Glossary

B

backward compatibility — A new version of a software library is backward compatible when existing applications using an earlier version of the library will continue to work with the newer version. A change in an application is backward compatible when new versions of the application do not introduce a change in existing behavior.

bidirectional streaming — Communication pattern in gRPC applications where both the client and the server can send and receive streams of data simultaneously.

bucket — A term used in object storage services (such as AWS S3) to serve as a container for objects, like a directory on a filesystem.

C

certificate authority (CA) — In the context of securing network communication, a certificate authority is an entity that signs certificates and acts as a trusted authority helping establish trust between a client and a server application.

channel — In general Go applications, a channel is a mechanism for goroutines to communicate with each other.

In gRPC applications, a channel is created when a client establishes a connection with the server. It is a long-lived connection, with all RPC method calls sharing the same channel for the lifetime of the client.

client-side interceptors — In gRPC applications, client-side interceptors are a way to perform tasks that are carried out for all RPC method calls.

Common use cases for client-side interceptors are exporting telemetry data, adding metadata, and implementing retry logic.

client-side streaming — In gRPC applications, client-side streaming allows a client application to send a stream of messages as part of an RPC method call to the server.

configuration data — Data that your application needs to perform its functionality, such as the database credentials or the address of your monitoring service.

connection — A communication channel between a client and a server application.

connection pooling — Connection pooling is a technique used to offset the cost of creating a network connection in an on-demand manner. Instead of a connection being created when one is required, it is retrieved from a pool of existing connections.

context — The `context.Context` type provides facilities for cancellation, time-out, and temporary storage of request-scoped key-value data within an application, as well as between applications, often across network boundaries.

counter metric — A measured value that increases monotonically during the lifetime of an application.

D

data producer — An application that continuously produces data, usually for another application to process further.

data reader — An application that reads the data from another source and either processes it further or simply forwards it to another application.

deep check — A special HTTP endpoint or gRPC method that includes logic to ensure that the application can perform the desired functionality. For example, a deep check may verify whether all dependencies are reachable from the application.

deserialization — The process of converting bytes of data, usually stored in a persistent store or received over a network, into a Go data structure, such as a `struct` type.

F

forward compatibility — A software package or application is forward compatible when the behavior exhibited by the older versions of the software is compatible with the behavior of the newer versions of the software. For example, if the newer version of a gRPC application successfully understands requests from an older version of the application, the application is said to be forward compatible.

G

gauge metric — A measured value that may increase or decrease during the lifetime of an application.

H

handler — The `Handler` interface is a type defined in the `net/http` package. It defines the method set that a type must implement to handle an HTTP request in a server.

handler function — A function that processes requests received by an HTTP server application.

health check — A special HTTP endpoint or an RPC method that is invoked by an external software, such as a load balancer, to assess the readiness of an HTTP or gRPC server to process requests.

histogram metric — A measured value that typically is used for recording sampled observations, such as the latency of an HTTP request or how long a database transaction lasted.

I

integration tests — Tests that examine and verify the functionality of an application that results in verifying its functionality which involves other applications and/or datastores.

interceptors — Application components used to implement common functionality in applications and thus avoid code duplication. Also known as *middleware*.

J

JavaScript Object Notation (JSON) — A language-neutral, human-readable data format commonly used as a data interchange format in HTTP applications.

JSON stream — Data containing more than one JSON object, delimited by a special character such as a newline or another way to indicate the end of one object and beginning of another.

M

marshalling — The process of converting a Go data structure, such as a `struct` type, to bytes of data to be transmitted over a network call or stored in a persistent store. Also known as *serialization*.

message — In gRPC applications, requests and responses are communicated as protobuf messages.

metadata — In gRPC applications, the `grpc/metadata` package enables transmitting arbitrary key-value pairs between client and server applications.

metrics — Measurements of key application behavior, such as the number of requests processed by a server application or the latency of requests processed.

middleware — Application components used to implement common functionality in applications and thus avoid code duplication. Also known as *interceptors*.

O

OpenTelemetry project — A collection of open-source specifications and libraries for instrumenting applications and exporting logs, metrics, and traces.

P

positional argument — In the context of applications using the `flag` package to parse command-line arguments, a positional argument is one whose position is significant to the application.

production readiness — The production readiness of an application is its assessment across multiple dimensions—user interface, user experience, robustness, resiliency, observability, and security.

protobuf — Protocol Buffers, or protobuf, is a language-neutral mechanism for serializing data so that they can be transmitted over the network or stored in a persistent store. It is the default serialization format used for gRPC applications.

pull model — In the context of software monitoring systems, the pull model refers to the communication pattern where the monitoring systems query the applications for the metrics. An example of such a system is Prometheus (`https://prometheus.io/`).

push model — In the context of software monitoring systems, the push model refers to the communication pattern where the applications push the metrics into the monitoring system. An example of such a system is statsd (`https://github.com/statsd/statsd`).

R

Remote Procedure Call (RPC) — A networking communication pattern where a client application communicates with a server application using specialized request-response data formats. For the client, it appears as a function call invoking code within the application itself, but the reality is that it is a request to another application over a network.

resilient — An application is considered resilient if it can withstand transitory unexpected situations and return back to its normal functioning.

robust — An application is said to be robust if it continues to function, perhaps in a degraded manner, when faced with unexpected situations in its runtime environment.

S

self-signed certificates — Transport Layer Security (TLS) certificates that are not signed by a certificate authority.

serialization — The process of converting a Go data structure, such as a `struct` type, to bytes of data to be transmitted over a network call or stored in a persistent store.

server-side interceptors — In gRPC applications, implementing server-side interceptors are a way to perform tasks that are carried out for all incoming RPC method calls. Common use cases for server-side interceptors are exporting telemetry data, adding metadata, and implementing retry logic.

server-side streaming — In gRPC applications, server-side streaming is a communication pattern where the server's response to a client request is sent as a stream of messages.

span — In distributed tracing, a span represents a specific operation in a transaction that is represented by a trace.

streaming — Data transmission pattern where an application is continuously producing data with no defined termination point.

structured logging — A logging pattern where log lines conform to a predefined format, such as JSON-formatted strings or key-value pairs.

sub-commands — A pattern of implementing command-line applications where distinct functionality is grouped as separate commands nested inside the original command.

T

trace — In distributed tracing, a trace encapsulates a complete transaction within the system.

trace's lifetime — The lifetime of a trace corresponds to the lifetime of a single transaction in an application.

Transport Layer Security (TLS) — A networking protocol designed to provide secure network communication using cryptographically signed certificates.

U

unit tests — A unit test verifies the functionality of a certain behavior of the application in isolation, avoiding any interactions beyond the part under test.

unmarshalling — The process of converting bytes of data, usually stored in a persistent store or received over a network, into a Go data structure, such as a `struct` type. This is also referred to as *deserialization*.