REMOT project

Astronomical Telescope Remote Operation Demonstration. System Design and Integration Report

M. Callegari, P. Marcucci, C. Vuerli

Osservatorio Astronomico di Trieste

Rapporto Tecnico N. 39/98

Pubblicazione O.A.T. N. 2026/98

Presentato nell'ambito del progetto UE REMOT (RE 1008) come documento D8.1.

D8.1: ASTRONOMICAL TELESCOPE REMOTE OPERATION DEMONSTRATION. SYSTEM DESIGN AND INTEGRATION REPORT

Massimo Callegari (e-Mail: *callegar@oat.ts.astro.it*, phone +39 - 40 - 3199232)
Paolo Marcucci (e-Mail: *marcucci@oat.ts.astro.it*, phone +39 - 40 - 3199214)
Claudio Vuerli (e-Mail: *vuerli@oat.ts.astro.it*, phone +39 - 40 - 3199213)

OAT: Osservatorio Astronomico di Trieste
Via G.B. Tiepolo 11
I - 34131 TRIESTE

1. INTRODUCTION

Within the context of this document, two items, related to the work of integrating the TNG software with the Teleoperation System one, will be discussed:

- The TNG simulation: has been designed in order to incarnate some of the various functionality related to the Telescope environment and has been used to evaluate, from the Telescope Users' point of view, the behaviour of the communication tools provided by the Teleoperation System;
- The TNG integration: has to be interpreted as the study and estimation of the results coming out from the work of "adding" the Teleoperation System to the TNG software, with the aim of providing the possibility of performing remote observations by introducing a more "reasonable" way to gain access to the network over which the data is dispatched (i.e. more efficiently, and without having to take care about any problems regarding the communication channels and the network traffic, which have to be optimized by the Teleoperation System).

The next sections will analyze these two arguments more deeply.

The whole structure of the Integrated System from the OAT's point of view, in the context of the REMOT project, can be summarized with the information related to Figure 1.

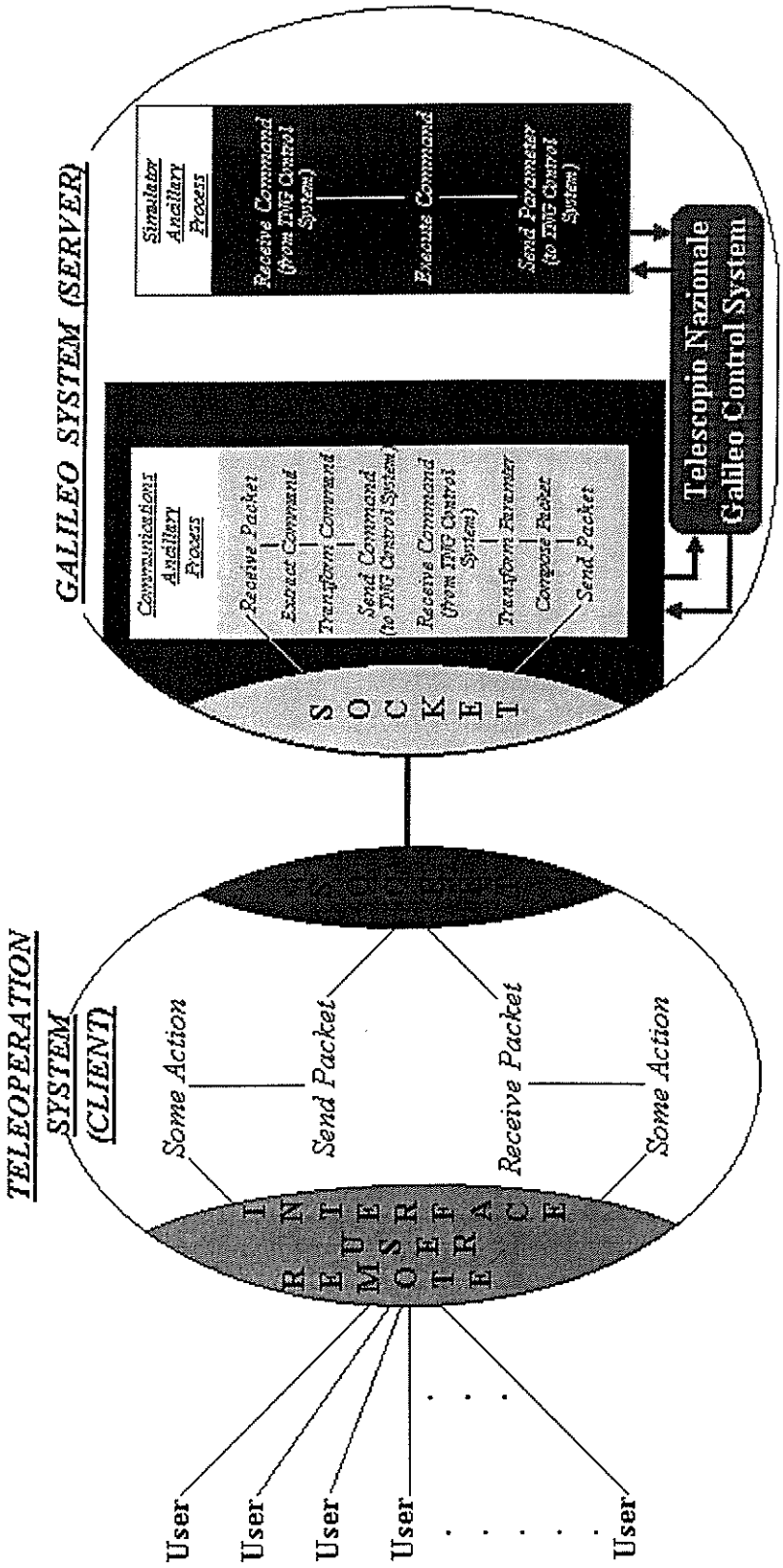

Figure 1: Teleoperation System Integration with the TNG Control System Software

The whole Experimental System can be seen as (logically) split into three main parts (considering the Telescope environment as running at the Local site):

- The *(Remote) Client*, which is made up of:
  - the *REMOT Remote User Interface* (that in the context of this project is seen as a static and integrated part of the Teleoperation System);
  - the *Teleoperation System*, running at the remote site;
  - the *REMOT Remote Communications* part (also seen as an integrated part of the Teleoperation System).
- The *(Local) Server*[1], which comprises:
  - the *REMOT Local Communications* part (the Communications Ancillary Process), used to handle the exchange of data from the Teleoperation System to the TNG environment and vice-versa; among its responsibilities it is worth stressing the following:
    - it handles the sending and receiving of information between the REMOT's remote client (connected to the Remote User Interface of the Teleoperation System) and the REMOT local server (the Communications Process itself), by providing a software interface between the Teleoperation System and the TNG Control System via a socket connection;
    - it defines a mapping and provides a translation between the commands/parameters designed in the context of the REMOT project and the corresponding ones related to the TNG environment;
    - it keeps track of the REMOT remote clients connected to the TNG System via the Teleoperation System and of the panels of the (static) Remote User Interface which are

[1] Openly speaking, the whole integrated Experimental System should be seen in a quite different way, particularly with the Teleoperation System software included in the server part too. This, because of the fact that the former includes, as part of its functionality, the handling of the remote communications (which have thus to be considered transparent to everyone using this software) and is running at the site of the latter too. With this kind of view, all the remote exchange of data and commands can be seen as responsibility of the Teleoperation System, while the server part only receives local information (coming from the Teleoperation System part that is running at its site). But the final result of the integration work is better explained with the diagram of Figure 1 because, from the local server point of view, everything it receives has to be considered as a remote communication.

currently waiting for an answer from the TNG Control System;
- it handles the sending and receiving of information between the REMOT local server (the Communications Process part itself) and the TNG environment, using the communications facility provided by the TNG Control System.

- the *Simulator* part (the Simulator Ancillary Process), used to test the "connection" of the TNG environment with the Teleoperation System software, before really accessing the Telescope Control System Software, and to provide, via software emulation, some functionality not currently available at the Telescope site; among its responsibilities it is worth stressing the following:
  - it handles the sending and receiving of information between itself and the Communications Ancillary Process, using the communications facility provided by the TNG Control System;
  - it executes the REMOT commands in the TNG environment by simulating the behaviour of the real subsystems, prepares and provides the parameters to be sent to the REMOT's remote client.
- the *Telescope Environment* part (the TNG Control System), which has been integrated with the two parts mentioned above.

- The *Communications Channel*, which is directly handled by an appropriate functionality of the Teleoperation System.

2. TNG SIMULATION DESCRIPTION

As far as we are concerned with the problem of simulating some of the functionality of a Telescope, a few points have to be stressed:

- The accuracy of the simulator can be moved from a simple emulator of actions, to be performed according to commands sent to the Telescope environment, to a complex process or

group of processes, each one designed in order to simulate the behaviour of the various instruments;

- The interconnection of the software implementing the simulator with the software used to control the Telescope can range from the simulator running alone, (thus completely substituting the Telescope Control System Software) to the simulator being integrated and running into the Telescope environment (thus behaving as a part of the whole software for the Telescope);
- The communications with the "rest of the world" (the remote access to the Telescope and the responses to the remote requests) can be committed to the simulator itself (if it runs "alone") or to some particular process of the Telescope environment, specially designed to handle the transfer of data between a local server (the simulator plus the Telescope software) and a remote client.

Having in mind these features and other characteristics, the simulator process of the Telescopio Nazionale Galileo, has been designed with the aim of reaching a good compromise between efficiency, reliability, likelihood and realism of the whole software running at the local site.

In order to reach these qualifications we have designed and implemented a simulator that takes into account the following:

- We needed to test the Quality of Service provided by the communications implemented with the Teleoperation System, before using it with the real Telescope, in order to avoid the damaging of the global TNG functionality;
- The various functionality of the Telescope, that are not currently available, have been emulated in a simple and straight way, having in mind the importance of the testing of the Teleoperation System behaviour in the largest context possible, thus involving the maximum number of commands available; as soon as we have some new real functionality available, we'll drop it out from the simulator and use it directly with the Telescope, accessing it via the Teleoperation System;
- At the Demo Level, (again, because of the fact that we are mainly interested in testing the quality of the communications), we consider sufficient, if no real instruments are available, the use of the simulator running in the TNG environment (so using the TNG functionality, but substituting the real instruments);

- The communications (between the Telescope Control System and the Teleoperation System) part has been developed apart from the simulation one, in order to have as much independent processes as possible running at the Telescope side[2];
- Because of the fact that the Demo part of the REMOT project has mainly the aim of testing the behaviour of the Teleoperation System in a simple real case, we have accepted the idea of having a static Remote User Interface, implemented as a part of the Teleoperation System client (following our specifications), instead of asking the use of a dynamical, auto-configuring (on the base of the one designed for the particular scientific application) Remote User Interface. This all has led to a particular design of the simulator, which has to maintain and handle some information related to those panels and commands of the Remote User Interface that a user is working on.

2.1 Actions of the Simulator corresponding to commands sent to the Telescope Environment.

In the following table we'll show how the TNG Simulator has been designed, by providing the general description of its actions when a particular command is sent to the TNG environment (and dispatched to the Simulator Ancillary Process).

COMMAND NAME	***COMMAND DESCRIPTION***
LOGIN	1) Receive information about the user name and the user type in order to allow the check of the login permissions. 2) Acquire the User Type from the operands of the command, in order to be able to control the particular session depending on the type of user connected to the system.
	3) Store information about the fact that the Dome Panel is open. 4) Send to the Remote User Interface a

[2] This is not only for the sake of simplicity, but even for modularity and interchangeability reasons: should it be the case that we need to remove the simulator software and/or substitute it with the software for a particular instrument, and still need to communicate with the Teleoperation System software, this can be done without "destroying" the communications part.

SHODOM	telemetry parameter, representing the current position of the doors of the dome.
SHOTLS	**5)** Store information about the fact that the Telescope Panel is open. **6)** Send to the Remote User Interface a telemetry parameter, representing the current position of the telescope (in right ascension and declination values), if the user is a Secondary one.
SHOCCD	**7)** Store information about the fact that the CCD Panel is open. **8)** Send to the Remote User Interface a parameter, representing the default filename that the Simulator environment is proposing for the image to be acquired. **9)** Send to the Remote User Interface (every 2 seconds) a telemetry parameter, representing the time that has elapsed from the start of the exposure.
LOGOUT	**10)** Inform the Local System (at the site where the Telescope is located) that the experimental session of the Remote User has been closed.
DOMOPN	**11)** Store information about the fact that the doors of the dome are currently being opened. No other actions can be made while this action is being performed. **12)** Change the position of the doors. **13)** Send a telemetry parameter, representing the actual position of the doors, to the Remote User Interface (every 2 seconds).
DOMCLO	**14)** Store information about the fact that the doors of the dome are currently being closed. No other actions can be made while this action is being performed. **15)** Change the position of the doors. **16)** Send a telemetry parameter, representing the actual position of the doors, to the Remote User Interface (every 2 seconds).
DOMBYE	**17)** Store information about the fact that

	the Dome Panel has been closed.
SESOPN	**18)** Store information about the fact that the Observing Session has been started (mirror covers of the Telescope have been removed, and initialising starting actions have been done). **19)** Send a telemetry parameter, representing the actual position of the Telescope, to the Remote User Interface in the Primary User case. **20)** Block the usage of the Dome Control Panel (by storing information about the current active status of the current session) to avoid improper use of the commands to be sent to the dome while the Telescope is being used.
SESCLO	**21)** Store information about the fact that the Observing Session has been closed (mirror covers o the Telescope have been put on, and the Telescope has been repositioned into its standard position) and prepare the Telescope Environment to a new observing session.
GOTOPS	**22)** Store information about the fact that the Telescope is currently moving. **23)** Change the position of the Telescope. **24)** Send a telemetry parameter, representing the actual position of the Telescope, to the Remote User Interface (every 2 seconds). **25)** Store information about the fact that the tracking facility is currently active. **26)** Send a telemetry parameter (when the Telescope has been pointed correctly), in order to inform the Remote User that the tracking facility has been activated.
STOPPT	**27)** Stop the action of pointing the Telescope. **28)** Store information about the fact that the Telescope is currently in a "stopped" position. **29)** Send a telemetry parameter, in order to inform the Remote User that the tracking facility has been deactivated.

NEWOBS	**30)** Send a telemetry parameter, in order to inform the Remote User that the tracking facility has been deactivated and a new observing session can be carried on. **31)** Store information about the fact that the tracking facility is not currently active.
TELBYE	**32)** Store information about the fact that the Dome Panel has been closed.
FILTER	**33)** Check the position required for the filters vs the range of values that the position itself can assume. **34)** If the required position is an acceptable one, mount the filter, else send an error message back to the Remote User Interface.
GOEXP	**35)** Store information about the fact that the exposing action is being carried on. **36)** Start the exposure action by incrementing the exposure time required to reach the Remote User Request. **37)** Send a telemetry parameter, in order to inform the Remote User about the elapsed exposition time (every 2 seconds).
ABEXP	**38)** Store information about the fact that the exposure has been aborted, by adding information of "stopped" exposure to one of its internal parameters.
SHOIMG	**39)** Send the name of the image currently being observed in order to let the users know its location if they need to save it. **40)** Send to the Remote User Interface the information needed to display the image.
CCDBYE	**41)** Store information about the fact that the CCD Panel has been closed.
SAVIMG	**42)** Send the name of the image currently being observed in order to let the users know its location if they need to save it. **43)** Send the image to be saved to the

	Remote User.
IMGBYE	**44)** Store information about the fact that the Graphic Panel has been closed.

3. INTEGRATING THE TELEOPERATION SYSTEM INTO THE TNG ENVIRONMENT

The work of integrating the Teleoperation System into the TNG environment has been based on the design of a particular communications process (the Communications Ancillary Process), which, allowing the connection of processes external to the TNG Control System Software, has provided the possibility of sending commands and receiving parameters to/from the Telescope without modifying dramatically the architecture of its software design.

In fact, being an Ancillary Process a particular module, that can be added to the TNG Control System Software with the aim of increasing its whole functionality, it has been used as a bridge between the Teleoperation System and the TNG Control System Software, although being part of the second one too.

The main idea used, while designing this process, has been to make it possible to let the Teleoperation System and the TNG Control System communicate via a socket connection. In this way, the only modification made on the TNG software has been the addition of the Communications Ancillary Process to its environment.

The way of working of this process is simply based on the forwarding of the data that it receives to the correct destination (the Teleoperation System or the Simulator of the Telescope), depending on the sender and on its requests:

- When the client sends a request to the Communications Ancilllary Process (in the format devised for the REMOT project and described in Appendix B of document D6.3), the latter one performs the following actions:

- extracts the command (written in the REMOT format) from the packet that it has received on the socket connected to the client;
- keeps track of the client that has invoked that request (because it has the capability of allowing multiple socket connections to different clients);
- transforms the command into a format suitable for its execution inside the TNG environment;
- sends the command to the process(es) appointed to its processing;
- waits for another request from a client process or for a reply from the TNG environment.

- When a process internal to the TNG environment has completed the processing of a request and supplies as available some parameter(s) that the client process is waiting for, these data are sent to the Communication Ancillary Process, which, in turn, performs the following actions:

  - recognises which is the client that is waiting for those data that the TNG environment has made available;
  - transforms the parameter(s) into a format siutable for the shipment to the client that is waiting for it (them);
  - incapsulates the parameter(s) into packet(s) having the REMOT format;
  - sends the packet(s) to the client via the socket connection.

- The communications between processes internal to the TNG environment (including the Communications Ancillary Process itself too) are handled by a library of functions which provides TCP/IP, UDP/IP, System V exchange of messages, depending on the processes involved in the communication and on the systems which they reside in.

Looking at this all, we can evaluate as good the work of integrating the Teleoperation System into the TNG environment, because of the following facts:

- the integration itself (with the policy chosen at this step) has been quite simple and immediate if compared to the efforts that

could have been faced while trying to plan, design and develop a completely new remote control software for the Telescope from scratch;

- we have verified that the TNG is remotely accessible while trying to use it with the aid of the software which sets up the Teleoperation System;
- the communications, when handled by the Teleoperation System, have shown to be reliable enough and the quality of service seems to have been improved;
- using the Teleoperation System, we have verified that we can profitably add the videoconferencing tool to our facilities; although not fundamental, it can be considered a good addition to the whole scientific instrumentation of the TNG.