FROM MONOLITH

TO

MICROSERVICE

Modernizing legacy codebases with grpc + go

(or gRPC for dummies)

Hi, I'm Cecy Correa

@cecycorrea
Software Engineer,
Context.IO


LEGACY CODE

gRPC to break up your monolith!

what is gRPC?

gRPC Remote Procedure Calls

oh ok

what is RPC?

Data exchange between 2 processes


TAKE DATA FROM ONE PLACE
PUSH IT SOMEWHERE ELSE
imgflip.com

so an API?

RPC or REST?

The “RPC” part stands for “remote procedure call,” and it’s essentially ***the same as calling a function in JavaScript, PHP, Python*** *and so on, taking a method name and arguments.*

Source: [Smashing Magazine]

RPC == good for actions

REST == CRUD
&
modeling your domain

[REST]

GET users/:id/photos/:photo_id

[RPC]

getUserPhotos(args...)

Why gRPC?

Why gRPC?

- Use HTTP/2

Why gRPC?

- Use HTTP/2
- Language agnostic

Why gRPC?

- Use HTTP/2
- Language agnostic
- Highly efficient / scalable

Why gRPC?

- Use HTTP/2
- Language agnostic
- Highly efficient / scalable
- Handle large amounts of data

HTTP2!

HTTP2!

HTTP2!

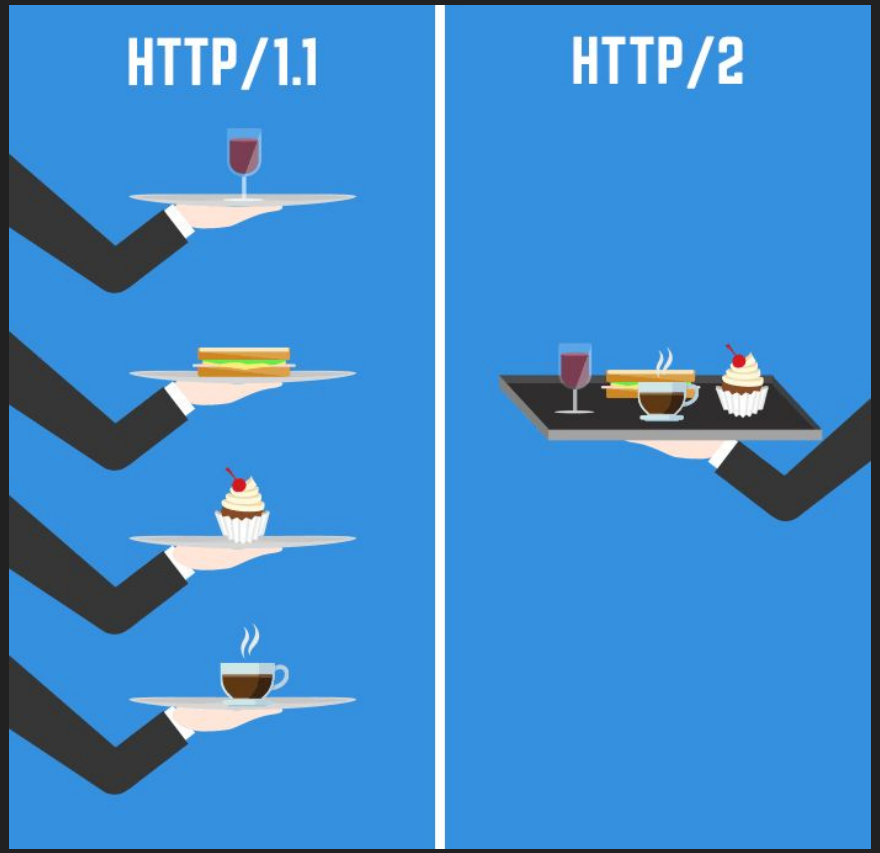
- Binary: moar data! Moar efficient!

HTTP2!

- Binary: moar data! Moar efficient!
- Multiplexed: multiple files at a time

HTTP2!

- Binary: moar data! Moar efficient!
- Multiplexed: multiple files at a time
- Same connection!



Source: [CSS Tricks]

Who is using it?

- Google
- Netflix
- Docker
- Square
- CoreOS
- Twitch (Twirp)


Pokémon
GO

Let’s get started!

gRPC + Protobuf

GRPC

The protocol

PROTOBUF

Define the service

Messages that are exchanged back and forth

What we will do

- Use protobuf to define your service
- Use protoc to generate client and server stubs based on your service definition
- Create a gRPC server to handle requests

Install all the things

go to grpc.io for options

Step 1: Create a proto file

Sample proto file

dad_jokes.proto

```
syntax = "proto3";

package DadJoke;

service DadJoke {
    rpc GetDadJoke (DadJokeRequest) returns (DadJokeResponse) {};
}

message DadJokeRequest {
    string keyword = 1;
}

message DadJokeResponse {
    string joke = 1;
}
```

This is a protobuf message that was generated while I was playing Pokemon GO. It's encoded with base64:


```
CAIYhICAgIDH9YMtIgIIAiICCH4iCwgEEgcI2uj5690qIgMIgQEiLggFEioKKDRhMmU5YmMzMz
BkYWU2MGU3Yjc0ZmM4NWI5ODg2OGFiNDcwMDgwMmUyqAQIBhKjBAqgBG7GaKm7Pw0q1Cc2QLtM
GMC5m4LGKR7jNoEgJhkOqyaUiU1vNExYDB7R0BanfBLTYKcTjgBSxpIj5kiDJj95L0+ri5SucL
sl1KRMuf/0N5A38gmJlKp/9KUhJ42J36NRCqtantd9bZw6r0VZq0/dH/GoK1xPx/lYi18NHlHM
BDdwB7sKh1LcQ3VRlPp9Se28SvG18kFrGXMi9W7U1HcWWsACtv7og3gSf1GVXyyA5C70y0BdOq
O7WP0I8cJjZ6i6W2fI+6CfBBxZMB+MNNIPdAW49dDitKk1cts/aHdcMnMjobLGaYye99nT25CC
mGaMyHl9KbRyu6HvwMBUQGu2qvmsZRvDoWFw33QaskRhB987DF5p7KeN12kODt7LAtZmnyHvzh
QT1qbolqKpQCwhZpdPGRkDTMoPVhzWOkoNhnQab1n8gK7GiHA5mWEBU4JKMvkRpi7wzciAiuTk
s1FFKT5ywRChW+TOJRZ0aNVyhJ8yg9yZABoe42rTo30mwL2Q3qG6oShyyrNPg9b1MrXwf+LcBz
QkGnC/RiGOwybWHuPCC5uP5PWqIIoDJP3ArGZMrNJwrEYK/aTlj7eAZEH/PO+VvlZkzdurBvKf
jb6sNO/z0POrzgR08FqqKGjIOcxysVbHlbs7vufiH6rIDKqnQh0Xm/UX/KApQCGQX9qAQ5P+yt
skrLmVEzrCPS42DqJRN7ACsJU5UOVfeGqz0RJMp6w5AAAAwGnmQsBBAAAAwNkeYkBJAAAA4Gg3
VkBaWwpAjak/Co8vA9rnxeP5tKpOApMdlWxWyeYUJyIftyjJ6EdIEotI0ueUrZQkcOCoR10J+B
xoocpdD6o4RxwTT7OcRBDD1+fs3SoaELr5MxmdQYKC7HQaxg5tat5gnzY=
```

```
1: 2
3: 3244797592550244356
4 {
  1: 2
}
4 {
  1: 126
}
4 {
  1: 4
  2 {
    1: 1468299899994
  }
}
4 {
  1: 129
}
4 {
  1: 5
  2 {
    1: "4a2e9bc330dae60e7b74fc85b98868ab4700802e"
  }
}
6 {
  1: 6
  2 {
    1: "n\306h\251\273?\r*\324\'6@\273L\030\300\271\233\202\306)\036..."
  }
}
```

Step 2: Create your server / client stubs

```
protoc -I proto --php_out=lib --grpc_out=lib
--plugin=protoc-gen-grpc=`which
grpc_php_plugin` service.proto
```

```
protoc -I DIR --LANG_out=DIR --grpc_out=DIR
      --plugin=protoc-gen-grpc=`which
grpc_LANG_plugin` WHERE_YOUR_PROTO_IS.proto
```

```
project-dir
├── client
|   └── client.go|js|rb|py ...
├── proto
|   └── service.proto
├── server
|   └── server.go|js|rb|py ...
└── libs
    └── ruby

    └── python

    └── ...
```

Step 3. Create a gRPC server

main.go

```go

const (
    port        = ":50051"
    placeholder = "How much does a hipster weigh? An instagram."
)

type server struct{}

func (s *server) GetDadJoke(c context.Context, req *pb.DadJokeRequest) (*pb.DadJokeResponse, error) {
    joke := &pb.DadJokeResponse{Joke: placeholder}
    return joke, nil
}

func main() {
    log.Println("starting DadJoke server🤓")
    lis, err := net.Listen("tcp", port)
    if err != nil {
        log.Println(err)
    }
    s := grpc.NewServer()
    pb.RegisterDadJokeServer(s, &server{})
    reflection.Register(s)
    if err := s.Serve(lis); err != nil {
        log.Fatalf("failed to serve: %s", err)
    }
}
```

Sample proto file

dad_jokes.proto

```
syntax = "proto3";

package DadJoke;

service DadJoke {
    rpc GetDadJoke (DadJokeRequest) returns (DadJokeResponse) {};
}

message DadJokeRequest {
    string keyword = 1;
}

message DadJokeResponse {
    string joke = 1;
}
```

Step 4: Write some clients!

dadJokeClient.php

```php
#!/usr/bin/env php
<?php

require dirname(__FILE__).'/vendor/autoload.php';

echo "***dad joke PHP client example***";

$client = new DadJoke\DadJokeClient('localhost:50051', [
    'credentials' => Grpc\ChannelCredentials::createInsecure(),
]);

$joke = new DadJoke\DadJokeRequest();

list($res, $status) = $client->GetDadJoke($joke)->wait();

echo "\n";
echo $res->getJoke();
echo "\n";
```

*Note: you will need to install some stuff to get started in PHP

- composer install grpc as a dependency
- pecl install grpc / protobuf
- Full instructions on getting started: https://grpc.io/docs/quickstart/php.html

dadJokeClient.php

```php
#!/usr/bin/env php
<?php

require dirname(__FILE__).'/vendor/autoload.php';

echo "***dad joke PHP client example***";

$client = new DadJoke\DadJokeClient('localhost:50051', [
    'credentials' => Grpc\ChannelCredentials::createInsecure(),
]);

$joke = new DadJoke\DadJokeRequest();

list($res, $status) = $client->GetDadJoke($joke)->wait();

echo "\n";
echo $res->getJoke();
echo "\n";
```

Now in Ruby just for the heck of it

dad_joke_client.rb

```ruby
# do some crazy stuff to load the lib...
this_dir = File.expand_path(File.dirname(__FILE__))
path = this_dir.split("/")
path.pop
lib_dir = path.join("/") + "/libs/ruby"
$LOAD_PATH.unshift(lib_dir) unless $LOAD_PATH.include?(lib_dir)

require 'grpc'
require 'dad_jokes_services_pb'

def main
    puts "***dad joke Ruby client example***"
    stub = DadJoke::DadJoke::Stub.new('localhost:50051', :this_channel_is_insecure)
    res = stub.get_dad_joke(DadJoke::DadJokeRequest.new())
    puts res.joke
end

main
```

Let’s see it in action!

[pray to demo gods]

Use case: Context.IO

context.io

accounts | explore | logs | settings | documentation | privacy | sign out

2.0 | lite | app

- users
- users/<id>
  - connect_tokens
  - connect_tokens/<token>
  - email_accounts
  - email_accounts/<label>
    - folders
    - folders/<folder>
      - **messages**
      - messages/<message_id>
        - attachments
        - attachments/<attachmer
        - body
        - flags
        - headers
        - raw
        - read
    - connect_tokens
    - connect_tokens/<token>
  - webhooks
  - webhooks/<webhook_id>
- connect_tokens
- connect_tokens/<token>
- discovery
- oauth_providers
- oauth_providers/<key>
- webhooks
- webhooks/<webhook_id>

GET | POST | DELETE | PUT

https://api.context.io/lite/users/ id /email_accounts/ label /folders/ folder /messages

send

id:		Unique id of a user accessible through your API key
label:		The `label` property of the email account instance. You can use `0` as an alias for the first email account of a user.
folder:		The full folder path using / as the path hierarchy delimiter.

optional parameters

delimiter:		string If `/` isn't fancy enough as a hierarchy delimiter when specifying the folder you want to obtain, you're free to use what you want, just make sure you set this `delimiter` parameter to tell us what you're using.
subject:		string Get messages whose subject matches this search string. To use regular expressions instead of simple string matching, make sure the string starts and ends with `/`.
to:		string Email address of a contact messages have been sent to.
from:		string Email address of a contact messages have been received from.
cc:		string Email address of a contact CC'ed on the messages.
bcc:		string Email address of a contact BCC'ed on the messages.
date_before:		string (mm/dd/yyyy) Only include messages before a given date (mm/dd/yyyy). Messages whose internal date (disregarding time and timezone) is earlier than the specified date.
date_after:		string (mm/dd/yyyy) Only include messages after a given date (mm/dd/yyyy). Messages whose internal date (disregarding time and timezone) is earlier than the specified date.
include_body:		integer Set to `1` to include message bodies in the result.

Pain points

- Legacy codebase (~10 years old)
- Monolith codebase
- Heavily coupled == hard to unit test
- Scaling problems

How gRPC helped

- Replace small pieces of functionality with microservices
- Easy to do by “resource”

From REST to microservice

GET /discovery	Discovery service
GET /user/:id/contacts	Contacts service
GET | POST /user/:id/webhooks GET | POST /webhooks	Webhooks service
GET /user/:id/messages	Messages services
GET /user/:id/folders	Folder service

Our PHP API became the frontend for a microservice ecosystem

The brave new future:
Completely generated client libraries & docs

This solves...

- Feature parity between API and client libraries
- Feature parity between API and docs
- Lack of knowledge in the team around certain languages
- Easy for other teams to consume your service

...and
gRPC can also generate REST stubs!

So you can gRPC -> REST -> gRPC

grpc-gateway

```
 syntax = "proto3";
 package example;
+
+import "google/api/annotations.proto";
+
 message StringMessage {
   string value = 1;
 }

 service YourService {
-  rpc Echo(StringMessage) returns (StringMessage) {}
+  rpc Echo(StringMessage) returns (StringMessage) {
+    option (google.api.http) = {
+      post: "/v1/example/echo"
+      body: "*"
+    };
+  }
 }
```

Generate stubs + gateway

```
protoc -I/usr/local/include -I. \
  -I$GOPATH/src \
  -I$GOPATH/src/github.com/grpc-ecosystem/grpc-gateway/third_party/googleapis \
  --grpc-gateway_out=logtostderr=true:. \
  path/to/your_service.proto
```

path/to/your_service.pb.gw.go

What does our API look like now?

Monolith -> Monorepo

godoc Reference

Godzilla

"You have your fear, which might become reality; and you have Godzilla, which is reality." - Lt. Hideto Ogata

Godzilla is a monorepo for our Go code and its dependencies. We use bazel for running tests and building docker images.

Some advantages of a monorepo are:

- Standardized process and tooling for testing, deploying and managing dependencies.
- Collaboration and code-sharing across teams.
- Easy to make atomic changes across projects and libraries.
- Single location for all projects.
- Avoids reinventing the wheel.

Questions? Visit #godzilla in slack or use the godzilla google group.

accounts	more mongo changes	20 hours ago
adi	more mongo changes	20 hours ago
apollo	apollo: subscription table schema, store protobuf (#4274)	2 days ago
ci-router	Remove services/ci-router/cmd/dump-parse-templates/ and services/vpa/...	a month ago
cio-account-status	cio-account-status: Ignore paribus accounts (#4017)	a month ago
cio-admin-api	CON-985 cio opt out search (#4265)	2 days ago
cio-contacts	Contacts: use right endpoint (#4282)	20 hours ago
cio-discovery	Add force flag to helm upgrade command (#4013)	a month ago
cio-events	Add force flag to helm upgrade command (#4013)	a month ago
cio-grpc-sandbox	*: replace deprecated grpc.{Code,Errorf} with status.{Code,Errorf} (#...	2 months ago
cio-internal-api	move 2 services to new mongo config	5 days ago
cio-kubelove	Updating with new outlook proto (#3528)	3 months ago
cio-oauth-gateway	Gateway: port (#4288)	18 hours ago
cio-opt-out	CON-985 cio opt out search (#4265)	2 days ago
cio-webhooks-api	move more things to new mongo strings	4 days ago
cio-webhooks	move more things to new mongo strings	4 days ago
email-sender	Add force flag to helm upgrade command (#4013)	a month ago
esp_service	sfmc: sftp connection error handling (#4283)	19 hours ago
geolocate	Add force flag to helm upgrade command (#4013)	a month ago

Cthulhu: Organizing Go Code in a Scalable Repo

Matt Layher on Engineering • October 10, 2017 • 27 Comments

[CODE]
github.com/cecyc/dad-joke-service

Questions?
Happy to talk in the hall!

or ping @cecycorrea on Twitter