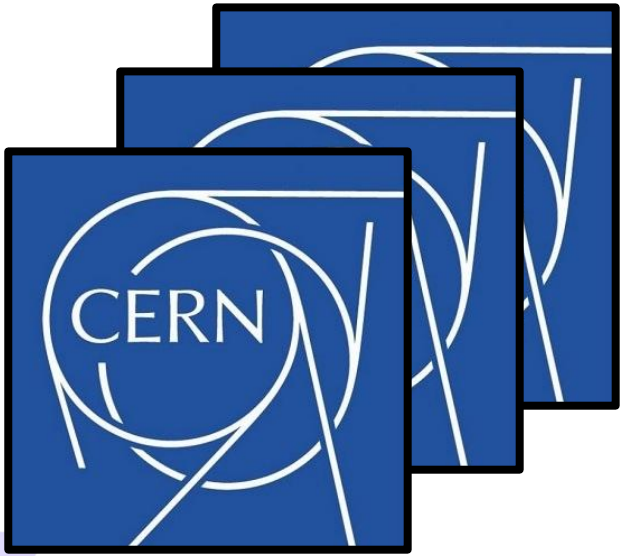
The Life of an Open-Source Project

David Garcia Quintas
Xoogler, gRPC C Core Team
dgquintas@gmail.com

How I Got Here



GRPC

Agenda

- **Why** develop [a new project] in open-source.
- **What** is being developed: gRPC.
- **How** is it being developed: the process.

A sampling of OSS at Google

A sampling of OSS at Google

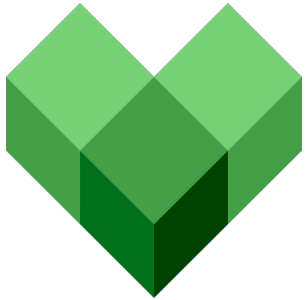












GRPC

Why?

- **Google has had 4 generations of internal RPC systems, called Stubby**
  - All production applications and systems built using microservices connected by RPCs
  - Over 10^{10} RPCs per second, fleetwide
  - APIs for C++, Java, Python, Go
  - Not suitable for open-source community (Tight coupling with internal tools)
  - Not suitable for open Internet (proprietary wire protocol won't work with firewalls, etc)

What?

- **g**RPC **R**emote **P**rocedure **C**alls.
- High performance, open source, general purpose, standards-based, feature-rich RPC framework.
  - **HTTP-2** based transport
  - Standards-based flow-control, auth, LB, etc.

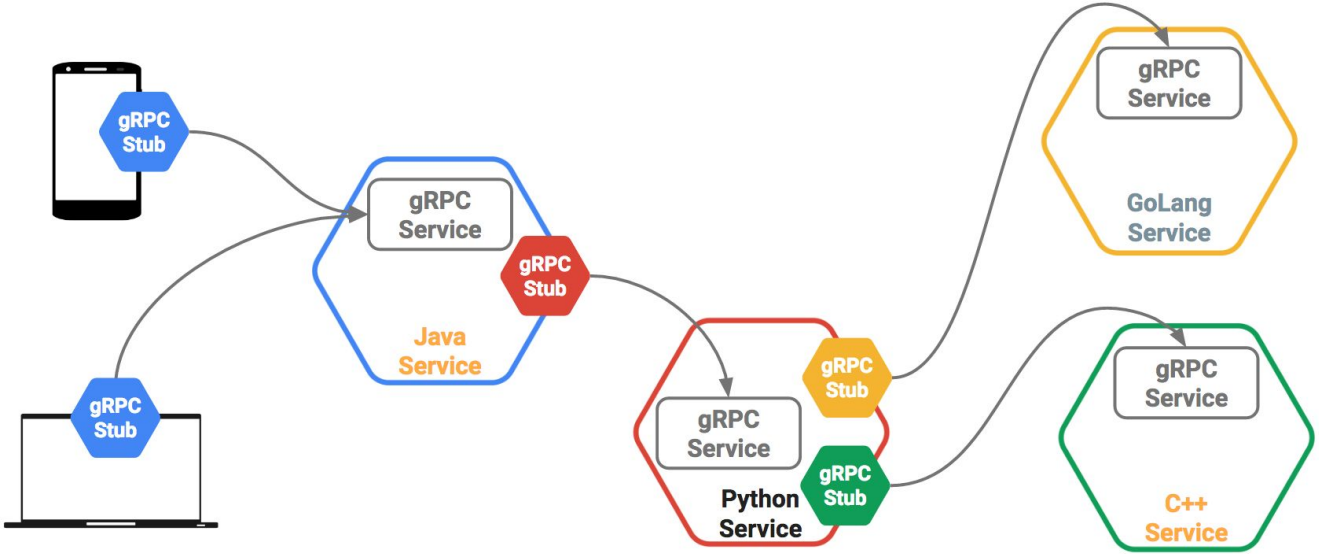
- Developed by Google, donated to CNCF
  - Development is Github-first
  - github.com/grpc

Service definitions and client libraries

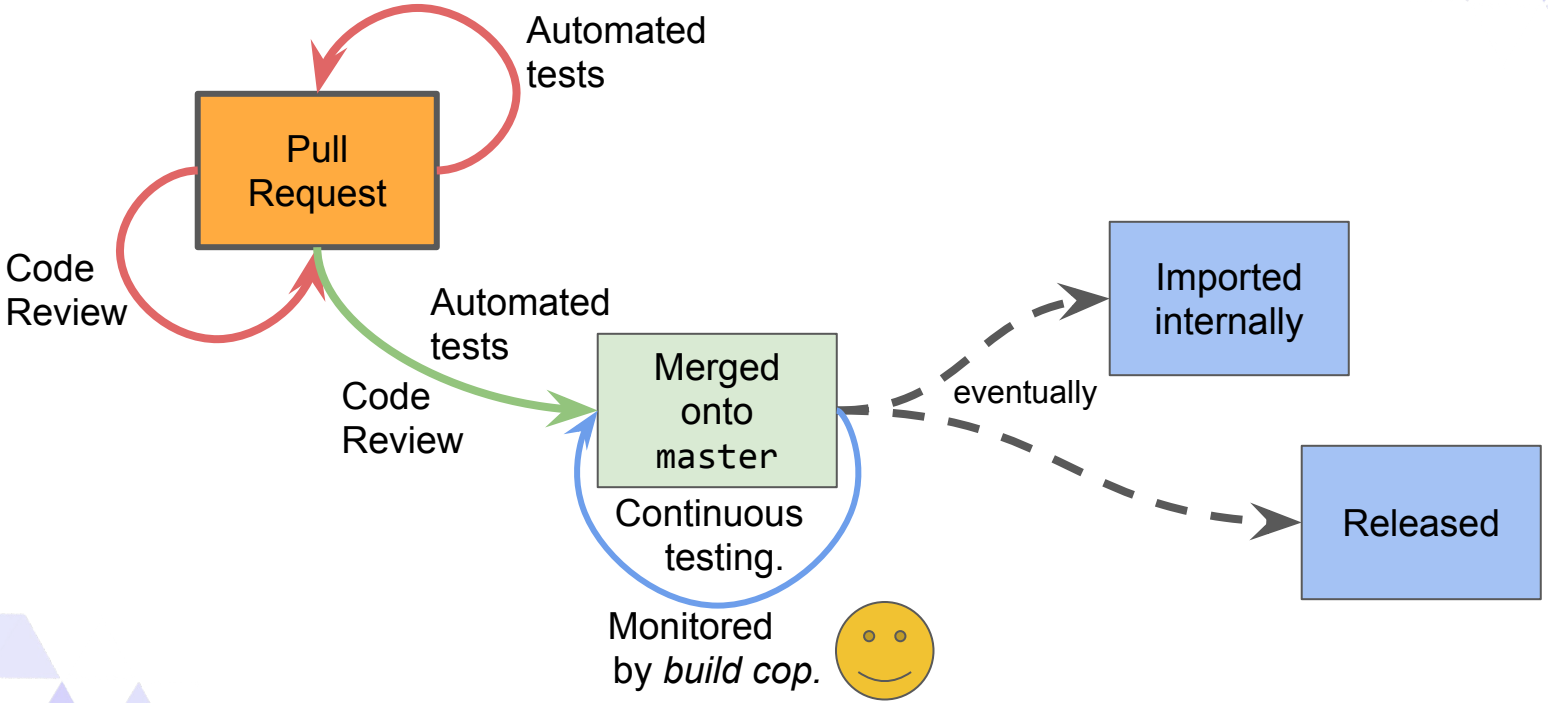
- Java
- Go
- C/C++
- C#
- Node.js
- PHP
- Ruby
- Python
- Objective-C

More Languages...

- Swift
- Haskell
- Rust
- Typescript
- ….

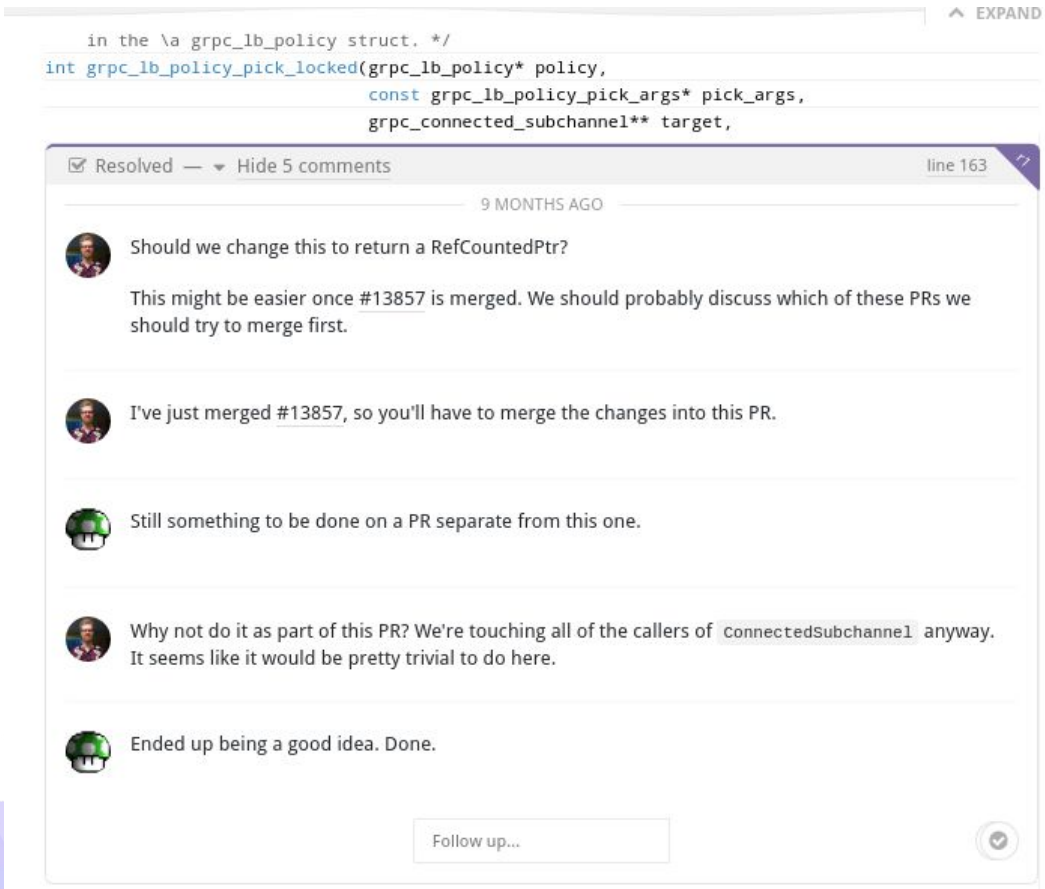


Development Process




RESPECT THE PROCESS

Development Process: **Code Review**



Development Process: **Testing**

Development Process: **Testing**

Tens of thousands of tests run **per PR** *and* **continuously on `master`**.

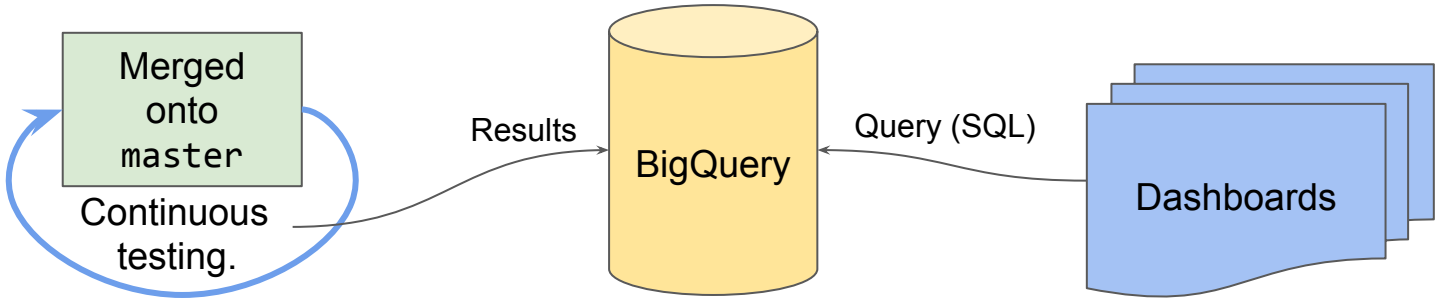
`{windows, mac, linux} x {bazel, make} x`
`{asan, msan, tsan, ubsan} x {opt, dbg} x`
`{C Core tests, C++ tests}`

+

`microbenchmarks + binary size`
`+ {Python, Ruby, iOS, PHP, C#, Node.js}`
`+ interop (all client/server pairs of Java, Go,`
`C-wrapping languages).`

Development Process: **Build Cop**

Monitors increases in test flakiness + performance from the master CI runs

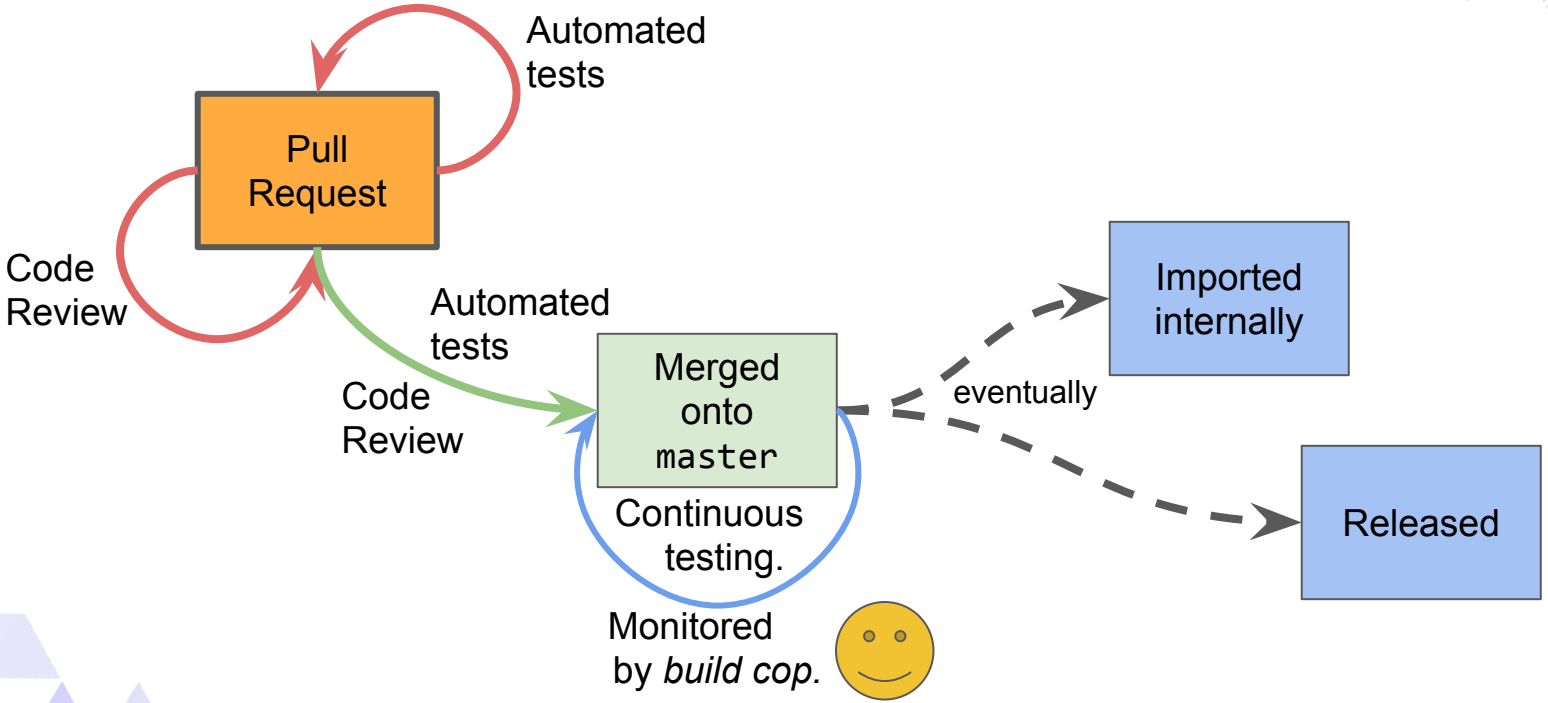


Total Failures over last 24 hours (sorted by %, then by time) (use go/grpc-test-failure-history for individual test history)

test_binary	pct_failed	job_name	latest_failing_build_id	latest_timestamp_MT
tools/run_tests/helper_scripts/run_grpc-node.sh	40	grpc/core/master/macos/grpc_basictests_dbg	[2596, 2594, 2593, 2588]	2018-10-29 19:41:23-(
tools/run_tests/helper_scripts/run_grpc-node.sh	37.5	grpc/core/master/macos/grpc_basictests_opt	[2096, 2094, 2092]	2018-10-29 18:42:44-(
objc-tests	14.29	grpc/core/master/macos/grpc_basictests_opt	[2092]	2018-10-29 10:48:17-(
h2_full_test hpack_size	10	grpc/core/master/macos/grpc_basictests_dbg	[2590]	2018-10-29 09:04:07-(
py27_native.test.unit._channel_ready_future_test.ChannelReadyFutureTest	5.88	grpc/core/master/linux/grpc_basictests_multilang	[3075]	2018-10-30 02:33:19-(
client_lb_end2end_test --gtest_filter=SubchannelForceCreation/ClientLbEnd2endWithParamTest.PickFirstManyUpdates	1.79	grpc/core/master/linux/sanitizer/grpc_cpp_tsan	[1251]	2018-10-30 04:37:39-(
end2end_test --gtest_filter=End2end/End2endTest.ClientCancelsRequestStream	0.83	grpc/core/master/linux/sanitizer/grpc_cpp_asan	[1634]	2018-10-29 12:18:17-(

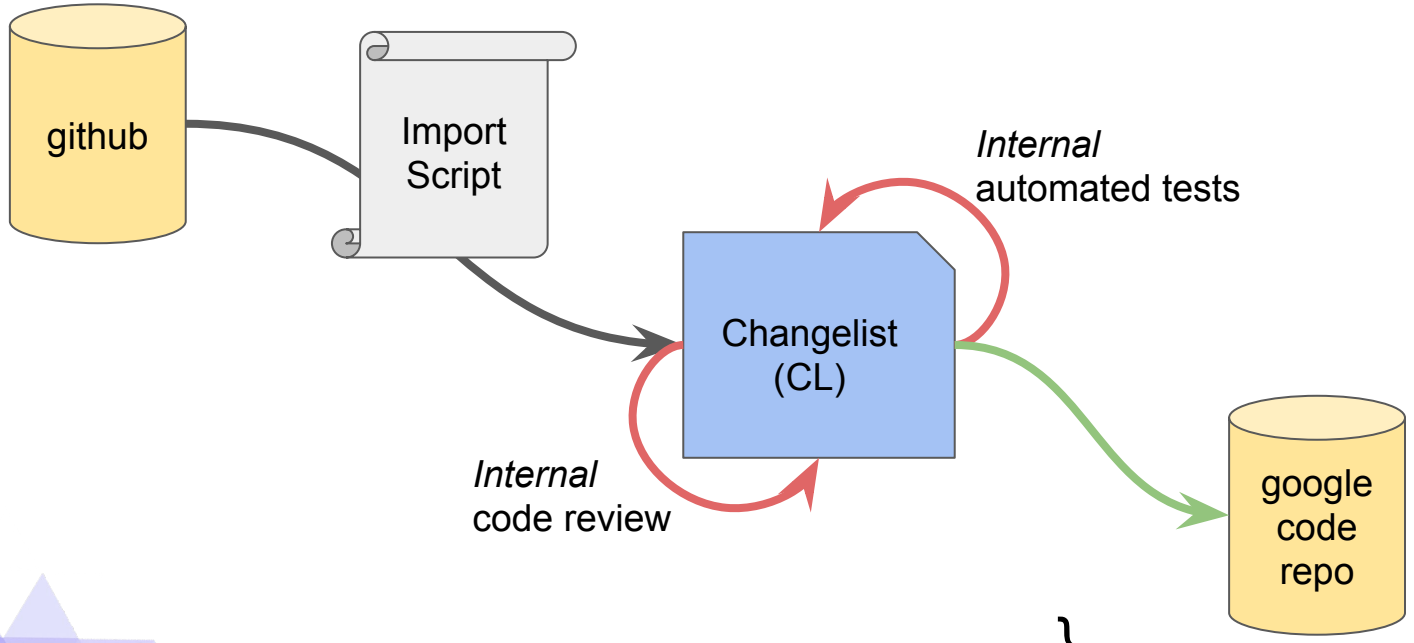
Development Process: **Build Cop**

Development Process



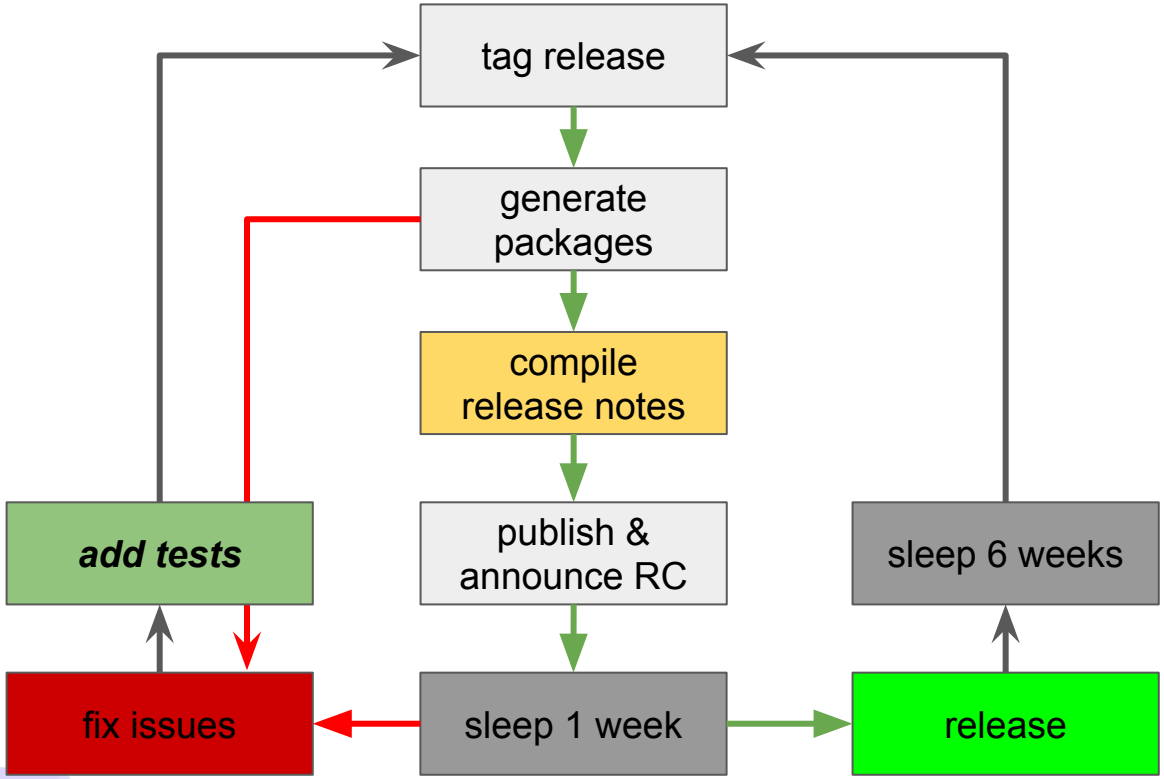
Development Process: **Internal Import**

```
while (next importer in rotation) {
```

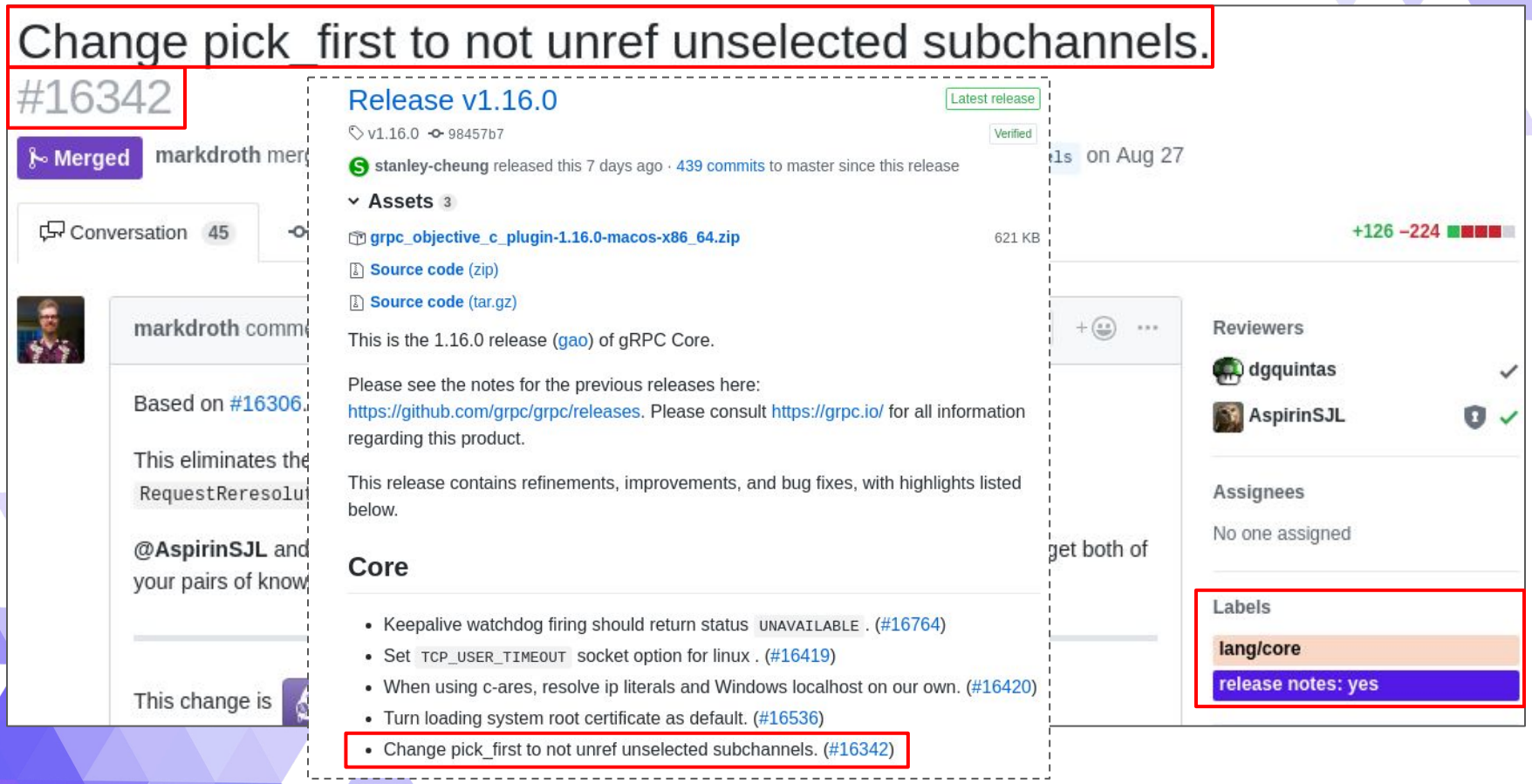



```
}
```

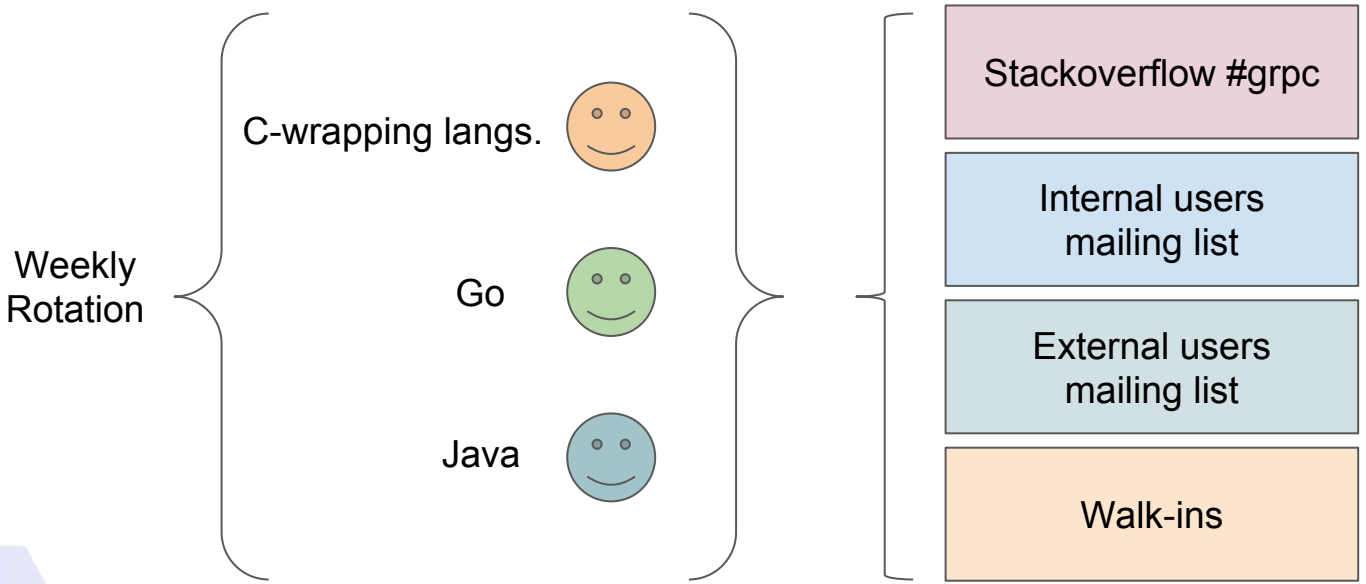
Development Process: **Releases**



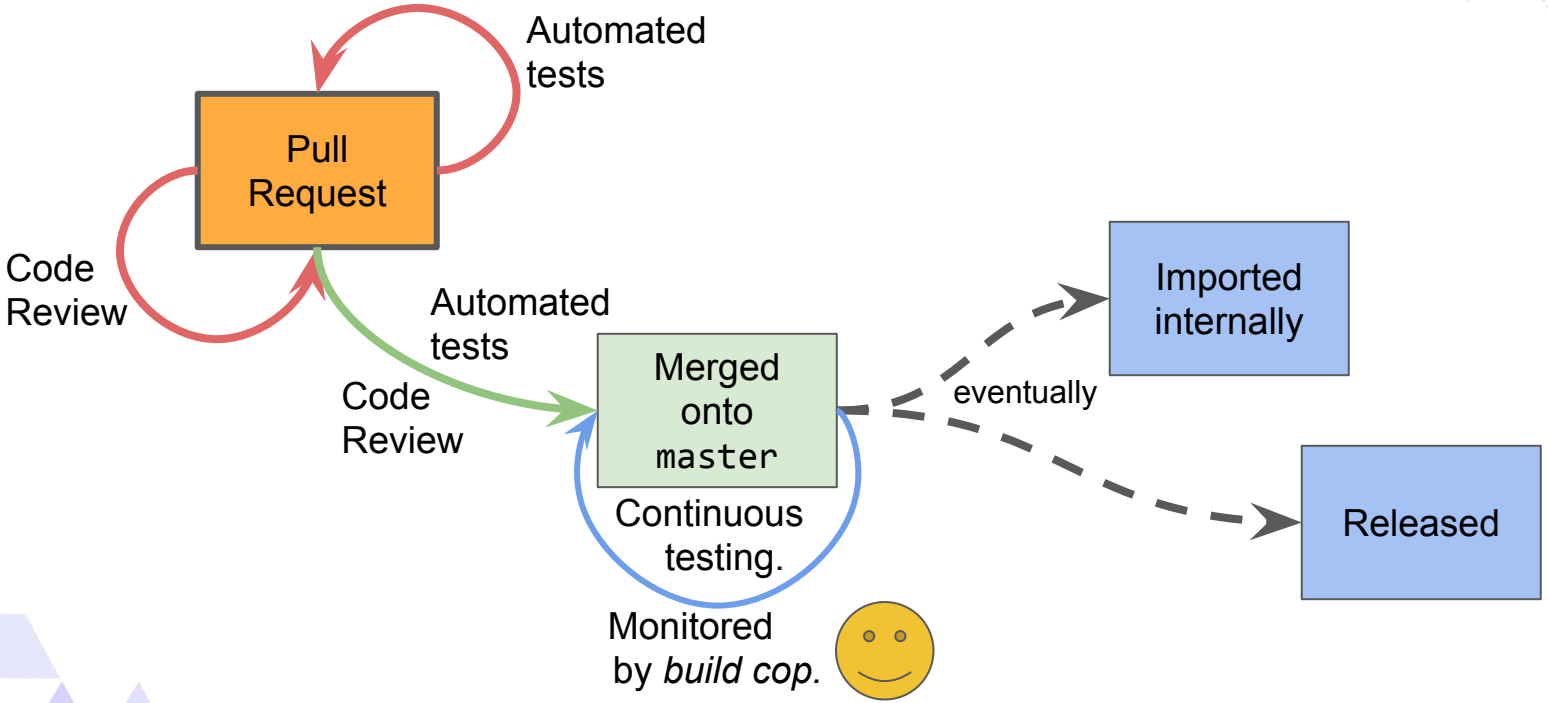
Development Process: **Releases**/Release Notes



Development Process: **Office Hours**



Development Process



Keeping the Community Engaged

- Foster understanding between devs and community
  - Set **expectations**
- Establish community **Meetups**
  - Place for community to meet with team, put faces to names, **press strongly for feature requests, bug fixes**, etc
  - Showcase for **interesting** and often **unexpected use cases**
- Establish processes for making changes
  - **gRFC** - seek **community input** for 2 weeks before converging on new features
- Improve documentation and empower community

An empowered community

From REST to gRPC:
An API Evolution Story

Joe Runde
IBM
@joerunde

Michael Keeling
IBM
@michaelkeeling

Why is gRPC Awesome?

Performance
Remote Procedure Calls
Strategic Direction of our Platform

Would we do it again?

Yes.

- Super easy to integrate with a service
- Promotes small polyglot services
- Difficult to do bad things
- Performance is 🔥 🔥 🔥

Adoption

Microservices: in data centers

Client Server comm. / Internal APIs

Streaming telemetry from network devices

Mobile Apps

What [IMO] makes a great OSS project

- Valuable software role **for** the community
- Clean **licensing** for maximum utility
- Is **well-supported**, **managed**, and **maintained**. Signal **commitment**.
- Code **quality** and **testing**
- Continuous improvement
- Prompt and **predictable release cycle**
- Easy to **get started**: Installation, documentation
- **Feedback from users** and understanding of their use cases

All of which facilitate an **engaged** and vibrant community

What [IMO] is probably not important

- Code contributor population
  - Most users of open-source never actually open the source
  - Being open-source > just free: community can extend & self-sustain

- Development model
  - OSS-first with internal uptake
  - Internal-first with external releases

In Summary

OSS development is harder

- The world is a big place, more systems/configs.
- Increased overhead from community.
- Restricted to external tools...
- ... or you'll need to roll your own.

OSS development is harder, **but...**

- ✓ Improves org's PR/goodwill.
- ✓ Attracts talent.
- ✓ External scrutiny: keeps you honest.
- ✓ Gateway to the organization.
- ✓ Enables contributions.

Thank you!


CERN
C-092927 -1
Valide du 12/07/2005
au 02/09/2005
Prénom
David
Nom
Garcia Quintas