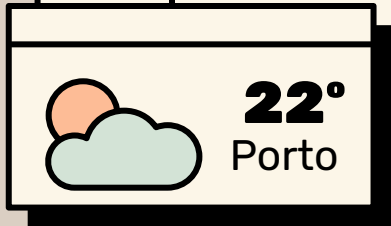
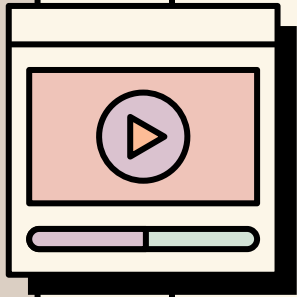
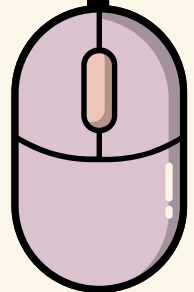


# gRPC Framework



>>>>>

EXADS Workshop



~~~~~  
.....



# Table of contents



**01** **RPC What?**  
A bit of the history about  
RPC.

**02** **Why gRPC?**  
What is the reasoning  
behind gRPC?

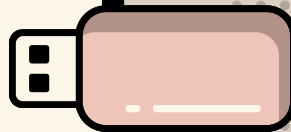
**03** **Get it to Work**  
Hands-on it.

**04** **Pros & Cons**  
Nothing is perfect.  
Technology is messy.

01

.....

>>>>



# RPC What?

Better to know the basics, huh?!



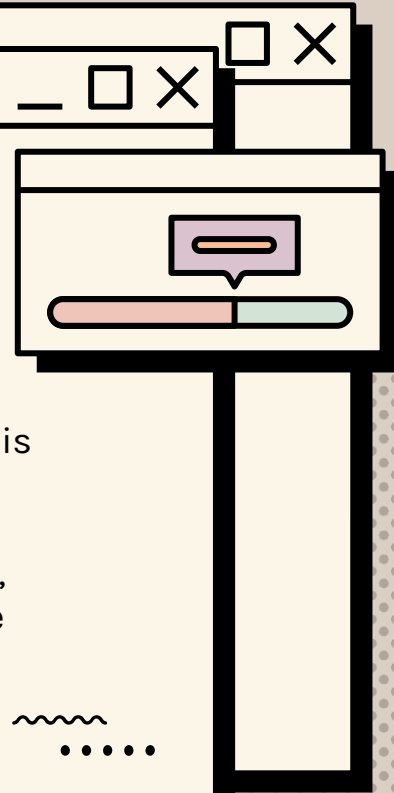


>>>>

# RPC Definition

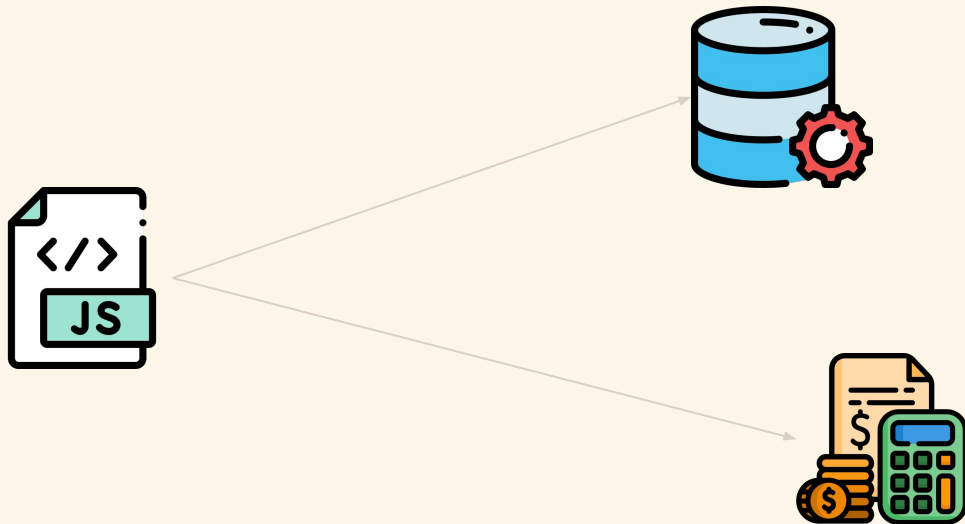
In distributed computing, a remote procedure call (RPC) is when a computer program causes a procedure (subroutine) to **execute in a different address space** (commonly on another computer on a shared network), which is written as if it were a normal (local) procedure call, **without the programmer explicitly writing the details for the remote interaction.**

~~~~~  
.....



.....

# Normal Application Flow



~~~~~  
>>>>



```
function main() {  
    const result = add(1, 2);  
    console.log(`The result is '${result}'`);  
}  
  
function add(a, b) {  
    return a + b;  
}  
  
// Run the application  
main();
```

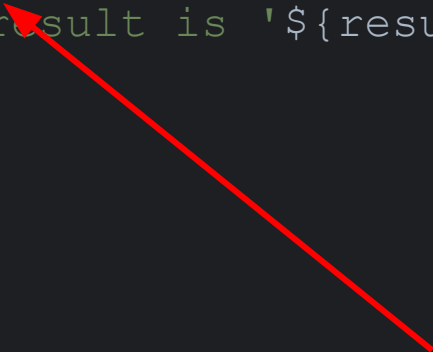


```
function main() {  
  const result = add(1, 2);  
  console.log(`The result is '${result}'`);  
}
```

```
function add(a, b) {  
  return a + b;  
}
```

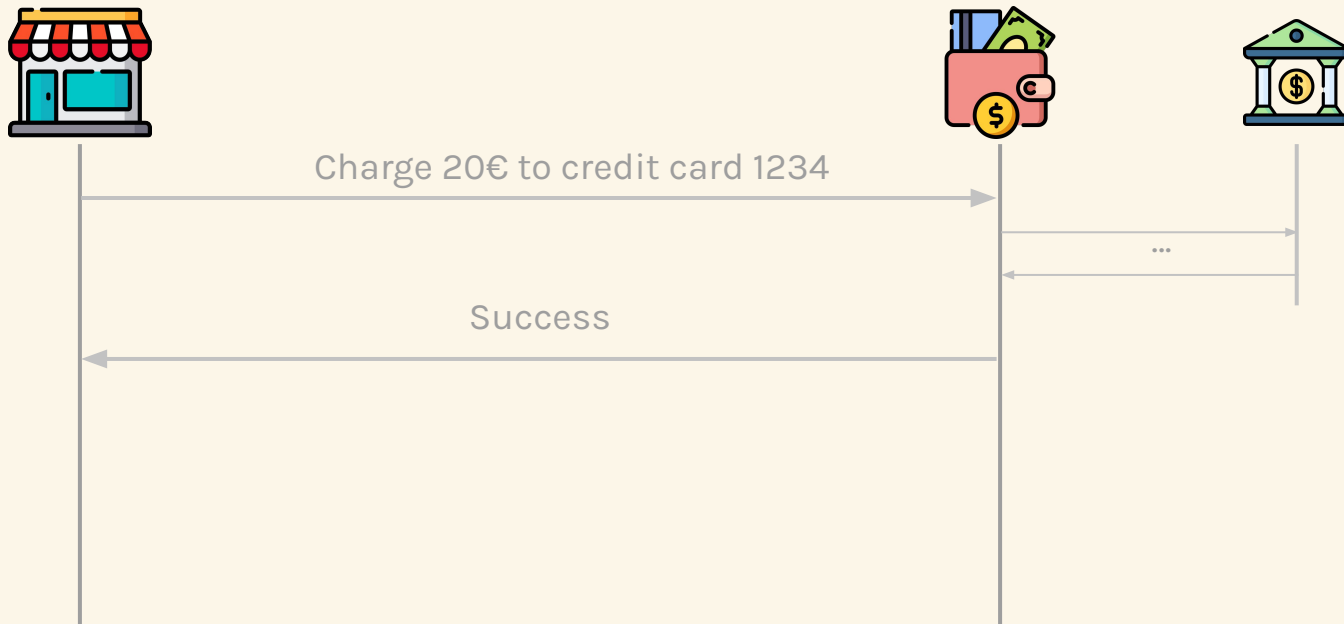
```
// Run the application  
main();
```

Implementation of this function is  
on another node



.....

# RPC Flow Example

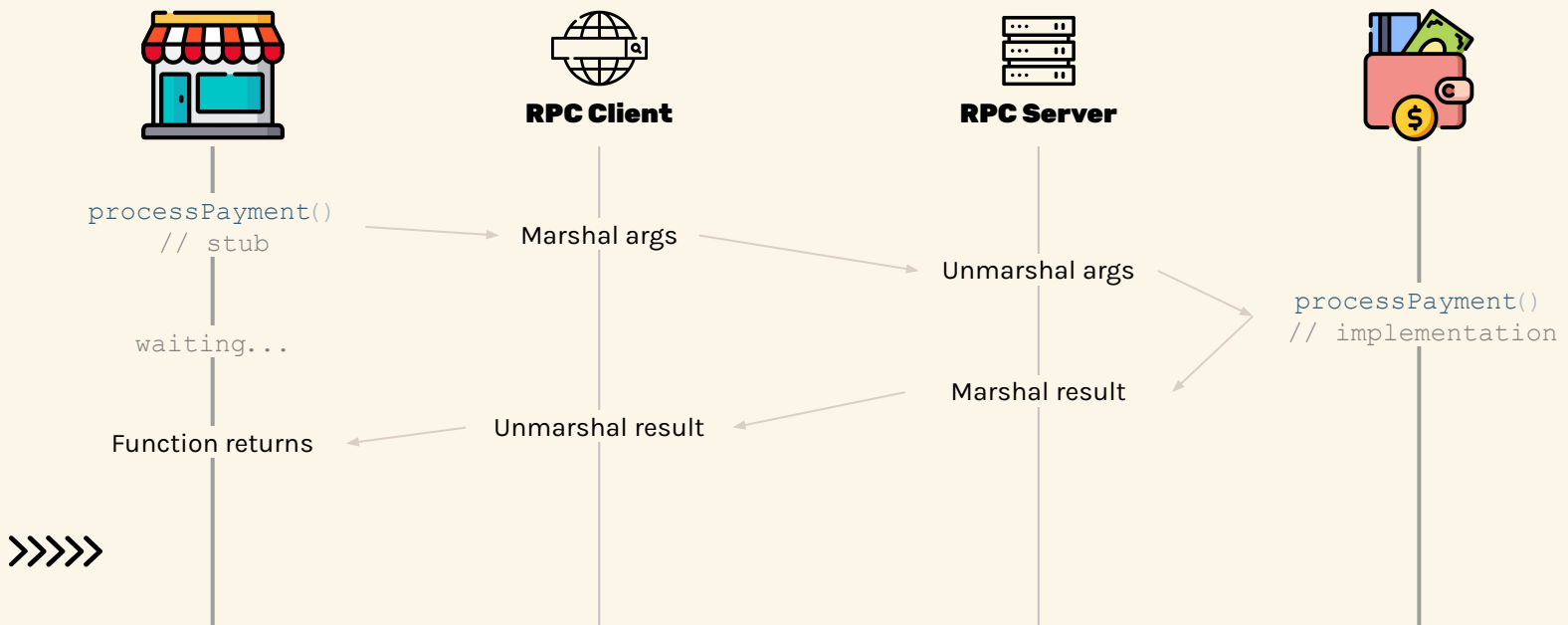


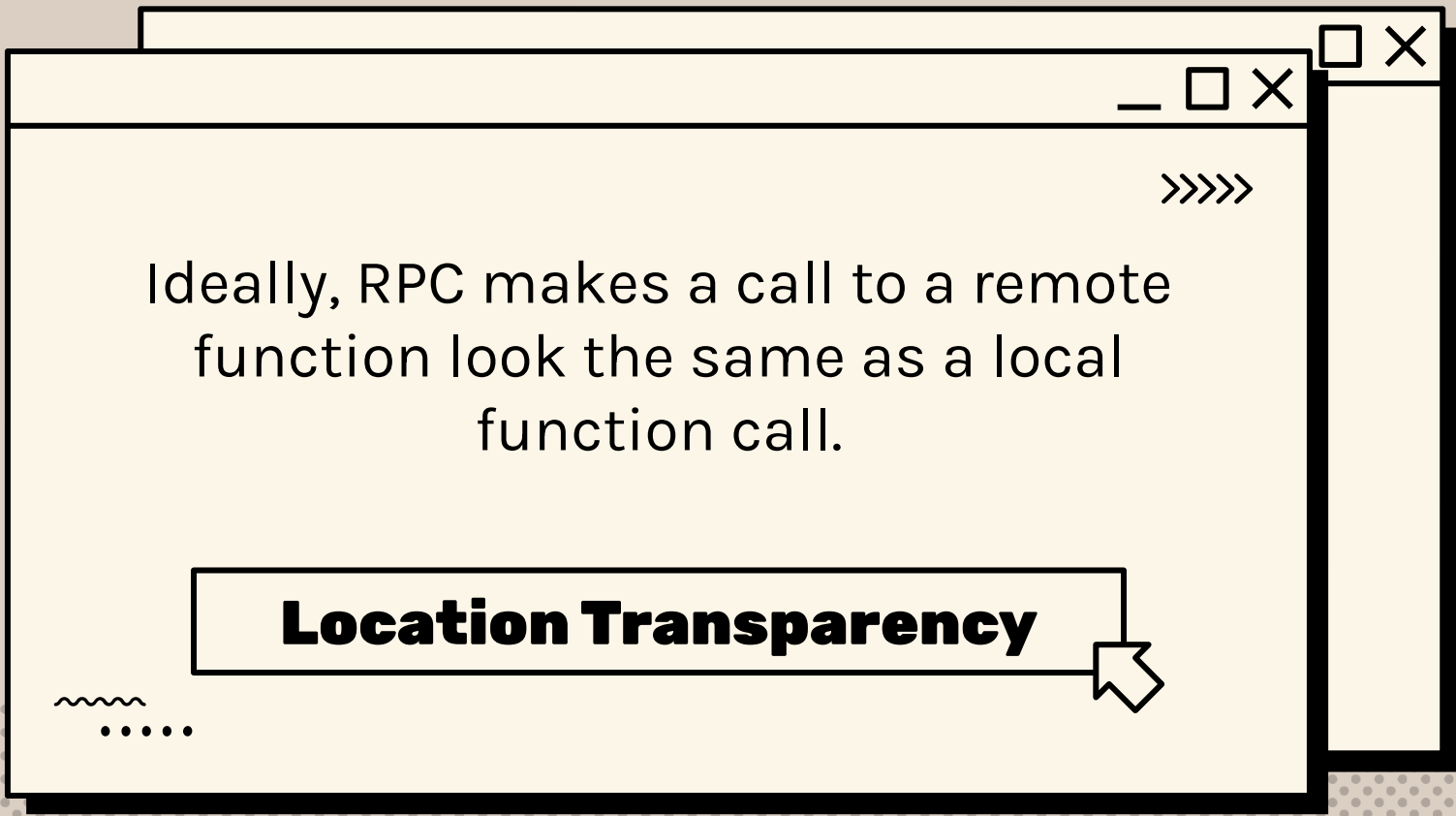
~~~~~  
>>>>>




....

# RPC Flow Example







```
function main() {  
    const result = add(1, 2);  
    console.log(`The result is '${result}'`);  
}  
  
// Run the application  
main(); // 3
```

.....

# RPC History

- SunRPC / ONC RPC (1980s, basis for NFS)
- COBRA: object-oriented middleware (1990s)
- Microsoft DCOM (1996)
- JAVA RMI (1999)
- SOAP/XML-RPC: RPC using XML and HTTP (1998)
- Thrift (Facebook, 2007)
- REST
- GraphQL
- **gRPC** (Google, 2015)

~~~~~  
>>>>>

Q Type to search



sts 232 Actions Projects 14 Security Insights

grpc / doc / g\_stands\_for.md



yijiem [Release] Bump version to 1.57.0-dev (on master branch) (#33311)

5a06780 · last month History

Preview

Code

Blame

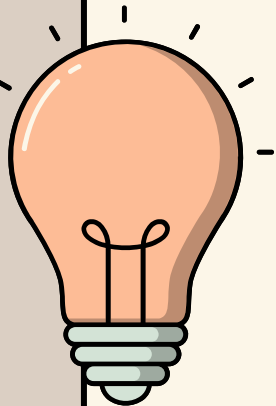
59 lines (58 loc) · 4.32 KB

Raw



'g' stands for something different every gRPC release:

- 1.0 'g' stands for 'gRPC'
- 1.1 'g' stands for 'good'
- 1.2 'g' stands for 'green'
- 1.3 'g' stands for 'gentle'
- 1.4 'g' stands for 'gregarious'
- 1.6 'g' stands for 'garcia'
- 1.7 'g' stands for 'gambit'
- 1.8 'g' stands for 'generous'
- 1.9 'g' stands for 'glossy'
- 1.10 'g' stands for 'glamorous'
- 1.11 'g' stands for 'gorgeous'
- 1.12 'g' stands for 'glorious'



# Why gRPC?

Isn't REST good enough?!

02

>>>>

# Why gRPC?



## Protobuf

Efficient Serialization;  
Language-neutral;  
Schema Definition  
Language



## HTTP2

Multiplexing;  
Header Compression;  
Bidirectional Streaming;  
Server Push

Official support for  
multiple programming  
languages:  
C++, Java, Python, Go, C#,  
Ruby, Node.js,  
Objective-C, PHP, Dart,  
Kotlin, Swift, Android,  
Java

.....

# Key Features

- Bi-directional streaming;
- Strongly-typed contracts;
- Support for multiple platforms and environments;
- Interceptors and middleware;
- Support for various authentication mechanisms;
- Code generation.







8

## gRPC Scenario



How much times faster can be gRPC over REST?



As of my last knowledge update in September 2021, it's challenging to provide a precise average speedup value for gRPC over REST because the performance difference can vary significantly depending on the use case and the specific implementation.

In certain scenarios, gRPC can be several times faster than REST due to the factors mentioned earlier (e.g., binary serialization, multiplexing, and streaming). Some performance benchmarks have shown gRPC to be up to 10 times faster than traditional REST APIs.



**Show me the code, human!**

04

## Pros & Cons

.....

>>>>>



## Pros

- Compact
- Fast
- One Client Library
- Progress Feedback (Upload UC)
- H2/Protobuf
- Strongly Typed Contracts
- Built-in Authentication and Security

## Cons

- Code Generation
- Error Handling
- Versioning
- Compression Overhead
- No native browser support



# Thanks!

Does anyone have any questions?  
francisco@exads.com

**CREDITS:** This presentation template was created by **Slidesgo**, including icons by **Flaticon**, and infographics & images by **Freepik**