Alexander Neidhardt et al.: E-control: First Public Release of Remote Control Software for VLBI Telescopes, IVS 2010 General Meeting Proceedings, p.439–443
http://ivscc.gsfc.nasa.gov/publications/gm2010/neidhardt2.pdf

E-control: First Public Release of Remote Control Software for VLBI Telescopes

Alexander Neidhardt [1], *Martin Ettl* [1], *Helge Rottmann* [2], *Christian Plötz* [3], *Matthias Mühlbauer* [3], *Hayo Hase* [3], *Walter Alef* [2], *Sergio Sobarzo* [4], *Cristian Herrera* [4], *Ed Himwich* [5]

[1)] *Forschungseinrichtung Satellitengeodäsie, TU München, Geodätisches Observatorium Wettzell*

[2)] *Max-Planck-Institut für Radioastronomie*

[3)] *Bundesamt für Kartographie und Geodäsie, Geodätisches Observatorium Wettzell*

[4)] *Universidad de Concepción*

[5)] *NVI, Inc., National Aeronautics and Space Administration, Goddard Space Flight Center (NASA/GSFC)*

Contact author: Alexander Neidhardt, e-mail: neidhardt@fs.wettzell.de

Abstract

Automating and remotely controlling observations are important for future operations in a Global Geodetic Observing System (GGOS). At the Geodetic Observatory Wettzell, in cooperation with the Max-Planck-Institute for Radio Astronomy in Bonn, a software extension to the existing NASA Field System has been developed for remote control. It uses the principle of a remotely accessible, autonomous process cell as a server extension for the Field System. The communication is realized for low transfer rates using Remote Procedure Calls (RPC). It uses generative programming with the interface software generator "idl2rpc.pl" developed at Wettzell. The user interacts with this system over a modern graphical user interface created with wxWidgets. For security reasons the communication is automatically tunneled through a Secure Shell (SSH) session to the telescope. There are already successful test observations with the telescopes at O'Higgins, Concepción, and Wettzell. At Wettzell the software is already used routinely for weekend observations. Therefore the first public release of the software is now available, which will also be useful for other telescopes.

1. Introduction

The future Global Geodetic Observing System aims at 40 globally distributed VLBI sites in the station network with two telescopes at each site in the best case scenario. These new VLBI2010 antennas should be distributed more regularly over the surface of the earth [5]. Some of these telescopes may be in remote locations and have limited staff, such as the current O'Higgins station. Therefore a method for remote control of existing sites was developed as an extension to the NASA Field System, which is currently used to control the equipment. The already existing tools for forward mouse, keyboard, and video signals are suboptimal, because they do not allow monitoring of the Internet connection itself to facilitate safety actions. The new software offers this Ethernet-based, safe, and stable remote communication. Since most of the new devices controlled by the Field System are also connected via Ethernet, the new concept also includes ideas to standardize such individual communication needs. This combination offers the appropriate elements for remote control as a new VLBI observing mode, named "e-control", which fits into the new observing strategies proposed in the VLBI2010 vision paper [3].

2. The Communication over Ethernet with Remote Procedure Calls

The communication in current communication systems over the Ethernet is usually a request and character-based (ASCII) client-server model. This means a service-requesting client starts the communication and sends an order request via message communication to a remote-service-offering server. On the server side the order is processed, and an answer message is returned to the client [6]. In classic communication networks each client-server-interaction is programmed individually during the software development process with the standard protocol stacks of either the Transmission Control Protocol over Internet Protocol (TCP/IP) or the User Datagram Protocol over Internet Protocol (UDP/IP). This proprietary approach makes it more difficult to set up a remote control and to include or adapt new properties of remotely usable devices.

A more modern attempt reduces the efforts of communication programming by defining a standardized way for the transmission of remote procedure calls (RPC). RPCs are comparable to local calls of procedures (functions) in a structured program, but are realized as standardized control and data flows over a communication network. The client calls a procedure or function without the concrete knowledge of the processing location. An additional RPC communication layer offers the transfer between the client and the remote processing server. The derived answer follows the same way but reversed to the client. This makes the procedure call seem local to the client. In the given context the Open Network Computing Remote Procedure Call (ONC RPC) is a preferred communication technique, because it is a standard part of each Linux Operating System and has been well tested since 1988. It also includes the platform-independent conversion of procedure parameter data using the External Data Representation (XDR) [8].

To reduce the programming effort, the communication layer can be created by using the generator utility “rpcgen”. This utility is available in each Linux system. It creates a low-level client-server communication based on ONC RPC. For sophisticated remote control implementations, higher level techniques are necessary, such as connection monitoring on client and server side and safety concepts. They realize stable states for the hardware even when the communication is broken. These higher level add-ons are used as middleware between the operating system and the application.

3. The Middleware Generator “Idl2rpc.Pl” and the Resulting e-control Stack

Common middleware systems extend the described communication with the needed higher level concepts and with advanced services. However these realizations are offered as huge, bulky, and abstract packages. The offered releases are correlated with operating system versions and are often commercial products. Additionally, they are not always flexible enough to be used in heterogeneous networks with different security guidelines and firewall implementations because of the additional proprietary protocol layers [1].

In order to deal with these issues a new middleware generator “idl2rpc.pl” was designed. This generator is just a single Perl script which uses only the ONC RPC realizations of standard Linux distributions. It extends “rpcgen” and reads a dedicated interface definition file to create several C++ adaptor classes for the C-coded RPC communication. They can directly be used in the application code. Dedicated modules offer threads to realize parallel tasks with semaphores to protect critical sections. A sophisticated communication control mechanism, such as a watchdog process, which always restarts the server after an unexpected crash, extends the stability. An Au-

tomatic Safety Device (ASD) controls the existence of a responsible client and forces safety actions when the connection to the client breaks down. It is open source code, and no additional, external packages are needed [2]. The basic usage of TCP/IP and UDP/IP without additional, proprietary layers allows the generated communication to easily tunnel firewall barriers using a Secure Shell (SSH). Therefore an additional program, the "sshbroker", is added to automatically control and monitor the SSH-connection to the VLBI systems or entire observatories. This helps to develop autonomous, distributed systems, consisting of several independent computers (processors), which are connected together to solve a collective task in a cooperative way [4].

Each hardware device can be represented as an individually programmed server which is autonomous in controlling the hardware and accessible via a generated idl2rpc communication. Higher level servers can contain several client modules to access and autonomously control different, subordinated device servers. The higher level combined servers themselves also offer services on a generated idl2rpc communication. The server accessing data on one platform can be found via registrations at a "portmapper", where all services and their ports are registered. It is also possible to directly contact the servers on their dedicated communication ports. In summary, a hierarchical architecture is realized which establishes hierarchical and autonomous control zones. On the top level of this hierarchy a user interface allows human interaction.

For e-control the resulting communication stack is shown in Figure 1. One of the main drivers in the given design is a strict separation of control, communication, and presentation logic. The complete arithmetic and workflow control logic reside in the server, defined as device control code. It is an autonomous working process which interacts with the remote controlled device (in the case at hand, the FS). The communication code independently connects the server to the outer world for requesting clients which can also be realized over the mentioned SSH tunnels with the "sshbroker". The clients are only used to realize a user interface with a presentation of the server-processed elements.

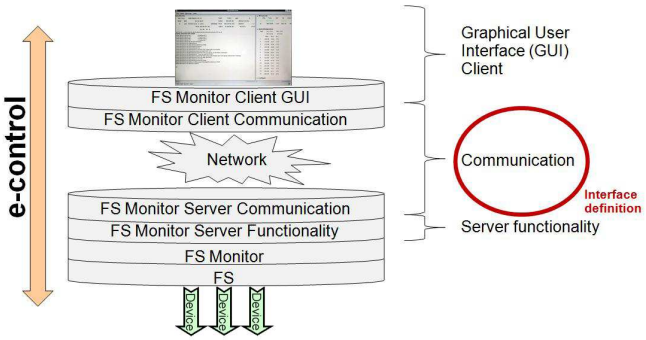

Figure 1. The communication stack of e-control.

4. The Features of the New Remote Control Software

It is helpful to provide a graphical user interface. For the described FS extension, wxWidgets is the preferred way to do this. It is a C++ based open source framework for platform-independent developments of graphical user interfaces [7]. Although the current RPC generator only supports Linux systems (32 and 64 Bit), the graphical user interface is modular enough to support different

platforms such as Windows, Linux, OSX, and others. In terms of the proposed FS extension a new graphical user interface was created using the wxWidgets framework for its realization (see Fig. 2). To keep usage similar, the display elements are organized to be like those of the current local interface for the FS. In addition a more graphical-oriented version can be chosen during runtime. The example in Figure 2 shows a realization with pie charts and line plots to present Mark 5 information and system temperatures over time.

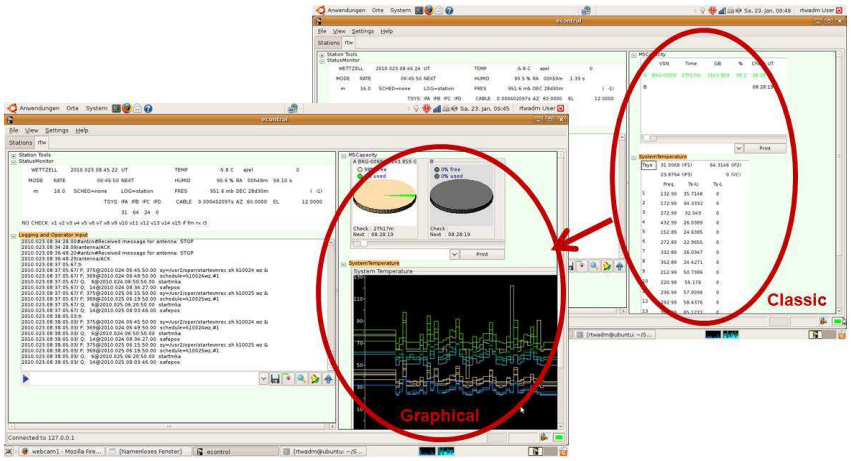

Figure 2. The new style of remote operator interaction with the NASA field system.

The entire interface is configurable during runtime, which allows switching between different telescope locations on-the-fly, just by reading a new configuration file. No additional software is needed, except the standard SSH client which is automatically controlled via different “sshbrokers”. The behavior of these “sshbrokers” can be set in the configuration mask, where also all of the other relevant configuration file parameters can be set. This is how the automated SSH-control and communication monitoring with re-establishment can be defined.

In addition to the standard behavior of the well-known FS windows, some additional features were added. This includes the implementation of a more sophisticated logging-command tool. With it, the operator can see the progress in the log-file and can enter FS commands. In addition, the user can also perform a search. This will open additional log tabs to present only lines where the defined search tag is found. It offers the ability to print only the error lines or especially interesting monitoring values such as the cryogenic parameters. The log output can be dumped into a local file copy. Furthermore it is also possible to generate acoustic “BEEP” signals for each error log via standard audio card.

To avoid a separate handling of the connection to a remote system, e.g., for additional software as terminals, it is possible to define hot keys in the configuration section. They are linked with remote programs and open them via separated and “sshbroker” controlled windows. These hot keys can also be defined during runtime. To complete the basic setup of tools, a direct output for the phase calibration monitor (MonPCal) and a chat tool for the inter-operator communication has been added.

5. Conclusion

The new software offers possibilities for new observing modes in line with the VLBI2010 project [3]. It should become an official part of the NASA Field System software. Additionally, a current version can be downloaded from the newly installed software servers at the Technische Universität München. As shown during the live demonstration at the 5th IVS General Meeting, it is easy to use the software. At the moment there are test sites in Hobart, Tasmania (both 26-m and 12-m antennas); in Wettzell, Germany; in Concepción, Chile; and in O'Higgins, Antarctica. It is planned to run shared shifts between these telescopes. Additional tests are planned at Westford and Kokee Park. For future developments to increase the security of access, the development team is participating in the NEXPRES project, funded by the European Union. Furthermore a corresponding software for controlling satellite laser ranging systems at Wettzell Observatory is in the final stage of development.

In summary, these new developments support the ability to check system states remotely, even for very remote sites. They allow shared observations and tele-working sessions and can facilitate remote specialists in assisting the local operators. They can form the new observing strategies. For the highest reliability, all telescopes need to support remote connections. A highly educated local staff will continue to be needed for maintenance and on-call service to react to critical situations.

References

[1] Neidhardt, A.: Verbesserung des Datenmanagements in inhomogenen Rechnernetzen geodätischer Messeinrichtungen auf der Basis von Middleware und Dateisystemen am Beispiel der Fundamentalstation Wettzell, Dissertation, Mitteilungen des Bundesamtes für Kartographie und Geodäsie, 37, Bonifatius GmbH, 2006.

[2] Neidhardt, A.: Manual for the remote procedure call generator "idl2rpc.pl", Geodetic Observatory Wettzell, 2009.

[3] Niell, A., Whitney, A., Petrachenko, B., Schlüter, W., Vandenberg, N., Hase, H., Koyama, Y., Ma, C., Schuh, H., Tuccari, G., VLBI2010: Current and Future Requirements for Geodetic VLBI Systems, In: IVS Annual Report 2005, NASA, Behrend, D., Baver, K., 13–40, 2006.

[4] Puder, A.; Römer, K.: Middleware für verteilte Systeme, dpunkt-Verlag GmbH, 2001.

[5] Rothacher, M., Beutler, G., Bosch, W., Donnellan, A., Gross, R., Hinderer, J., Ma, C., Pearlman, M., Plag, H.-P., Richter, B., Ries, J., Schuh, H., Seitz, F., Shum, C.K., Smith, D.; Thomas, M., Velacognia, E., Wahr, J., Willis, P., Woodworth, P., The future Global Geodetic Observing System (GGOS), In: The Global Geodetic Observing System. Meeting the Requirements of a Global Society on a Changing Planet in 2020, Springer-Verlag, Plag, H.-P., Pearlman, M. (eds.), 2009.

[6] Singhal, M., Shivaratri, N.G., Advanced Concepts in Operating Systems, McGraw-Hill, 1994.

[7] Smart, J., Hock, K., Csomor, S., Cross-Platform GUI Programming with wxWidgets, Prentice Hall International, 2005.

[8] Stevens, R.W., Programmieren von UNIX-Netzen. Grundlagen, Programmierung, Anwendung, Prentice-Hall International, 1992.