APERICODER https://montacchiello.dev

Montacchiello
Cittadella dell'Innovazione

gRPC: Nei microservizi quando usarlo (e quando no)

Connessioni moderne, efficienti e tipizzate

Autore: Luca Bigoni

Data: 6 Aprile 2026

Da infrastruttura interna a standard open source

Le origini (2001)

Stubby (Google Internal)

Utilizzato internamente da Google per oltre 15 anni per connettere la propria immensa rete di data center.

L'Evoluzione (2015)

Open Source

Rilasciato al pubblico nel 2015 come framework RPC universale ad alte prestazioni.

Lo Standard (Oggi)

Donato alla CNCF da Google nel 2017. CNCF Incubation.

Oggi è un progetto cardine della Cloud Native Computing Foundation.

gRPC = Google Remote Procedure Call. Un framework ad alte prestazioni progettato per scalare a milioni di RPC al secondo.

Governance Attuale (Aggiornamento 2025)

In preparazione per la Graduation nella CNCF, gRPC ha recentemente rinnovato la propria struttura di governance (agosto 2025) per passare da un modello "piatto" a uno più strutturato e trasparente, garantendo la neutralità rispetto ai vendor.

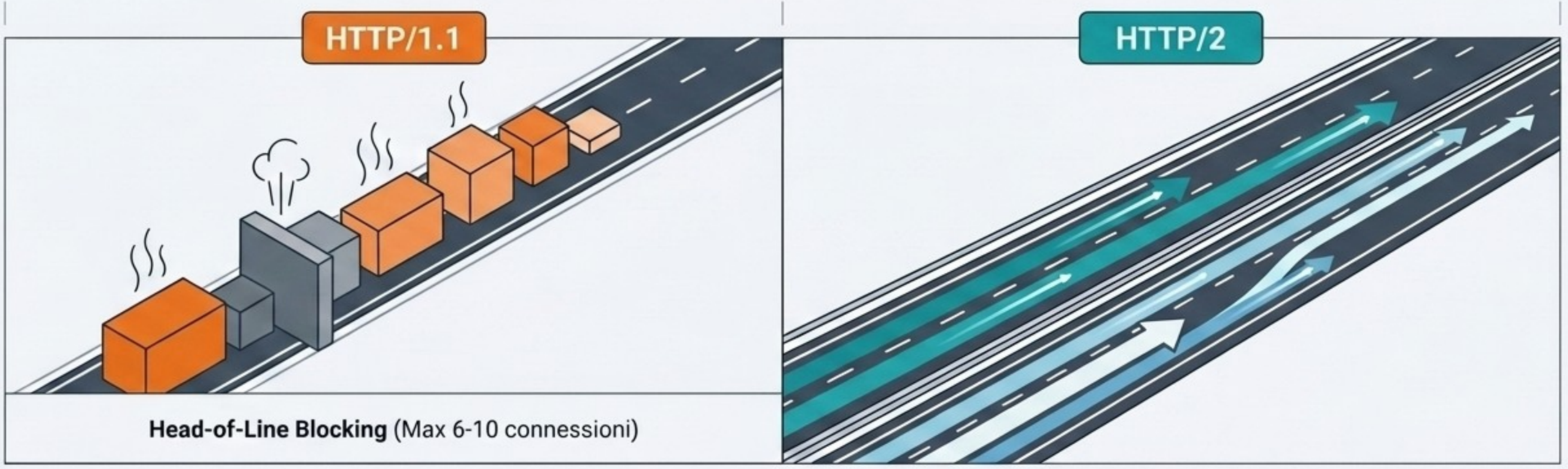
Il "Contributor Ladder" (Scala dei Contributori)

Role	Requisiti e Responsabilità
Org Member	Minimo 4 Pull Request (PR), correzioni di bug o revisioni all'anno. Richiede la sponsorizzazione di due membri esistenti.
Core Contributor	Esperti in un dominio specifico (es. sicurezza, osservabilità, un repository specifico). Responsabili della revisione della maggior parte delle PR nel loro dominio.
Maintainer	Il livello più alto. Esperti senior responsabili di intere implementazioni di linguaggio e linguaggio della definizione della strategia tecnica globale del progetto.

Comitato Direttivo (Steering Committee)

- Comunicazione con la CNCF.
- Determinazione della direzione generale per il marketing e l'advocacy.
- Emissione di dichiarazioni ufficiali a nome del progetto.
- Gestione di riunioni pubbliche e trasparenti.

Il motore delle performance: HTTP/2



Multiplexing

Più richieste e risposte viaggiano in parallelo su una singola connessione TCP persistente, azzerando l'overhead di apertura/chiusura.

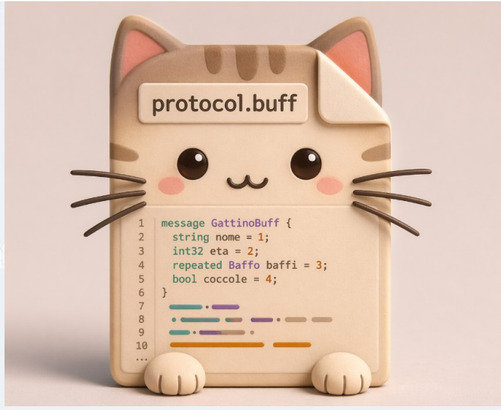
Compressione HPACK

Gli header vengono compressi e i dati ripetuti omessi, riducendo drasticamente la latenza.

Server Push

Il server invia risorse al client prima che vengano richieste, anticipando le necessità e migliorando il caricamento.

Protocol Buffers: L'efficienza del binario



Protobuf è il linguaggio di serializzazione di Google. Definisce dati strutturati in modo indipendente dalla piattaforma.

JSON **Protobuf Compiler** **Binario**

Vantaggi

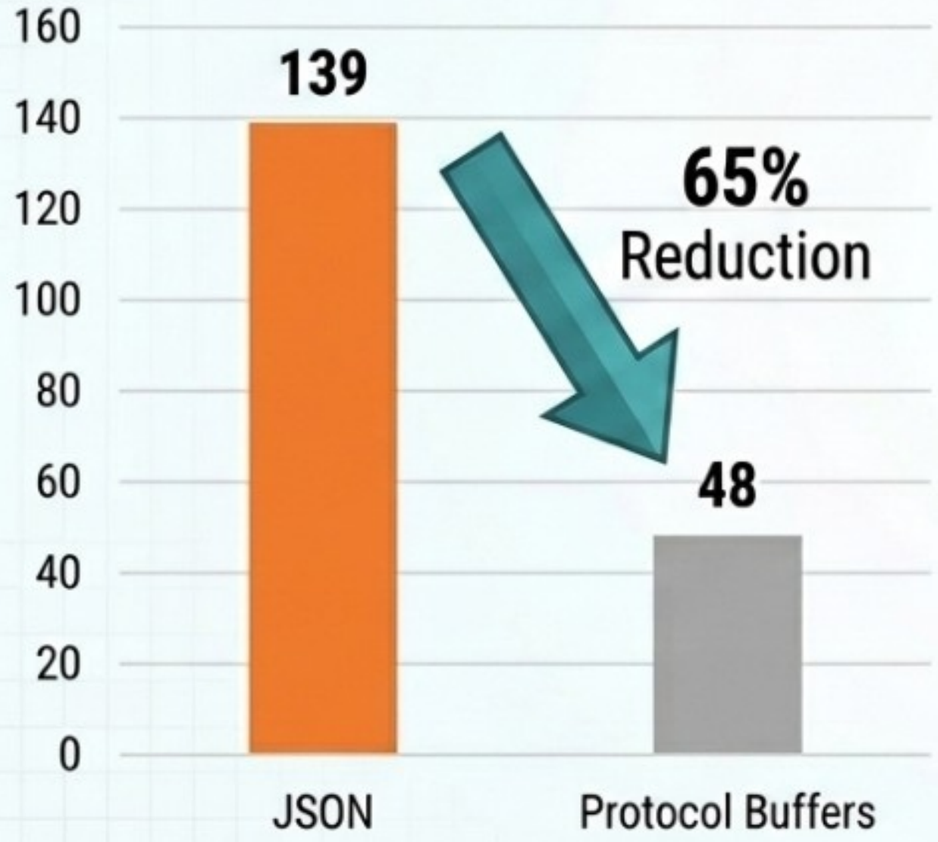
- Estremamente compatto (molto più leggero di JSON/XML).
- Serializzazione/parsing ad altissime prestazioni.
- Forte tipizzazione dei dati e schema rigoroso.

Il Trade-off

Non leggibile da umani. Il formato è interamente binario, rendendo il debugging manuale più complesso rispetto alle tradizionali API REST.

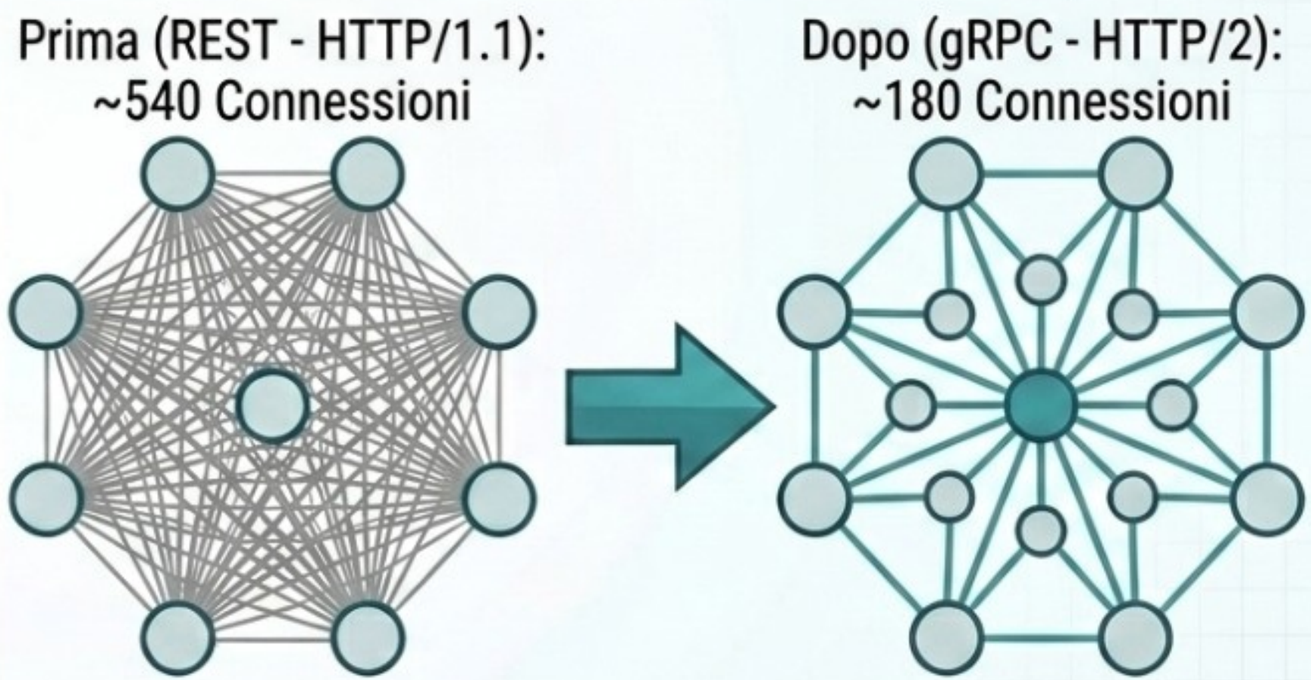
Performance: Il Vero Vantaggio

Dimensioni Pacchetto



(Fonte: Studio BME OMIKK, su struttura dati complessa)

Efficienza Connessioni TCP
(Scenario: 10 Microservizi)



67% Reduction in Connections

(Fonte: Arpit Bhayani, "Why gRPC Uses H

I 4 Modelli di Comunicazione

Unary
1 richiesta ➔ 1 risposta.
(Come una classica chiamata REST).

Server Streaming
1 richiesta ➔ Molteplici risposte
(Flusso di dati dal server, es. aggiornamenti in tempo reale).

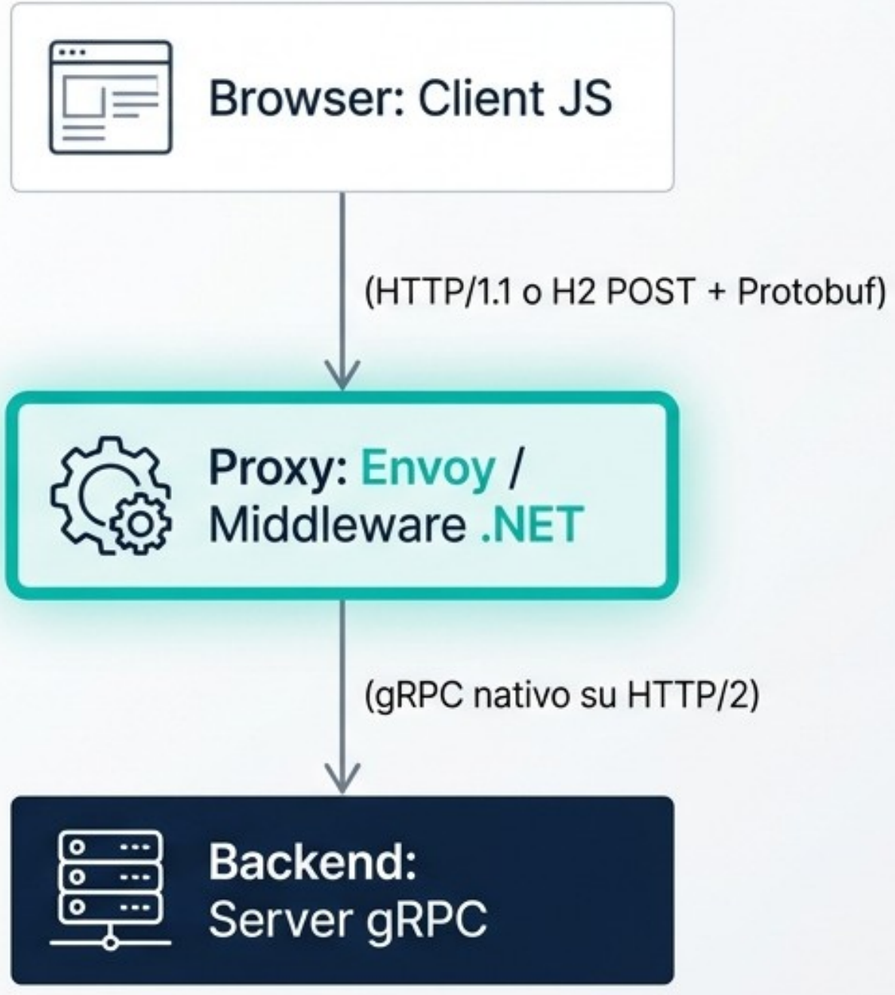
Client Streaming
Molteplici richieste ➔ 1 risposta
(Caricamento di flussi di dati, es. upload a blocchi).

Bidirectional Streaming
Molteplici richieste ↴ Molteplici risposte
(Comunicazione continua a due vie, es. chat in tempo reale).

Il grande confronto: gRPC vs REST

Caratteristica	gRPC	REST
Filosofia	RPC (Remote Procedure Call). Action / procedure-oriented (es. GetUser(123))	Stile architetturale. Resource-oriented (es. /users/123)
Protocollo	HTTP/2	HTTP/1.1 (prevalentemente)
Formato	Binario (Protobuf)	Testuale (JSON)
Streaming	Nativo (4 modelli)	Assente / Limitato
Contratto	Rigido (.proto)	Opzionale (Swagger/OpenAPI)
Performance	 Elevatissima	 Overhead maggiore
Supporto Browser	 Limitato (Richiede proxy)	 Universale

gRPC-Web: Il ponte Protocol-Aware per il browser



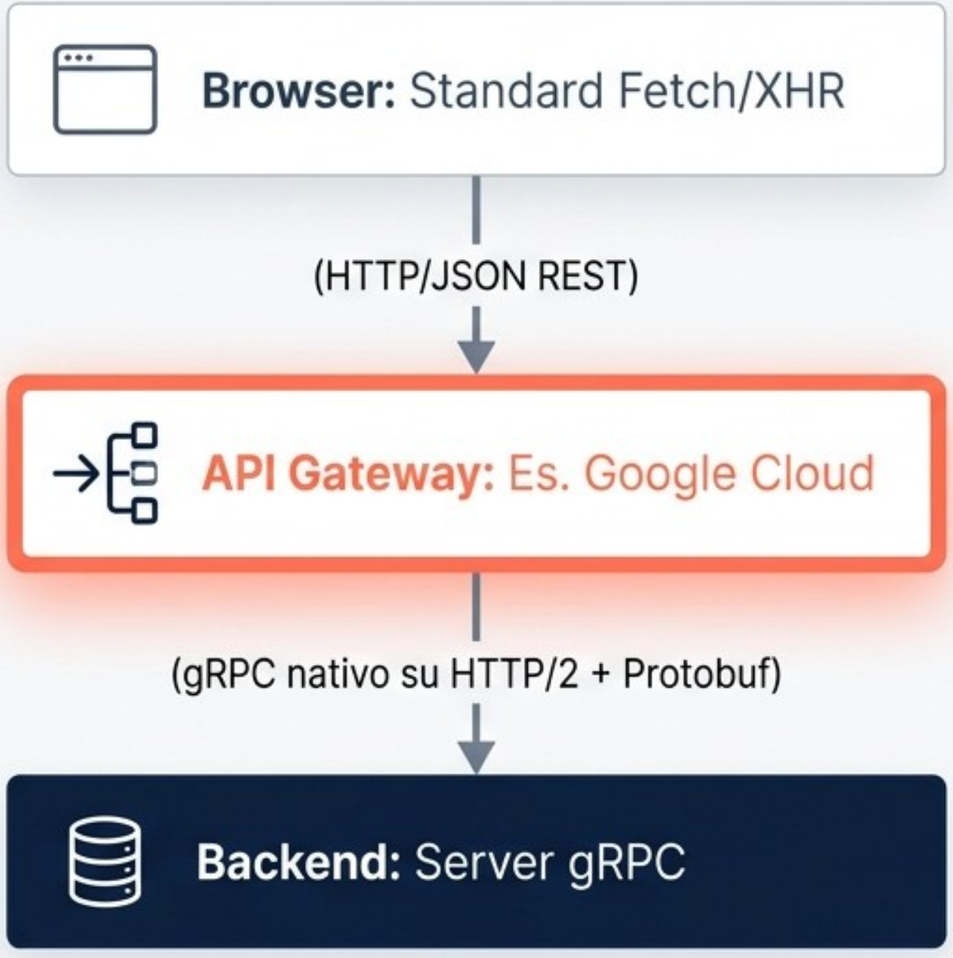
Funzionalità Core

- Estende l'ecosistema gRPC direttamente alle applicazioni web front-end.
- Supporta nativamente chiamate Unary e Server Streaming.
- Utilizza invariabilmente il metodo HTTP POST per tutte le transazioni.

Vincoli Architetturali

- **Streaming Limitato:** Impossibile eseguire streaming bidirezionale o client-side a causa delle restrizioni di base dei browser.
- **Passthrough Opaco:** Utilizza un formato di framing binario chiuso. L'infrastruttura di rete non può ispezionare, riscrivere gli header o comprimere il payload in volo.

Gateway di Transcodifica: Il traduttore RESTful per i sistemi legacy



Funzionalità Core

- **Mappatura REST-to-gRPC**: Trasforma servizi gRPC ad alte prestazioni in API REST tradizionali.
- **Flessibilità Assoluta**: Accessibile da qualsiasi client HTTP standard, ideale per sistemi legacy.
- **API Management**: Integra nativamente funzioni di autenticazione, rate-limiting e tracciamento.

Vincoli Architetturali

- **Hard-Limit del Payload**: Errore 500 Internal Server Error critico se il corpo della richiesta o risposta supera 1 MB.
- **Degradazione Funzionale**: Supporto limitato esclusivamente a chiamate Unary (niente streaming) e nessuna compressione del payload.
- **Rigidità di Configurazione**: Richiede regole di mappatura manuali nel file di servizio tramite IDL Protobuf.

Streaming a confronto: gRPC vs WebSockets

	WebSockets	gRPC Bidirectional
	Pipe grezza	Contratto + tipi + tooling
🌐	⚖ **Supporto universale nativo nei browser.** **Streaming generico e non tipizzato** (invia stringhe o blob binari senza schema). **Ideale** per applicazioni web pure e chat basate su browser.	⇄ Streaming **fortemente tipizzato grazie al contratto .proto.** **Multiplexing nativo su HTTP/2.** **Ideale** per comunicazione da **server a server e app native** (mobile/desktop). Richiede gRPC-Web (e proxy) per i browser.

Quando NON usarlo: Svantaggi e Attriti

Debugging Complesso

Essendo un formato binario, non è possibile ispezionare i payload raw con la stessa facilità con cui si legge un file JSON nei DevTools.

Limiti dei Browser

Impossibile implementare un client gRPC puro direttamente nel browser a causa di limitazioni storiche delle API HTTP. Richiede sempre un web proxy (es. Envoy).

Setup e Tooling

Setup iniziale più complesso. Richiede la configurazione del compilatore dei protocolli e step di build aggiuntivi rispetto a una semplice API REST.

Il Poliglottismo e l'Unica Verità

Il file .proto funge da contratto universale. Da un singolo file è possibile autogenerare il codice client e server in oltre 12 linguaggi.

Nessuna discrepanza tra backend e frontend.

Se il contratto compila, la comunicazione è garantita in qualsiasi linguaggio.

gRPC è il nuovo SOAP? (No)

WSDL / SOAP (Il Passato)

- **Formato:** XML (Estremamente verboso e pesante).
- **Tipizzazione:** Rigorosa, ma complessa da mantenere.
- **Prestazioni:** Lente, con enorme overhead di rete e parsing.

gRPC / Protobuf (Il Presente)

- **Formato:** Binario compresso.
- **Tipizzazione:** Rigorosa tramite .proto elegante e leggibile.
- **Prestazioni:** Velocità elevatissima, progettato per microservizi cloud-native.

L'Anatomia di un Contratto .proto

Definizione chiara della versione syntax (proto3).

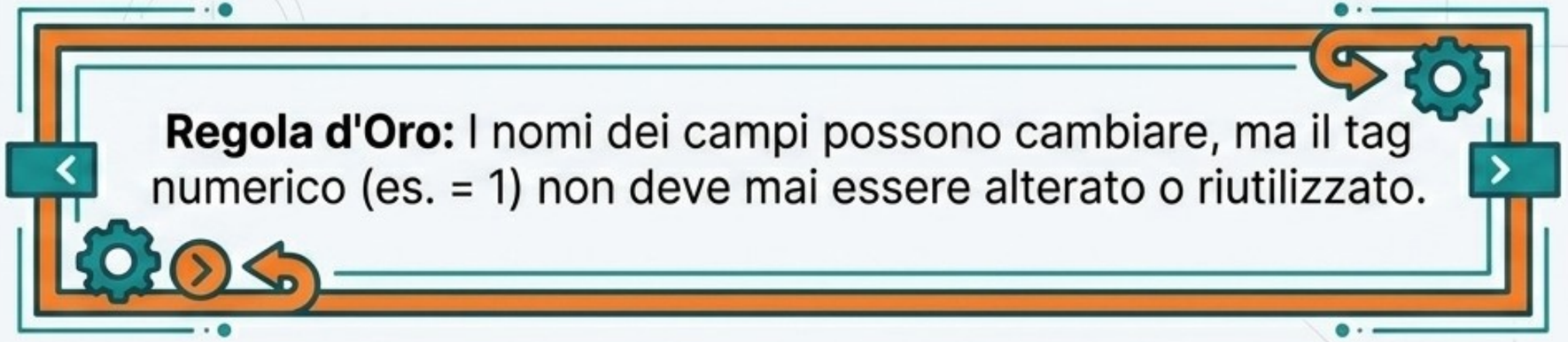
```proto
syntax = "proto3";

package communication;

service CommunicationService {
    // 1. Unary (una richiesta, una risposta)
    rpc SendUnaryMessage (RequestMessage) returns (ResponseMessage);

    // 2. Server streaming (una richiesta, molte risposte)
    rpc StartServerStream (RequestMessage) returns (stream ResponseMessage);

    // 3. Client streaming (molte richieste, una risposta)
    rpc SendClientStream (stream RequestMessage) returns (ResponseMessage);

    // 4. Bidirectional streaming (molte richieste, molte risposte)
    rpc ChatBidirectional (stream RequestMessage) returns (stream ResponseMessage);
}

message RequestMessage {
    string content = 1;
}

message ResponseMessage {
    string content = 1;
}
```

Dichiarazione dei metodi RPC e della direzione degli stream.

Definizione fortemente tipizzata dei messaggi.

Evoluzione del Contratto e Versionamento

gRPC garantisce la backward e forward compatibility attraverso un rigido sistema di tag numerici immutabili.



Old: `string name = 1;`

↓

New: `string full_name = 1;` (SAFE - Compatibile)

↓

Deprecated: `reserved 2;` (Previene il riutilizzo accidentale del tag 2).

Developer Experience: Integrazione in Visual Studio

In ambiente .NET, l'integrazione è nativa e invisibile. Visual Studio interpreta automaticamente la direttiva nel file di progetto.

```
<ItemGroup>
  <Protobuf Include="..\\Shared\\Protos\\greet.proto" GrpcServices="Server" />
</ItemGroup>
```

Ad ogni build, il compilatore genera automaticamente le classi C# per il client e le classi base per il server, pronte per essere implementate o consumate.

Adozione Enterprise e Sicurezza Nativa

NETFLIX Uber Spotify Dropbox docker CISCO

Utilizzato per gestire milioni di chiamate RPC al secondo nei loro core system.

- **TLS Nativo**: Connessioni cifrate end-to-end (obbligatorie di default per la connessione ai servizi Google).
- **Autenticazione**: Supporto nativo per Token-based Auth (OAuth2, JWT) tramite i Call Credentials.
- **Interceptors**: Middleware nativo per autorizzazioni globali a livello di Server e Client.

Risorse Utili per Iniziare

Sito Ufficiale

`https://grpc.io`

Guide, Quickstarts e architettura core.

Documentazione Protocol Buffers

`https://developers.google.com/protocol-buffers`

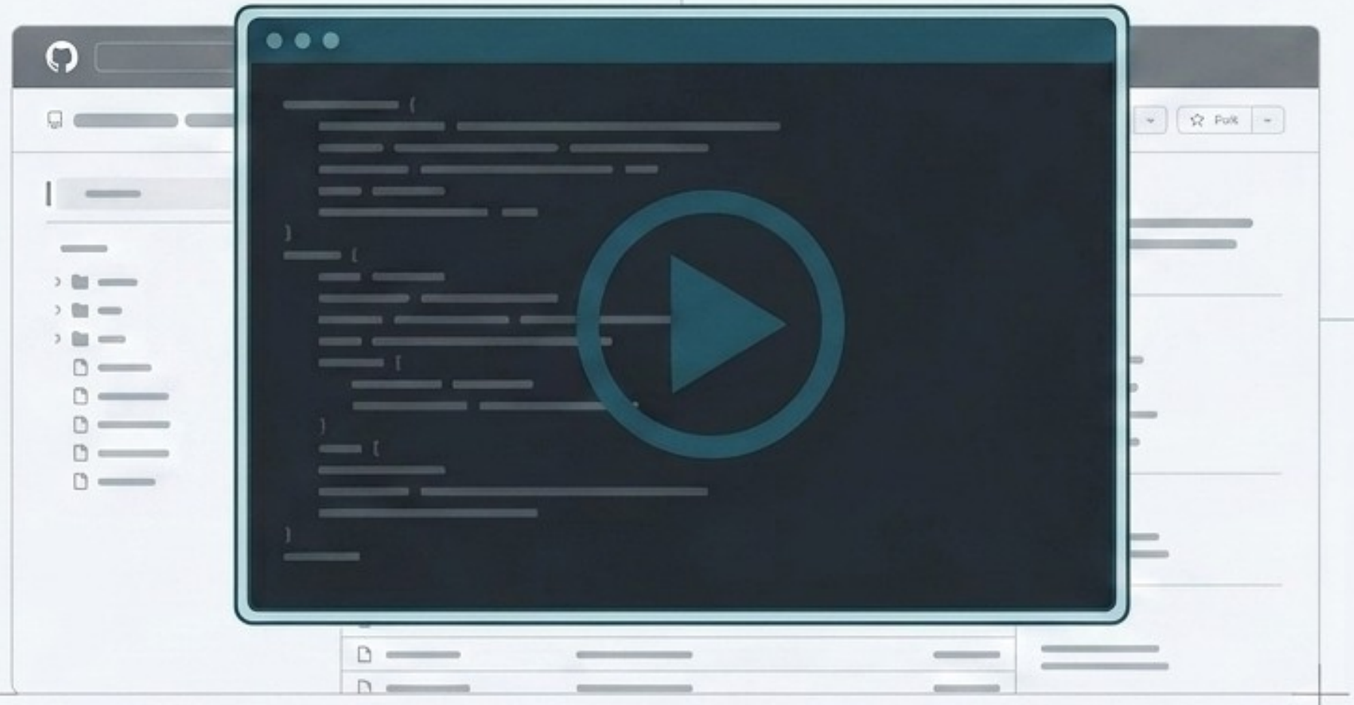
Sintassi, best practices e reference.

Repository GitHub

`github.com/grpc/grpc`

Progetto open source CNCF.

Demo Time!



Demo: gRPC Bidirectional Communication.

- Un server in C# che parla con due client.
- Uno in C# e uno in Go.
- Il tutto in streaming simultaneo bidirezionale.

Riferimento Codice: github.com/lbigoni92/GrpcDemoComplete

Q&A e Contatti

LinkedIn

Connettiti per discutere di architetture e microservizi.

https://www.linkedin.com/in/luca-bigoni/

Repository GitHub

Esplora il codice sorgente della demo bidirezionale in Go.

https://github.com/lbigoni92/GrpcDemoComplete

Feedback Form

Lascia un commento sulla sessione (Microsoft Forms).

https://forms.cloud.microsoft/Pages/ResponsePage.aspx?…

Spazio Q&A