Upgradation and Modernization of
The Tango based GMRT Control (TGC) System

Presenter – Jitendra Kodilkar

40th Tango Community Meeting at ALBA
June 8th , 2026

Operation Group, GMRT-NCRA members, India

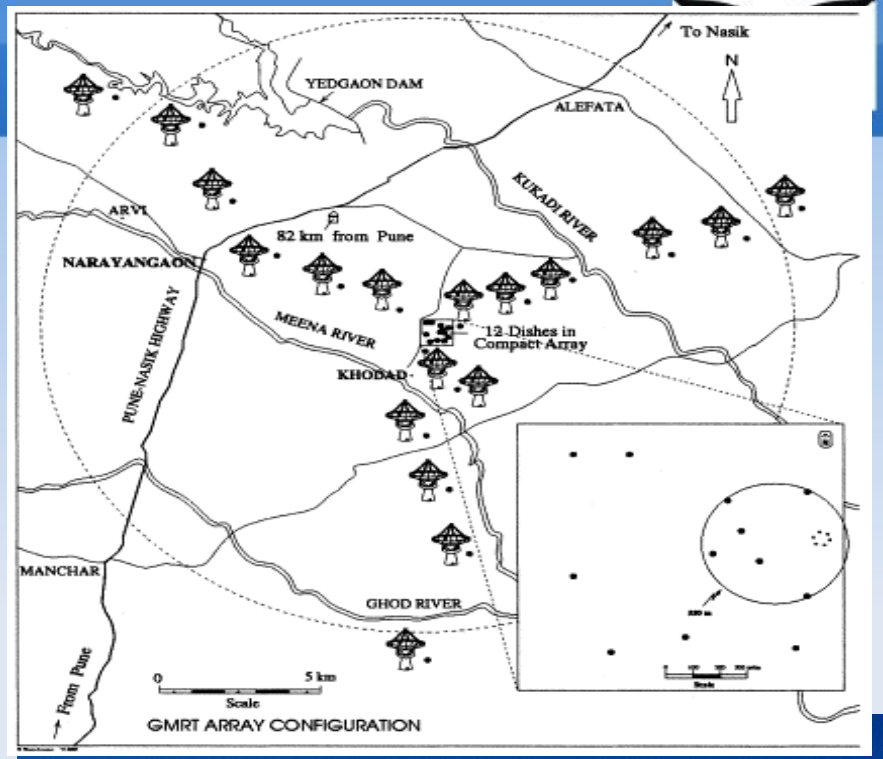
Jitendra Kodilkar, Raj Uprade, Bhavesh Kunbi, Mayur Bhagade, Isha Gothwad, Anil Raut, Ishwara Chandra, Yogesh Wadadekar, Yashwant Gupta

uGMRT (Giant Metrewave Radio Telescope)

- Located at Khodad (Lat 19.1◦ N , Long 74.05◦ E) Pune, India.
- Total 30 Parabolic Dishes each of 45 diameter.
- 14 Antennas in Central Sq. area of ~ 1 x 1 km
  Remaining spread over ~30 km diameter
  Shortest baseline ~ 100 meter.
- Observing Frequency bands are 150 to 1500 MHz with instantaneous bandwidth of 100-400 MHz.
- Support Interferometry and Beam forming modes.
- Accorded the status of SKA pathfinder.

Observations are scheduled twice in a year each of ~5.5 month duration.

More than factor of 2 oversubscribed, ~50% users are from outside India.



Outline of the talk

- Introduction : Tango Based GMRT Control (TGC) System
- Upgradation & Modernization Goals
- Feasibility Study
- Methodology
- Functional & Validation Testing
- Tango Client-Server : Aggregation Node Problem resolving
- Conclusion

Introduction : TGC System

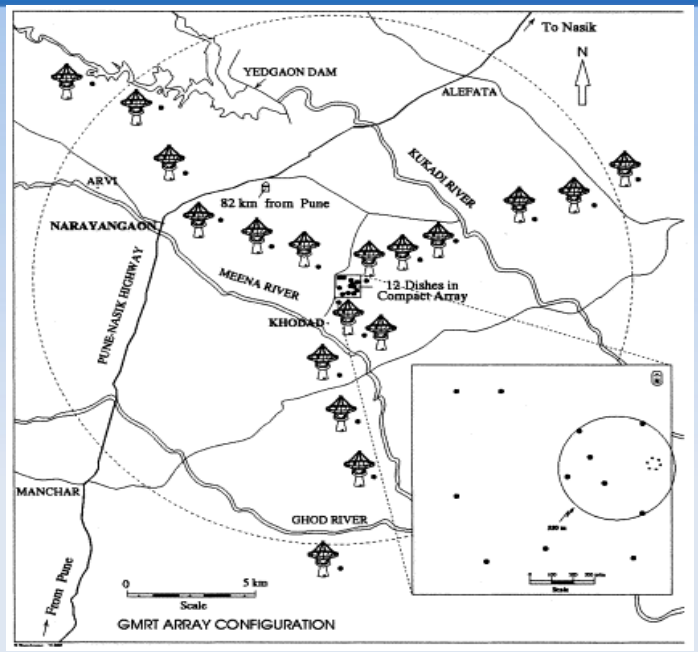
- 30 GMRT antenna comprises 30 x 5 = 150 subsystems :

  - Two servo systems : For antenna, and Feed rotation (FPS).
  - RF Receiver chain systems – FECB, OPT-FIBER, SENTINEL

At the GMRT Central Electronic Building ~ 40 Subsystems :

  - 30 GAB (GMRT Analog Backend) control Units, one per antenna
  - GMRT Wideband Backend : 1 correlator + 4 beam former(s)

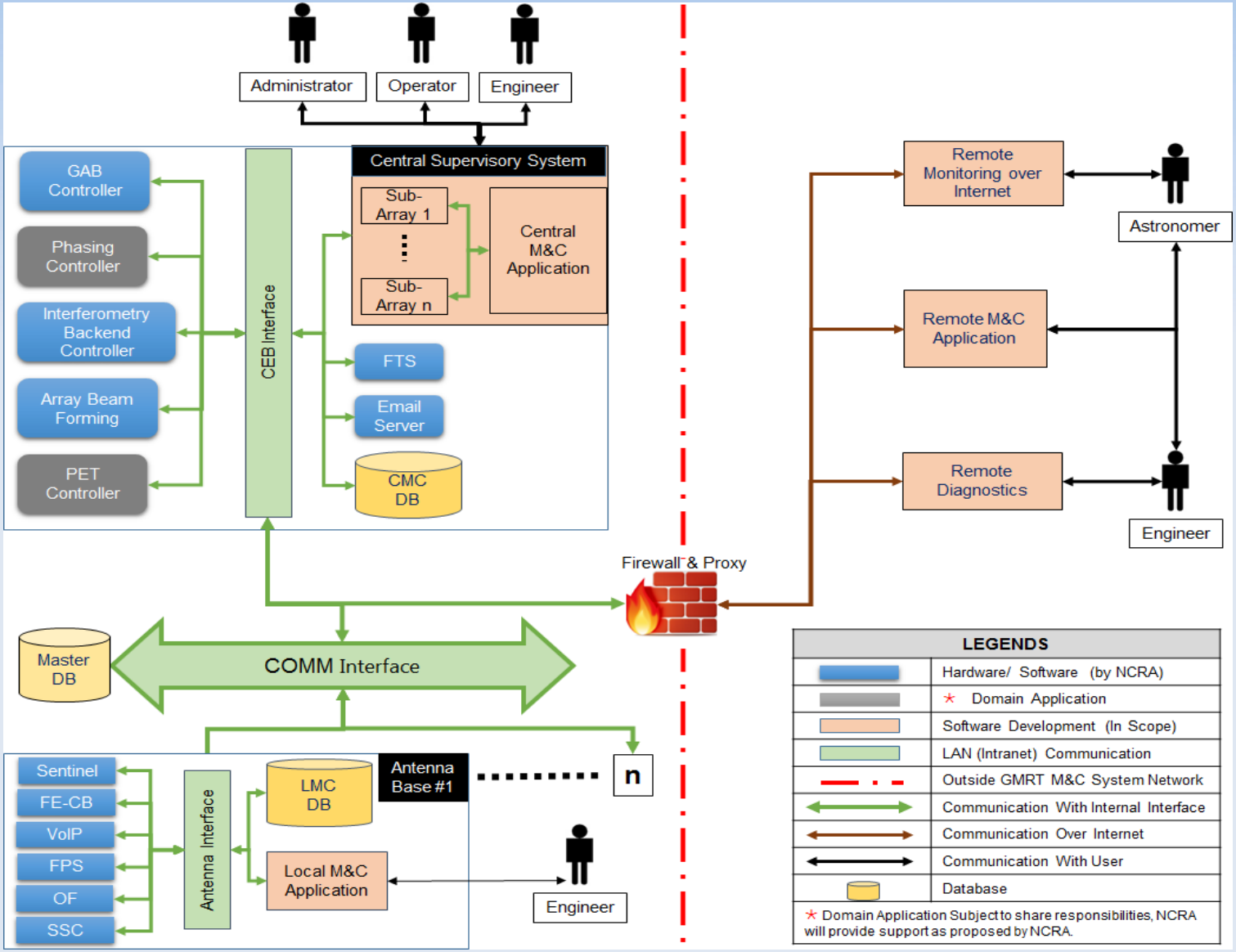
    (*Each Backend unit consist of separate data Acq. System*.)



- Total ~190 GMRT sub-systems are being monitor and Control by the TGC system in real time.

Link	Device Interface	Control Cards	Central and Local M&C machines	OS and Softwares	User Interface
Ethernet 1 Gbps	(i) Ethernet TCP/IP – 100 Mbps / 1 Gbps (i) USB/RS 485	(i) Rabbit 4300 (ii) Servo PC104	(i) Dell precision Tower 5810 (Application & Data server) (i) Dell Optiplex 7000 Series	Ubuntu 16.04 / 18.04 LTS, TANGO frame work, C++, Java and Python	GUI – Taurus, PyQt4, PyTango CLI – python Scripting Environment

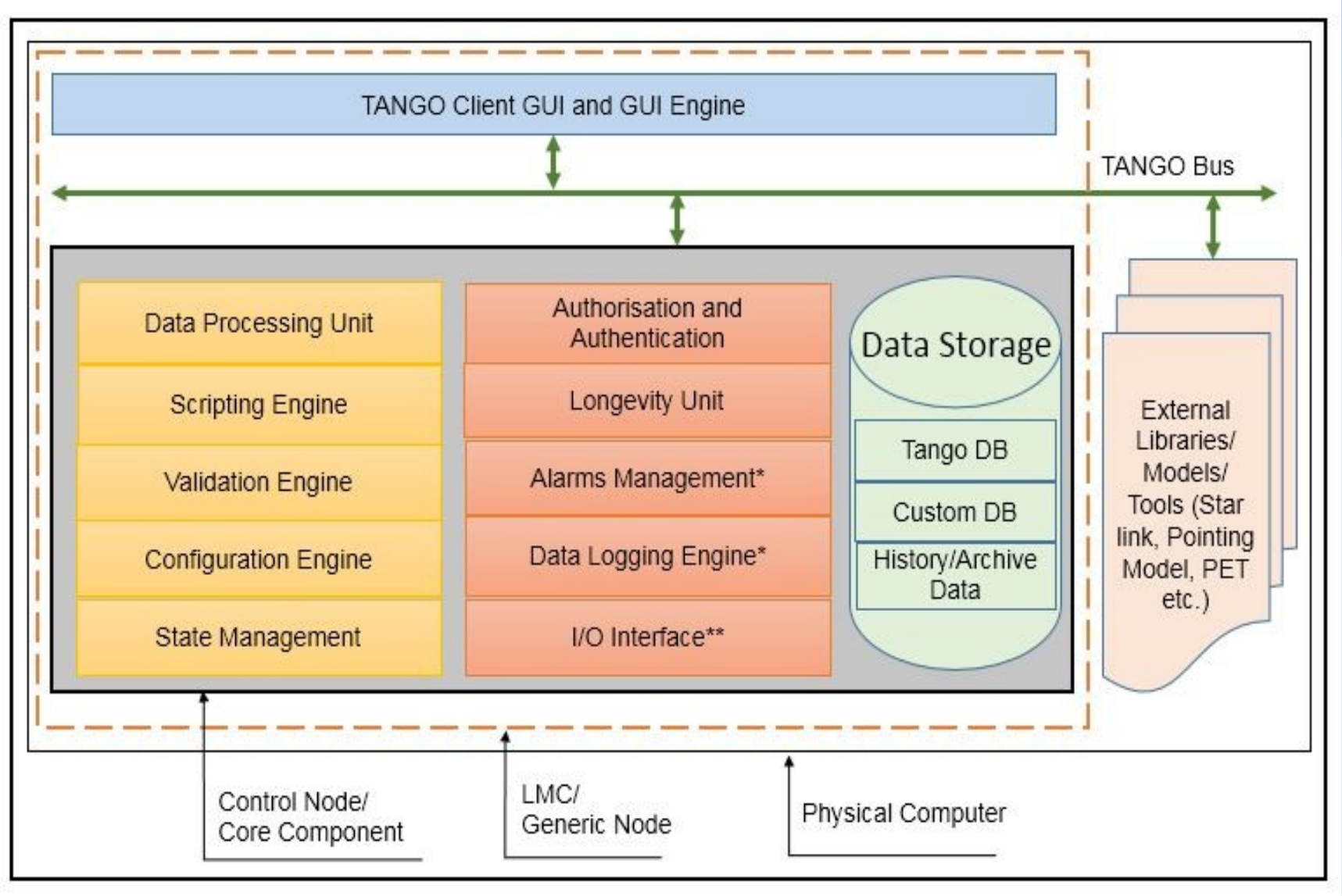
Next Generation GMRT M&C System
(Overview)



- **Role Based GUI with A&A**
- **Supports Five Observation arrays and one Maint. array**
- **Support Local M & C**
- **Automated Observing schedule Execution**
- **CLI with Python Scripting Environment**
- **Auto Alarm & Exception Handling**
- **M&C System data Archiving/Retrieving**
- **Web Interfaces for Absentee observing**

A Generic Control-Node Architecture

Quality Attributes : Modular, Configuration driven, Reusability, Modifiability
(*Component based Architecture - Segregation of Interface and Implementation*)



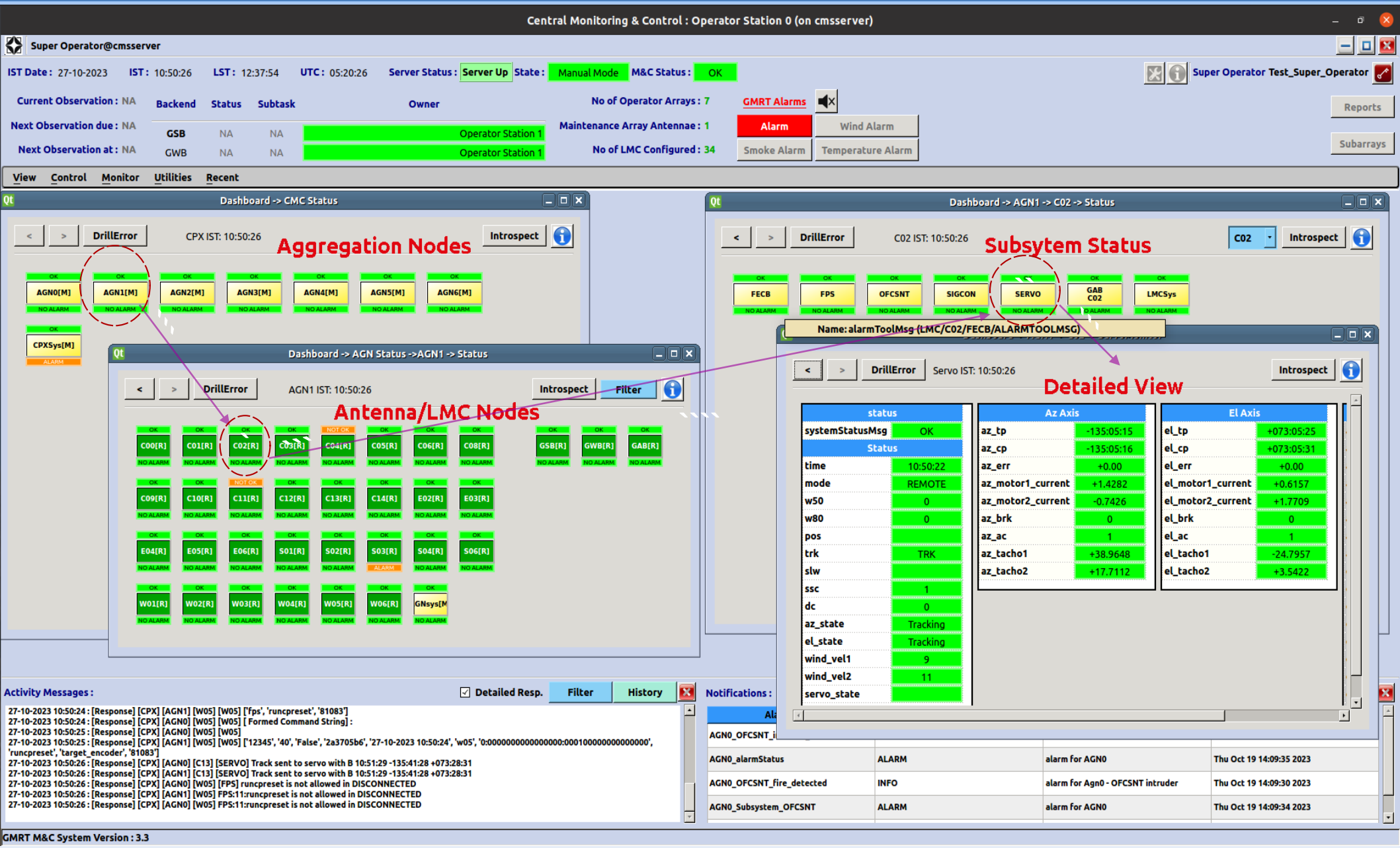
Tango based M&C System implementation

(Continued..)

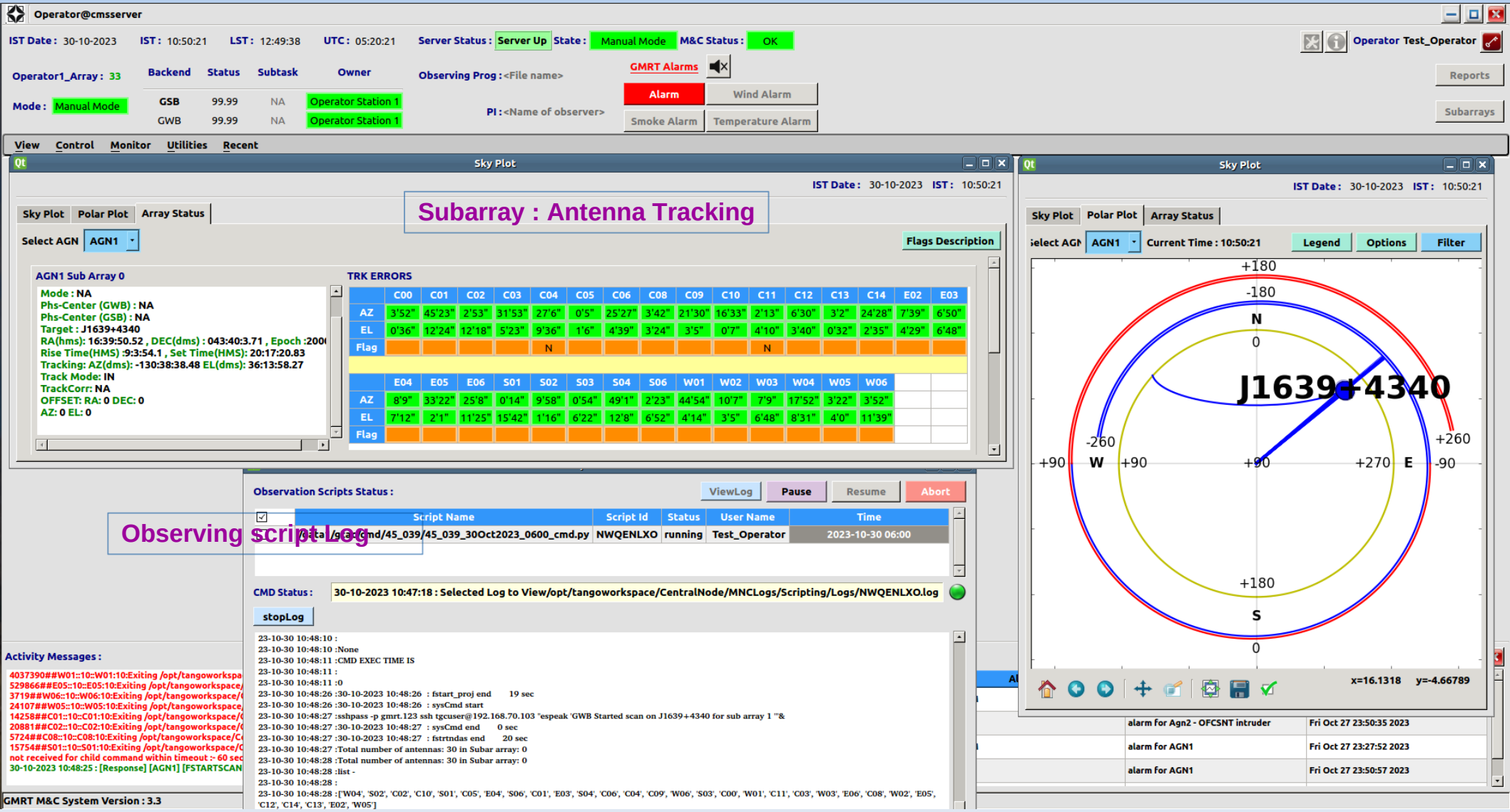
Tango Host	Number of Tango-Devices/ Servers per Machine	Devices	Attributes
Local M&C Machine	23	LMCs , FECB, FPS, OPFSNT, Servo, GAB, HDB++ (ES, CM) , PyAlarm, Scripting, and Starter	~450 to 500 (polling 1-3 sec)
Central Application server	40	AGN(0-6), CPX, HDB++ (ES, CM), Scripting , Starter, Pyalarm	~1200 (Polling 1-3 sec)

- M&C Comprises, at present at 33 LMCs
- Total DS = LMC DS ~759 + CMS DS 40 = 799
- Total Attributes : ~ 17000

TGC Features : Observability



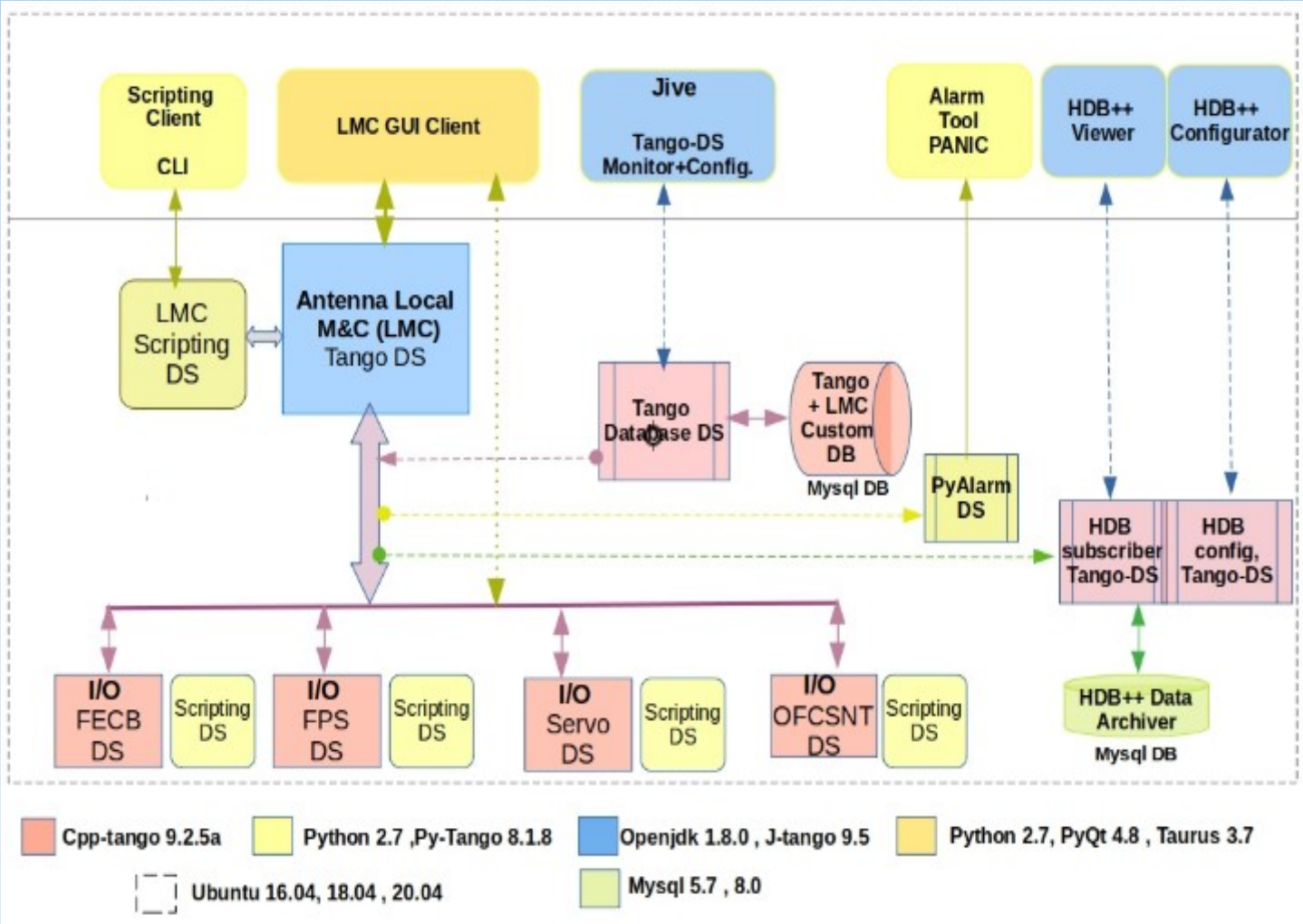
TGC Features : Usability, Functional suitability



Upgradation and Modernization Goals

- OS : Ubuntu 16.04 LTS
- Tango 9.2.5a (Release 2017)
- Python 2.7.18 support Ended at Jan, 2020
- PyQT4 reached EOL Aug, 2018
- Taurus 3.7 relied heavily on Python 3.7 (EOL Jun 2023)
- OpenJDK8 public updates ended in 2019 (*Officially ends in Nov 2026)*

TGC Components Comprising the LMC of GMRT Antenna



Current TGC platform is based on Tango 9.2.5a (2017) and legacy Python 2 / OpenJDK 8 technologies, creating maintainability, security, and long-term support challenges.

Upgradation and Modernization Goals

(Continue..)

Objective :

Support GMRT's long-term operational goal for conducting Science observations through reduced downtime, easier maintenance, and the future-ready architecture of the TGC system.

Ensure Compatibility , Long-term support, and Maintainability :

- TGC S/W shall be compliant with modern Linux environments.
- Migrate **Python 2 → Python 3, PyQt5/6, Taurus 5**
- Upgrade **OpenJDK 8 → Latest OpenJDK platform**
- Upgrade Java libraries: Hibernate ORM/JPA, Apache Networking, Log4j, Jython etc.

Tango Ecosystem Upgradation :

- Upgrade Tango 9.2.5a (Jan 2017) to the latest supported version.
- Update Jive, Astor, POGO, Archiving etc. toolkit components to the latest OpenJDK.
- Refactor TGC code to current Tango C++ / Python / Java standards.

Performance, Scalability & Security (Python-3 and Latest OpenJDK platform) :

Feasibility Study

Machine : Dell Optiplex SFF 7020 with Ubuntu 24.04 LTS OS

Tango 10.x Vs Tango 9.4.x Selection :

- **Tango 10.X :** Release with C++17, includes Stricter APIs for PyTango.
  Recommended for the New projects or Upgrade.
- **Tango 9.4 :** is more flexible with Non-Method based Function calls for Attribute Access.
  For Backward Compatibility with older style APIs, avoid breaking changes.
- **Hence, Tango 9.2.5a → Tango 9.4 Framework is selected.**

Python 2 to 3.9 Change Impact :

- **Python 3 → PyTango >= 9.4 → cppTango >= 9.4**
- **Python 3 → PyQT5** (PyQT4 Compatible upto Python 3.7, deprecated now)
- **Python 3 → PyQT5 → Taurus 5** (Taurus 4 is no longer supported since Jul 2021)

Java OpenJDK 1.8 to OpenJDK 17 :

- **OpenJDK17** Java platform is a balance of modern features and long-term support.
- Java 21 not selected as it offers virtual thread & thread concurrency which may be difficult to handle in the old code.
- **To avoid package dependency errors, we find OpenJDK 17 is a viable option.**

Feasibility Study (continue..)

The CONDA is identified as a most suitable Package-Manager and Environment Management Tool.

- ✔ Provides broad package availability for the Tango Control System.
- ✔ Simplifies installation by automatically managing dependencies.
- ✔ It enables reproducible development environments, and allows the same configured environment to be shared across multiple Tango-related projects and tools.

- ✗ To down-grade or up-grade packages, or Sub-package :
  Lot of complexities because of dependency verification tree, some time make it impossbile.

- ✗ **MySQL Library Integration problem :**
  - Installing the Tango via Conda also installs MySQL Libraries as a dependencies which conflicts with the Ubuntu's default MySQL libraries.
  - To resolve this, Ubuntu's MySQL Libraries were removed, and the CONDA provided MySQL libraries were configured as the system default by reconfiguring the *AppArmor* Security Module.

Feasibility Testing Performed :

- Created and tested multiple Conda Env. to resolve dependencies among already installed packages.
- Successfully validated Tango database, and device communication using the Jive and PyTango.
- Verified PyQt/Taurus functionality with simulated Tango devices for application migration and development.

Feasibility Study (continue..)

- **Using CONDA ver.24.11.3 (anaconda3) :**

◆ **Around ~90 S/W Tools, Libraries, amd Packaes were installed in the 'tangodefault' environment**

Basic (Compiler tools, libraries , MySql) – 37 , Tango (cpp, python, java, Qt) - 46 , Taurus- 9

TGC Application	Tango Library/Tools	New	Old
Common for all apps	MySQL	8.3.0	5.7
Sub-System I/O at antennas + TangoBaseDS +All tango tools/apps	Tango-Controls (cpp)	9.4.2	9.2.5a
	OmniORB (cpp)	4.2.5	4.2.1
	zeromq	4.3.5	4.0.0
Scripting Manager (DS) + CLI & GUI	Python, cpython	3.9.22	2.7.x
	PyTango	9.4.2	8.1
	Boost, boost-cp	1.78.0	1.58
Antenna Local M&C DS, HDB++ archive	OpenJDK	17.0.3	1.8
	Jtango, JtangoServer lib	9.7.4	9.5
GUI, PANIC Alarm	Qt	5.15.8	4.8.x
	PyQT	5.15.9	4.11.x
	Taurus	5.2.1	3.7

Methodology for the Upgradation

Python-2 to 3 Code Migration :

(TGC System code: MNCScriptManager DS, QT-GUI, Script Environment, and Astronomical Libraries)

2to3-3.9 -W -o /opt/tangoworkspace/ControlNode/Scripting *# -W keep backupfiles.*

Classification	Manual Code corrections in Python3 files
Code Formatting	• Indentation and Spaces were corrected to avoid logical execution errors, or misleading method execution. • **"from . import <module-name>"** → **import <module-name>** • For documenting, **'''** replaced with double quote **"""**
Data Type Errors	• **division()** return type casted to **int** • **filter() and map()** return type casted to **list** *# returns objects*
API Changes	• **sort()** → **sorted()** ; for dictionary **itermitems()** → **dict.items()** • Log4py APIs : **get_instance()** → **Logger()** ; I/O Stream buffer : sys.stdout.encoding = sys.getdefaultencoding() → **sys.stdout.reconfigure('utf8')**
Unicode 'utf-8' Data-types changes	• For **split(), find(), match()** methods byte code comparison is required, hence string prefixed with **b"...."** • **str(obj).rstrip('\n')** return type changed : byte-data to 'str'
Around 66 python files corrected manually.	

Methodology for the Upgradation

PyQT4 to PyQT5 Migration : *Total 37 PyQT5 and Tarus files corrected manually.*
(TGC System code: GUI , PANIC Alarm Tools)

pyqt4topyqt5 –nolog ./<pyqt4_code> -o ./<pyqt5_code>

(i) PyQT4 to PyQT5 Signal Handling :

The method associated with the GUI Object (*such as button*) can be called using the connect call**.**

```
# PyQT4 - for the 'button' widget , upon clicked() event, on_button_click() method is called.
self.connect(self.pushButton, SIGNAL("clicked()"), self.on_button_click)

# In PyQt5 new style of signal-slot connection is used
self.pushButton.clicked.connect(self.on_button_click)
```

- **Signal Connect : self.connect(obj, SIGNAL(...)) → obj.signal.connect(...)**
- **Signal Emit : self.emit(SIGNAL("sig"), args) → self.signal.emit(args)**

(ii) Phonon library removal in Qt5 : Multimedia framework for Audio/Video playback, replaced with **QMultiMedia /QmultiMediaWidgets** code

(iii) Widget SuperClass changed in PyQT5 : In converted code, many QTWidget Objects were missing their super/parent class. The GUI widget objects parent class found using the PyQt5 documents' web-sites or with the google-help.

(iv) Attribute Model in PyQT5 + Taurus 5.x : Required FQDN for the attributes.
viz. *attributeModel=* ***tango://<hostname>:****1000/domain/family/<attribute_name>*

Methodology for the Upgradation

Java OpenJDK 8 to OpenJDK 17 :

(TGC System Code : LMC DS , HDB++ Archive, Astronomical Libraries)

Using the ANT builder, the TGC code was compiled successfully, but run-time Bugs were occurred :

(i) MultipleBagFetch Exception Errors : Bug arised due to the Hibernate's confusion about whether duplicate records are suppose to be there in the DB , or it has become multiple due to the cartetian products when **@ManyToOne** / **@OneToMany** relations are used.

The entity relationship mappings in the **"DatabaseTx"** modules were updated by changing the fetch strategy from **FetchType.EAGER to FetchType.LAZY.**

e.g.

```
@OneToMany (mappedBy = "argIdInstanceFrmCmdAssocArgJpa", fetch = FetchType.LAZY)
  private List < CommandAssocArgumentJpa > cmdAssocArgObjFrmArgJpa;
```

(ii) 'ANTLR' (*ANother Tool for Language Recognition*) Upgrading from 2.7.x to 3.x :

BUG : java.lang.NoClassDefFoundError: antlr/collections/AST

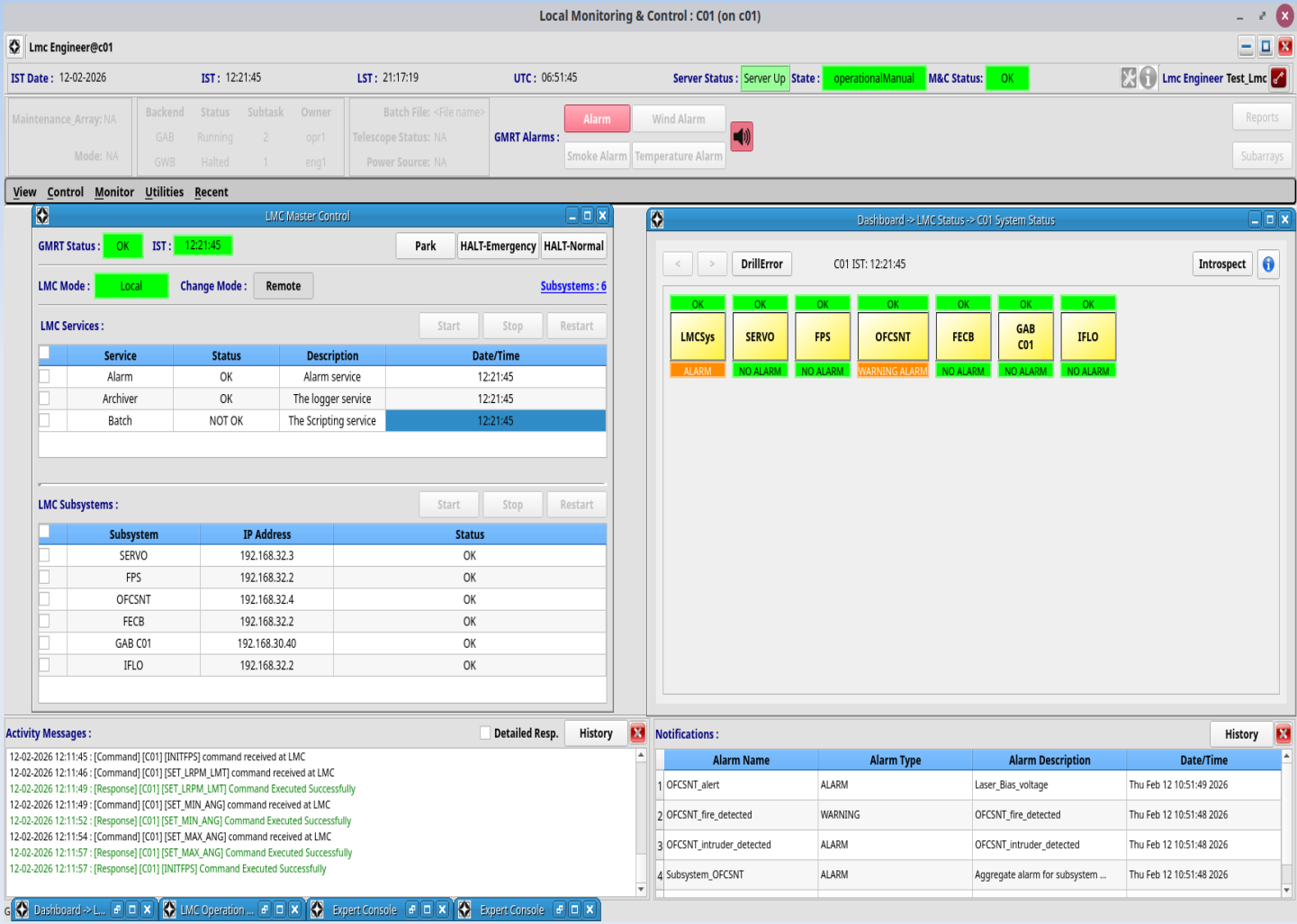
Jtango-9.7.4.jar consist the AST (Abstract Syntax Tree) class, but it was clashing with the Hibernate's and Maven-repository antlr-2.7.7.jar.

(iii) Database Exception Handling : Incomplete Data transactions is rollback() first before closing the Hibernates EntityManager instance i.e. this.em.close() This avoided stale connections.

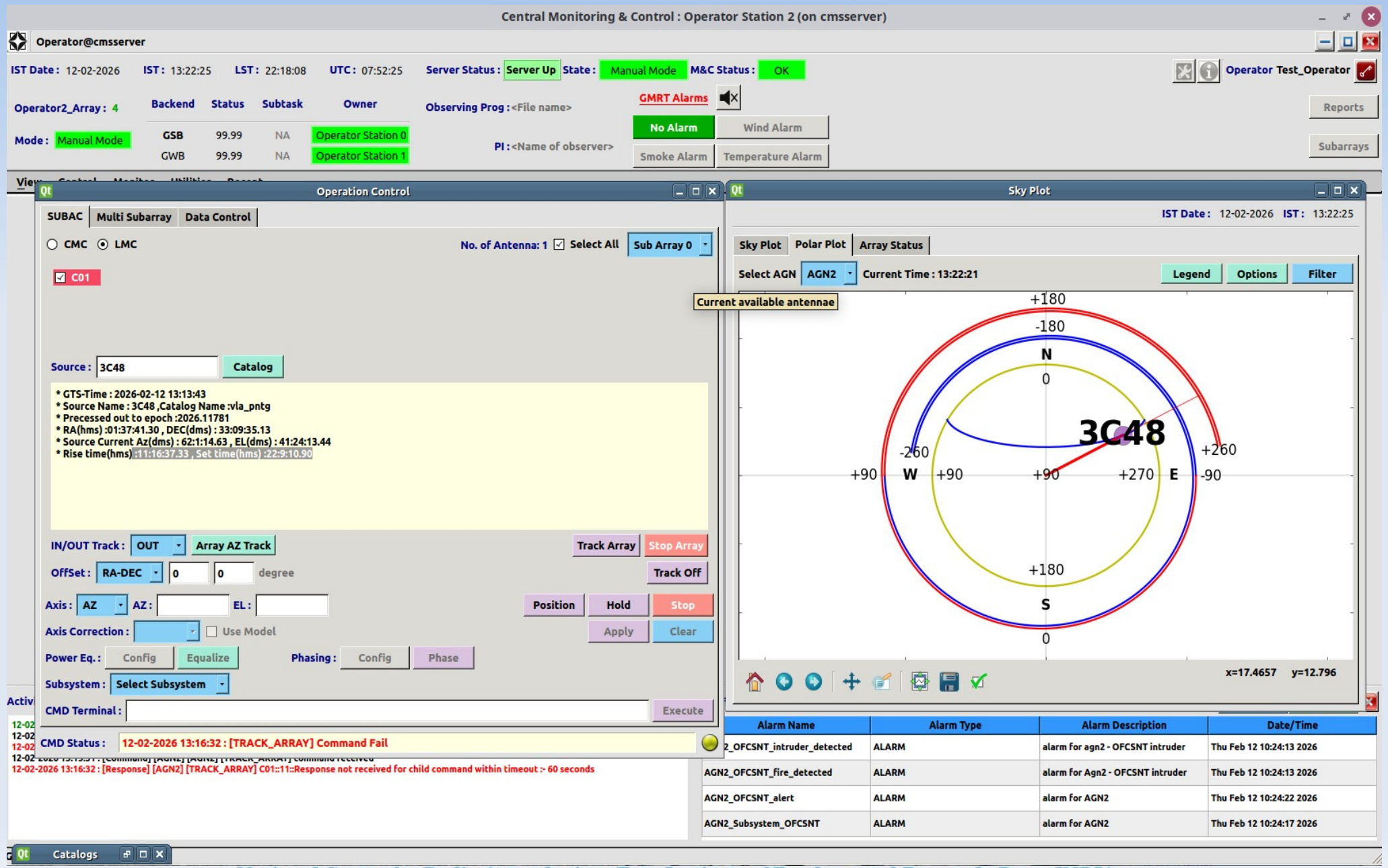
Functional & Validation Testing

Functional Testing done at the Lab, and at the "C01" GMRT Antenna

- **TGC-LMC Communicated with the antenna Control H/W.**
- **All subsystems (FPS, Servo, GAB, OFCSNT) control & Monitoring response checked.**
- **Antenna Tracking was verified with the correct Astronomical to Alt-Az coordinate conversion on 3C48 source.**
- **Receiver configuration to the desired observing band with expected power-level checked,**
- **Data Arching, Alarming validation testing done.**



Functional & Validation Testing



Tango Versions implementation at various facilities

#	Facility	CPP_TANGO	PyTANGO	JTANGO	OS
1	ALBA Synchrotron	7.2.x/8.1.x/9.2.x/9.3.3	7.2.x/8.1.x/9.2.x	7.2/8.1.x	Linux/Windows
2	DESY	**9.2.5**	9.3.2	9.1.x	Linux
3	ESRF Accelerators	**9.3.6-rc2**	9.3.6	9.6.6	Linux/Windows
4	MAX-IV Laboratory	**9.3.5**	9.3.3	9.3.x	Linux
5	NSRC Solaris	8.1.x/9.2.x/9.3.3	9.2.X	-	Linux/Windows
6	SKA Organisation	**9.3.4-rc**	**9.3.2**	-	**Linux**
7	SOLEIL	6.x/7.2.x/9.2.x	9.2.x	**9.5.x**	Linux/Windows
8	SKA	**9.3.3**	9.3.1	9.5.x	Linux
9	IJCLab	9.2.5	9.2.0	9.2.5	Linux
10	GMRT	**9.4.2**	**9.4.1**	**9.7.4**	**Linux Ubuntu 24.04.3**

Tango Client-Server :
Aggregation Node Problem resolving

AGN (Aggregation Node) :

- AGN is a Top-Level Controller managing antenna groups (**subarrays**) for Observations and experiments.

- Java-based Tango DS, but acts as a Tango-client for the remote LMC nodes at the GMRT antennas.

- **AGN hang observed every 2-3 days whenever there is LMC communication timeouts because of the network disconnections, Power supply failures, or LMC host PC issues.**

Tango Client-Server :
Aggregation Node Problem resolving

Solutions :

(1) Tango client–server communication simulation performed to investigate the root cause.
Result is documented on the Tango Community Forum.

set_transparency_reconnection(Transparency=<bool>) controls automatic reconnection.

- **Transparency = true (default) :**
  - ➔ Automatically retries to establish connection when the server is unavailable.
  - ➔ Client may hang for several minutes during reconnection attempts.

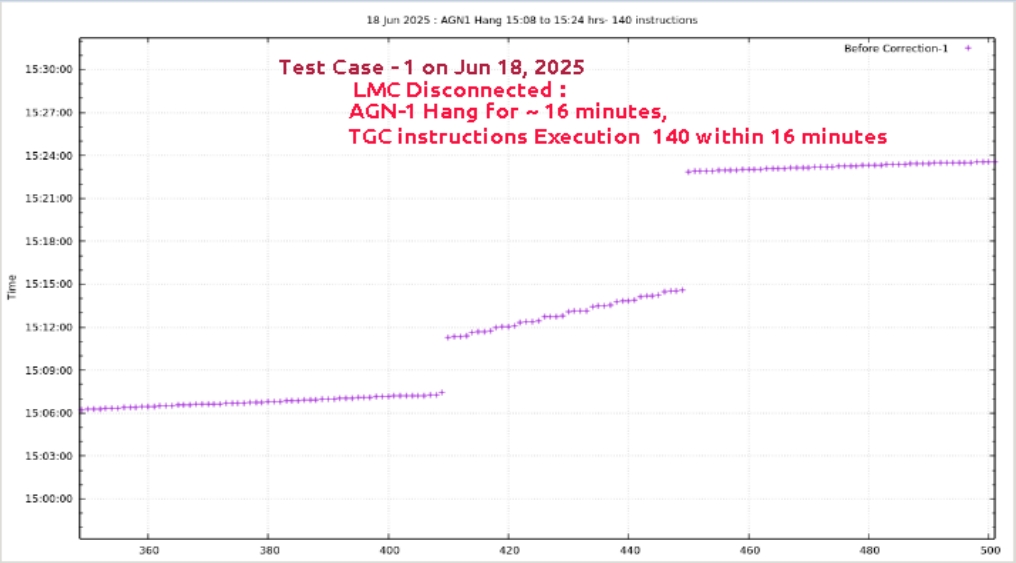
- **Transparency = false :**
  - ➔ Immediately throws a CommunicationFailed exception if the device is unreachable.
  - ➔ Prevents client hangs. Requires application-level exception handling.

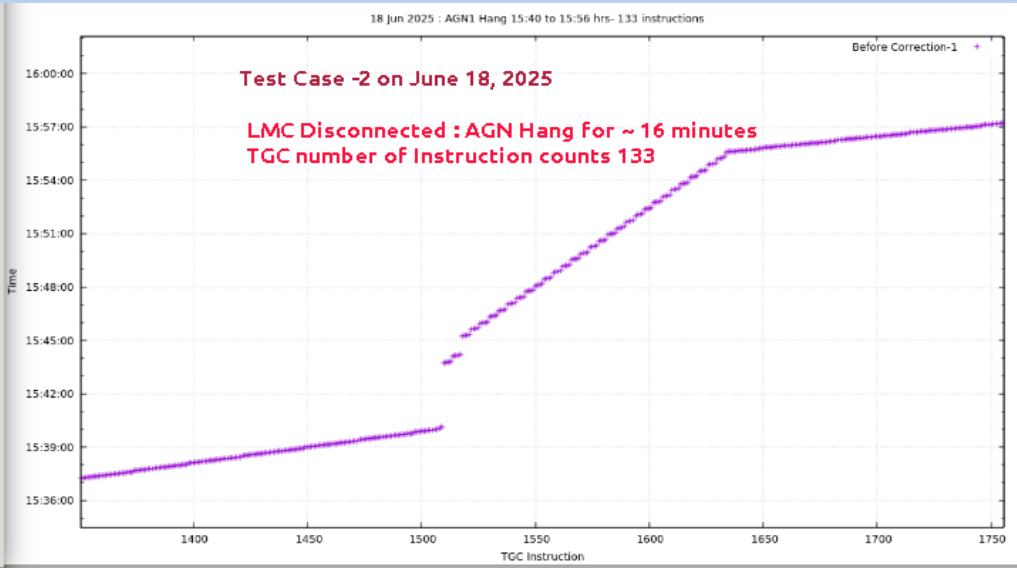
(2) MySQL Database Configuration tuned for Maximum Connections, InnoDB cache memory increases, connection timeout etc.

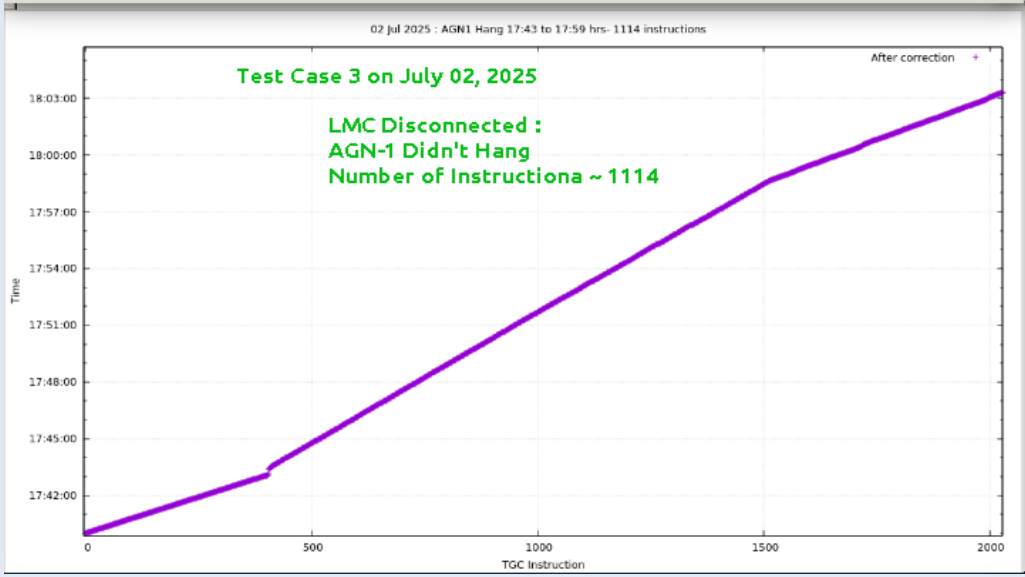
(3) In the DatabaseTx (Database Engine) Transaction management code, a careful exception handling taken care with a rolled back of pending transactions before closing the Hibernate EntityManager instance in the **Final loop of Try-Catch.**

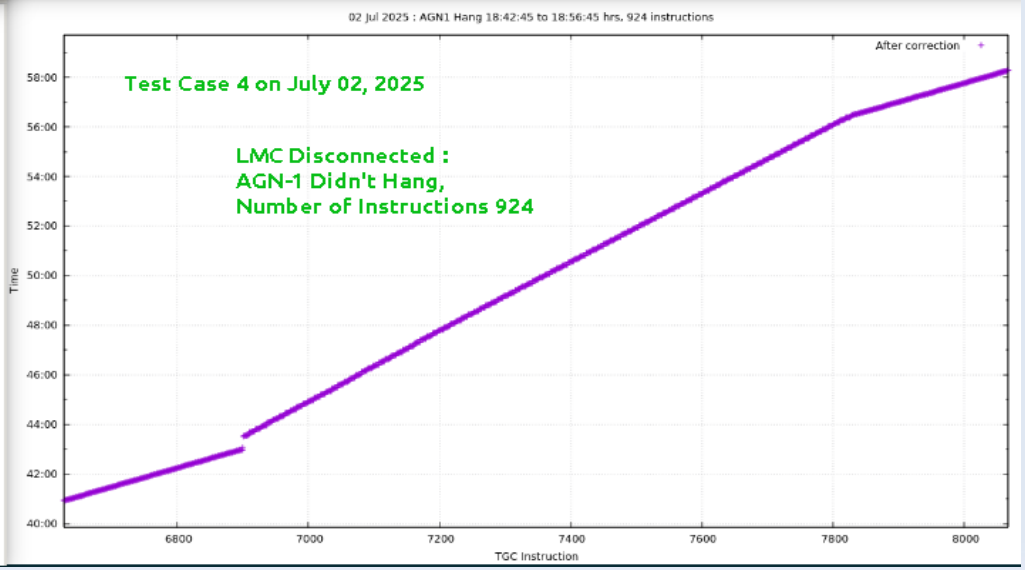
Tango Client-Server :
Aggregation Node Problem resolving (Continue..)

LMC Timeout Test case study without code correction (Jun 18, 2025) and With Code Correction (July 02, 2025)



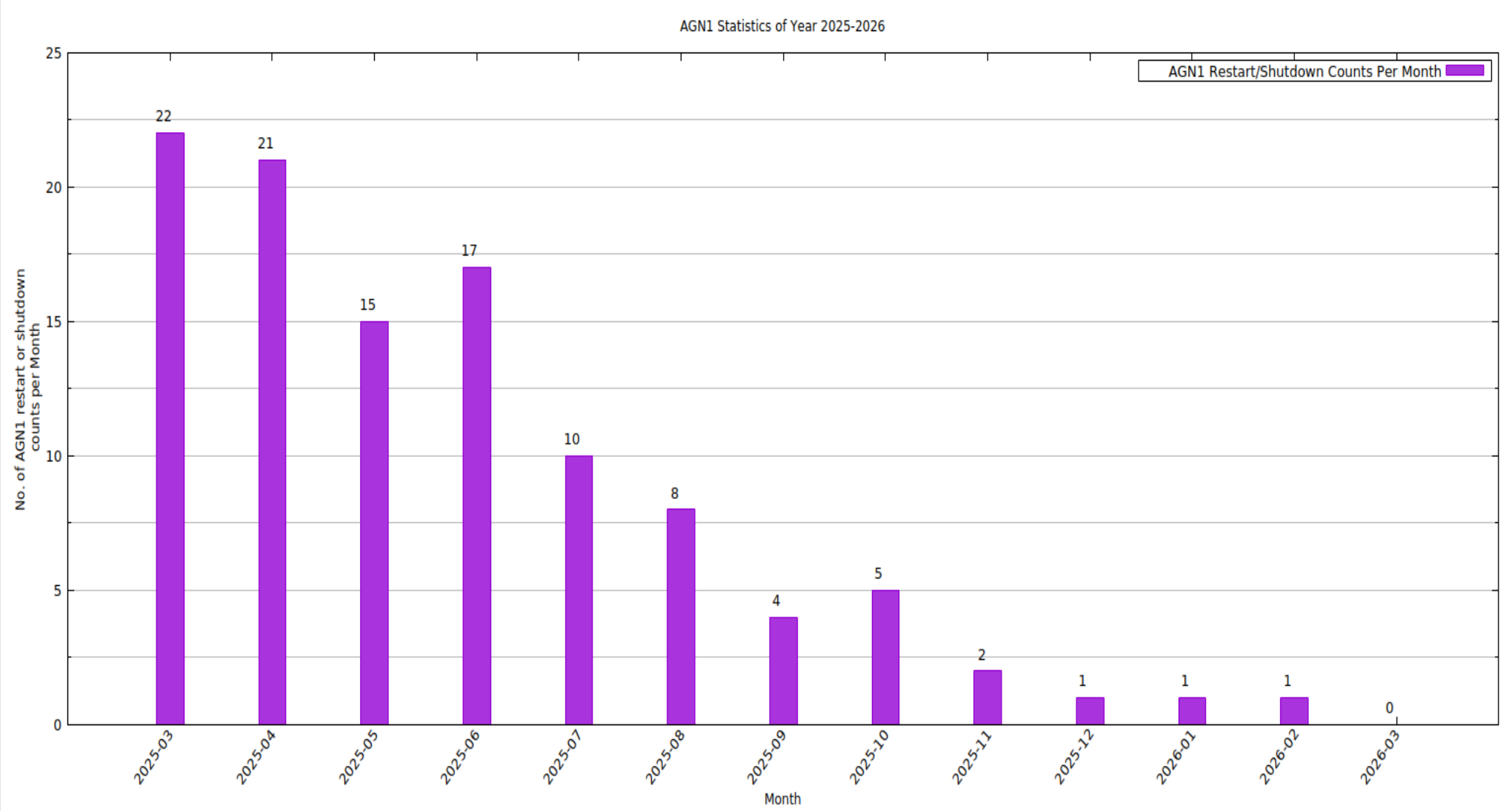






Tango Client-Server :
Aggregation Node Problem resolving (Continue..)

AGN Restart/Shutdown Counts per Month March 2025 to March 2026



Conclusion

- The TGC Local Monitoring and Control (LMC) software was successfully upgraded to the latest Linux distribution, Tango 9.4 framework, Python 3, and OpenJDK 17, ensuring long-term maintainability and compatibility.

- In the TGC system, Aggregation bug resolving increased MTBF from ~50 hours to ~372 hours, and thus reduced agn1-hang frequency from 15–21 times per month to 1–2 times per month.

Thanks...

References : [1] Development of Next Generation Monitor & Control System for Radio Telescopes, 2013 IOP Conf. Ser.: Mater. Sci. Eng. 44 012026

[2] GMRT M&C System Software Architecture Description Document, December 2014

[3] Report On Tango Control System Framework Evaluation, Version 1.3, October, 2015

[4] Tango Controls, http://www.tangocontrols.org

[5] https://tango-controls.readthedocs.io/projects/pytango/en/stable/versions/migration/to-9.4/deps-install.htm

[6] https://wiki.python.org/moin/PyQt4

[7] https://conf1.ncra.tifr.res.in/event/5/attachments/87/186/ngmnc_demo_nov15_2019-Jitendra.pdf

[8] https://www.geeksforgeeks.org/python-introduction-to-pyqt5/

[9] https://github.com/rferrazz/pyqt4topyqt5