

gRPC

A shorter way to Microservice Architecture?

Balazs Nagy

Senior Field Applications Engineer at NI CLA, CLED, CTD

balazs.nagy@ni.com

Microservice Architecture – Intro
gRPC – Intro
gRPC – As architectural building block
gRPC – Use cases
gRPC – Use in LabVIEW

N



Microservice Architecture – Intro

או When you outgrow your monolithic architecture

Sometimes an application can outgrow its monolithic architecture and become an obstacle to rapid, frequent and reliable software delivery.

This typically happens when the application becomes large and complex and is developed by many teams. For example, its deployment pipeline become a bottleneck.

When this occurs, you should consider migrating to microservices.



ה What are microservices?

Microservices - also known as the <u>microservice</u> <u>architecture</u> - is an architectural style that structures an application as a collection of services that are:

Independently deployable
Loosely coupled
Organized around business capabilities
Owned by a small team

The microservice architecture enables an organization to deliver large, complex applications rapidly, frequently, reliably and sustainably - a necessity for competing and winning in today's world.





gRPC – Intro

GRPC Overview

gRPC is a Google-invented, open-source, remote procedure technology built for cloud process communication.

It offers many advantages such as:

- OS-Agnostic
- Language Agnostic
- Transport Layer Agnostic (ethernet, PCIe, Reflective Memory, etc.)
- High Bandwidth Streaming Enabled
- Secure (HTTP2)
- Open Source
- Can Directly Call Drivers or Application SW
- Works with native error handling



A high performance, open-source universal RPC framework







Request aRPC Stub Response Client gRPC Server Request LabVIEW C# C Service gRPC Stub Node C++ • Response(s) **Objective-C** Dart • PHP Go • Client 💽 Python Java •

Key Feature - Interoperability

- Communicate across . processes or systems
- Client and server can be on • different platforms and written in different languages
- Supported Languages ٠

•

•

•

.

Ruby Kotlin •

https://grpc.io/docs/languages/

ni.com



Key Feature - Cross-language

.proto file compiles to language of choice for client/server stubs, message definitions, etc.

.proto file compiles to different languages for client(s) and server



What is a Proto File?

The ProtoBuf interface describes the structure of the data to be sent. Payload structures are defined as "messages" in a .proto file.



Review the gRPC **Core concepts, architecture and lifecycle** document here <u>[https://grpc.io/docs/what-is-grpc/core-concepts/]</u> for more details.

High-Level Workflow of gRPC



gRPC – As architectural building block

gRPC – Use cases

gRPC – As architectural building block



INTERNAL - NI CONFIDENTIAL

ni.com



gRPC – Use cases

gRPC – Applications Examples

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

Workflow 2: Allow to make driver calls remotely.



■ gRPC – Detailed Applications Examples

LabVIEW Server Support





Customized GUI Server







Custom Developed UI aRPC Custom

code

Client

Examples:

Remote interface for IV Application

Benefits:

- Easily accessible through e.g. Python
- Will open the door for fully remote applications.



Examples:

Device Driver Support

Multiple instruments being controlled by a single test machine

Benefits:

- No drivers installed on test machine
- Can integrate NI instruments without rework



Examples:

Different groups are sharing the same equipment in a lab.

Benefits:

- Single Driver install on server
- Groups can use different languages and OS's

Examples:

gRPC Server is added to the Datalogger to expose a set of functionality to users.

Benefits:

- Light-weight client.
- Flexibility due to range of supported OS and Languages.
- Quick timeline turnaround due to minimal rework

ni.com

gRPC App. Ex. 1: NI gRPC Device Server and Client APIs

Use test systems from anywhere, with any language, on any OS





ni.com

NI Provided Components for gRPC Device Server and Client APIs



- gRPC Server
 - The gRPC Server is installed on the machine directly connected to the test instrumentation.
 - The gRPC Server provides remote clients with a link to the actual instrument drivers installed on the tester.
 - A single gRPC Server can support multiple instrument drivers.
- Remote gRPC API for each supported driver
 - NI will provide a gRPC .proto file that describes the API client machines can use in order to interact with their instrumentation. This API will mirror the C API of the driver.
 - The .proto file can be used to generate the remote API into any programming language you prefer.
 - Ex.) The .proto file for the NI Scope driver will allow you to use the remote NI Scope API in C#, Python, Go, and many other languages.
- Basic instrument discovery capability
 - User on client side will be able to programmatically query for instruments connected to the gRPC Server.

N

Current Support

- NI-DAQmx (in-progress)
- NI-DCPower
- NI-Digital Patter
- NI-DMM
- NI-FGEN
- NI-SCOPE
- NI-SWITCH
- NI-Sync
- NI-TClk

Future Support

- RF Drivers
- NI-XNet
- NI VRTS

ni.com





NI MeasurementLink, NI InstrumentStudio, NI TestStand

gRPC App. Ex. 2: Plug-ins Shared Between Applications





gRPC – Use in LabVIEW

Custom LV gRPC Server: Development Workflow

- 1. Download and Install the gRPC tools for LabVIEW: <u>https://github.com/ni/grpc-labview</u>
- 2. Define Messages for communication between Sever and Client
- 3. Create and Validate Protofile
- 4. Generate LV gRPC Server Interface
- 5. Generate Client Interface

- 6. Integrate LV gRPC Server Interface into existing LabVIEW Codebase
- 7. Integrate Client Interface into Client Codebase

Where to get gRPC for LabVIEW?

						gRPC Support for LabVIEW by NI - Toolkit for LabVIEW Downlos		
						This repo contains nece	ssary C++ code and support VIs to implement a gRPC server in LabVIEW.	
C 🛆 🖩 github.com/ni/gr	pc-labview							
Q Search or jump to	7 Pull requests Issues	7 Pull requests Issues Codespaces Marketplace Explore				* gRPC		-{
i / grpc-labview Public				Version	Version	0.4.0.0	Ver	
				Released	Released	Nov 09, 2021	0.4	
ode ⊙ Issues 🚮 🗋 Pu	Il requests 👔 💿 Actions 🖽 Projects	⑦ Security ∠ Insights				Publisher	NI	Put
	V master - V U houndher	D 39 taos	Go to file Add file	<> Code -	About	License	MIT License	- 12
		e maxer • • • • • • • • • • • • • • • • • • •		✓ e536d28 yesterday ③301 commits	gRPC client and server support for LabVIEW	LabVIEW Version Operating System	LabVIEW>=19.0 Windows, Mac, and Linux	Featured
	ni-sujain fix enum parsing for							
	github/workflows				gtpc labview	Project links	Homepage Documentation Repository	
	build-it				🖽 Readme	gRPC Support for LabVIEW		(
	docs	docs Updated QuickStart.md with updating I			母 MIT license	This repo contains necessary C++ code and support VIs to implement a gRPC server in LabVIEW.		
	i examples				Activity	Open examples/query_serve /examples/query_server/Prot	er/Query Server.lvproj for examples on creating a query server. tos/query server.proto defines a simple query service example that can be used for a variety of	
	labview source				15 watching	purposes.	· · · ·	
	src Src				♀ 47 forks	You can either use the servic	ce as defined to implement a generic server via gPRC or use the implementation as a pattern to	
	🖿 tests				Report repository	The project supports Window	n your design. ve Linux and Linux PT	
	third_party					The project supports window	s, Linux, and Linux KT.	
					Releases			
					⊘ 39 tags			
	CMakeLists.txt							
	CONTRIBUTING.md				Packages			
					No packages published			
	D aright i							

VIPM Q Discover LabVIEW tools.

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

https://www.vipm.io/package/ni_lib_grpc_labview/

ni.com

או

LabVIEW Server / Client code generation based on .proto file with Scripting Tool

gRPC Demo.lvproj * - Project Explorer	-		\times									
File Edit View Project Operate Tools Window	Help											
🌇 😂 😝 🗶 🖻 🗊 🗙 📾 🗸 🖗 🖼 🕶 🚰		s 🐮										
Items Files	- [11											
Project items												
🖨 🜉 My Computer												
Generated_server												
🖨 🕟 Generated_server.lvlib												
🖨 🔂 Custom_VIs.Ivlib												
🕂 💭 Controls												
E SubVis												
Continous_Async_Acquisition_VI.vi												
Generated_server.ivclass												
Accessors												
RPC Messages												
🗄 📁 End RPC Call												
🖶 💭 P_AcquistionRequest												
P_AcquistionResponse												
P_DataRequest_Multi												
P_DataRequest_Single												
P_Entrode												
P SingleDataResponse												
🕀 📁 P_StopRequest												
🗄 📁 P_StopResponse												
🖨 📁 RPC Methods												
P_Cont_Acq_Measure_Multi_D	ata											
P_Cont_Acq_Measure_Single_D)ata											
P_Cont_Acq_start_Acquisition												
Server API												
Server typeDefs												
L 💽 Main.vi												
Berndencies												
🗠 🛬 Build Specifications												



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

ni.com

Π



LV Server: Method VI

To implement the functionality when a gRPC client makes a request.

e.g. send a start acquiition request to the suitable thread, send response the client.



וח

•

•

•

•

•

Additional References

- NI gRPC-Device Repo
 - https://github.com/ni/grpc-device
 - gRPC Repo
 - <u>https://github.com/grpc/grpc</u>
 - NI gRPC Device Wiki:
 - <u>https://github.com/ni/grpc-device/wiki</u>
 - NI Device gRPC Server and Client API Wiki:
 - <u>https://github.com/ni/grpc-device/wiki</u>
 - gRPC Overview
 - <u>https://grpc.io/</u>
 - Proto language guide
 - https://developers.google.com/protocol-buffers/docs/proto3
 - Getting started with GoogleTest
 - http://google.github.io/googletest/

Thank you!

Questions?