GLD TODO

* **Format instances thread safety**. Remove public static final SimpleDateFormat and other Formats everywhere, and replace them with static method that create the instances on demand. We need to do this because Format instances are not thread safe. See io.novaordis.events.api.event.DateProperty. getDefaultDateFormat() as example.
* GLD is not coded to perform a transparent fail-over with HornetQ. Look to how it’s done <https://github.com/NovaOrdis/playground/tree/master/jboss/hornetq/failover-testing-framework> and then port the appropriate functionality to GLD.
* Test and document session-per-thread.
* Make gld use events-csv for data generation.
* Document and test “receiving unlimited messages”
* **(!)** Ctrl-C on a running gld should cleanly close the on-going connections – I see connections lingering on the queue after closing.
* I need a “foreground mode” for short loads – where I start it, get the stats at stdout, and it dies.
* Implement toString() for all other Operations to be similar with Write.
* The content of TO-DISTRIBUTE-extensions should be depleted.
* Consolidate mocks. Too many mocks, too confusing, we should have a set of each, if possible.

# Second Pass

* Implement Redis support.
* **Longs.** TODO Ps772s correctly handle longs, now we just do implicit conversion.
* **Gld Wrapper Classpath Extraction**. The current implementation appends everything it finds under ./extensions/<…>/ to classpath. If the same gld instance has two extensions for two different versions of the same product (jboss-datagrid-6, jboss-datagrid-7), then the common JARs will clash and the common libraries will clash. Functionality to detect this, warn and possibly allow selection.
* Uncomment // \*@Test and make all tests pass
* Why do I explicitly need to specify   
    
   <dependency>

<groupId>org.infinispan</groupId>

<artifactId>infinispan-commons</artifactId>

<version>${jdg.infinspan.client.hotrod.version}</version>

</dependency>

in top of the infinispan-client-hotrod dependency? Why did transitivity did not work? It is repository broken?

* After release a stable JDG 7 version, document the final <dependencySet> solution here: <https://kb.novaordis.com/index.php/Custom_Maven_Assembly_Descriptors#Dependencies>
* Two interdependent load driver instances:
  + Coordinate common configuration from a single source.
  + Coordinate startup
  + Coordinate statistics collection and merging. Events functionality? Merge?
* Make sure the current equivalent of --attribute-size is implemented and works.
* **extensions child-parent relationship**
  + Added <parent> in jboss-datagrid-7 POM to fix this:   
      
    release snapshot  
    [error]: failed to read POM file /Users/ovidiu/projects/gld/./pom.xml: we only support lockstep versioning mode, yet the project GLD seems to contain independent module versions (/Users/ovidiu/projects/gld/./extensions/jboss-datagrid-7/pom.xml)
  + Review <https://kb.novaordis.com/index.php/Multi-Module_Maven_Projects> in the light of the finding that a module POM does not need a <parent>. What is <parent> for, then? Differences noticed so far:
    - I need to specify <build> again and reconfigure all plugins in the child POM. It seems the parent’s <build> does not propagate down.
* The old pom.xml is available under “TO-DISTRIBUTE-extensions”. Keep it around as long as I am still working on JDG6, Redis, ActiveMQ and everything else. I may need it to figure out dependencies.
* Clarify CommandLineConsole presence and usage in MultiThreadedRunnerImpl.run(). Probably need to get rid of it.
* **Review and remove gld-api io.novaordis.gld.api.todiscard** 
  + **Understand and Refactor RedisFailure.java**
  + **Understand and deplete SystemStatistics.**
* **Operation Refactoring -** We should not Operation.perform(Service s). The Service reference is internal and inextricably related to the Operation instance. We should only have a perform(), and possibly Operation.getService().
* **Separate Sampler Module.** Separate the Sampler and use the types from events instead of creating our own. Currently they’re in API. Sampler implementation probably belongs to its own module. Reuse as much as I can from events.
* Understand DeprecatedStatistics and get rid of it if I don’t need it anymore.
* **snakeyaml Handling of Longs**. (read snakeyaml FAQs) LoadConfiguration.getOperations() parses and return Integer. Fix to return longs.
* **Consolidate ReadThenWriteOnMissLoadStrategy and WriteThenReadLoadStrategy**. There is a lot of common copied-over code, must refactor and consolidate.
  + **Transfer ReadWriteRatio javadoc** from io.novaordis.gld.api.cache.load.ReadWriteRatio to NOKB.
* **DeleteLoadStrategy, Delete** not fully implemented yet, no tests.
* **Collocated** – continue to build a fully working load driver into an embedded instance -> once that is ready, extract the topology into the Architecture diagram, and replicate the topology for the CacheService, JmsService and HttpService.   
  + **Refactor “local” to “collocated”**, “local” is misleading.
* HtttSession LoadStrategy: Default HTTP strategy bug: it looks like in some cases WRITE arrives after INVALIDATE so sessions leak in memory. Also, the trace below:

com.novaordis.gld.strategy.load.cache.http.HttpSessionSimulationException: session with ID "VTbych4K7Eidyk7yGbZvfbEf" already found in cache: {0=0, 1=0, 3=com.novaordis.gld.strategy.load.cache.http.DistributableSessionMetadataSimulation@66a8777, TEST-KEY=TEST-VALUE}

at com.novaordis.gld.strategy.load.cache.http.operations.HttpSessionCreate.performInternal(HttpSessionCreate.java:65)

at com.novaordis.gld.strategy.load.cache.http.operations.HttpSessionOperation.perform(HttpSessionOperation.java:65)

at com.novaordis.gld.SingleThreadedRunner.loopUntilStoppedOrOutOfOperationsOrDurationExpired(SingleThreadedRunner.java:210)

at com.novaordis.gld.SingleThreadedRunner.run(SingleThreadedRunner.java:102)

at java.lang.Thread.run(Thread.java:745)

* Extension ZIPs should be auto-installable into gld core by running

gld install <extension-file.zip>

There should also be an alternative manual installation. Document it in NOKB.

* Extension installation problem: we run into problems if we develop and release an extension that relies on newer versions of novaordis-utilities or other common gld core/extensions dependencies. This is because the installed gld core has the old version in its library, and we don’t ship the extension with any dependency. The solution is A) upgrade novaordis-util in gld core B) release and install a new gld core, then C) re-install install the extension. Find a more elegant solution.
* Go over the NOKB documentation <https://kb.novaordis.com/index.php/Gld> and clean it up, make sure it is in sync with the code base.
* '''Outcome of JDG testing'''
  + Make sure I can run two gld instances in parallel without interference. Currently there's just one log file ($(pwd)/gld.log) and this gets overwritten.
  + Make sure the java memory can be configured individually per instance - on the command line. <tt>--memory 1024m</tt> that should override the value hardcoed in the wrapper.
  + Possibility to configure gc logging from command line <tt>--gc-logging /a/b/c-gc.log</tt>.
  + Memory leak analysis.