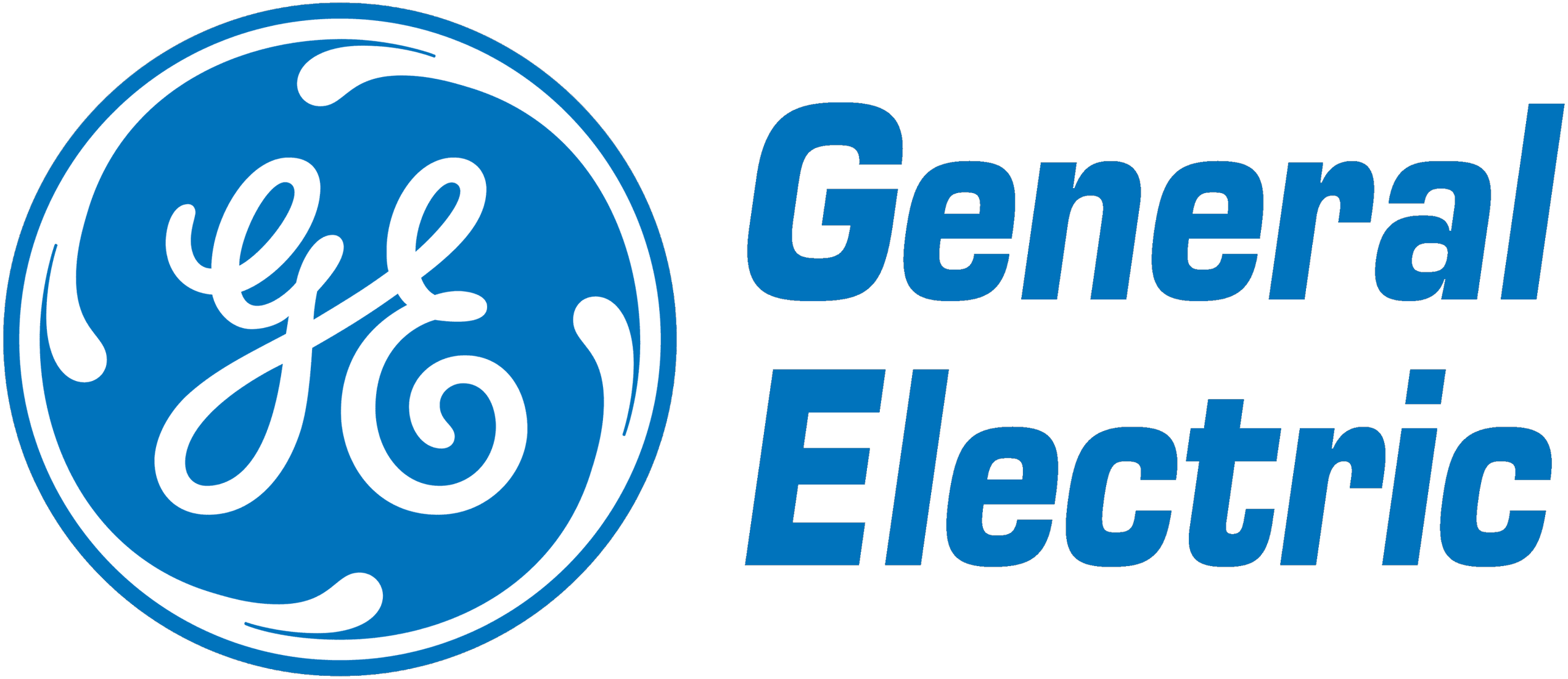

General
Electric

GE Historian

- The GE Historian is a building automation system
- It uses a RESTful API with a username and password to generate a token

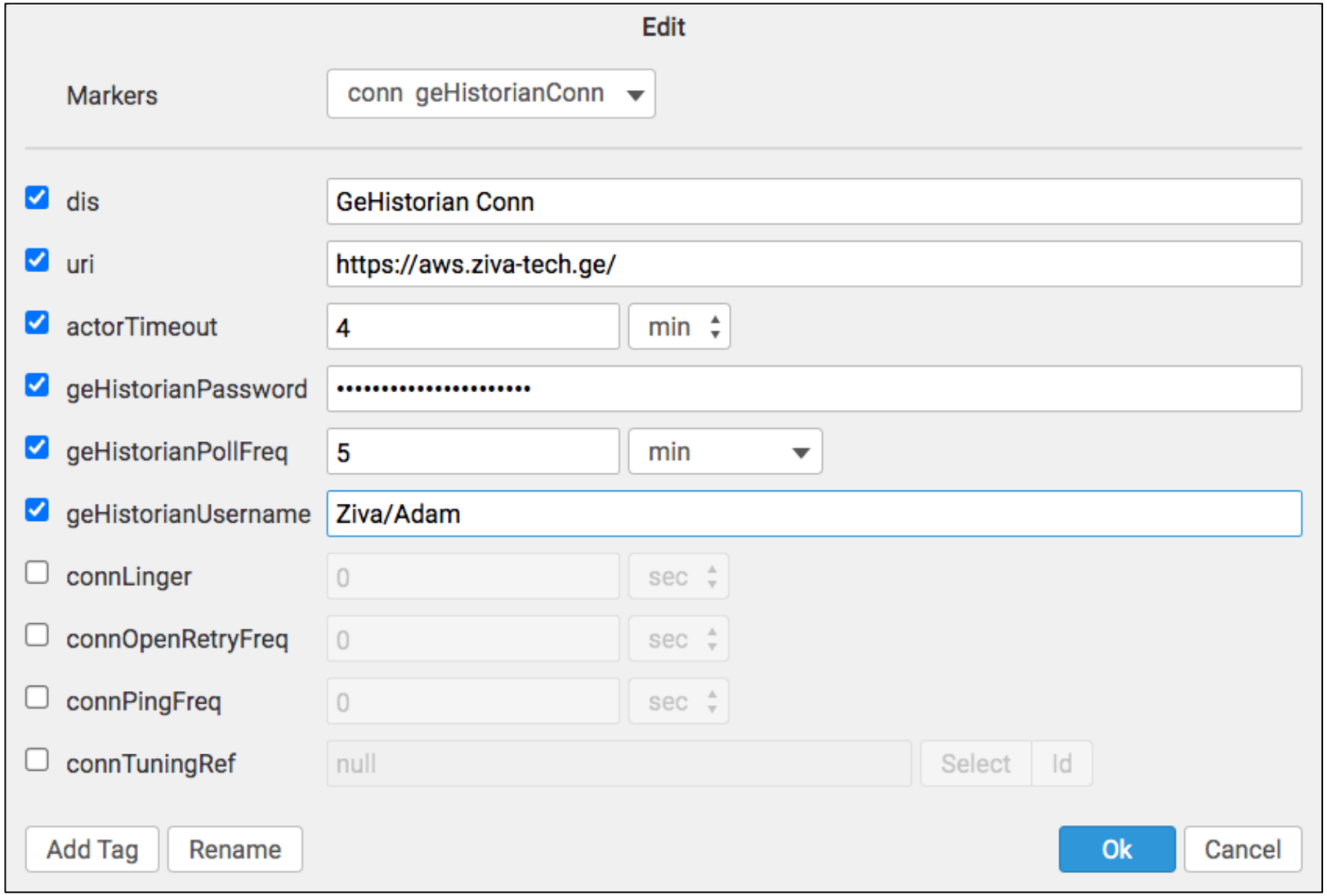
Licensing

- Each GE Historian connector extension is licensed to a single SkySpark Node Id
- Each GE Historian connector extension is licensed to a set maximum number of points
- If either of these states goes into fault, the issue must be corrected and SkySpark must be restarted

GE Historian Connectors in SkySpark

- Creating a GE Historian connector:
  - dis: Any Readable Name
  - uri: `https://host/` ie `https://gehistorian.net/`
  - actorTimeout: How long a busy connector will wait to respond to a new message
  - geHistorianUsername: This will be provided by your GE Historian Team
  - geHistorianPassword: This will be provided by your GE Historian Team
  - The geHistorianPollFreq tag determines the minimum amount of time before the cache can be refreshed and how often the curVal values will be updated (which is from where both polling with intervals and COV come). The default value of 1min is good for detecting COV changes. The curVals do not matter for trend log synching.
  - Of course, these records will have the geHistorianConn and conn tags
- These can be created in the connector app

GE Historian Connector

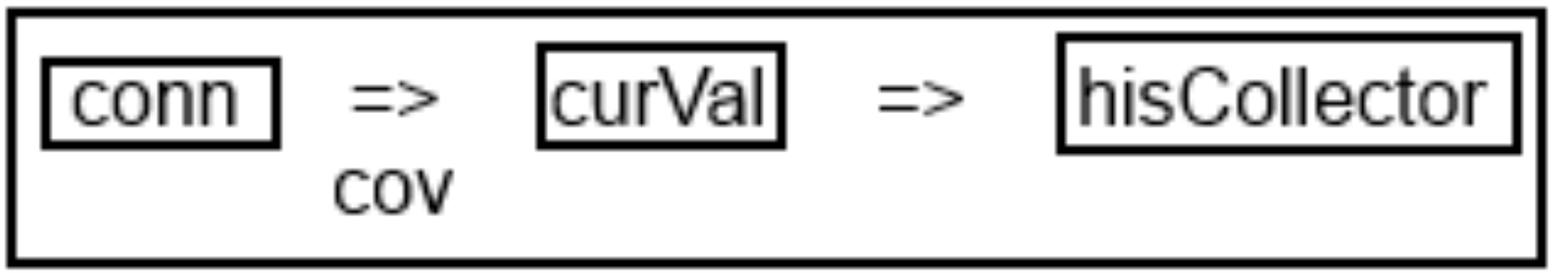


GE Historian Connector Set Up

- You must set up the API on the ALC side, then SkySpark will be able to talk to it. As long as the uri, username, and password are correct, it should work.
- You can run read(geHistorianConn).geHistorianToken if you think you might not have received a token

Polling of curVal

- geHistorianPollFreq (duration)
  - How often you want SkySpark to poll the connector for COV
  - The default is 5min
  - It is a tuning parameter
  - Note that this determines how often the curVal in SkySpark changes. You would still need to set how often the data is collected with the hisCollect tags.



GE Historian Functions Part I

- geHistorianSyncHis – sync history from a trend log
  - Use the geHistorianHisInterval tag on the point to specify what interval data you want. Supported values are: 1min, 5min, 10min, 15min, 20min, 30min, and 60min. The default is 15min.
  - readAll(geHistorianHis).geHistorianSyncHis(null)
- geHistorianSyncCur – force one or more points to sync
  - readAll(geHistorianCur).geHistorianSyncCur
- geHistorianPing – send ping to connector to check connection
  - read(geHistorianConn).geHistorianPing
- geHistorianHisRead – attempt to do a raw read from the geHistorian trend without synching (will tell you if point is trended)
  - Adding the debug marker tag to the connector will show you the raw response which is usually secondly data; otherwise, the default is minutely data
  - read(geHistorianConn).geHistorianHisRead(read(geHistorianHis), today)
  - read(geHistorianConn).geHistorianHisRead("address1", today)
- geHistorianPointChecker – look for points that are not in the GE Historian server and mark them with a bad marker tag

GE Historian Functions Part 2

- geHistorianVals – return a list of addresses and curVals (good for testing if an address exists since 1 bad address seems to cause a failure for the whole read block)
  - Adding the debug marker tag to the connector will show you the raw response
  - read(geHistorianConn).geHistorianVals(readAll(geHistorianCur))
  - read(geHistorianConn).geHistorianVals(null, ["pointAddr1", "pointAddr2"])
- geHistorianCmd – run a custom command in GE Historian
  - read(geHistorianConn).geHistorianCmd("historian-rest-api/v1/tags")
- geHistorianScadas – return the available scadas
  - read(geHistorianConn).geHistorianScadas()
- geHistorianPoints – return all points or a filtered subset
  - Adding a debug marker tag to the connector attempts to add the units tag to this function as well as the builder learn tree. It makes this connector very slow.
  - read(geHistorianConn).geHistorianPoints("Building1")
  - read(geHistorianConn).geHistorianPoints()
- geHistorianProperties – return the GE Historian properties of a given point
  - read(geHistorianConn).geHistorianProperties("Building1.Zone1.AirTemp")

GE Historian View Functions

- geHistorianDupFinder – find points with the same addresses (default true will add dupPoint tag to point)
  - geHistorianDupFinder(true)
- geHistorianMakePointsToBuckets – assigns tuning records to GE Historian curVal points (use geHistorianTuningBuckets view)
  - Number of points per connTuningRec
  - geHistorianMakePointsToBuckets(100)
- geHistorianMakeTuningBuckets – makes GE Historian Tuning records (use geHistorianTuningBuckets view)
  - Number of points per connTuningRec, starting pollFreq, max pollFreq
  - geHistorianMakeTuningBuckets(100, 5min, 9min)
- geHistorianViewStatus – see GE Historian points
  - geHistorianViewStatus()
- geHistorianViewSummary – see status of GE Historian points
  - geHistorianViewSummary()
- geHistorianViewTuning – display GE Historian tuning records
  - geHistorianViewTuning()

Getting Data for GE Historian Points

Manually or Automatically

- Collecting current values (from curVal):
  - Each point must have these tags:
    - geHistorianConnRef (ref) which is a ref that points to the GE Historian Connector
    - geHistorianCur (str) which points to the GE Historian point object
    - his (marker)
    - cur (marker)
    - hisCollectCov (marker or number) for cov collection OR
    - hisCollectInterval (duration) for polling
- Getting history data from Trend Logs:
  - Each point must have these tags:
    - geHistorianConnRef (ref) which is a ref that points to the GE Historian Connector
    - geHistorianHis (str) which points to the GE Historian Trend Log object
    - his (marker)
    - Run the geHistorianSyncHis(timeRange) function
      - generally as a task
      - readAll(geHistorianHis).geHistorianSyncHis(null)

Note: we recommend also using a hisCollectInterval of 24hr when using a hisCollectCov to catch faulty sensors.

Basic Troubleshooting

- Note that it is normal for a connection to close when it has no points in a watch for a given period of time
- For larger systems, you want to set the poll frequency to a higher value to avoid network congestion
- Make sure time zones, kinds, and units are the same in SkySpark and GE Historian
- The GE Historian seems very sensitive towards buckets. The absolute limit in testing appeared to be around 220 points / bucket.
- Note: If a point kind is Bool, 0 and 1 will automatically be converted to false and true for curVals and trends in SkySpark.

Getting Points via Code

```
() => do
  read(geHistorianConn).geHistorianPoints.map r => do
    r.merge({equipRef: read(equip)->id
      siteRef: read(site)->id,
      hisCollectInterval: 15min, })
  end.each r => commit(diff(null, r, {add}))
end
```

Run as a task in large systems to avoid a timeout error

Getting Points with GUI

- We recommend using the Builder App's drag and drop capability to select which points are of interest as there are often many superfluous points