SOLARIS
CENTRE

SOLARIS
CENTRE

Summary of the work and tools done at SOLARIS synchrotron for managing MySQL Tango archiving database

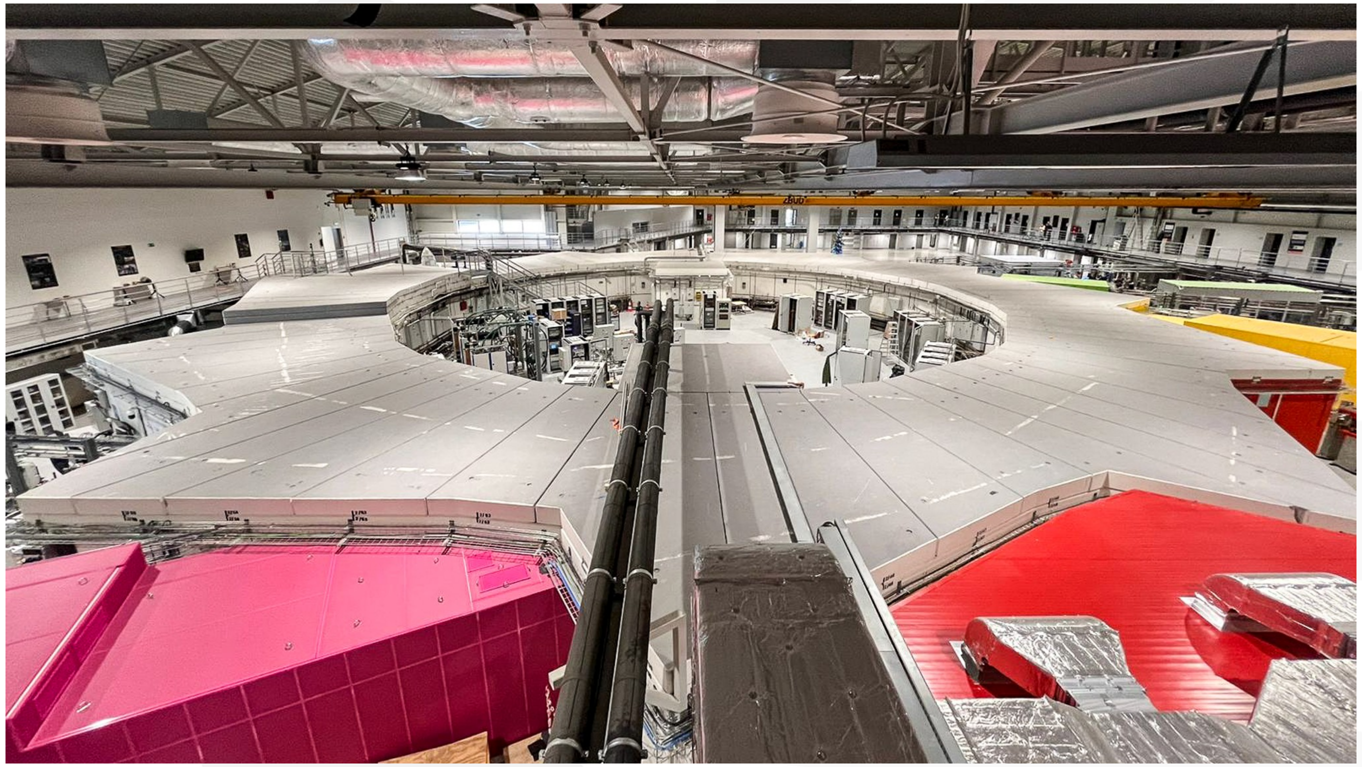
Giulianova, 39th Tango Community Meeting 21-23.05.2025

Krzysztof Madura, SOLARIS, CS-IT Section

SOLARIS National Synchrotron Radiation Centre

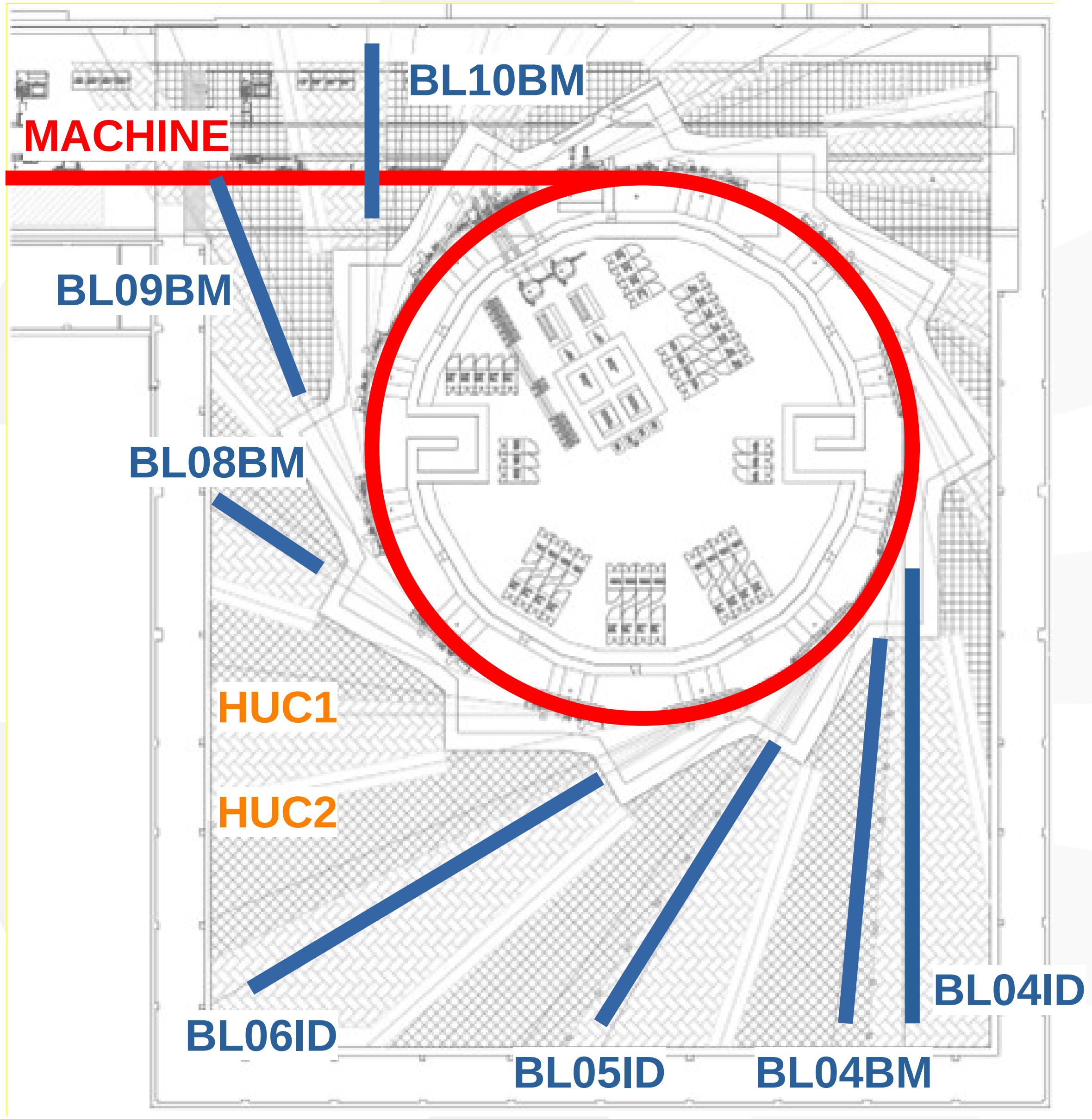
Specifics of Solaris Synchrotron

- What is a Synchrotron
- Linear accelerator and a storage ring
- Experimental end stations
- Other scientific equipment in Solaris (Cryo-EM)



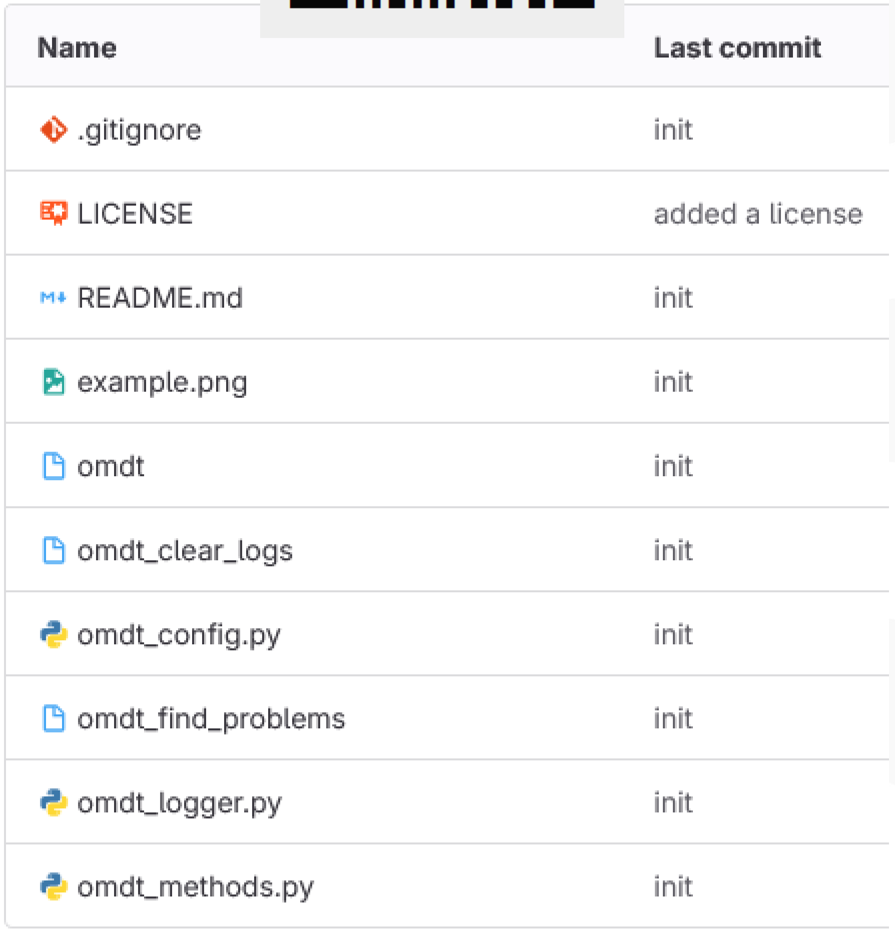
Tango Archiving in Solaris, how it is managed?

- Multiple MySQL databases for different components of the synchrotron, hosted on individual virtual machines
- Beam-line attributes are stored independently in their separate archiving databases
- Few devices such as HUC's have their own internal MySQL databases, SQLite back-end could be useful here
- Code names are used in domain names as opposed to official beam-line names
- Archive event periods are not lower than one second, usually every 5-60 seconds or more with additional absolute change event
- Cryo-electron microscopes are not part of the tango infrastructure, no need for archiving
- Possible migration towards TimescaleDB
- Current amount of data is at manageable level



OMDT – Old Machine Data Transfer

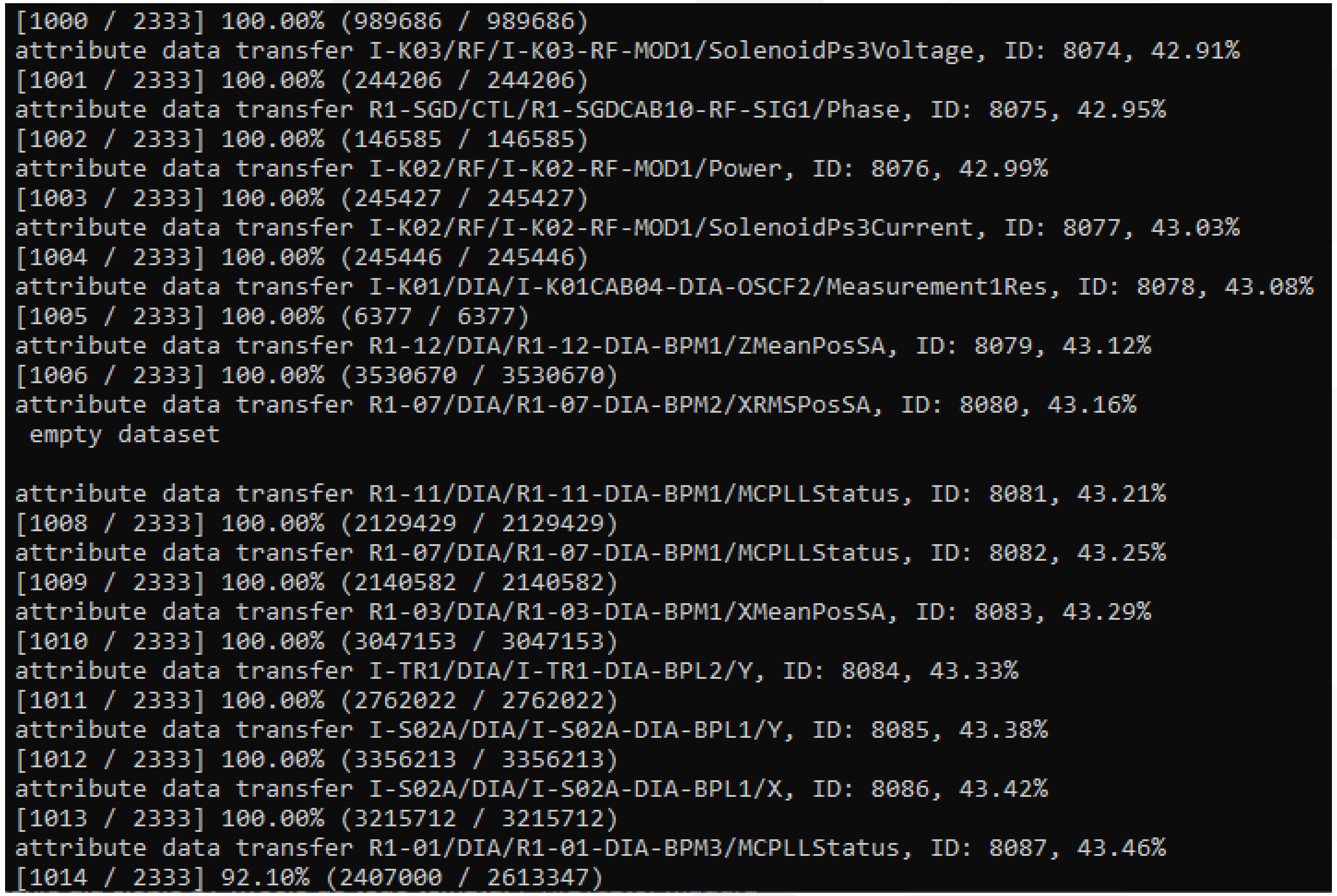
- A tool for transferring data from old MySQL HDB database to new HDB++ database
- Modify database connection parameters in omdt_config.py before running this program – the program will detect lack of change or it will fail to connect to the databases
- Accepts two optional arguments, -sf <number> to resume copying process from specific location and -tm for testing and debugging purposes, where it only copies 1000 rows per attribute
- Three phases:
  - Copy attribute data to att_conf table
  - Copy attribute settings to att_parameter table
  - Copy all data rows of an attribute
- There are no duplicates, when script is running for the second time, first and second phase will only update internal lists and variables of the tool
- Every time a program is run it creates a log file
- Log file is automatically deleted if it doesn't contain any errors
- omdt_clear_logs is used to clean logs
- omdt_find_problems can locate some issues within old HDB database



Name	Last commit
.gitignore	init
LICENSE	added a license
README.md	init
example.png	init
omdt	init
omdt_clear_logs	init
omdt_config.py	init
omdt_find_problems	init
omdt_logger.py	init
omdt_methods.py	init

OMDT – Old Machine Data Transfer

- During copying process each attribute takes two lines in the console
- First line contains location, id in HDB database, and current progress in %
- Second line shows current position (important when restarting with -sf option), progress in % and progress as number of copied rows
- At the end the program will print out performance statistics
- Maybe not extremely efficient considering how often it will be used (only once)
- Advice to use screen or other terminal multiplexer in order to keep it running when terminal is closed (ssh connection)

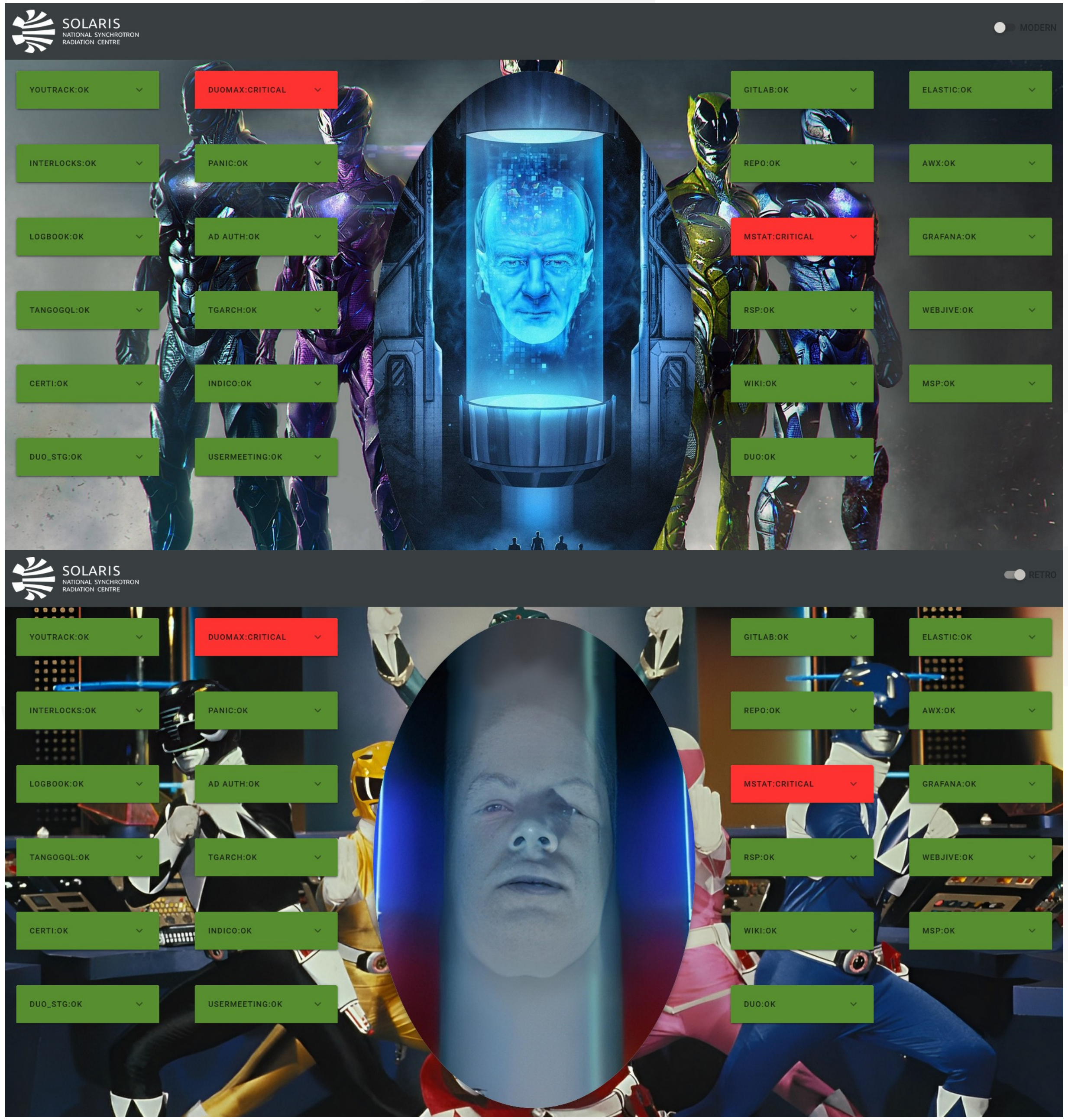


```
[1000 / 2333] 100.00% (989686 / 989686)
attribute data transfer I-K03/RF/I-K03-RF-MOD1/SolenoidPs3Voltage, ID: 8074, 42.91%
[1001 / 2333] 100.00% (244206 / 244206)
attribute data transfer R1-SGD/CTL/R1-SGDCAB10-RF-SIG1/Phase, ID: 8075, 42.95%
[1002 / 2333] 100.00% (146585 / 146585)
attribute data transfer I-K02/RF/I-K02-RF-MOD1/Power, ID: 8076, 42.99%
[1003 / 2333] 100.00% (245427 / 245427)
attribute data transfer I-K02/RF/I-K02-RF-MOD1/SolenoidPs3Current, ID: 8077, 43.03%
[1004 / 2333] 100.00% (245446 / 245446)
attribute data transfer I-K01/DIA/I-K01CAB04-DIA-OSCF2/Measurement1Res, ID: 8078, 43.08%
[1005 / 2333] 100.00% (6377 / 6377)
attribute data transfer R1-12/DIA/R1-12-DIA-BPM1/ZMeanPosSA, ID: 8079, 43.12%
[1006 / 2333] 100.00% (3530670 / 3530670)
attribute data transfer R1-07/DIA/R1-07-DIA-BPM2/XRMSPosSA, ID: 8080, 43.16%
 empty dataset

attribute data transfer R1-11/DIA/R1-11-DIA-BPM1/MCPLLStatus, ID: 8081, 43.21%
[1008 / 2333] 100.00% (2129429 / 2129429)
attribute data transfer R1-07/DIA/R1-07-DIA-BPM1/MCPLLStatus, ID: 8082, 43.25%
[1009 / 2333] 100.00% (2140582 / 2140582)
attribute data transfer R1-03/DIA/R1-03-DIA-BPM1/XMeanPosSA, ID: 8083, 43.29%
[1010 / 2333] 100.00% (3047153 / 3047153)
attribute data transfer I-TR1/DIA/I-TR1-DIA-BPL2/Y, ID: 8084, 43.33%
[1011 / 2333] 100.00% (2762022 / 2762022)
attribute data transfer I-S02A/DIA/I-S02A-DIA-BPL1/Y, ID: 8085, 43.38%
[1012 / 2333] 100.00% (3356213 / 3356213)
attribute data transfer I-S02A/DIA/I-S02A-DIA-BPL1/X, ID: 8086, 43.42%
[1013 / 2333] 100.00% (3215712 / 3215712)
attribute data transfer R1-01/DIA/R1-01-DIA-BPM3/MCPLLStatus, ID: 8087, 43.46%
[1014 / 2333] 92.10% (2407000 / 2613347)
```

Zordon – archtango module

- Zordon is Solaris internal tool for displaying current status of several systems on the screen, usually high resolution monitor in the CS-IT section room
- It is web-based, you can view it in your browser in solaris internal network
- Uses pytest
- Has a retro mode!

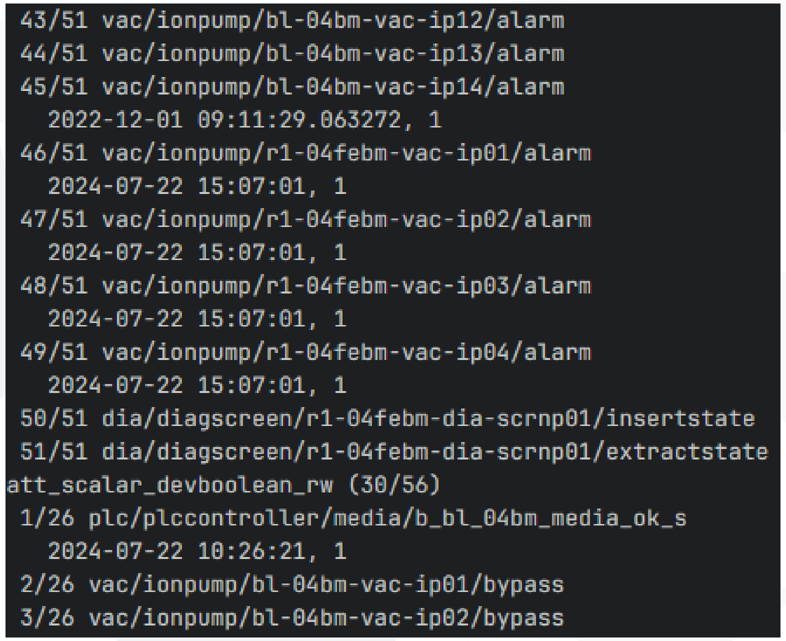


Zordon – archtango module

- Uses SQLite database file to store information about attributes that we want to check
- Not all attributes should be analysed (some are only used for testing or are temporary)
- First tests are used to check database connection and if lists of attributes are valid
- Pytest loops through attributes and checks if their archiving works properly
- There are two testing modes depending on how an attribute is stored in the database
  - For attributes with event periods it checks how many rows were stored during last 8 hours
    - Each such attribute described in SQLite database has minimum amount of rows set
    - Attributes with abnormal amount of rows are also detected
  - For attributes without event periods it fetches last row from archiving database and compares dates with most recent reading on a device, and if those dates match, we assume everything is working correctly
- Zordon provides only an insight on what may be wrong, doesn't explain exactly what is the issue
- A better monitoring tool made specifically for archiving could be a nicer option – zordon module acts more like a proof of concept

Other tools - 2024-07-29_value_error_null_detector

- The hdbpp-viewer tool in the version we are currently using has a bug preventing from loading any further data when it encounters a row with “value_r” and “att_error_desc_id” columns both containing a NULL value
- Haven’t tested if this error exists in newer versions of hdbpp-viewer
- When hdbpp-viewer encounters this row, if “value_r” contains NULL value, it assumes something went wrong and will fetch error message from att_error_desc table with id which is also NULL. An error dialog is shown, and any subsequent search will show no results making it look to our users as if there is no data
- In order to fix this temporarily on some attributes I made a simple tool which scans the entire database looking for any rows with such an issue, so I could manually remove them



```
 43/51 vac/ionpump/bl-04bm-vac-ip12/alarm
 44/51 vac/ionpump/bl-04bm-vac-ip13/alarm
 45/51 vac/ionpump/bl-04bm-vac-ip14/alarm
   2022-12-01 09:11:29.063272, 1
 46/51 vac/ionpump/r1-04febm-vac-ip01/alarm
   2024-07-22 15:07:01, 1
 47/51 vac/ionpump/r1-04febm-vac-ip02/alarm
   2024-07-22 15:07:01, 1
 48/51 vac/ionpump/r1-04febm-vac-ip03/alarm
   2024-07-22 15:07:01, 1
 49/51 vac/ionpump/r1-04febm-vac-ip04/alarm
   2024-07-22 15:07:01, 1
 50/51 dia/diagscreen/r1-04febm-dia-scrnp01/insertstate
 51/51 dia/diagscreen/r1-04febm-dia-scrnp01/extractstate
att_scalar_devboolean_rw (30/56)
 1/26 plc/plccontroller/media/b_bl_04bm_media_ok_s
   2024-07-22 10:26:21, 1
 2/26 vac/ionpump/bl-04bm-vac-ip01/bypass
 3/26 vac/ionpump/bl-04bm-vac-ip02/bypass
```

Other tools - 2024-08-28_status_cleaner

- A sudden increase in size of the att_scalar_devstring_ro table
- A bug in a device server was found. When physical device was shut down during summer maintenance, a device server tried to connect to it over and over again, each time executing an archive event for Status attribute. Status string contained “DevFailed” text concatenated with traceback, which was over 16MB in size.
- The “sizeCounter” script scans the entire table and categorises strings length into several groups: 0B, <=1KB, <=10KB, <=100KB, <=1MB, >1MB. This way we know how much data various attributes store in this table.
- The “locate” script looks for any rows starting with “DevFailed” with a massive text next to it. “drop” option in this script removes all such rows freeing disk space

Other tools - 2024-11-27_rf_dumper

- Our RF team asks for a quick and easy tool to dump data from several attributes in the database to CSV file
- This data is planned to be used for AI and Machine Learning purposes
- There are two scripts:
  - “rf_dumper_hdbpp_correlated” – it mimics “correlated” option as seen in hdbpp-viewer
  - “rf_dumper_full_dump” – downloads rows of data as it is, without manipulating them
- Both scripts accept three optional arguments:
  - Starting date – YYYY-MM-DD format, default value: midnight, over 7 days ago
  - Ending date – YYYY-MM-DD format, default value: previous midnight
  - File name – default: automatically generated CSV file name

Other tools - 2024-06-18_table_cloning_tool

- A table contained an abysmal amount of stored unnecessary images to be deleted
- Deletion process always failed at specific place and as it turned out, several rows were corrupted at some time in the past, but it had no performance effect on any later inserts
- We couldn’t execute REPAIR TABLE command due to space constraints, and IT department would had to find a way to reallocate more disk space to solve it
- All rows from the problematic attribute were manually removed except those very few broken rows, but as you probably know, disk usage will be still the same if we won’t repair this table
- To solve this problem I have made a tool which replicates table schema, creates a new table and transfers all the good rows between two tables. “cloningTool” script does just that, while “verifyTool” script ensures the data is all correct
- In the and we were left with a DROP TABLE and ALTER TABLE
- commands successfully replacing broken table with a new one,
- freeing over 1TB of disk space

Thank you for your attention!