**ISC** - (Inter-service communication) là quá trình trao đổi thông tin giữa các dịch vụ (service) riêng lẻ trong một hệ thống phân tán.

Có 2 cách để chia ra các loại trao đổi thông tin giữa 2 dịch vụ (service) riêng lẻ:

**Thứ nhất:** Chia theo giao thức gồm 2 loại là đồng bộ (synchronous) hoặc không đồng bộ (asynchronous) cách này phổ biến hơn

1. **Đồng bộ (synchronous)**

Có thể hiểu đơn giản là trong giao tiếp đồng bộ, dịch vụ (service) thực hiện yêu cầu sẽ chờ phản hồi từ dịch vụ khác trước khi tiếp tục.

1 số phương pháp thông dụng :

**HTTP/REST APIs**: Các dịch vụ giao tiếp thông qua yêu cầu HTTP (ví dụ: GET, POST, …) với dữ liệu ở định dạng JSON hoặc XML. ( Ở bài sau sẽ viết về cái này sau, cái này là phổ biến nhất)

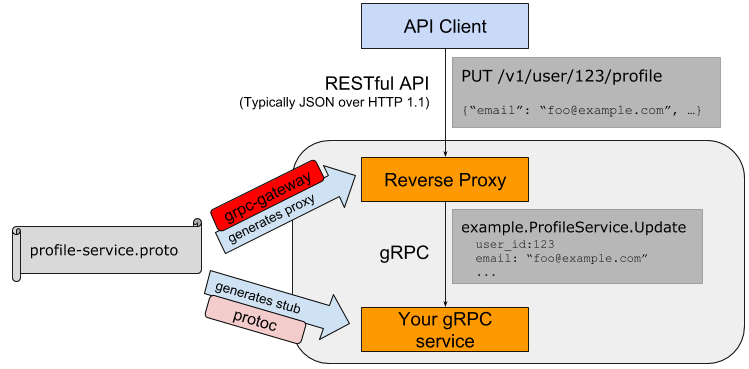
**gRPC**: Sử dụng HTTP/2 để giao tiếp nhanh hơn và Protocol Buffers (Protobuf) để tuần tự hóa dữ liệu.

Hiệu quả, hỗ trợ truyền dữ liệu dạng streaming, phù hợp với trường hợp yêu cầu hiệu năng cao.

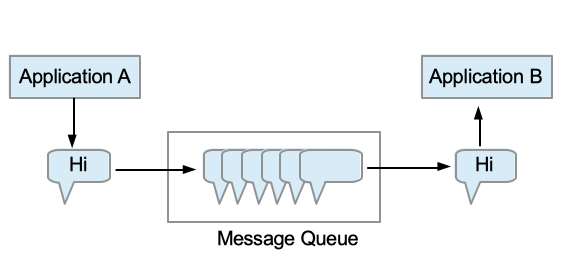
**SOAP**: Một giao thức trao đổi thông tin có cấu trúc sử dụng XML qua HTTP, SMTP, v.v.

Tiêu chuẩn nghiêm ngặt, thường dùng trong các hệ thống cũ (legacy).

* Lưu ý : Điểm quan trọng ở đây là giao thức đồng bộ thì mã client chỉ có thể tiếp tục công việc sau khi nhận được phản hồi từ máy chủ ( điểm đến khi gọi và nhận response về ) và các dịch vụ (service) sẽ trao đổi thông tin trực tiếp với nhau.

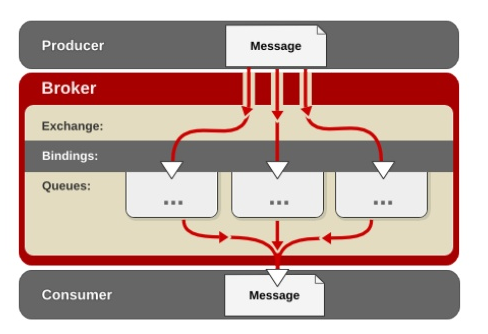


1. **Bất đồng bộ (asynchronous)**

 Trong giao tiếp bất đồng bộ, dịch vụ (service) gửi yêu cầu không cần chờ phản hồi ngay lập tức.

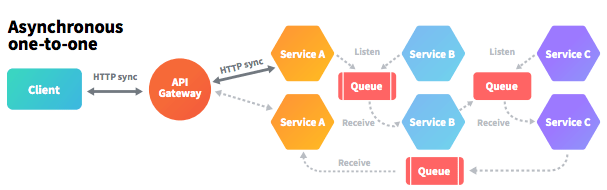
Một cải tiến đối với Message communication chính là **Advance message queue protocol (AMQP)** (Giao thức xếp hàng tin nhắn nâng cao ) . **AMQP** là loại giao thức được cung cấp bởi multiple OS hoặc cloud sử dụng asynchronous message (hệ thống message bất đồng bộ). **AMQP** được chuẩn hóa để truyền và nhận message giữa các service hay ứng dụng một cách tin cậy hơn.

Ở loại giao tiếp này các services không giao tiếp trực tiếp với nhau, mà chúng giao tiếp thông qua Message Broker. VD như bạn gửi thư bạn sẽ thông qua người đưa thư để đưa thư tới người nhận.



**Thứ hai:** Chia theo bên nhận : Một hoặc nhiều bên nhận

1. **Một bên nhận:** Yêu cầu được xử lý bởi đúng một bên nhận, ví dụ như mô hình Command.( Khi ạn gửi một lệnh (Command) đến một dịch vụ, chỉ một dịch vụ có trách nhiệm thực hiện lệnh đó.)



**Giao thức thường sử dụng:** HTTP hoặc AMQP (dùng cơ chế hàng đợi - queue).

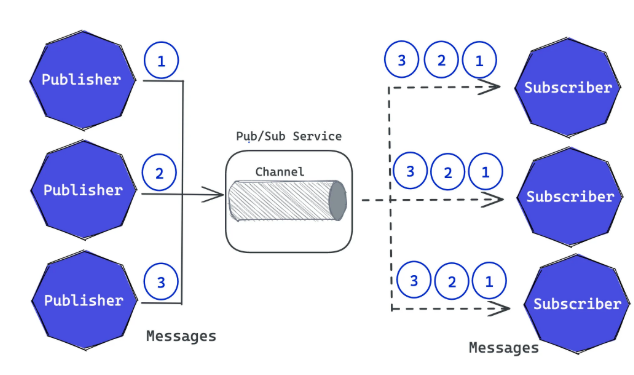
Trong trường hợp AMQP hoặc các hệ thống hàng đợi (queue), thông điệp thường được đưa vào một hàng đợi và chỉ một dịch vụ sẽ lấy thông điệp đó để xử lý (còn gọi là cơ chế *point-to-point messaging*).

**Ưu điểm:**

* Đơn giản và hiệu quả khi chỉ cần một bên nhận xử lý.
* Phù hợp với các tác vụ có tính chất xác định và không thể xử lý song song.

**Nhược điểm:** Nếu dịch vụ nhận bị gián đoạn hoặc quá tải, yêu cầu có thể bị trễ.

1. **Nhiều bên nhận:** Yêu cầu có thể được xử lý bởi nhiều bên, sử dụng cơ chế publish/subscribe.



**Cơ chế hoạt động :**

* Dựa trên cơ chế *event-driven architecture* (kiến trúc hướng sự kiện).
* Một sự kiện được phát hành (published) trên một **event bus** hoặc **message broker**.
* Các dịch vụ đăng ký (subscribed) với sự kiện này sẽ tự động nhận được thông báo và xử lý chúng.

**Giao thức thường sử dụng:**

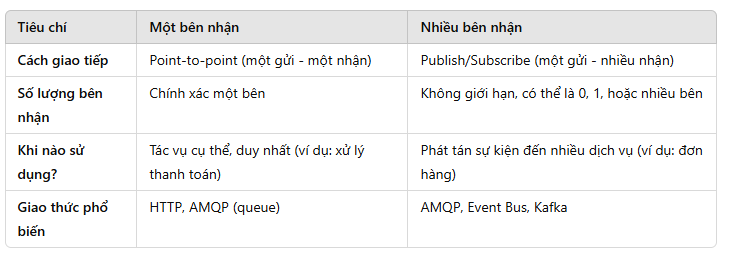
* **AMQP** (RabbitMQ, Kafka) hoặc các message brokers khác như Azure Service Bus, Google Pub/Sub.
* **Event Bus:** Một công cụ để quản lý và truyền tải sự kiện.

**Ưu điểm:**

* Linh hoạt và mạnh mẽ, dễ mở rộng khi thêm dịch vụ mới mà không cần thay đổi dịch vụ phát hành sự kiện.
* Các dịch vụ có thể xử lý song song, cải thiện hiệu suất tổng thể.

**Nhược điểm:**

* Phức tạp hơn trong việc quản lý và xử lý lỗi.
* Cần đảm bảo cơ chế *idempotency* (tính chất xử lý một lần duy nhất) để tránh việc xử lý trùng lặp khi nhiều bên cùng nhận sự kiện.



Còn 1 số câu hỏi như khi giao tiếp, các thông điệp sẽ có định dạng như nào, tiêu chuẩn ra làm sao, hồi sau sẽ rõ.

Ngoài những giao thức trên thì còn 1 số giao thức nữa như :

Giao tiếp dựa trên tệp tin (File-Based Communication)

Giao tiếp thông qua cơ sở dữ liệu (Database-Based Communication)

Giao thức gọi từ xa (Remote Procedure Calls - RPC)

Giao tiếp qua WebSocket

Service Mesh