because the collector is doing it asynchronously, some of the keys will stay in memory until "press the key" so we may end up with less keys on disk.

* Change the name of Connect to "interactive" or something similar.
* **ExitGuard** 06/05/15 send must stay up indefinitely until shutdown – especially for embedded broker. Currently implemented with an ExitGuard that always block. It works fine for the time being, but it needs more thought – should allow behavior configuration from the command line.

# Release Procedure

Turn Console Logging to INFO

**Build the Local Release for Integration Testing**

./util/make-zip; ./util/install-locally --force

This will run the test suite and won’t be able to build the local release if the tests don’t pass.

**Run Integration Testing**

cd ~tmp

mkdir cld

cd cld

From a different terminal, start the Infinispan instance.

cd ~/runtime/jboss-datagrid-6.3.2/profiles/node01/

./node01.sh

Then:

cld version

Then walk through scenarios described in “**Error! Reference source not found.**” above.

**Save/Restore from Hierarchical Storage (Embedded Cache)**

cld content --nodes embedded[50000] --storage-strategy hierarchical --root ./hierarchical

find ./hierarchical -type f | wc -l

cld load --nodes embedded --threads 10 --read-to-write 0 --storage-strategy hierarchical --root ./hierarchical

cld load --nodes localhost:11222 --threads 10 --read-to-write 0 --storage-strategy hierarchical --root ./hierarchical

## Release

Run test suite – all tests must pass.

Write down the number of tests that pass.

464/7/0.

Update pom with the version.

mvn clean

Check in.

Specify version, a short summary of features and the number of tests.

mvn clean

git add ...

git commit -m "release 1.0.1. 464/7/0."

git push

Make the zip. Place it in “releases” and also install it locally for reference.

./install --clean [--no-tests]

./install --clean [--no-tests] --zip-only

Verify that the release was installed locally by running gld version.

Push the new release to GitHub:

cp ./target/gld-1.0.1.zip ./releases

git add ./releases

git commit -m "published release 1.0.1"

git push

Not sure releasing in in GitHub is the best idea – it’s very large.

Increase the version in pom and append –SNAPHSOT

svn ci -m "starting 1.0.10-SNAPSHOT"