Connecting To Remote Test Systems

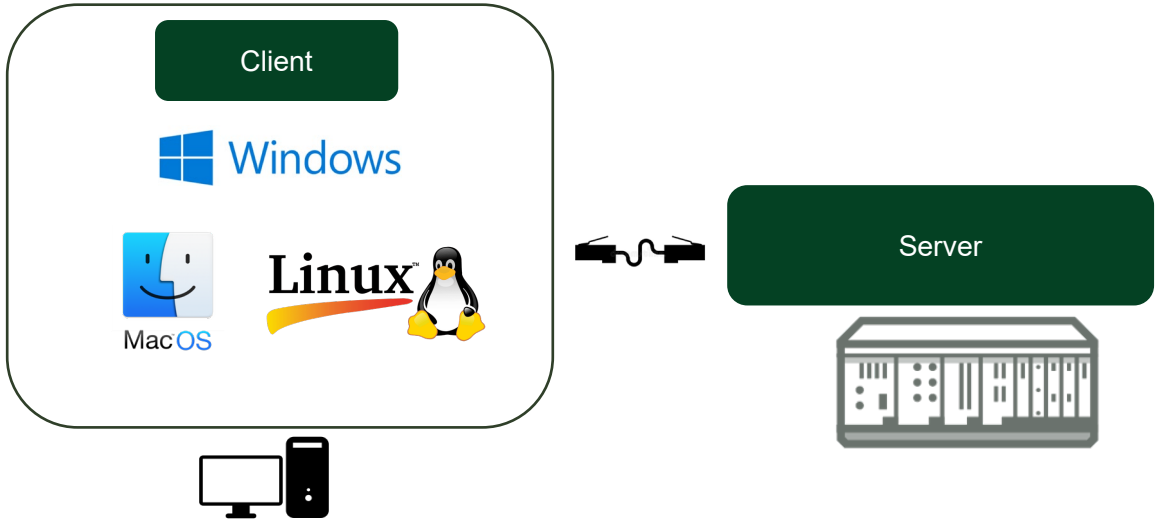
Christopher Cifra

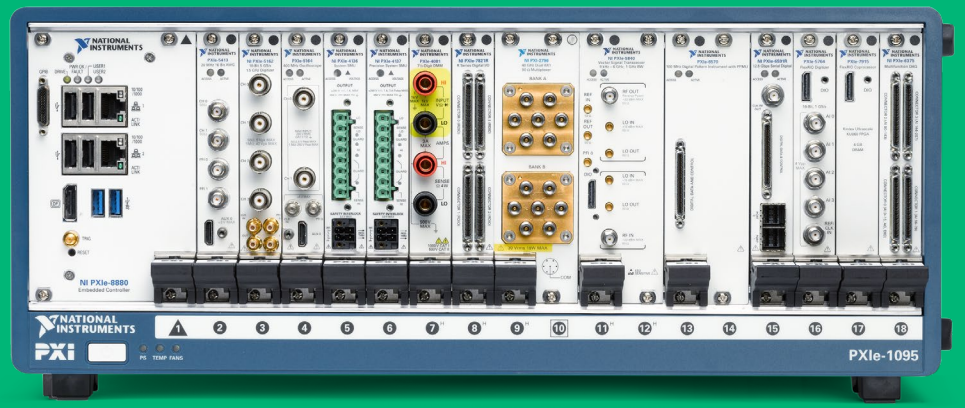
Agenda

1. Introduction
2. What is Remote-ability
3. Core Technologies
4. NI’s Investment
5. What’s Next
6. Q&A / Discussion

Remote-ability

Software capability that allows instrumentation and services to be accessed and controlled through a remote **network** interface via pre-defined standalone applications and/or API(s)/commands in a holistic and OS agnostic way using a **client / server** architecture.





FEATURES

NI Software Remote-ability Goals

Remotely Interact with NI Hardware and Services

Take high-quality, low-latency measurements remotely

Eliminate Redundant Driver Installs

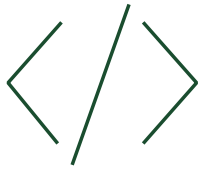
Require driver installation only on the server

Leverage Existing Workflows

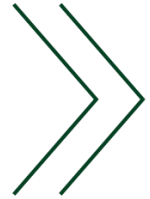
Wide range of supported OSs and programming languages

Powered by gRPC

gRPC is an open-source remote procedure call (RPC) framework that can run anywhere



Language and OS Agnostic



Low latency, highly scalable

Reliable and secure

Open-source

gRPC – Design Choices

- Transport: HTTP/2
  - Easily traverse common internet infrastructure
  - Firewalls, load balancing, encryption, authentication, compression, etc
  - Fast, multiplexing, supports bidirectional streaming
  - 1 TCP connection can handle many multiplexed messages

- Wire Format: Protocol Buffers (binary)
  - Default, but can be replaced (JSON, FlatBuffers, …)

- API Definition: Protocol Buffers IDL

gRPC – Key Features

- Simple service definition
  - Enables code generation of API for easy use
- Support any Language
  - Official support for 11 different languages
  - “Unofficial” support for many more languages
  - Code generate APIs for your language
- Cross platform (Windows, Mac, Linux, LinuxRT, …)
- Authentication: SSL/TLS – client/mutual – pluggable auth
  - Credentials per channel and/or per call
- Unary call and bidirectional streaming support
- Cancellation/Deadlines/Retries

Authentication / Authorization

Authentication can be enabled when needed for specific services

- Enabling authentication can affect overall performance

Client connect and per-method authentication support

Utilize HHTP headers for authorization tokens

OpenSSL based

- TLS - server certificate
- Mutual TLS (Zero Trust) - Client validation with client and server certificate

Software Industry Wide Use of gRPC

ASAM OSI

https://www.asam.net/standards/detail/osi

TensorFlow

Apache Arrow Flight

ZeroMQ – Protobuf support

https://zeromq.org/

Kafka (Protobuf support)

.NET Core

Google

Netflix

https://www.cncf.io/case-study/netflix/

Spotify

https://www.youtube.com/watch?v=qmlh9Co54lA

Square

https://www.youtube.com/watch?v=-2sWDr3Z0Wo

CockroachDB

https://github.com/cockroachdb/cockroach

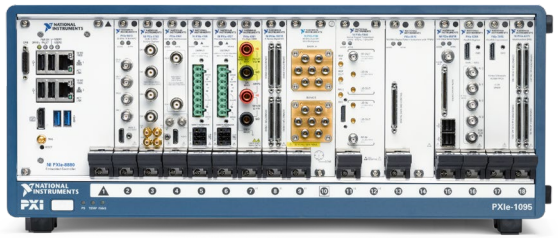
DropBox

Courier: Dropbox migration to gRPC – Dropbox

Dapr

Dapr's gRPC Interface | Dapr Docs

How gRPC Works



gRPC Server

Client API

Author provides a .proto file for each of their services

.proto File

gRPC

Contains all the details about the commands / queries / messages (called methods and messages)

Auto-creates a **full client API** ready for any programming language

gRPC – Diving In – The Service Definition

```
syntax = "proto3";
package greet;
option csharp_namespace = "GrpcGreeter";

service Greeter {
   rpc SayHello (HelloRequest) returns (HelloReply);
}

message HelloRequest {
   string name = 1;
}

message HelloReply {
   string message = 1;
}
```

Provides namescoping for everything in the file

RPC Service Definition

RPC Service Method

RPC Request Object

RPC Response Object

gRPC Performance Benchmark

- Typical Round-Trip performance
  - 10 Gb: 124 µs
  - 10 Gb Optimized: 77 µs
  - Localhost: 24 µs

- For Comparison DAQmx Read Call is 16 µs

Streaming Throughput

- Localhost: > 4 GB/s
- 1Gb/s Ethernet: 112 MB/s
  - 93% of theoretical max
- 10 Gb/s Ethernet: 1.2 GB/s

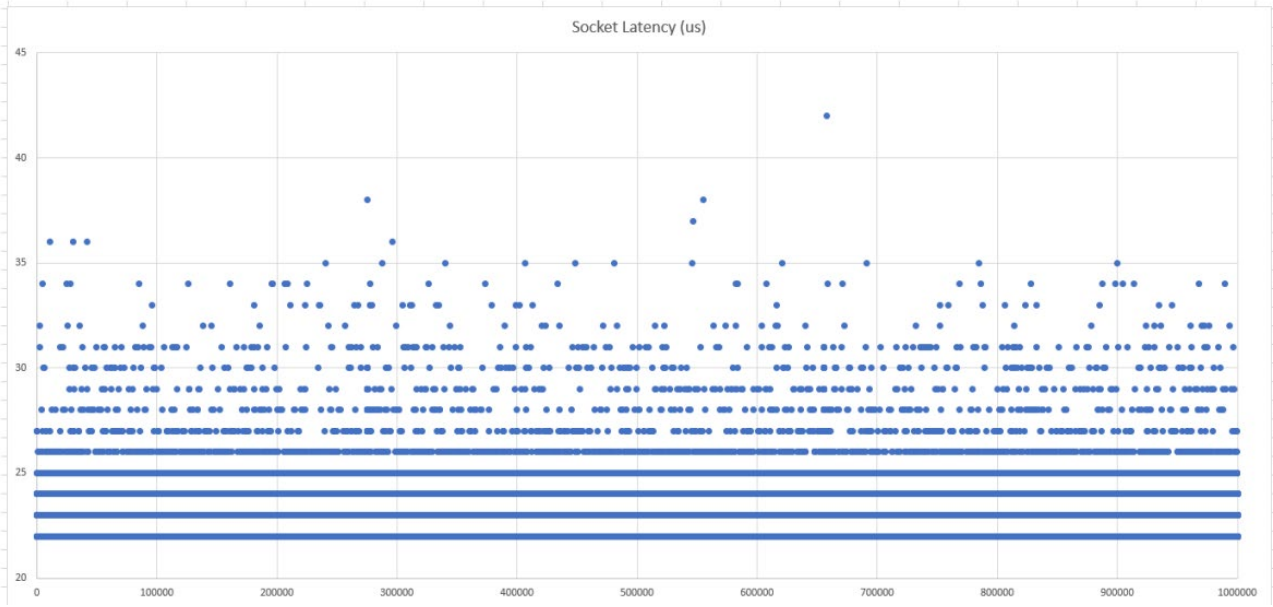
	10 Gb Link Local	10 Gb optimized	localhost
Messages / s	7903.45	11161.9	39685.1
MB / s	1122.17	1122.24	4915.76
RPC Latency (µs)	124	77	24
RPC Stream Latency (µs)	103	50	15

RPC Streaming Latency (µs)

	Payload Size							
	1	8	16	32	64	128	1024	32768
10 Gb	104	106	106	111	109	112	133	484
10 Gb optimized	51	50	52	51	52	54	66	324
localhost	15	15	15	15	15	15	15	116

RPC Call Latency (µs)

	Payload Size							
	1	8	16	32	64	128	1024	32768
10 Gb	124	125	125	127	126	130	151	544
10 Gb optimized	77	78	78	79	80	90	151	322
localhost	24	24	24	24	25	25	27	104



Simple Requests

RPC method - 1 int32 parameter and returns int32 result

- 1Gb ethernet through switch
- Insecure credentials
- 1 Client connection

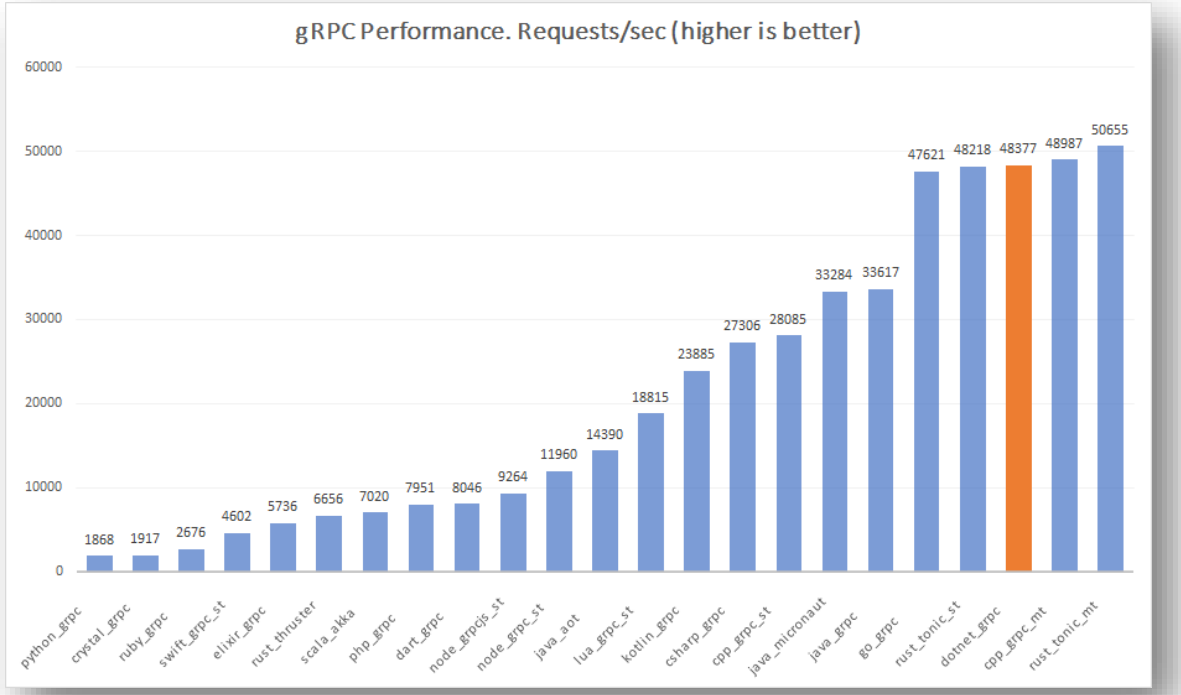
Linux localhost – 36,000 Requests/s

Linux to Linux – 4,400 Requests/s

Windows localhost – 10,000 Requests/s

10Gb

- Windows to Linux – 7,000 Requests/s
- Linux to Linux – 7,300 Requests/s



High Performance Use Cases (Experimental)

Continued research in supporting high performance use cases

ni/grpc-sideband: High bandwidth low latency data movement for gRPC based services. (github.com)

RDMA support for high throughput and low latency

- Support for real-time max latencies of <30us
- Full use of 40-100 Gb/s networking

Shared memory for localhost use cases

gRPC Ecosystem

Awesome-grpc

https://github.com/grpc-ecosystem/awesome-grpc

Official Libraries and Tools

- gRPC Core - C, C++, Ruby, Node.js, Python, PHP, C#, Objective-C
- gRPC Java - The Java gRPC implementation. HTTP/2 based RPC
- gRPC Node.js - gRPC for Node.js
- gRPC Go - The Go language implementation of gRPC. HTTP/2 based RPC
- gRPC Swift - The Swift language implementation of gRPC
- gRPC Dart - The Dart language implementation of gRPC
- gRPC C# - The C# language implementation of gRPC
- gRPC Web - gRPC for Web Clients
- gRPC Ecosystem - gRPC Ecosystem that complements gRPC
- gRPC contrib - Known useful contributions around github
- Homebrew gRPC - gRPC formulae repo for Homebrew
- grpc_cli - gRPC CLI tool

Protocol Buffers

Documentation

- Website - Official website and documentation
- Third-Party Add-ons for Protocol Buffers - List of add-ons for Protocol Buffers in main github repository

Tools

- buf - Protobuf tool that includes linting and breaking change detection. Allows many types of input including directly checking remote repositories and tarballs, and has a built-in compiler as well.
- prototools - Documentation generator & other tools for protobuf/gRPC
- protoc-gen-doc - Documentation generator plugin for Google Protocol Buffers
- Protoxygen - Doxygen plugin to generate documentation for protobuf/gRPC
- openapi2proto - A tool for generating Protobuf v3 schemas and gRPC service definitions from OpenAPI specifications
- Wireshark Protobuf Dissector - A Wireshark Lua plugin for decoding Google protobuf packets. Relevant PR and discussion.
- protoc-gen-lint - A plug-in for Google's Protocol Buffers (protobufs) compiler to lint .proto files for style violations
- prototool - Compile, lint, and format Protobuf files, and generate stubs for any lanuguage/plugin, along with Vim/IDE integration
- protoc-gen-validate - Protoc plugin to generate polyglot message validators
- go-proto-validators - Generate message validators from .proto annotations, used in `grpc_validator` Go gRPC middleware.
- protolock - Protocol Buffer companion tool to `protoc` and `git`. Track your .proto files and prevent changes to messages and services which impact API compatibilty.
- protoc-gen-map - SQL data mapper framework for Protocol Buffers.
- api-linter - A linter for APIs defined in protocol buffers.
- protoc-gen-struct-transformer - Transformation functions generator for Protocol Buffers.
- pbvm - Protocol Buffers Version Manager

Similar

- gogoprotobuf - Fork of golang/protobuf with extra code generation features
- MessagePack - It's like JSON, but fast and small
- Thrift - Thrift is an interface definition language and binary communication protocol
- TChannel - Network multiplexing and framing protocol for RPC
- Cap'n Proto - Think Protocol Buffers, except faster
- FlatBuffers - An efficient cross platform serialization library
- RSocket - Application protocol providing Reactive Streams semantics
- Twirp - A simple RPC framework with protobuf service definitions
- Greenpack - Serialization format similar to MessagePack, but adds field versioning and type annotation

Tools

CLI

- polyglot - A gRPC command line client written in Java
- grpcc - Node.js grpc command-line client
- gcall - Simple Node.js gRPC command line interface
- Evans - more expressive universal gRPC (CLI) client
- grpcurl - Like cURL, but for gRPC: Command-line tool for interacting with gRPC servers
- protodot - Transforming your .proto files into .dot files (and .svg, .png if you happen to have graphviz installed)
- grpc-client-cli - interactive gRPC client

GUI

- letmegrpc - Generate a web form gui from a grpc specification
- omgRPC (Deprecated) - A GUI client for interacting with gRPC services, similar to what Postman is for REST APIs
- grpcui - An interactive web UI for gRPC, along the lines of postman (also, a Go library for embedding these web UIs into Go HTTP servers)
- BloomRPC - A nice and simple GUI Client. Exploring and interacting with gRPC services has never been simpler, Inspired By GraphQL-Playground and Postman
- gRPCox - Like Postman, but for gRPC. web based GUI Client for gRPC, extremely easy to use.
- Milkman - Extensible alternative to Postman for crafting all kinds of requests, not only for gRPC, also http, sql etc.
- MuninRPC - Protobuf request and response testing application under the gRPC system.
- Delivery - A simple electron app for gRPC that uses gRPCurl to autodetect all endpoints/methods and their request bodies, just modify the JSON body. Simplicity in mind.
- (Yodelay.io) - A browser GUI Making sure your outbound 🤷 'yodelay' returns the 'IiiOoo' 📣 that you expect.

Testing

- ghz - Simple gRPC benchmarking and load testing tool inspired by hey and grpcurl
- gatling-grpc - A Gatling stress test plugin for gRPC.
- strest-grpc - A load tester for stress testing grpc intermediaries.
- hazana - A Go package for creating load test tooling. Supports gRPC.
- fortio - A microservices (http, grpc) load testing library and tool from Istio project.
- grpc-swagger - Debugging gRPC application with swagger-ui.
- grpc-tools - A suite of gRPC debugging tools. Like Fiddler/Charles but for gRPC.
- jmeter-grpc-plugin - A plugin supports load test gRPC service with Jmeter

Other

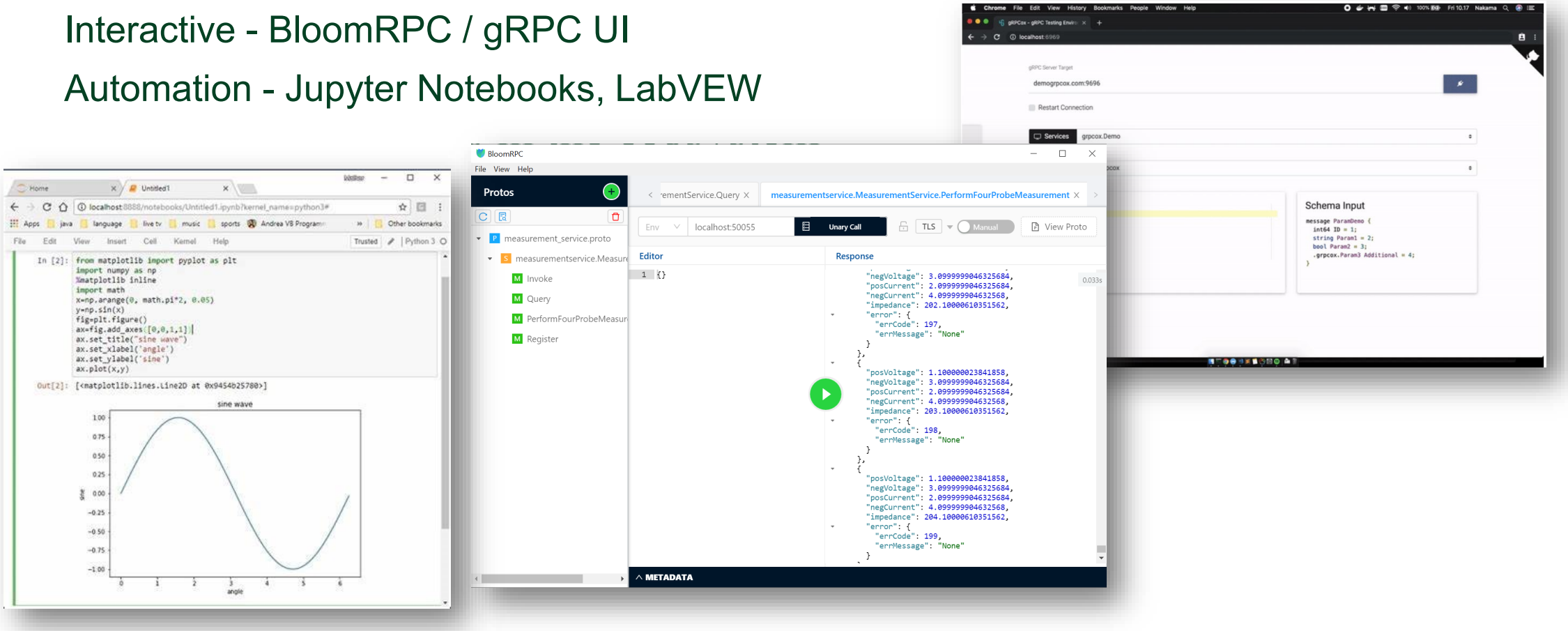
- kafka-pixy - gRPC/REST proxy for Kafka
- grpc-proxy - gRPC reverse proxy with the goal of making it easy to expose gRPC services over the internet
- ratelimit - Go/gRPC service designed to enable generic rate limit scenarios from different types of applications
- ProfaneDB - A Protocol Buffers database with gRPC API, built in C++ on top of RocksDB
- danby - A grpc proxy for the browser
- docker-protoc - Dockerized protoc, grpc-gateway, and grpc_cli commands bundled with Google API libraries
- grpc-json-proxy - A proxy which allows existing tools like Postman or curl to interact with gRPC servers
- protoc-gen-gotemplate - Generic generator based on golang's template system
- grpc-http-proxy - A reverse proxy server which translate JSON HTTP requests to gRPC calls based on protoreflect
- grpc-mate - A dynamic proxy server that translates JSON HTTP requests into gRPC calls
- jawlb - An unsophisticated grpclb load balancer implementation for Kubernetes and gRPC
- protoc-gen-hbs - Fast and easy protobuf generation with handlebars and some helpers

Interactive gRCP

Leverage open-source tools to enable web-based interactive use of devices

Many open-source tools to choose from

- Interactive - BloomRPC / gRPC UI
- Automation - Jupyter Notebooks, LabVEW

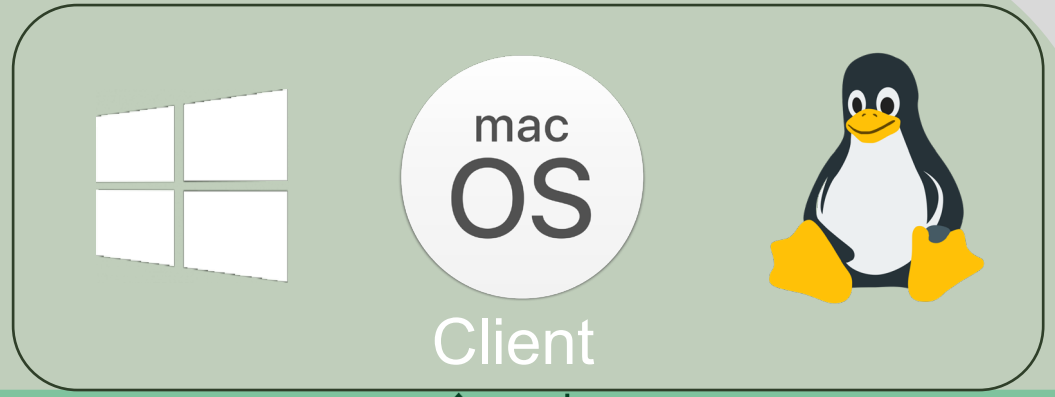


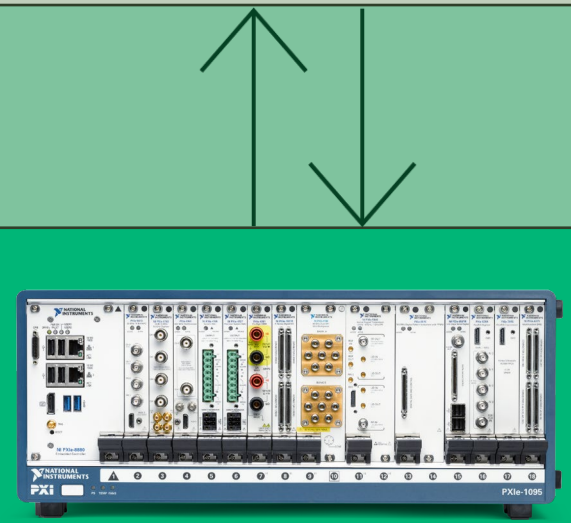
NI Remote-Ability

Test systems tailored to your workflow. From anywhere, with any language, on any OS

OS, Language Agnostic

Low Latency, Highly Scalable



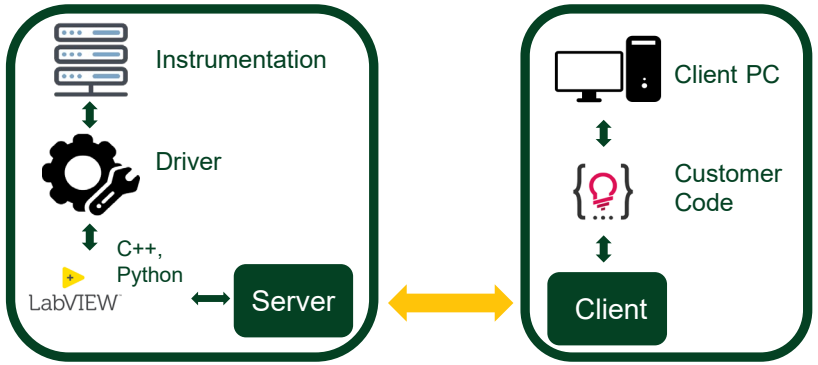


Benefits

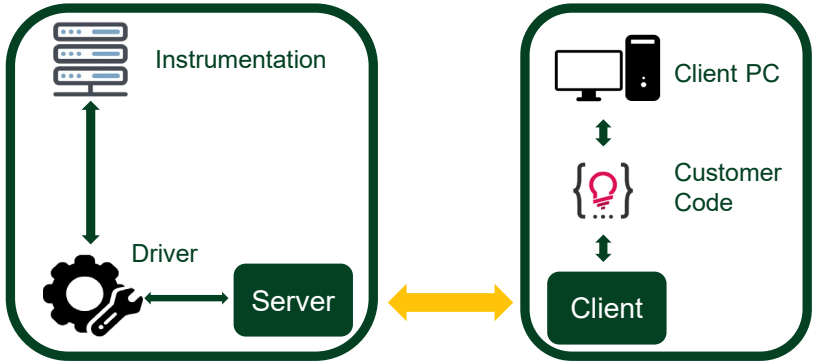
- Remotely control NI hardware and software
- Minimize time to first measurement
- Leverage existing workflows
- Avoid driver installation on client

Remote-ability Workflows

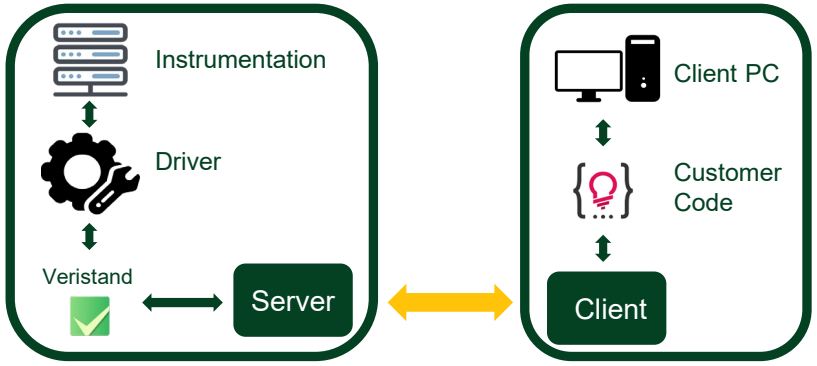
Workflow 1: Allow users to control test execution and pass data through LabVIEW between Tester and Client machine



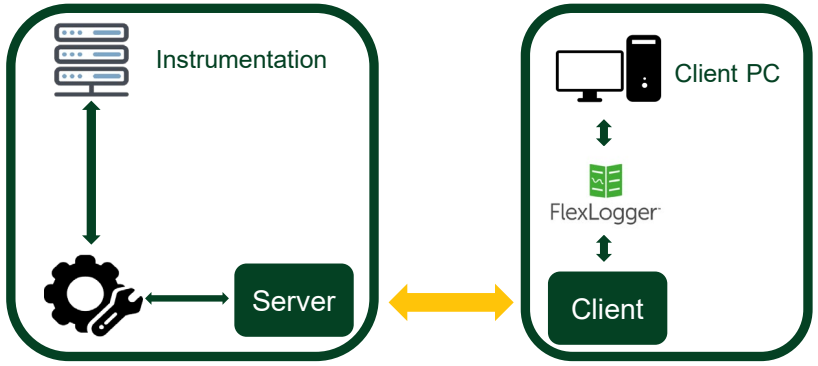
Workflow 2: Allow users to make driver calls remotely



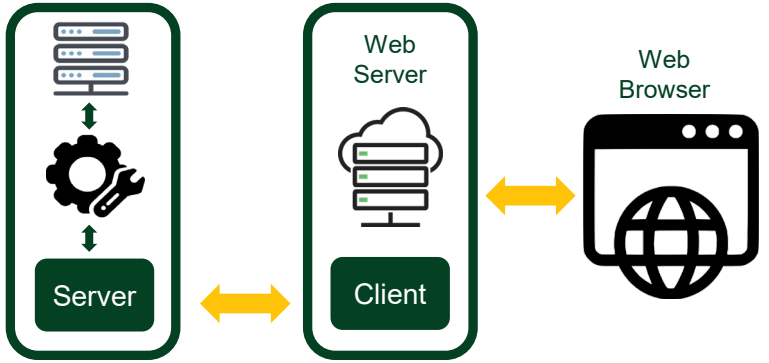
Workflow 3: Allow users to remotely control App Software



Workflow 4: Allow users to remotely control instrumentation from App Software



Workflow 5: Web-based instrument control



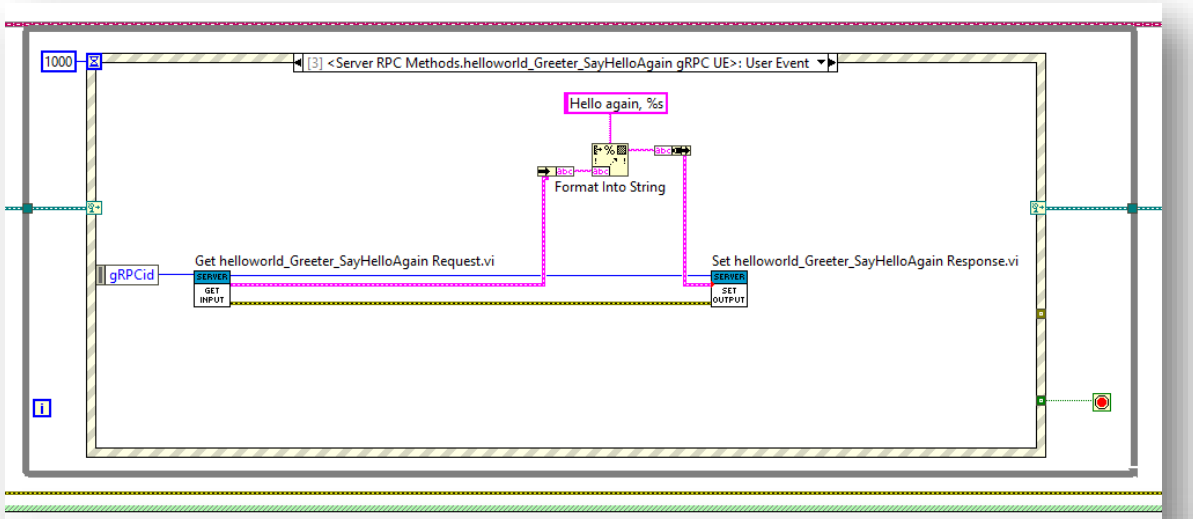
Two Current Workflows

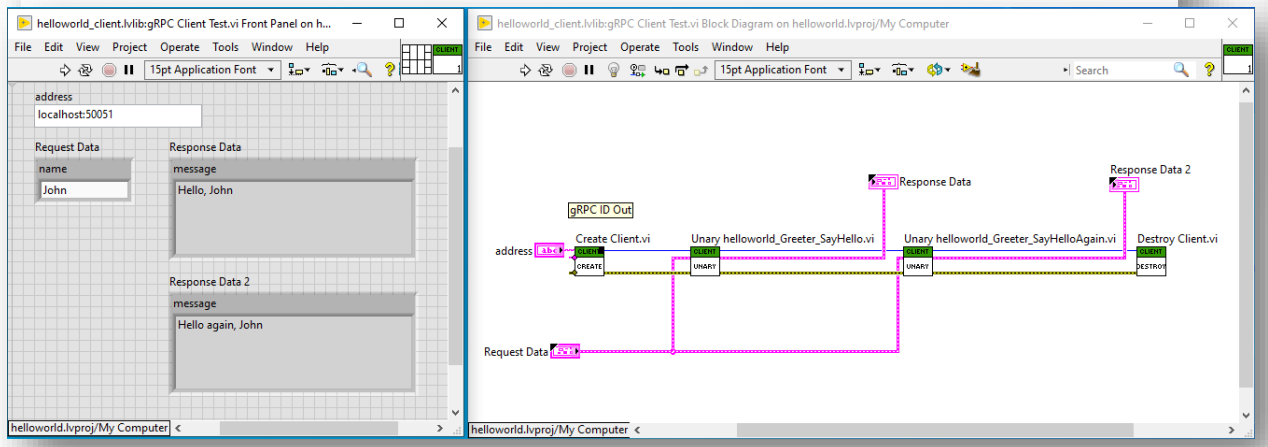
Device Driver Support	Workflow Name	LabVIEW Server Support
Customer can now control supported NI instruments remotely using their preferred language and operating system	**Description**	Enabling users to build custom remote-ability APIs on top of pre-existing LabVIEW applications
Allows instruments to fit into a wider variety of workflows, accessible from any OS and any language	**Why it matters**	This is a more efficient way to create a remote API and will enable users to call LabVIEW code from any application
• Centralized control of instrumentation • Group sharing of instrumentation	**Use Cases/ Examples**	• Remote VI interface

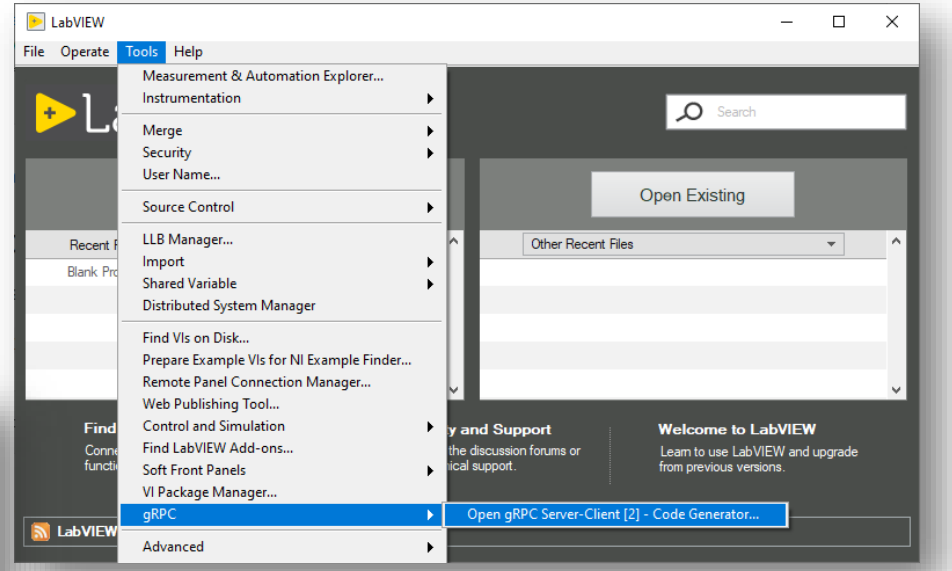
grpc-labview

gRPC Client and Server support for LabVIEW

https://github.com/ni/grpc-labview







Demo

labview-grpc

You Can Help

- labview-grpc is not yet 1.0
  - Client and server are stable
  - Still needs feature completion and code generation improvements

- You can help
  - Provide feedback on the server templates
  - Create examples and documentation
  - Create automated tests
  - File issues for any bugs you run into

grpc-device

Open source API for NI devices

https://github.com/ni/grpc-device

Enables:

- Remote access to device I/O
- Device I/O in containers
- Device I/O in VMs

Supported NI drivers

NI Driver	Minimum Version Tested (Windows)	Minimum Version Tested (Linux)	Minimum Version Tested (Linux RT)
NI-DAQmx	21.0.0	21.0.0	21.0.0
NI-DCPower	20.6.0	20.1.0	20.7.0
NI-Digital Pattern Driver	20.6.0	Not Supported	Not Supported
NI-DMM	20.0.0	20.1.0	20.5.0
NI-FGEN	20.0.0	Not Supported	Not Supported
NI-RFmx Bluetooth	21.5.0	Not Supported	Not Supported
NI-RFmx LTE	21.5.0	Not Supported	Not Supported
NI-RFmx NR	21.5.0	Not Supported	Not Supported
NI-RFmx SpecAn	21.5.0	Not Supported	Not Supported
NI-RFmx WLAN	21.5.0	Not Supported	Not Supported
NI-RFSA	21.0.0	21.0.0	Not Supported
NI-RFSG	21.0.0	21.0.0	Not Supported
NI-SCOPE	20.7.0	20.1.0	20.7.0
NI-SWITCH	20.0.0	20.1.0	20.5.0
NI-TClk	20.7.0	20.0.0	20.7.0

NI Remote-Ability Support & Compatibility

gRPC

LabVIEW

Server

mac OS

Client

Supported NI Drivers

- NI Scope
- NI Switch
- NI DCPower
- NI DMM
- NI TClk
- NI FGEN
- NI Digital Pattern
- NI DAQmx
- NI XNET
- RFMX
- RFSA
- RFSG

Supported OS

- Windows
- Linux Desktop
- Linux RT

Supported OS

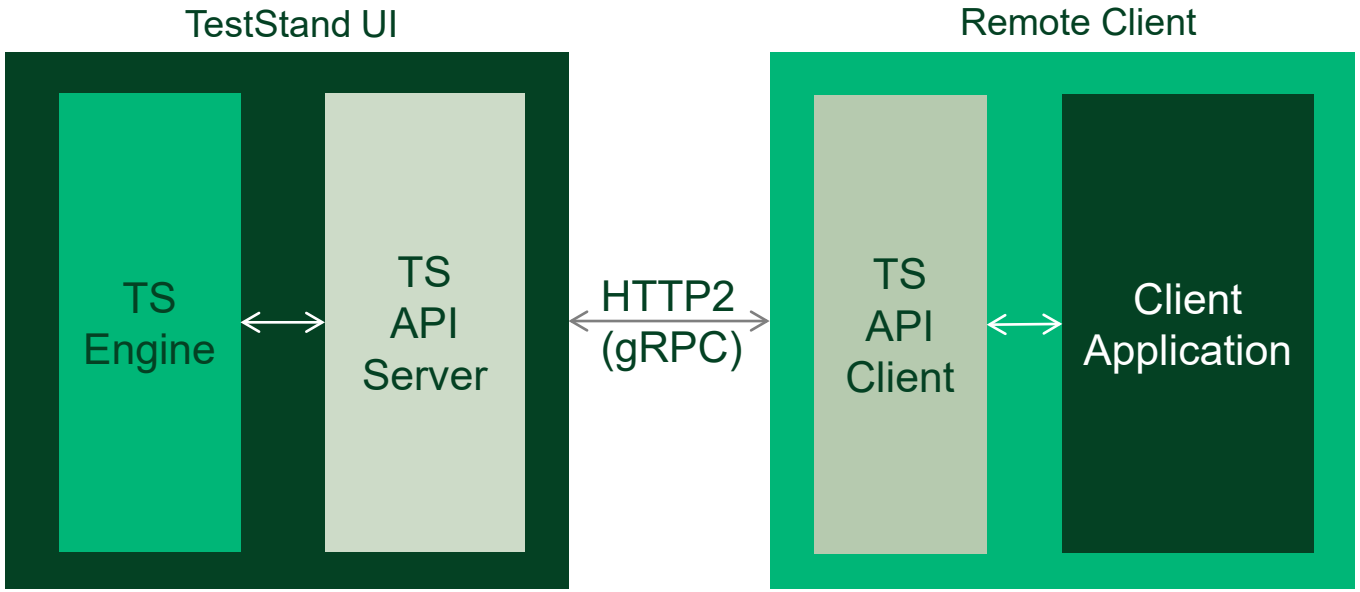
- Windows
- Linux RT
- Linux Desktop
- MacOS
- iOS
- Android
- …

Supported Languages

- Python
- C#
- C++
- LabVIEW
- Swift
- Go
- Java
- Kotlin
- Node
- Objective-C
- …

Demo

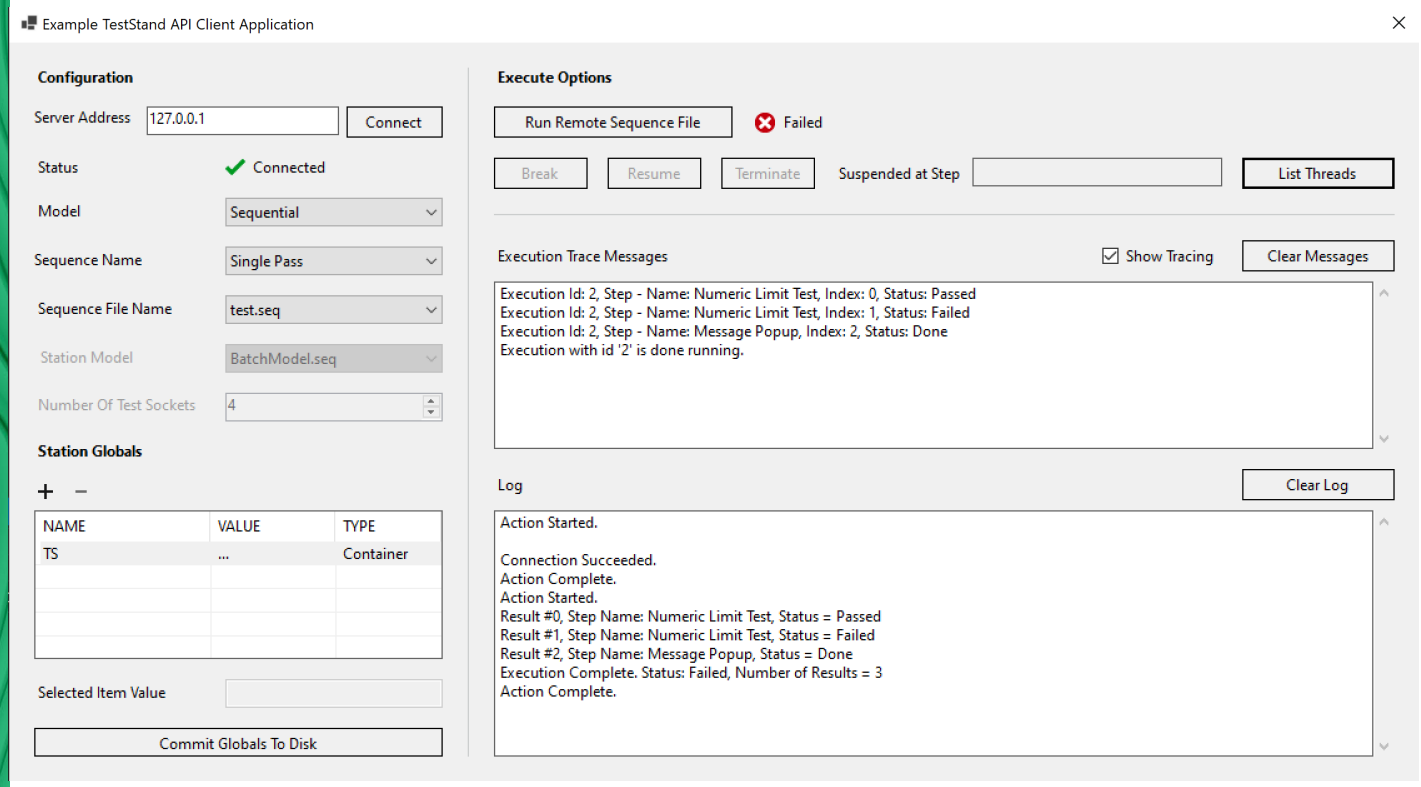
grpc-device



gRPC - TestStand

Early Access Coming Soon

API Example available as private Beta in early H2 2022



Connect remotely to server hosted in a TestStand UI

Select from a list of sequence files

Select process model and control execution

Use trace messages to understand execution status

Event support, headless server, and more will follow

Summary

grpc-device

- 16 drivers supported, more coming soon
- Robust and production ready

grpc-labview

- Ready for most use cases
- Full support coming soon

gRPC APIs coming to more NI products

Let us know what you think!

Q&A / Discussion