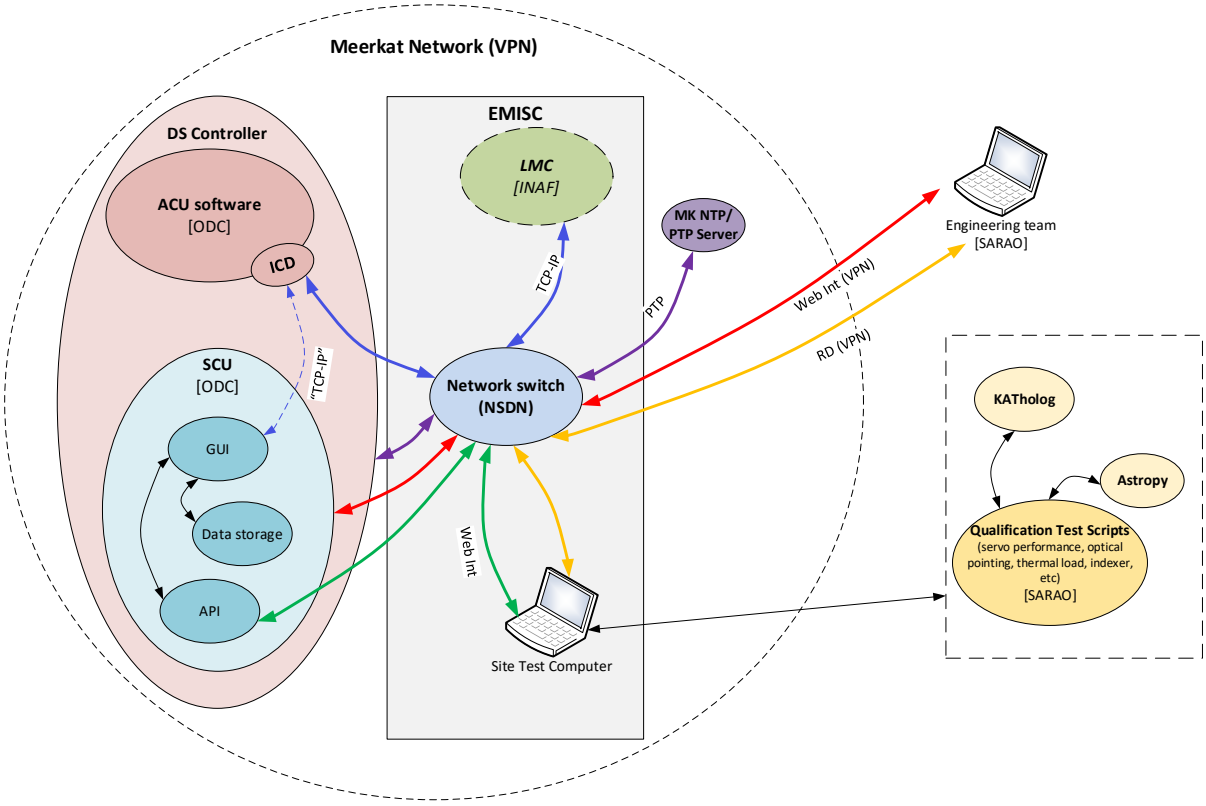
Requirements for the DS Test Software

The DS Test GUI is intended to be used by SKAO and SARAO Engineers to control and test the Dish Structure performance during DS Qualification.

The DS Test GUI typically runs on a machine (notebook/PC/LMC hardware) located inside the EMISC cabinet. It connects to the network switch (NSDN) inside the EMISC, and with the DS Controller also being connected to this network switch. The network switch is setup to provide remote access to the DS Test GUI. The DS Test GUI has the following main functions/features:

- Remote access from SARAO & SKAO networks
- Control the Dish Structure (DS)
- View detail status of data on the DS-LMC interface
- Record data
- Running “user defined” test scripts (python) OR provide an interface (API) to run and accept commands from the test scripts.

Typical configuration as used on the SKA-MPI Dish:



Refer to the appendix for more details on the actual Test Scripts as background information.

Examples of the SKA-MPI SCU GUI……..

Main functions/features of the DS Test GUI (software)

A. Remote access
   1. GUI accessible remotely from the MeerKAT & SKAO networks, via the network switch in the EMISC

B. Control the DS
   1. High level DS Control
      a. Take & Release command Authority
      b. Select combined DS State (Activate/Deactivate/Stop/~~Track~~/Slew2AbsPos)
      ~~c. Select individual axis State (Activate/Deactivate)~~
      d. Set combined Az/El position command (Position) and "GO"
      ~~e. Set individual axis (Az/El) position commands (Position&Rate) and "GO"~~
      f. Select Feed Indexer band positions (1/2/3/4/5a/5b/6)
      g. Select Stow and Unstow
      ~~h. Set Power State (Low/Normal)~~
      ~~i. Set Power Limit threshold value~~
      ~~j. Configure the Pointing Model(s) (On/Off/Update&StoreParameters parameters)~~
      ~~k. Configure the Tilt Sensor(s) correction (On/Off/FilterTC/Update&StoreStore parameters)~~
      ~~l. Configure the Thermal correction (On/Off/CorFactor/Update&Store parameters)~~

   2. Show the status of high-level DS Status Feedback (visual, simple graphics)
      a. Active Command Authority
      b. DS State (Locked/Standby/Stowed/Locked&Stowed/SIP/Stop/Slew/Track/Jog)
      c. Azimuth/Elevation/FeedIndexer actual State per axis (Locked/Standby/Stowed/Locked&Stowed/SIP/Stop/Slew/Track/Jog)
      d. Azimuth/Elevation/FeedIndexer position (set/actual/error)
      e. Azimuth/Elevation/FeedIndexer actual velocity
      f. Pointing Corrections applied (Pointing Model, Tilt sensor, Thermal)
      g. Safety (All limit switches, Emergency stop, Stow pin)
      h. Power status (PowerConsumption/PowerMode/PowerLimiting)
      i. Feed Indexer active Band (based on position status)
      j. DS time (ISO & TAI)
      k. High level Error / Warning status
      ~~l. Time-based, real-time trace of selected data (optional)~~

~~C. View detail status~~
   1. ~~Display the status of all commands to, and status from the DS on the DS/LMC interface.~~ (We can use the standard OPC UA capability for this)

D. Record data
   1. ~~Configure the data (parameters/sampling rate) to be recorded (config/setup files)~~ Could be done manually – no GUI
   2. Start/Stop/Abort the data recording

3. Recording “busy” status
4. Set the file name etc for storage. Preferably csv format. Could be done manually by renaming an automated timestamped datafile name
5. Provide storage space to allow for sampling rates of up to 10Hz for a duration of up to 12 hours of all data on the OPC-UA interface (location & size TBD)
6. ~~Capability to copy/delete/move data files remotely to a remote PC~~
7. Background revolving buffer OPC UA (PLC) does have some functionality – to be investigated further
    a. typical buffer of last 5 minutes at maximum sampling rate
    b. can be saved to a file upon command
    c. automatically start when DS GUI is running (TBC)

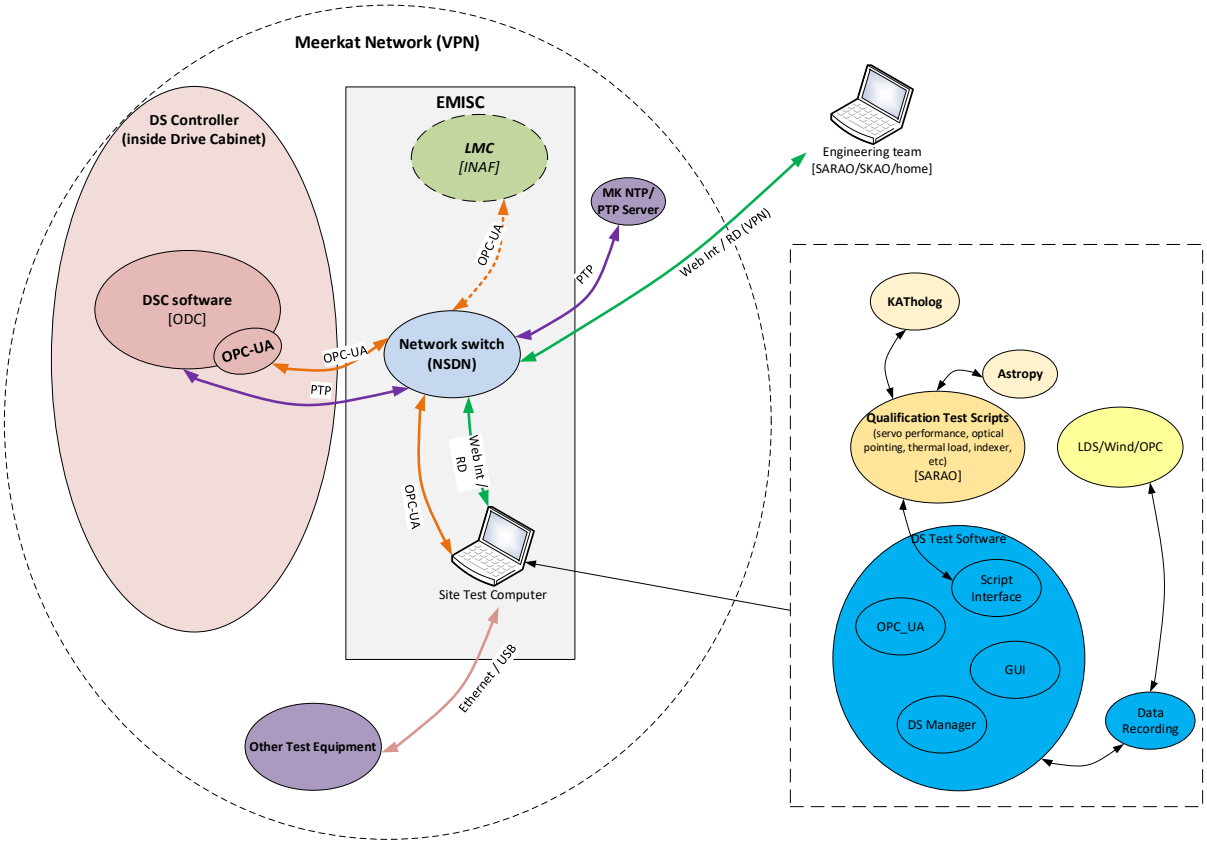
E. Running “user defined” test scripts
1. ~~Possible to upload “user defined” test scripts and associated input files (csv, VOTable XML)~~
2. ~~Select/start/stop a test script (file) to be executed from the GUI vs providing an API to allow commands from the Test Scripts (simplest option to be discussed)~~
3. Accept commands from the GUI in parallel with the test script (such as Activate/De-activate/stow) – mostly for safety intervention purposes
4. The “user defined” test scripts are python based (Jupyter Notebook)
5. Examples of typical test scripts can be found here.
6. Exact interaction (functions) between the “user defined” test script and the DS Test GUI to be defined
7. Implement “background” functions/handshaking to handle the lower level interaction with the DS Controller to execute the “user defined” test scripts – typically handling the timeously populating/appending of the track tables.

F. Provide functionality to access the WWW and MeerKAT network

Other options to discuss / investigate:

1. Read, timestamp & store data from a local weather station (To be discussed)

SKA AA0.5: (wip)


Meerkat Network (VPN)
EMISC
DS Controller
(inside Drive Cabinet)
LMC
[INAF]
MK NTP/
PTP Server
DSC software
[ODC]
OPC-UA
Network switch
(NSDN)
OPC-UA
PTP
OPC-UA
PTP
Web Int / RD (VPN)
Web Int / RD
OPC-UA
Site Test Computer
Ethernet / USB
Other Test Equipment
Engineering team
[SARAO/SKAO/home]
KATholog
Astropy
Qualification Test Scripts
(servo performance, optical pointing, thermal load, indexer, etc)
[SARAO]
LDS/Wind/OPC
DS Test Software
Script Interface
OPC_UA
GUI
DS Manager
Data Recording

https://jira.skatelescope.org/browse/SP-3432

Appendix: Typical architecture / functional diagram of a user defined Test Script