KIP-510: Metrics library upgrade

- Status
- Motivation
- Public Interfaces
- Proposed Changes
- Compatibility, Deprecation, and Migration Plan

Status

Current state: *Under discussion*

Discussion thread: thread

JIRA: *here*

Please keep the discussion on the mailing list rather than commenting on the wiki (wiki discussions get unwieldy fast).

Motivation

The current metrics library which Kafka is using is pretty old (version 2.2.0 from Yammer -latest version in 2012- and now we have its evolution -version 4.1.0 from Dropwizard-).

In the latest versions of the Dropwizard library, there are a lot of bugfixes and new features included which could be interesting for these metrics (ie: reservoris, support JDK9, etc). It's recommended to have a look into its changelog.

Public Interfaces

In the current metrics library version, it includes the rate unit when creating meters and timers. In this new library version, this info is removed and everything has as a rate unit "seconds". All meters in Kafka are using seconds except the meter "`*RequestHandlerAvgIdlePercent`" (in `KafkaRequestHandlerPool`) with unit "nanosecs". Additionally, in the value of the metrics you can see its unit.

On the other hand, the following metrics should be renamed to be more consistents:

- All metrics with the suffix "`RateAndTimeMs`" to "`RateAndTime`".
- Metric named "`yammer-metrics-count`" to "`dropwizard-metrics-count`" (or similar).

Proposed Changes

The way of adding metrics from the `KafkaMetricsGroup` could be simplified to something like this:

Sample code in KafkaMetricsGroup.scala

```
  def newGauge[T](name: String, gauge: () => T, tags: scala.collection.Map[String, String] = Map.empty): Gauge
[T] = {
    val supplier = new MetricSupplier[Gauge[_]] {
      override def newMetric(): Gauge[T] = new Gauge[T] {
        override def getValue: T = gauge()
      }
    }
    kafkaMetricRegistry.gauge(metricName(name, tags), supplier).asInstanceOf[Gauge[T]]
  }

  def newMeter(name: String, tags: scala.collection.Map[String, String] = Map.empty): Meter =
    kafkaMetricRegistry.meter(metricName(name, tags))

  def newHistogram(name: String, biased: Boolean = true, tags: scala.collection.Map[String, String] = Map.
empty): Histogram = {
    val supplier = new MetricSupplier[Histogram] {
      override def newMetric(): Histogram = {
        //TODO evaluate adding other kind of reservoirs
        val reservoir = if (biased) new ExponentiallyDecayingReservoir() else new UniformReservoir()
        new Histogram(reservoir)
      }
    }
    kafkaMetricRegistry.histogram(metricName(name, tags), supplier)
  }

  def newTimer(name: String, tags: scala.collection.Map[String, String] = Map.empty): Timer =
    kafkaMetricRegistry.timer(metricName(name, tags))

...
```

Sample code adding new metrics

```
// With these new changes
  newGauge[Long]("MemoryPoolAvailable", () => memoryPool.availableMemory())
  newGauge[Long]("MemoryPoolUsed", () => memoryPool.size() - memoryPool.availableMemory())



// Old style
newGauge("MemoryPoolAvailable",
  new Gauge[Long] {
    def value = memoryPool.availableMemory()
  }
)
newGauge("MemoryPoolAvailable",
  new Gauge[Long] {
    def value = memoryPool.size() - memoryPool.availableMemory()
  }
)
...
```

Compatibility, Deprecation, and Migration Plan

- Metrics ending in "`RateAndTimeMs`" will be deprecated in favor of the ones with the same name but ending in "`RateAndTime`".
- Metric named "`yammer-metrics-count`" will be also deprecated.
- Meters "`RequestHandlerAvgIdlePercent`" and "`ControlPlaneRequestHandlerAvgIdlePercent`" will have as a rate unit "seconds" instead of "nanoseconds".
- Changes in attributes for meters:
    - The current value for "`RateUnit`" attribute in meters is "SECONDS". With this change, the value will be "`events/seconds`".
    - "`EventType`" attribute is removed.
    - "`LatencyUnit`" attribute is renamed to "`DurationUnit`". Its value changes from "MILLISECONDS" to "`milliseconds`".