Comunicación con instrumentos con Python

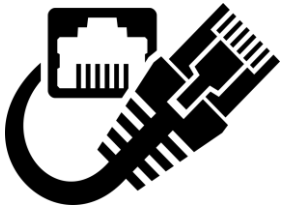
Hay una variedad de instrumentos con distintas conexiones y protocolos de comunicación:

USB

VXI

GPIB – IEEE-488

Serie - RS232



Ethernet – TCP/IP

Hay una variedad de instrumentos con distintas conexiones y protocolos de comunicación:

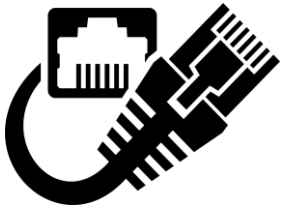
USB

VXI

GPIB – IEEE-488

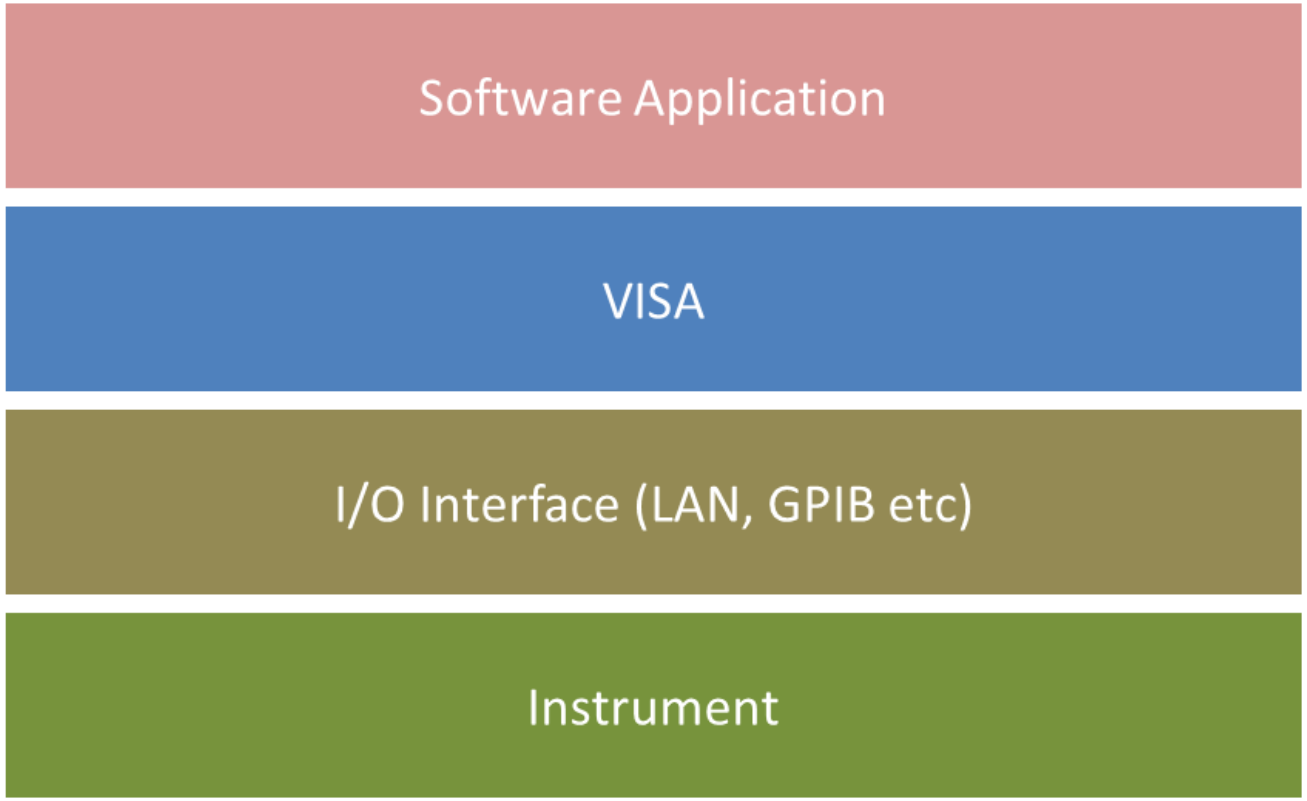
Virtual Instrument Software Architecture (VISA)

Serie - RS232



Ethernet – TCP/IP

VISA funciona cómo una capa que unifica la comunicación:

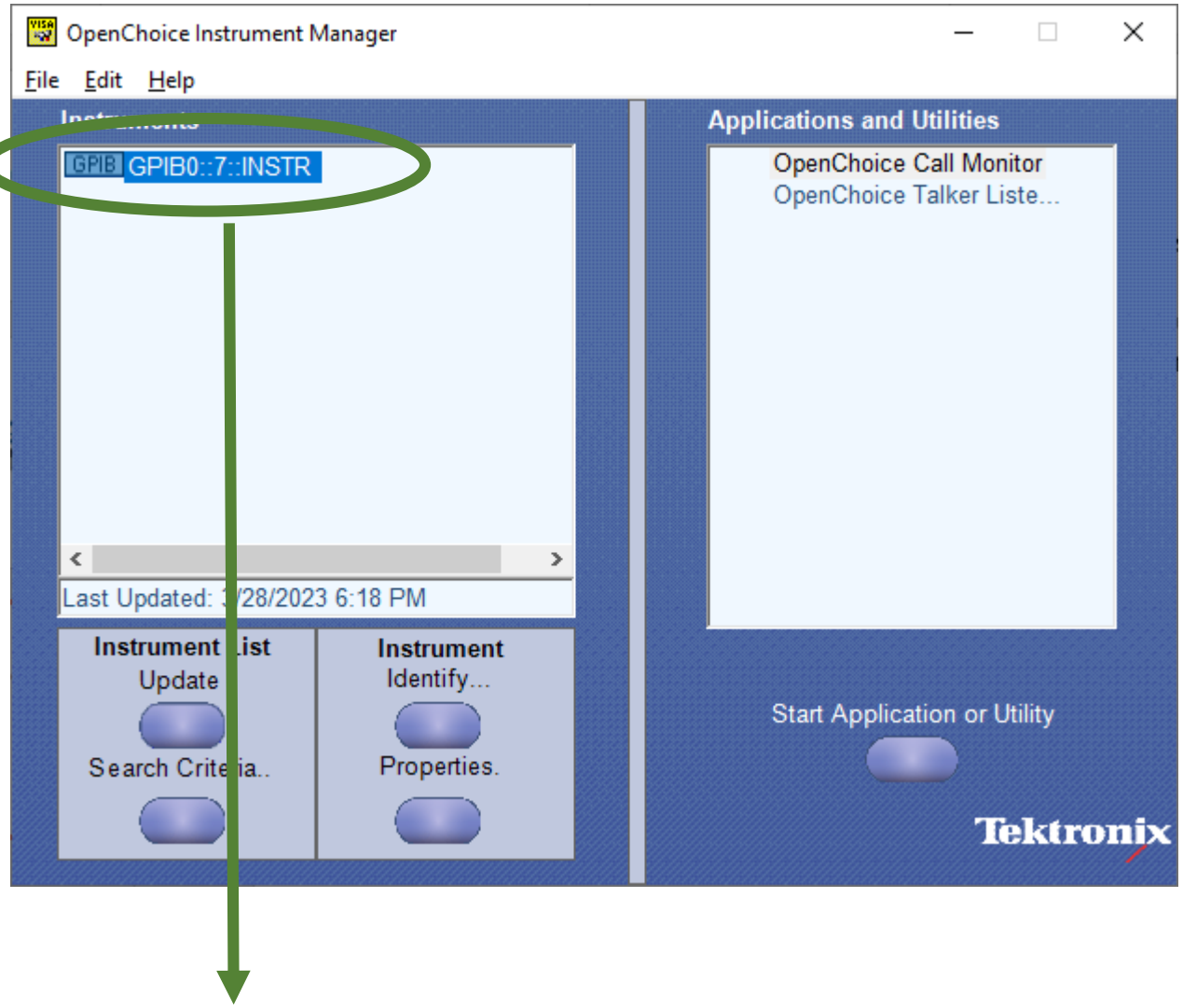


*

Hay muchas implementaciones: National Instruments, Tektronix, Keysight etc.

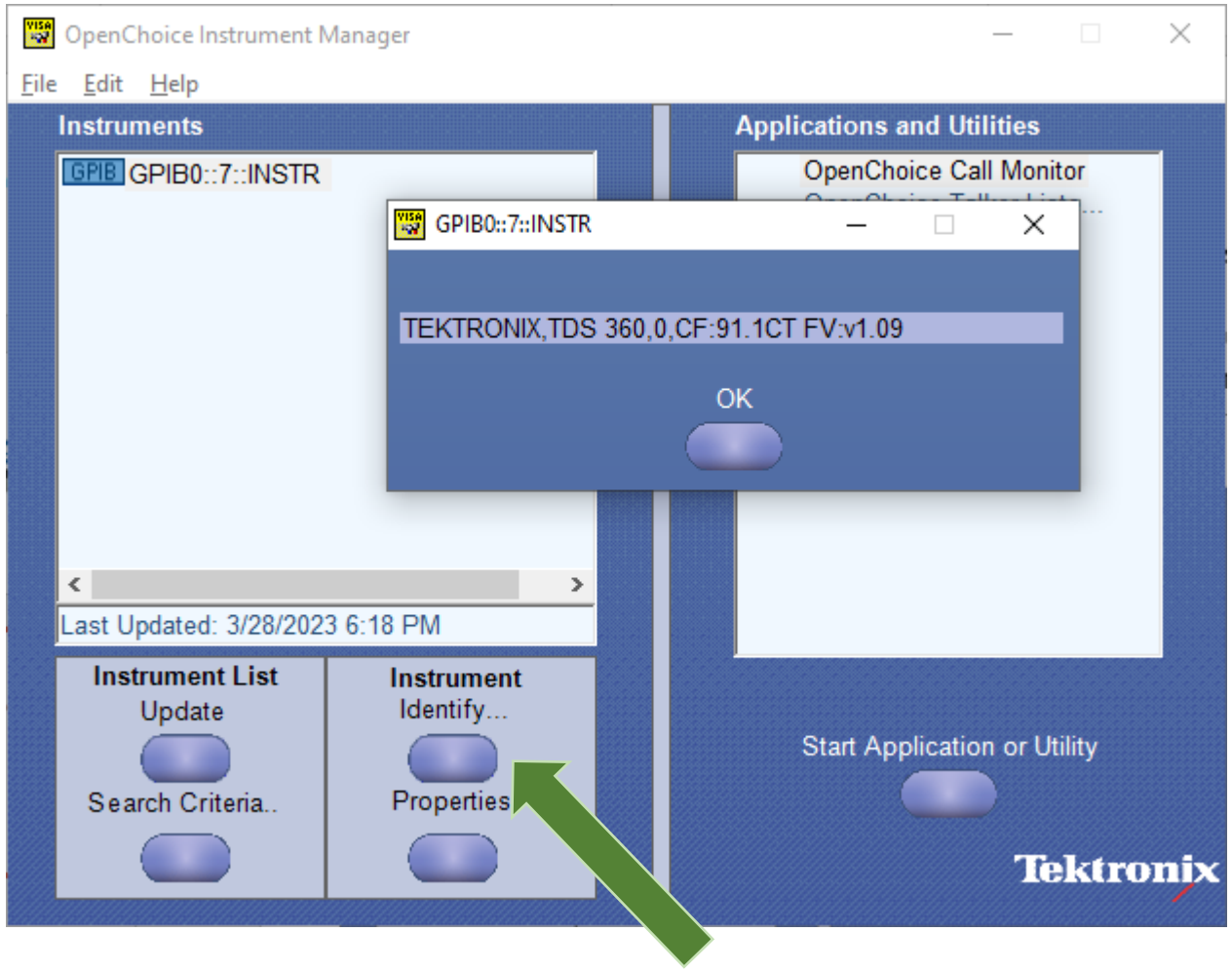
*CC Attr 4.0: http://navjodh.com/test-and-measurement/virtual-instrument-software-architecture-visa-basics-and-interoperability/

Suelen incluir algún cliente para **chequear que todo esté bien:**
probar comandos básicos, etc



Identificador del recurso **VISA**

Suelen incluir algún cliente para **chequear que todo esté bien:** probar comandos básicos, etc



*IDN?

Standard Commands for Programmable Instruments (SCPI)

*IDN?: identifica al dispositivo

?: esperamos respuesta

*: es un comando estandar

Mnemonic	Name	488.2 Section
*CLS	Clear Status Command	10.3
*ESE	Standard Event Status Enable Command	10.10
*ESE?	Standard Event Status Enable Query	10.11
*ESR?	Standard Event Status Register Query	10.12
*IDN?	Identification Query	10.14
*OPC	Operation Complete Command	10.18
*OPC?	Operation Complete Query	10.19
*RST	Reset Command	10.32
*SRE	Service Request Enable Command	10.34
*SRE?	Service Request Enable Query	10.35
*STB?	Read Status Byte Query	10.36
*TST?	Self-Test Query	10.38
*WAI	Wait-to-Continue Command	10.39

Ahora en Python

```
In [3]: import pyvisa as visa
   ...: rm = visa.ResourceManager()
   ...: rm.list_resources() # Qué recursos hay disponibles?
Out[3]: ('GPIB0::7::INSTR',)
```

Ahora en Python

```
In [3]: import pyvisa as visa
   ...: rm = visa.ResourceManager()
   ...: rm.list_resources() # Qué recursos hay disponibles?
Out[3]: ('GPIB0::7::INSTR',)
In [4]: # Abrimos el recurso de VISA, en este caso es un osciloscopio
   ...: osc = rm.open_resource('GPIB0::7::INSTR')
   ...: osc.query('*IDN?') # Este comando está dispobible en TODOS los dispositivos
Out[4]: 'TEKTRONIX,TDS 360,0,CF:91.1CT FV:v1.09\n'
```

Ahora en Python

```
In [5]: # Probamos otro queries que dependen del disposivo `query`
   ...: osc.query('CH1?')
Out[5]: '1.0E0;-2.04E0;0.0E0;DC;FULL;0\n'
```

Qué comandos tiene: hay que mirar el manual

CH<x>? (Query Only)

Returns the vertical parameters. Because CH<x>:SCAle and CH<x>:VOLts are identical, only CH<x>:SCAle is returned.

Group Vertical

Syntax CH<x>?

Examples CH1?
might return the string :CH1:SCALE 1.0E-2;POSITION 0.0E0;
OFFSET 0.0E0;COUPLING DC;BANDWIDTH FULL for channel 1.

Ahora en Python

```
In [5]: # Probamos otro queries que dependen del disposivo `query`
   ...: osc.query('CH1?')
Out[5]: '1.0E0;-2.04E0;0.0E0;DC;FULL;0\n'
In [6]: osc.query('HOR?')
Out[6]: 'MAIN;1000;5.86E1;2.5E-4;50;2.5E-4;RUNSAFTER;1.0E-8;1.0065E-6;LOCK;LOCK;0\n'
```

Qué comandos tiene: hay que mirar el manual

HORizontal? (Query Only)

Returns all settings for the horizontal commands. The commands HORizontal:MAIn:SCAle, HORizontal:MAIn:SECdiv, HORizontal:SCAle, and HORizontal:SECdiv are equivalent, so HORizontal:MAIn:SCAle is the only value that is returned.

Group Horizontal

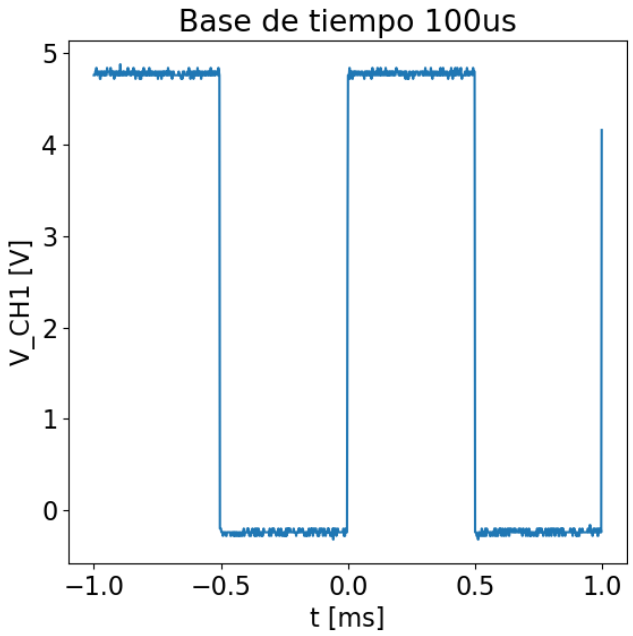
Syntax HORizontal?

Examples HORIZONTAL?
might return the string :HORIZONTAL:MODE MAIN;RECORDLENGTH 1000; POSITION 5.0E0;TRIGGER:POSITION 50;:HORIZONTAL:MAIN:SCALE 1.0E-6;:HORIZONTAL:DELAY:MODE RUNSAFTER;SCALE 1.0E-6;TIME:RUNSAFTER 1.6E-8;:HORIZONTAL:REF1 LOCK;REF2 LOCK;FITTOSCREEN 0

Ahora en Python

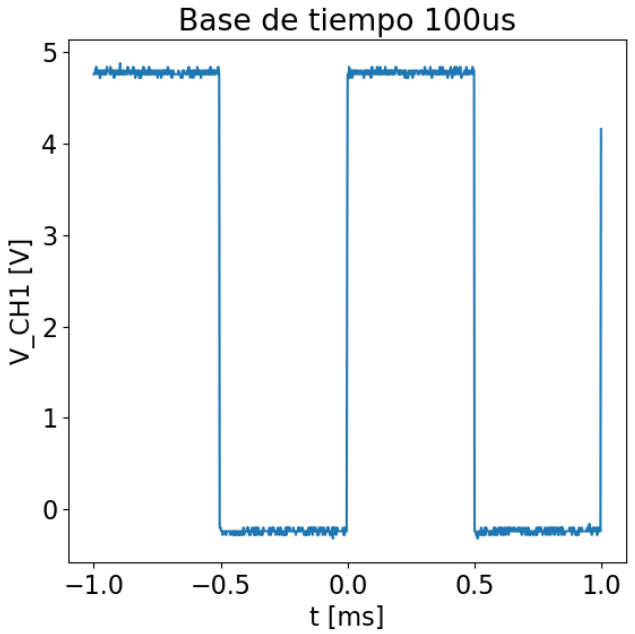
respuesta binaria

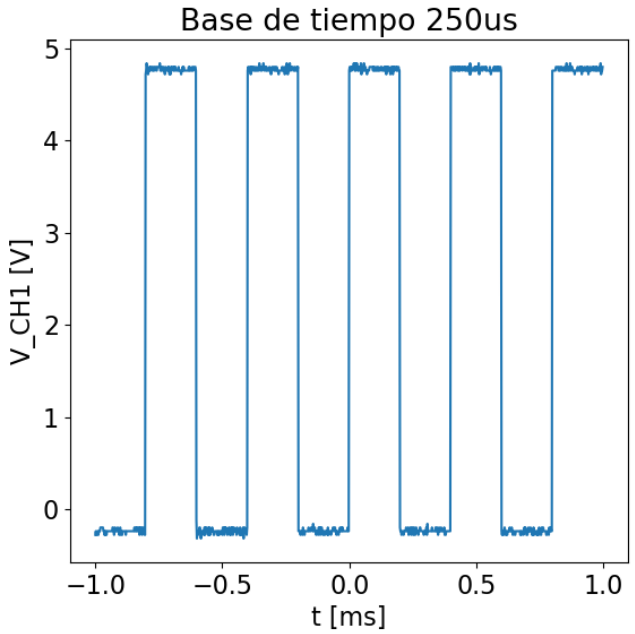
```
In [8]: osc.write('DAT:SOU CH1') # Adquirimos del canal 1
   ...: # La lectura se puede hacer en modo binario, lo que la hace más eficiente
   ...: conf_parameters_a = parse_parameters(osc.query('WFMP?'))
   ...: data_a = osc.query_binary_values('CURV?', datatype='B', container=np.array)
   ...: data_a
Out[8]: array([196, 196, 196, ...,  72,  71, 181], dtype=uint8)
```



Ahora en Python

```
In [9]: # Cambiamos la escala temporal y volvemos a adquirir
   ...: scale = '250e-6'
   ...: osc.write(f'HOR:SCA {scale}')
   ...: conf_parameters_b = parse_parameters(osc.query('WFMP?'))
In [10]: data_b = osc.query_binary_values('CURV?', datatype='B', container=np.array)
    ...: data_b
Out[10]: array([ 70,  71,  70, ..., 196, 196, 197], dtype=uint8)
```





Muchas cosas ya resueltas en el repositorio de Github de Laboratorios Superiores

Seguimos probando en vivo con un osciloscopio y un generador de funciones