AUTOMATION INTERFACES OF COPPER MOUNTAIN TECHNOLOGIES' NETWORK ANALYZERS

08/03/2018

Introduction

Copper Mountain Technologies' VNAs can be readily automated and there are numerous options for users who wish to do so. To help illustrate the different automation methods and interfaces available, consider the components in an automated VNA measurement system. There is hardware: the VNA itself and a computer. There is software, as well: the VNA control software loaded onto the computer. While non-automated use of these components consists of each of these components in this basic configuration, automation options alter this configuration.

For instance, an instrument can be locally or remotely automated. In local automation, an automating program executes on the same computer that is connected to the VNA and has the VNA control software. This is the same basic configuration. On the other hand, in remote automation, another computer, connected to the first by a network, runs the automation program.

There are requirements that determine which program needs to be installed where, but that varies depending on the automation interface used. Automation of Copper Mountain Technologies' VNAs can be achieved by using either of two interfaces:

- Component Object Model/Distributed Component Object Model (COM/DCOM) interfacing is supported for easy local automation on a Windows PC or remote automation across a network of Windows PCs.
- Transmission Control Protocol (TCP) interfacing is supported for local automation on a Windows or Linux PC, or remote automation across networks of computers whose operating systems may vary using standardized instrument automation commands.

This application note describes the relative advantages of each interface and demonstrates some simple examples of how each can be utilized in practice.

Interface Details

COM/DCOM Interface

After installation of the VNA control software, the last step of the installation wizard prompts users with the option whether to register the software's COM server. When this is selected, Windows creates an association between the VNA control software and an "Application Name." For example, in the case of the 2-port 2-path software (S2VNA) installer, the name "S2VNA.Application" is mapped to S2VNA.exe, wherever it was installed (i.e. C:\VNA\S2VNA\S2VNA.exe). Without the COM server and this association, COM/DCOM automation is not possible.

Automation through COM/DCOM is accomplished by creating an instance of a COM object in the automation software. When this software is executed, Windows automatically launches the VNA control software if the COM server is not already running. Then, the server-client connection is established. The COM object has properties that can be read from or written to and methods that can be invoked directly in the programming language in which the automation software is written.

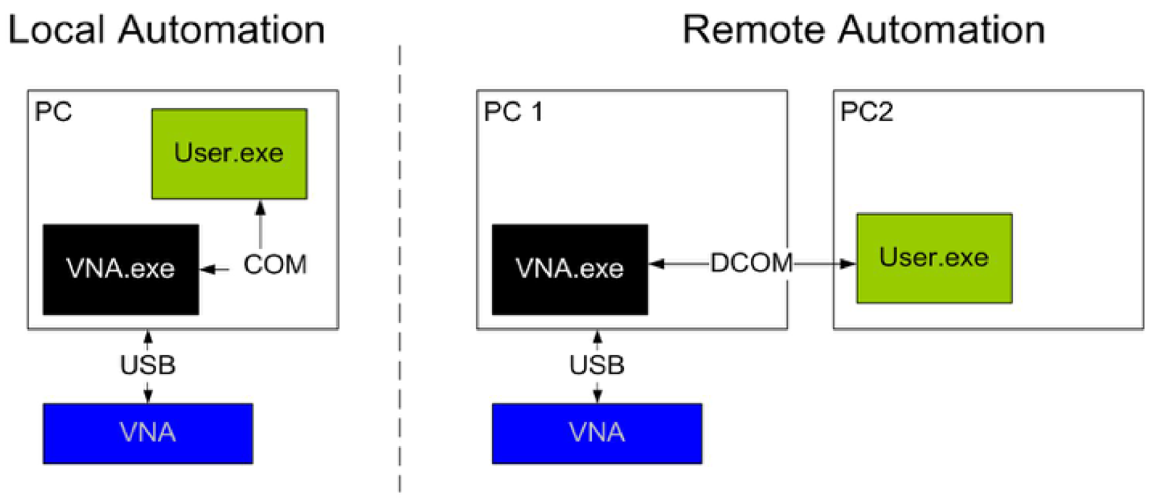


Figure 1: COM/DCOM Automation Block Diagrams

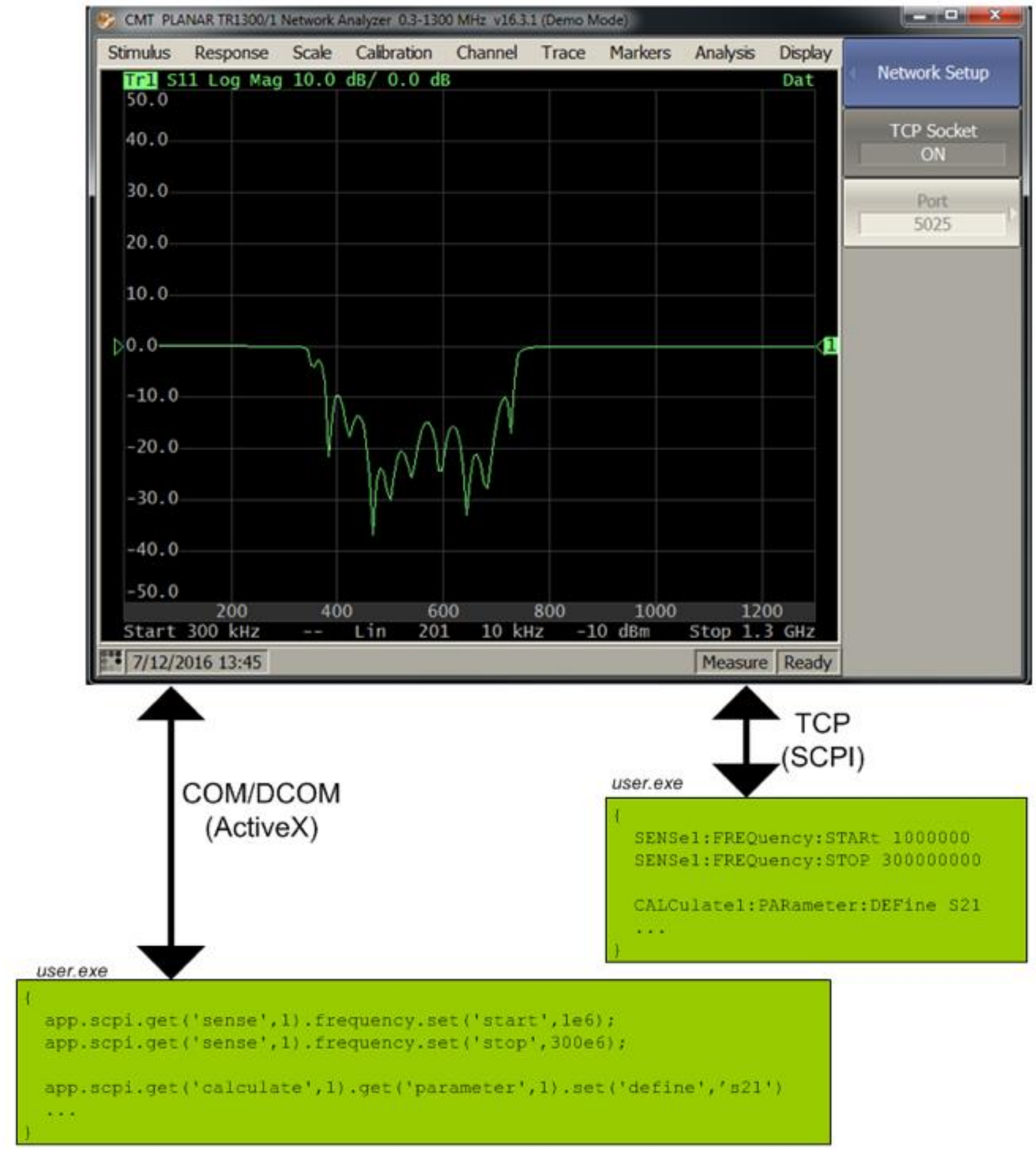
COM automation is well suited for local automation, where the automation software and the VNA control software both execute on the same Windows PC. This PC must be located within a USB cable's reach of the VNA being automated, generally meaning within a 5-meter radius, though there are technologies available for extending USB over distances of many meters (i.e. conversion to fiber optic or use of a wired or wireless USB sharing hub). Some of these are also explored in application notes available at http://coppermountaintech.com/technical-library/.

DCOM automation is suited for remote automation. Copper Mountain Technologies has an application note that demonstrates DCOM automation network configuration, which can be found at http://coppermountaintech.com/dcom-configuration-guide/. The procedure is quite involved, requiring various user account authorization settings and a firewall opening which might not be practical in some network environments. If a network between Windows PCs can be configured as a Workgroup – typically found in home network scenarios – DCOM may be a practical approach.

The COM/DCOM interface is a Microsoft product and can only be used in a Windows or virtual Windows environment. COM/DCOM automation is supported by LabVIEW, MATLAB, Python (with the PyWin32 package), VB.NET, VBA, C++, C#, VEE, Octave, and more. Because it is a software-to-software interface, data transfer times are negligible. Additionally, depending on the programming environment, the COM properties structure might be directly explored and command syntax auto-completed - a convenient addition to help avoid typos.

For more information on how to program COM/DCOM interface automation, please see the COM Programming Manual available with the relevant VNA control software, which can be downloaded from http://coppermountaintech.com/automation/. The documentation can additionally be found in the "Doc" subdirectory in the directory that the VNA control software is installed.

In summary, The COM interface works well for local automation in a Windows or virtual Windows environment and can even have unique benefits. However, it is difficult to set up DCOM for remote automation.


CMT PLANAR TR1300/1 Network Analyzer 0.3-1300 MHz v16.3.1 (Demo Mode)
Stimulus Response Scale Calibration Channel Trace Markers Analysis Display
Tr1 S11 Log Mag 10.0 dB/ 0.0 dB
Start 300 kHz -- Lin 201 10 kHz -10 dBm Stop 1.3 GHz
7/12/2016 13:45
Measure Ready
Network Setup
COM/DCOM (ActiveX)
TCP (SCPI)
user.exe
user.exe

TCP Socket: ON
Port: 5025

```
{
  SENSe1:FREQuency:STARt 1000000
  SENSe1:FREQuency:STOP 300000000

  CALCulate1:PARameter:DEFine S21
  ...
}
```

```
{
  app.scpi.get('sense',1).frequency.set('start',1e6);
  app.scpi.get('sense',1).frequency.set('stop',300e6);

  app.scpi.get('calculate',1).get('parameter',1).set('define','s21')
  ...
}
```

Figure 2: Comparison of SCPI and COM syntax as they relate to automating the TRVNA.exe

TCP Interface

In automation under the TCP interface, the connection between the automation program and the VNA control software is established through TCP/IP sockets and the network stack of the computer. Through these, the automation program can send industry-standard Standard Commands for Programmable Instruments (SCPI) commands to the VNA control software, either locally on the same computer or remotely across a network.

SCPI is a standard protocol for communicating with instruments using ASCII-based commands to control the device and ASCII encoding to send and receive data. It is a relatively easy to use protocol because of its straightforward encoding. Being a standard, it has many commands that are common between different instruments; because the SCPI commands of Copper Mountain Technologies' VNAs align closely with popular legacy equipment from other vendors, code reuse can be maximized, allowing for previously written code to be utilized with minimal modification.

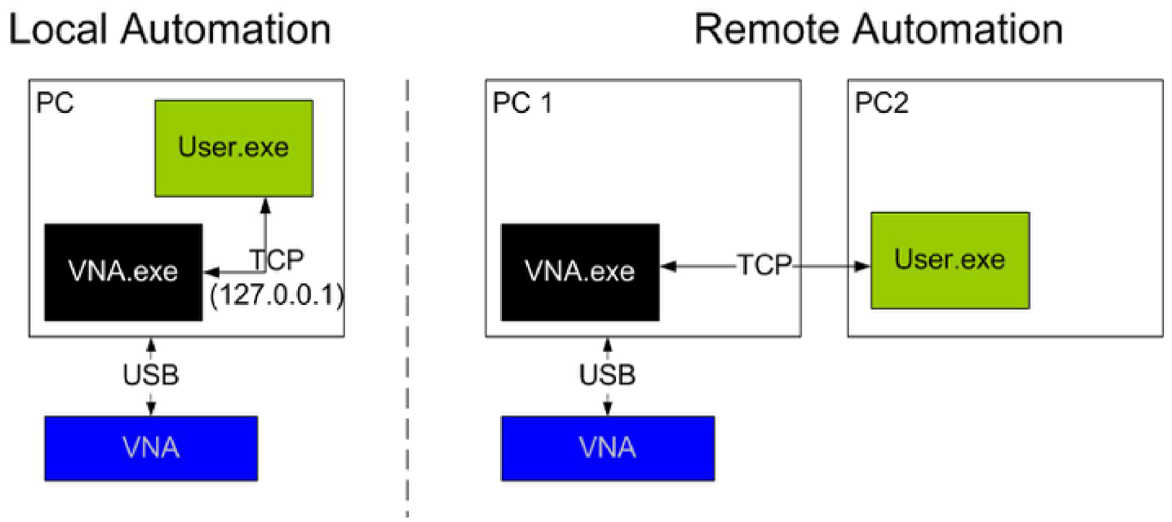


Figure 3: TCP Automation Block Diagrams

Local automation can be accomplished via the special IP address 127.0.0.1 (also known as 'localhost') together with port number 5025. This makes it simple to connect to a VNA over LAN, because no special network configuration is typically required beyond possibly port forwarding for remote connections across the internet.

Remote automation simply requires knowledge of the IP address of the PC co-located with the VNA, a port number through which commands will be received and processed, and a network to connect the two computers. TCP automation can be used locally and/or remotely with roughly equal ease. The method of connection is through the socket interface, which is supported by most programming languages. Connecting locally and remotely both follow the same procedure and utilize the same commands. Finally, it should be noted that for remote automation, the remote PC is not limited in its operating system so long as it supports TCP socket interfacing.

TCP has its own drawbacks. To query the device, you must first send a command and then read the response. SCPI uses ASCII to transfer data, so all data will be read as strings and must be later converted. Generally, programming environments will not be aware of valid and invalid SCPI command syntaxes, so errors due to typos are more likely to occur. Finally, using the TCP socket will introduce delays due to the latencies and bandwidth limitations of the network stack.

Another Option for Local Automation: VISA

National Instruments' Virtual Instrument Software Architecture (VISA) is a communication API that is used to communicate with measurement devices. This API is supplied by National Instruments, Keysight, and/or open source projects like pyvisa-py, depending on how a user intends to implement it. The command syntax in VISA is standardized without regard to the physical connection between the computer and the instrument to be automated. To communicate with the same instrument connected through a different interface, simply specify the new device connection. For the most part, VISA sends SCPI commands that are standardized for many different instruments, which makes converting code written for one instrument to another simple. In the context of the VNA'S COM and TCP interfaces, only the TCP socket falls within the scope of VISA-supported interfaces.

Choosing an Automation Interface

For local automation, generally, COM automation is easier to use and debug because of the ability of the IDE to interact directly with the programmer in terms of available properties and methods. If reuse of legacy code is not a significant consideration, our advice is to consider using COM over TCP automation.
For remote automation or outside of the Windows operating system, on the other hand, TCP automation will likely be a preferred option due to the simplicity of connection. If legacy code written using SCPI commands is available, e.g. for an ENA series instrument from Agilent/Keysight like the E5071C VNA, this also leads to a suggestion to use the TCP interface. Legacy code will likely will be easier to adapt to automate a Copper Mountain Technologies' VNA over TCP than to add support for COM.

Conclusion

Regardless of which technique you choose to use, resources are available to help you. On the Copper Mountain Technologies website, at http://coppermountaintech.com/automation/, you can find programming manuals for both SCPI and COM/DCOM interfaces as well as examples for both methods in multiple programming languages. If you are still having trouble, you can contact the Copper Mountain Technologies' support team at support@coppermountaintech.com or through the "Ask An Engineer" buttons on the website. Any question is appreciated by the support team, as Copper Mountain Technologies wants to extend your reach by helping you maximize the productivity of your metrology-grade VNA.