# 20220426 - REST

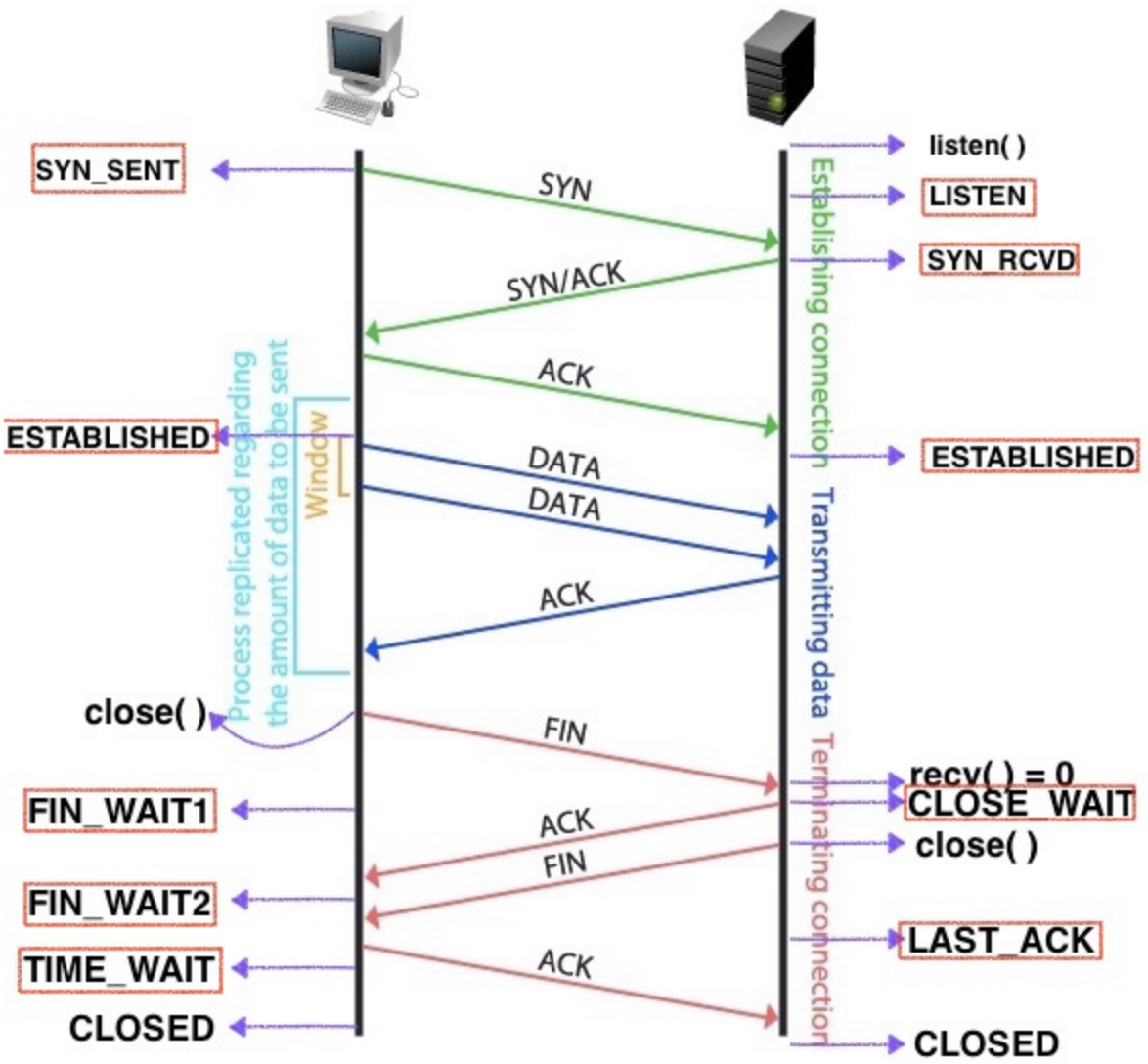
| | |
|---|---|
| Created | @2022년 4월 26일 오후 10:11 |
| Tags | Personal Study |
| Property | |

# REST 아키텍처 원칙

- 인터페이스 일관성: 일관적인 인터페이스로 분리되어야 함
- 무상태(Stateless): 각 요청 간 클라이언트의 컨텍스트가 서버에 저장되어서는 안된다.
- 캐시 처리 가능(Cacheable): 클라이언트는 응답을 캐싱할 수 있어야 한다.
- 계층화(Layered System): 클라이언트는 보통 대상 서버에 직접 연결되었는지, 또는 중간 서버를 통해 연결 되었는지를 알 수 없다. 중간 서버는 로드 밸런싱 기능이나 공유 캐시 기능을 제공함으로써 시스템 규모 확장성을 향상시키는 데 유용하다.
- Code on demand(optional): 필요한 경우 클라이언트는 서버가 전송한 코드를 수행할 수 있어야 한다.
- client / server 구조

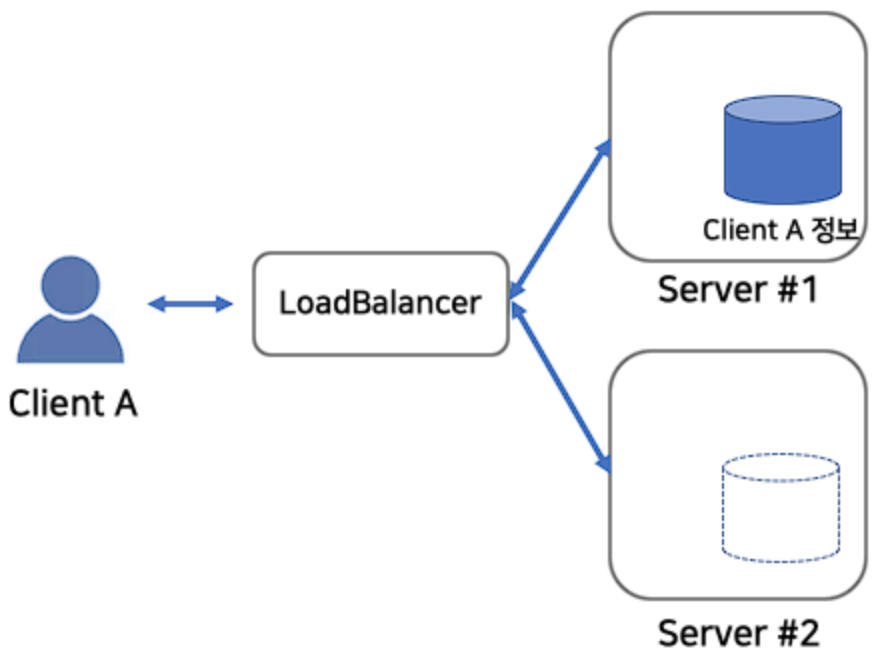
## Stateless

- **Stateful Service**
  - 서버와 클라이언트 간 세션의 'State(상태)'에 기반하여 클라이언트에 response를 보낸다.
  - 이를 위해 세션 상태를 포함한 클라이언트와의 세션 정보를 서버에 저장하게 된다.
  - 대표적 프로토콜: **TCP**(3-way handshaking)
  - TCP에서 서버와 클라이언트는 `SYN`, `SYNACK`를 주고 받으며 세션 상태를 'ESTABLISHED' 상태로 만든다.
  - 이때, 클라이언트와 서버는 데이터를 주고 받을 수 있다.
  - 위와 같이 세션 상태에 따라 서버의 응답이 달라지는 경우 ***Stateful*** 프로토콜이라 한다.




  - 데이터를 전송할 때 서버는 클라이언트로부터 송신된 패킷 헤더를 파싱하여 앞으로 수신해야할 데이터의 크기(윈도우 크기), 시퀀스 번호 등을 저장하고 모든 데이터가 수신될 때까지 연결 상태를 유지한다.

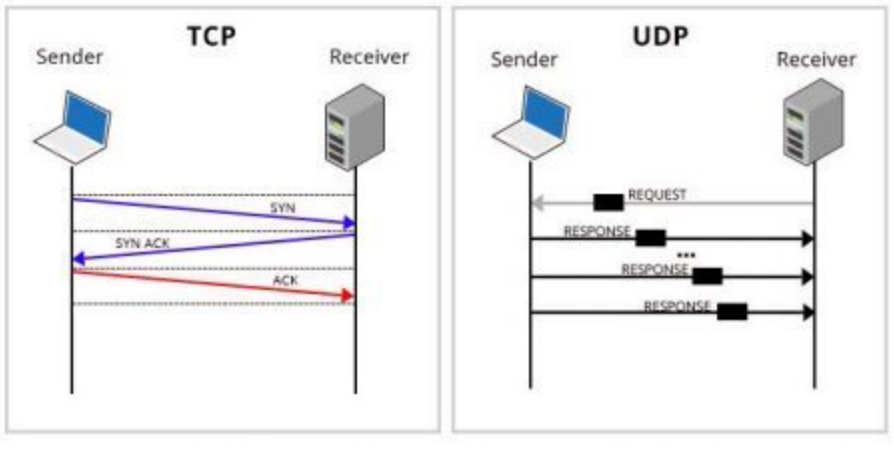
- 이 과정에서 서버는 앞으로 수신할 데이터 크기, 시퀀스 번호 등과 같이 클라이언트와의 세션 정보를 저장하게 된다.
- 정리하자면, Stateful Service는 ***1) 세션 정보를 서버에 저장***하고, ***2) 세션 상태에 따른 응답***을 만족하도록 설계된 서비스 구조이다.
- Stateful Service 구조에서 발생할 수 있는 문제점



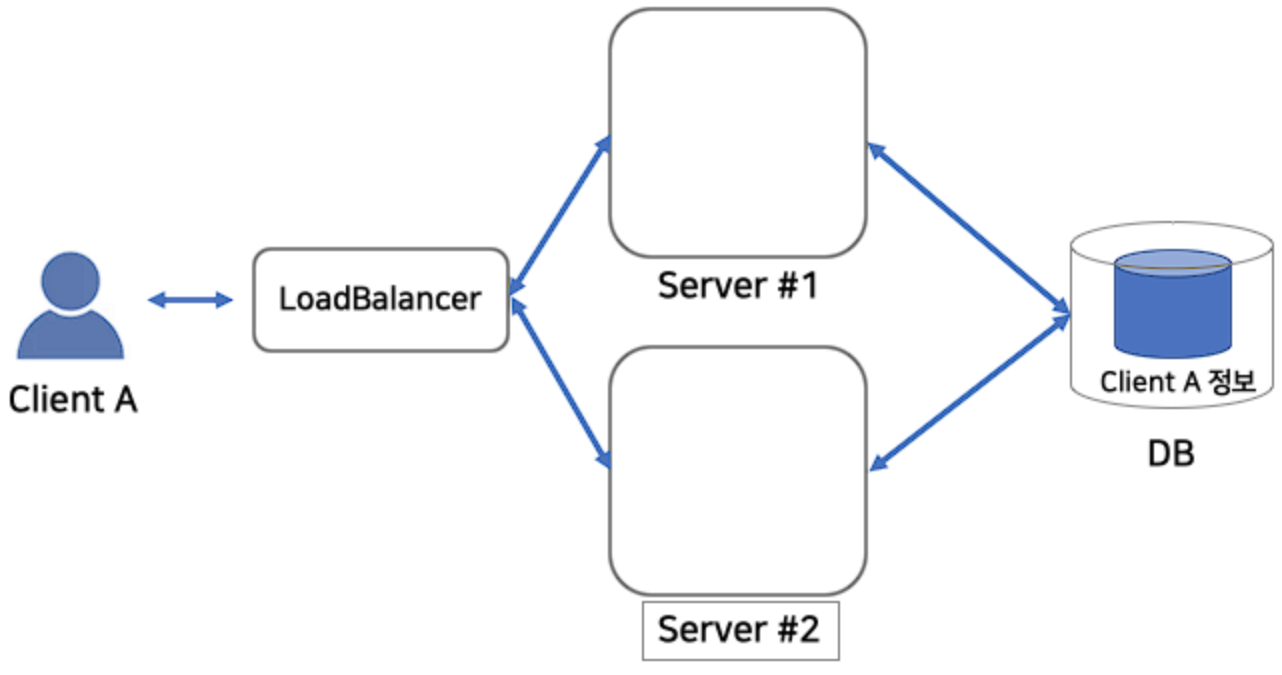

Stateful Service Architecture

  - ‘Client A’의 요청이 로드밸런서를 통해 임의의 서버(Server #1)에 전달되면 이 서버에 클라이언트의 정보가 저장이 되나 다른 서버(Server #2)에는 클라이언트의 정보가 없으므로 연결할 경우 세션 인증을 처음부터 다시 해야하며 이러한 작업을 피하기 위해 해당 클라이언트는 항상 서버 1에서 처리되도록 관리할 필요가 있다.

- **Stateless Service**
  - 서버의 응답이 클라이언트와의 세션 상태와 **독립적**이다.
  - 서버는 단순히 요청이 들어오면 응답을 보내는 역할만 수행하며, 세션 관리는 클라이언트가 책임을 진다.
  - 이러한 구조는 클라이언트와의 세션 정보를 기억할 필요가 없으므로 서버가 이러한 정보를 저장하지 않는다. 필요에 따라 DB에 저장하여 관리할 수는 있다.
  - 대표적 프로토콜: **UDP**, **HTTP**
  - UDP는 TCP와 달리 handshaking 과정을 통해 연결 세션을 인증하는 절차를 수행하지 않는다.
  - 세션 상태에 관계없이 단순히 데이터그램(datagram)을 source에서 destination 쪽으로 전송한다.
  - 클라이언트가 송신하려 했던 데이터가 서버 쪽에 수신되었는지 확인하지 않는다. 따라서, 서버 쪽은 클라이언트와의 세션 정보를 전혀 저장하지 않는다.
  - 따라서 UDP는 클라이언트와의 세션 상태에 관계없이 요청에 대한 응답만을 수행한다.




  - 클라이언트와의 세션 정보를 서버가 저장하지 않는다는 점에서 ***Stateless*** 하다고 말할 수 있다.
  - 정리하자면, Stateless Service는 ***1) 세션 정보를 서버에 저장하지 않고***, ***2) 세션 상태에 무관한 응답***을 만족하도록 설계된 서비스 구조이다.
  - TCP를 사용할 때의 문제 상황을 다시 살펴보자.




Stateless Service Architecture

    - Stateless 서비스 구조에서는 ‘Client A’의 정보를 서버에 저장하지 않고 외부 DB에 저장한다. 따라서 클라이언트의 요청이 Server #2에 전달되더라도 DB를 통해 클라이언트 정보에 접근이 가능하다.
- Stateless Service의 장점
    - Scaling이 자유로움(스케일 아웃, 스케일 업... 추후 학습 필요)

## 용어 정리

- HTTP: Stateless한 프로토콜
- REST: HTTP 프로토콜 상에 구현된 Resource Oriented Architecture(ROA) 설계 구조
- HTTP와 REST 모두 Stateless한 성격을 가짐
- HTTP는 Stateless한 성격을 가진 ‘프로토콜’, REST는 Stateless한 성격을 가진 ‘설계 구조’