Elettra Sincrotrone Trieste

# HDB++: a new archiving system for TANGO

L. Pivetta

on behalf of the HDB++ team

R.Bourtembourg
JL.Pons
C.Scafuri
G.Scalamera
G.Strangolino
P.Verdier
L.Zambon

# Outline

Requirements

Event-based archiving

HDB++ EventSubscriber (Archiver)

HDB++ ConfigurationManager

Back-end database interface

Historical data extraction

HDB++ configuration GUI

HdbExtractor++ GUI

HdbViewer GUI

Current status

Conclusions

# Requirements

- **Written in C++**
- **Event-driven**: exploit the TANGO publish/subscribe mechanism
- Architecture based on:
  - One or more archivers (EventSubscriber TANGO ds)
  - Configuration management (ConfigurationManager TANGO ds)
  - Libraries for data insertion and extraction (C++ and Java)
  - Data extraction TANGO ds / clients
- **Fast**
  - One database for slow and fast archiving (up to 1 Khz, possibly more)
- **Flexible**
  - Easy to manage and maintain even without GUI frontends
- **Self contained**
  - Single source for all configuration parameters (TANGO database)
- **Modular**
  - Abstraction+implementation libraries to support different database engines and schema
    - Support for existing HDB schema on MySQL
    - Support for **hdb++ new schema** with improved features (µs timestamp)
    - Support for **noSQL** back-end (Apache Cassandra, see WEM310)
    - Easily extensible to additional database/schema
- **Scalable**: same as TANGO, deploy as many DS as needed
- **GUI**: for HDB++ configuration and data extraction as well

# Event-driven archiving

- TANGO provides specific events for archiving purposes

- The **archive** event can be sent:
  - on value change → specify absolute or relative threshold
  - periodically → specify period

- Choosing the right thresholds is mandatory:
  - if the threshold is too large no events are sent → no archiving
  - if the threshold is too small too many events are sent → “noisy” archiving

- The right threshold is **strictly related to the variable/signal** to be archived (type, bandwidth, sampling rate...)

# HDB++ archiver: EventSubscriber

The EventSubscriber TANGO device server is the core of the HDB++ archiving system

- Event based; TANGO provides **archive events** on change and periodic basis
- Configuration stored in the TANGO database (device)
- One thread in charge of event(s) subscription and callback execution: fills a FIFO acting as producer
- One thread in charge of pushing data into the database; reads the FIFO as consumer

- Device methods allow to perform the following per-instance operations:
  - **add/remove** an Attribute to/from archiving
  - **start/stop** the archiving for all Attributes
  - start/stop the archiving for one Attribute
  - read the status of an Attribute
  - read the number/list of Attributes currently archived (started)
  - read the number/list of Attributes currently not archived (stopped)
  - read the number/list of Attributes in charge
  - read the configuration parameters of each Attribute
  - read the number/list of working Attributes
  - read the number/list of faulty Attributes with diagnostics
  - read the number/list of Attributes pending in the FIFO

} Archiver Management

- The EventSubscriber exposes some **additional figures**:
  - **for each instance**, total number of records per time
  - for each instance, total number of failures per time
  - **for each attribute**, number of records per time
  - for each attribute, number of failures per time
  - for each attribute, time stamp of last record
  - for each attribute, min and max processing and storing times

} Per Archiver on-line statistics
Much useful to spot anomalies

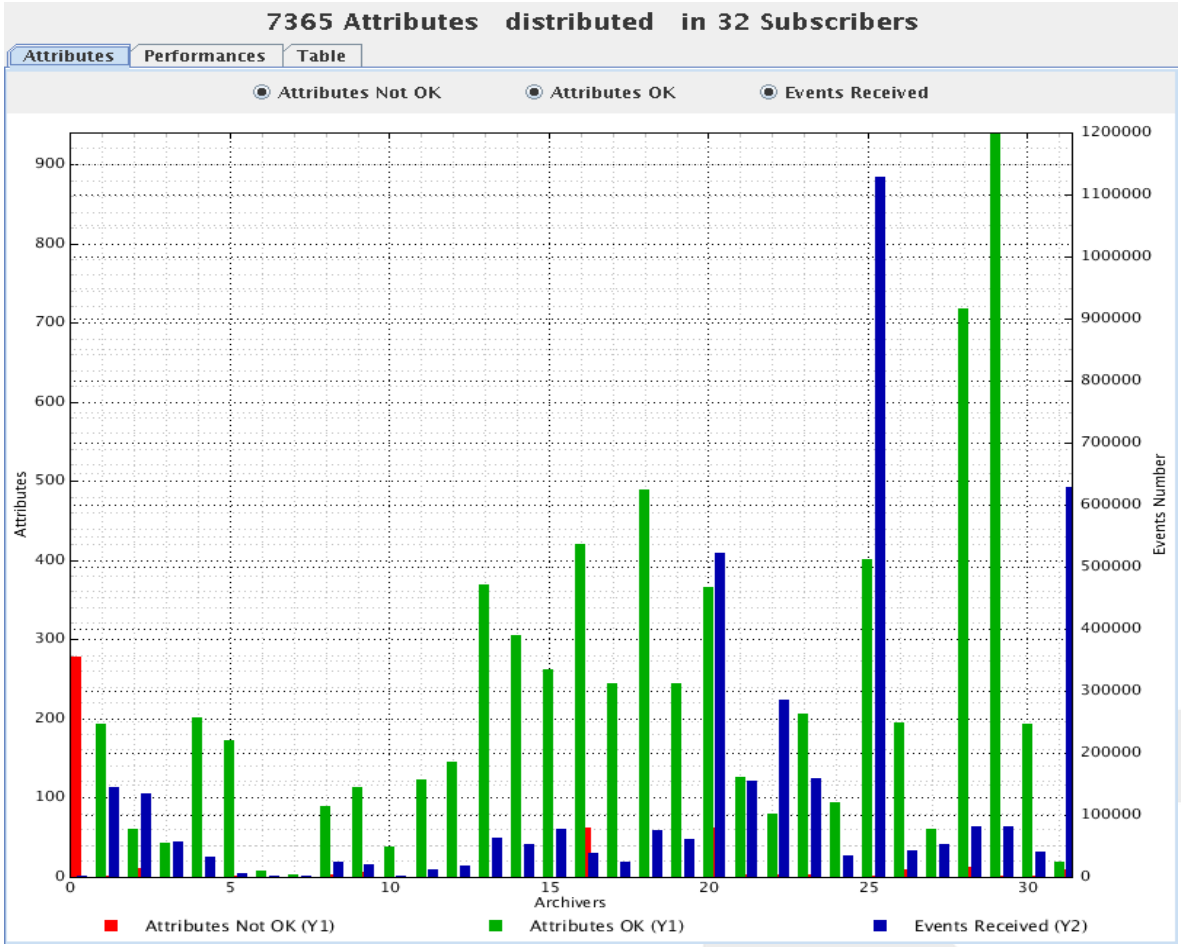
# HDB++ ConfigurationManager

The ConfigurationManager TANGO device server allows the global HDB++ archiving system management:

- **handle** the request of archiving a new Attribute
  - create an entry in the database if not already present
  - setup the Attribute archive event configuration
  - assign the Attribute to one of the archivers
- **move** an Attribute from one archiver to another
- keep trace of which Attribute is assigned to which archiver
- **start/stop** the archiving
- **remove** an Attribute from archiving

The Configuration manager exposes some **global statistics**:

- total number of Archivers
- total number of working/faulty attributes
- total number of events per second
- overall minimum and maximum processing and storing time



# HDB++ Database interface

A C++ API to address writing to the database from the archiver

- **libhdb++** : database abstraction layer
- **libhdb++mysql** : implementation, HDB++ schema support, MySQL back-end
- **libhdb++cassandra** : implementation, HDB++ schema support, Cassandra back-end
- **libhdbmysql** : implementation, legacy HDB schema support, MySQL back-end

The libraries allow reusing the EventSubscriber, the ConfigurationManager and the GUIs without changes

HDB++ is easily extendable to support additional back-ends(*) just writing the specific implementation library

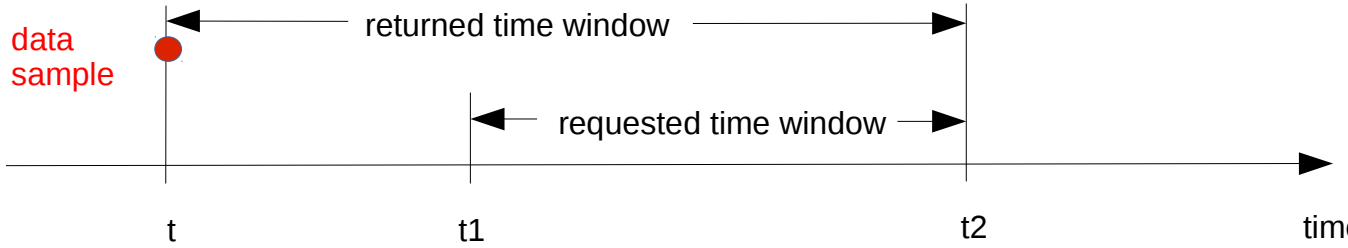
(*) not limited to database engines... HDF5 format on file?

# Historical data extraction

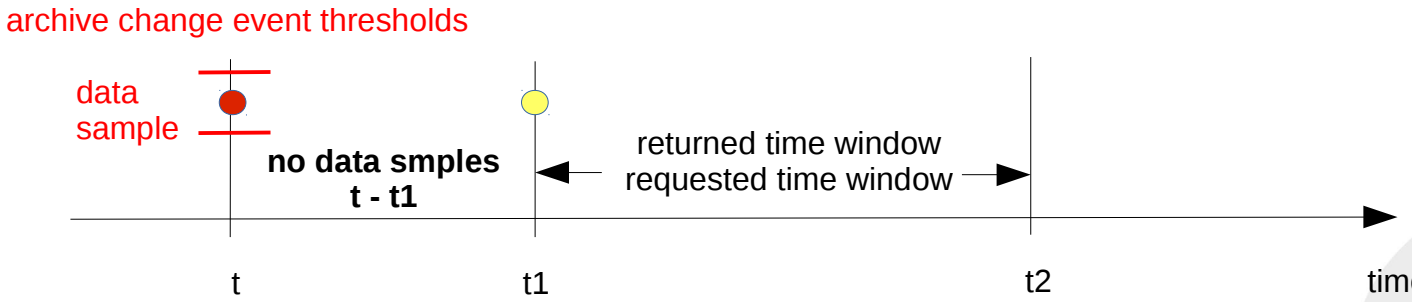
C++ and Java native extraction libraries have been developed

The data extraction library shall be able to **deal with event based archiving, i.e. data value change with respect to specified thresholds;** the possible lack of data in the requested time window shall be properly managed:

- returning some no-data-available error: in this case the reply contains no data
- enlarging the time window to include some archived data; no fake samples have to be introduced



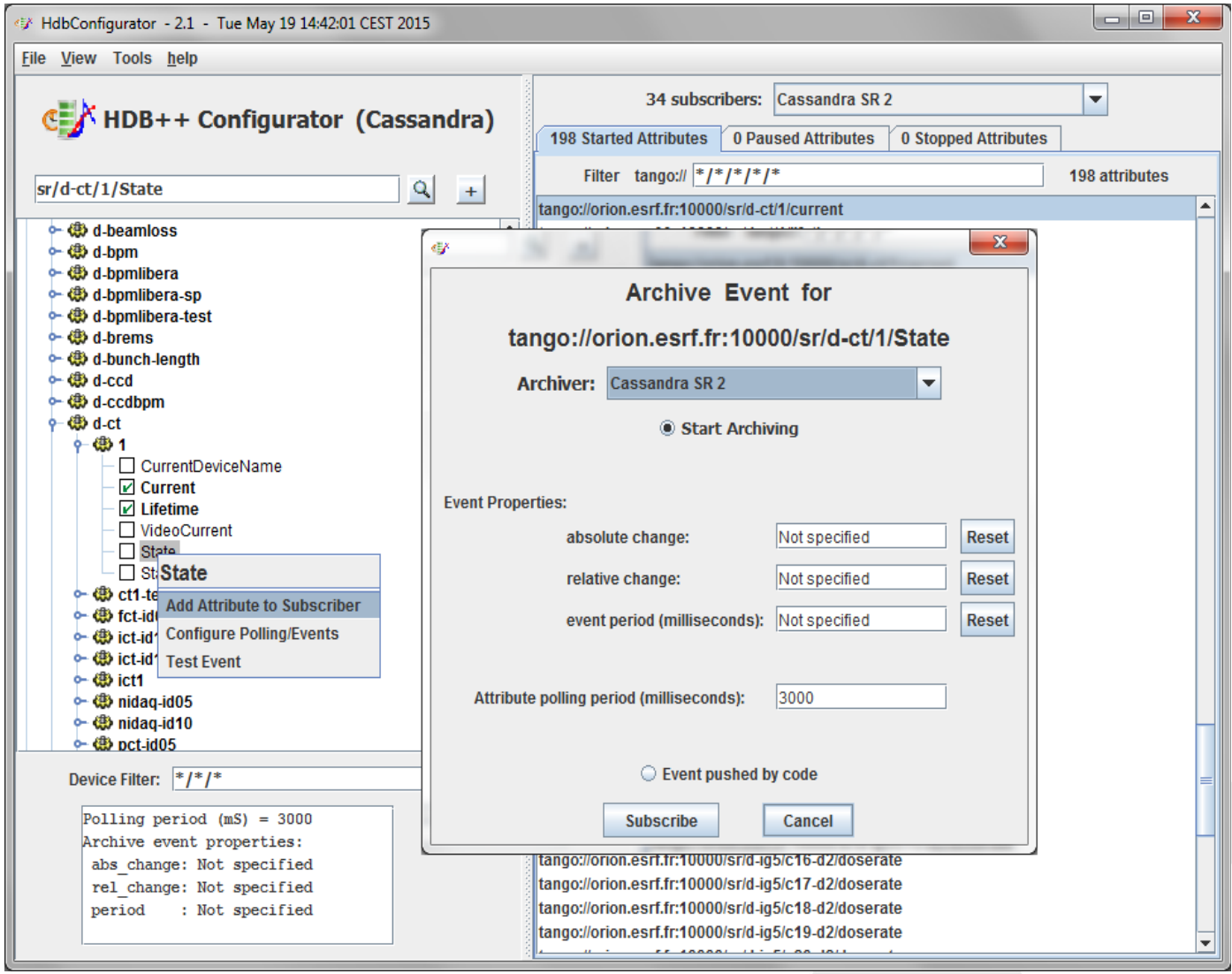

- returning the value of the last archived data anyhow; the requested time interval is kept and the last available data sample returned; the data value is guaranteed when **archiving on change,** care must be taken in case of **periodic archiving**



# HdbConfigurator

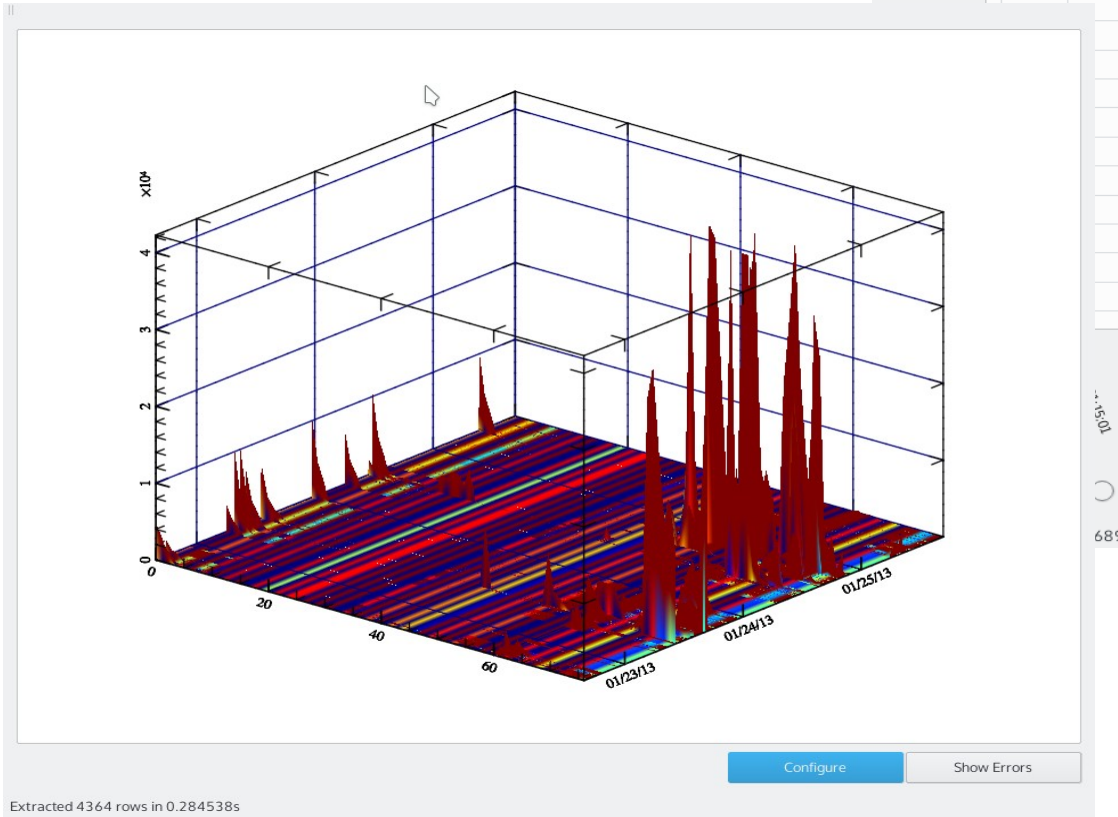
HdbConfigurator: a graphical user interface for the ConfigurationManager device server

- Jive-like device tree
- Selected Attribute archive event parameters bottom left
- Started, stopped, paused attribute lists
- Pop-up to select archiver and parameters



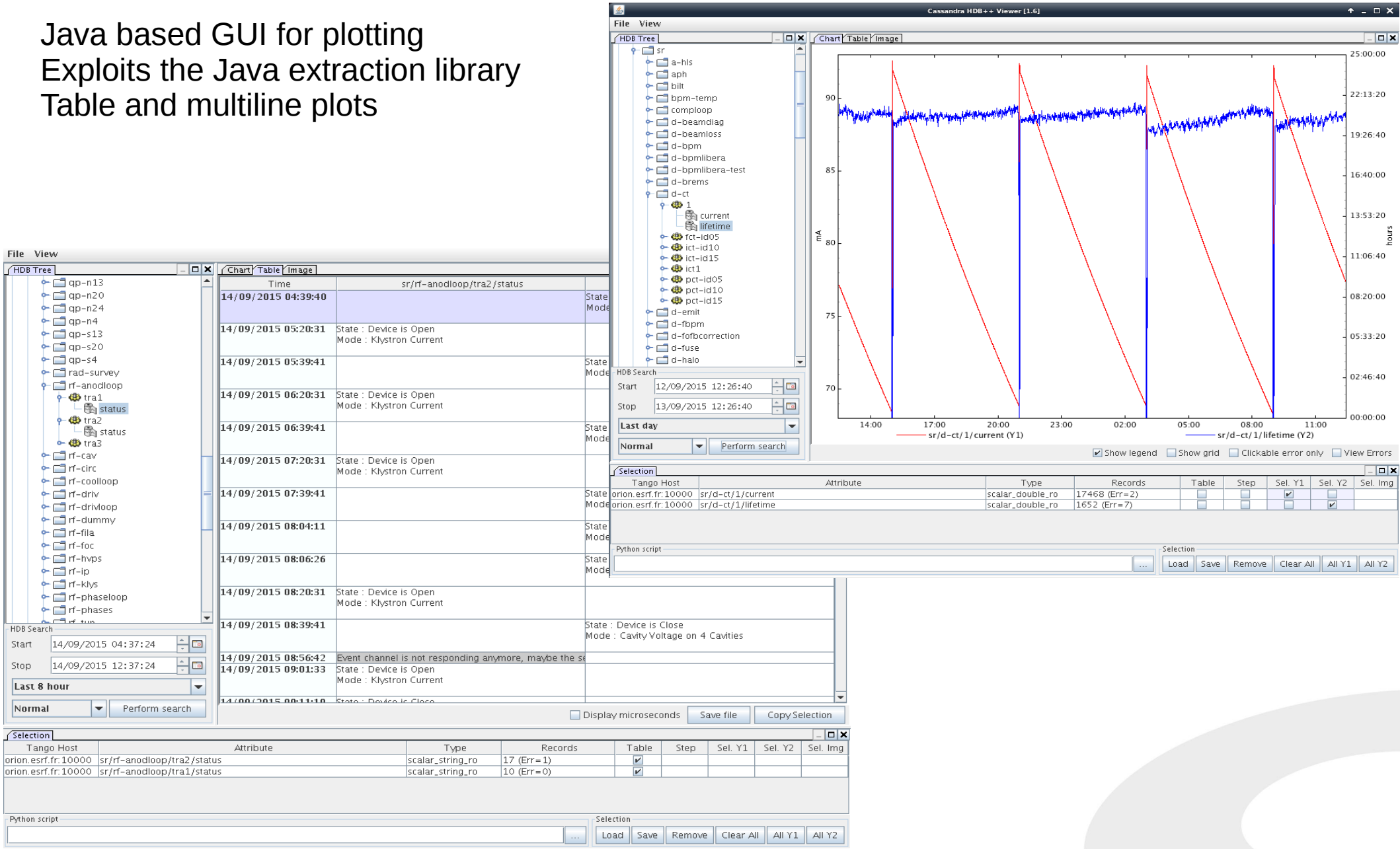
# HdbExtractor++ GUI

Qt based GUI using the MathGL framework for plotting
Exploits the C++ extraction library
Supports multiline and surface plots

# HdbViewer GUI

Java based GUI for plotting
Exploits the Java extraction library
Table and multiline plots



# Status

The HDB++ is still in active development, but **production ready**.

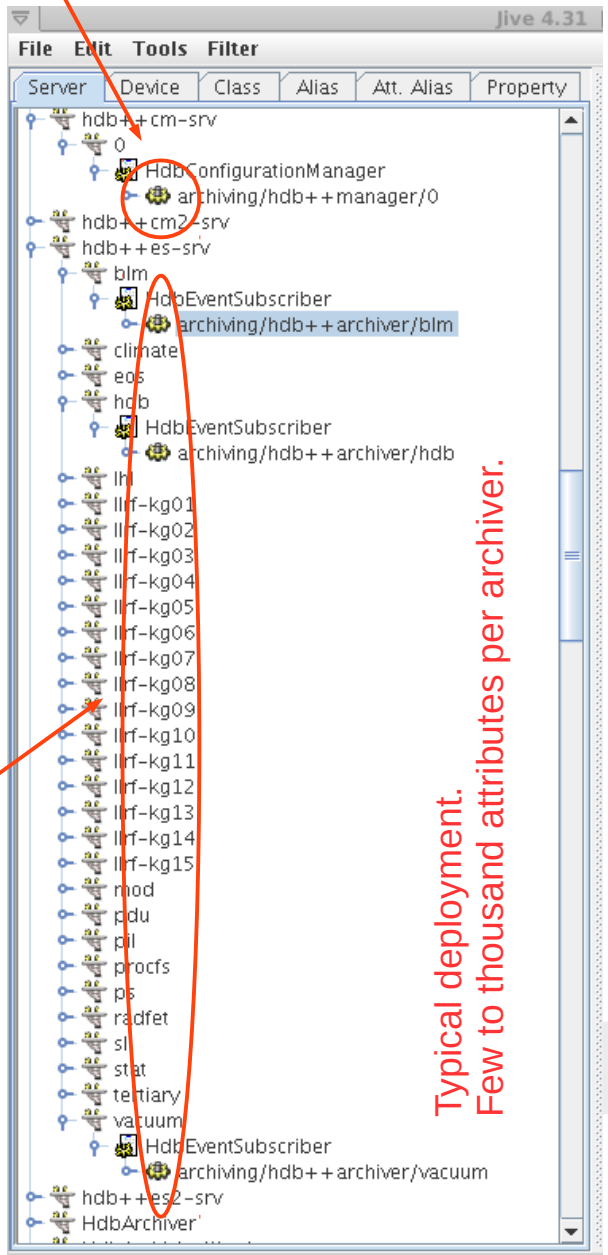
Running

- At ELETTRA
  - on FERMI since fall 2013
  - on ELETTRA since spring 2014
  - More than 6800 Attributes archived with both HDB legacy schema and HDB++ new schema on MySQL back-end
- At the ESRF
  - since July 2014 with MySQL back-end
  - Since October 2014 with Cassandra back-end
  - More than 7300 Attributes archived with HDB++ new schema on both MySQL and Cassandra back-end

Release: update almost twice per year

- Bugfix
- New functionalities

Tarball source distribution available since the beginning
Debian packages since few weeks



Elettra Sincrotrone Trieste

# Conclusions

ESRF The European Synchrotron

- HDB++: a new archiving system for TANGO has been developed
- Event based: exploits the full TANGO capabilities
- Modular by design: easily extensible to additional back-ends
- Historical data extraction libraries for C++ and Java are available to simplify data retrieval from db
- GUI for configuration
- Qt and Java based GUIs for plotting
- Debian package available

Elettra Sincrotrone Trieste

# Thank you!

TANGO Connecting things together

ESRF The European Synchrotron


**www.elettra.eu**

**www.esrf.eu**