Netty

NIO 기반 고성능 네트워크 어플리케이션 프레임워크

Netty

약 100여 개의 조직에서 사용 중

Asynchronous Event-driven

- 이벤트에 기반하여 요청을 비동기 처리
- Reactor Pattern, EventHandler, EventListener, Future, Callback 등을 사용하여 구현

thread-based vs event-driven (Server Model)

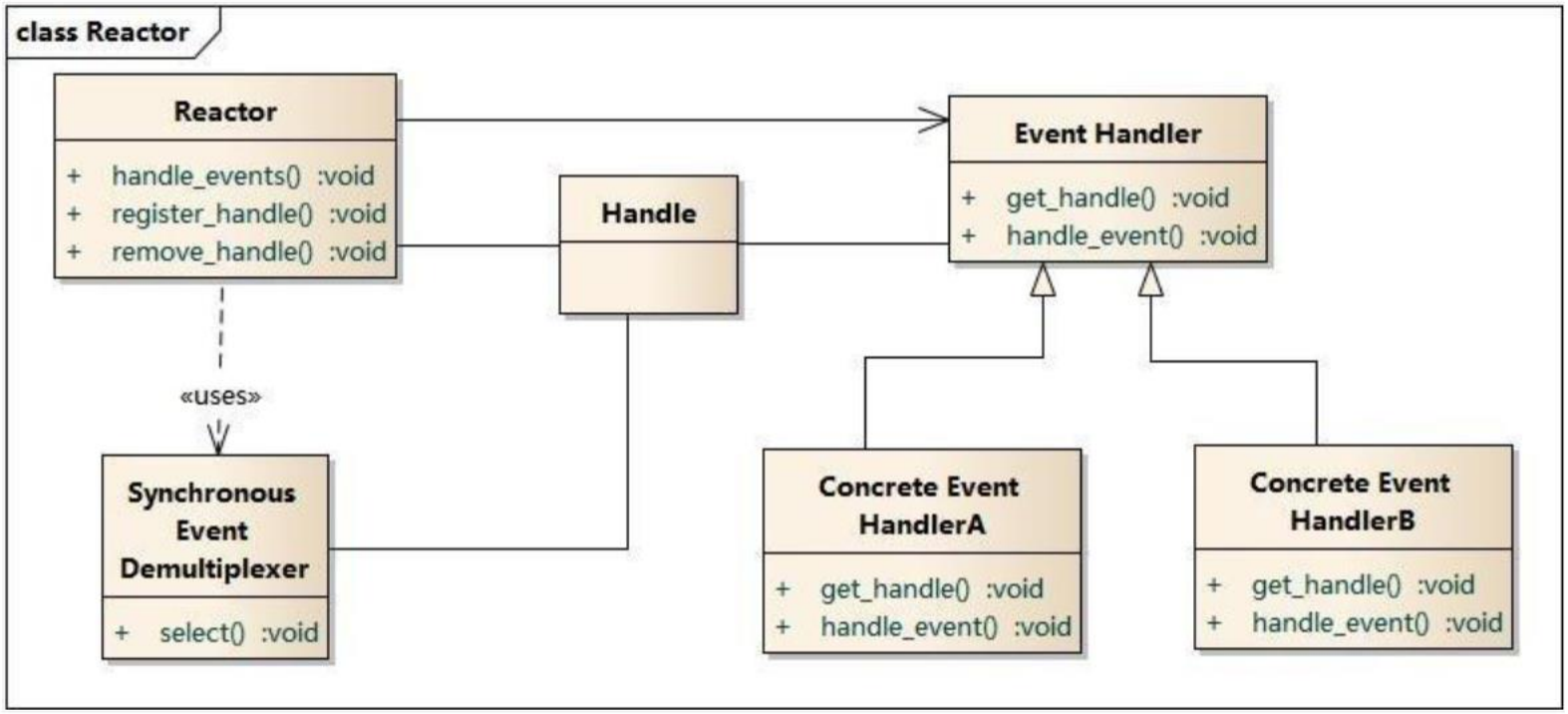
thread-based	event-driven
thread context	state machine/continuation
synchronous/blocking	asynchronous/non-blocking
thread-per-connection	events and associated handlers

Reactor Pattern

동시에 서버로 들어온 **Request**를 처리하기 위한 **Event Handling Pattern**

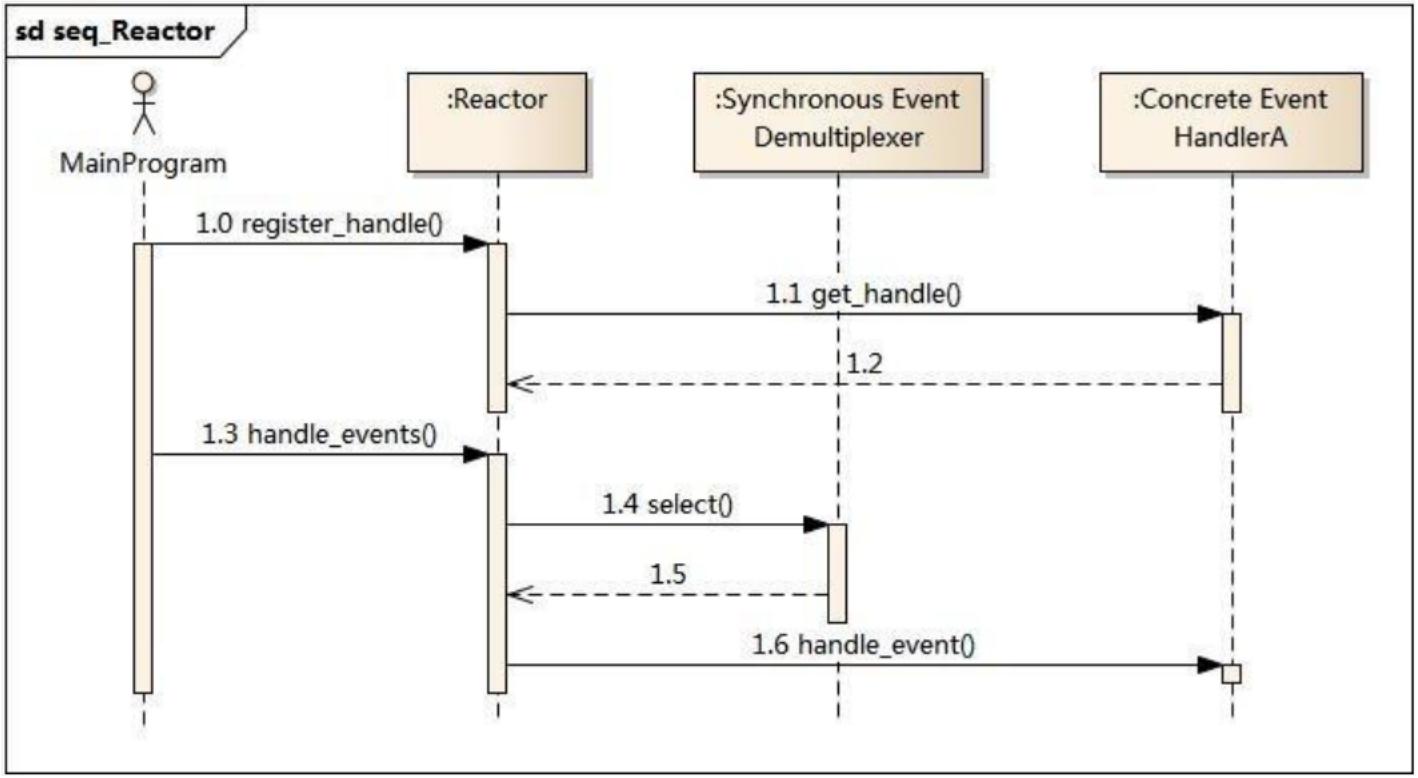
Netty

Reactor pattern structure



Netty

Reactor sequence diagram



The application launches and registers the event handler of interest to the Reactor;
calls Reactor to enter an infinite event loop, waiting for the registered event to arrive;
The event arrives, select returns, and Reactor distributes the event to the previously registered callback function.

Threading Model

EventLoop: 연결의 수명 기간 동안 발생하는 이벤트를 처리하는 작업을 나타내는 프로그래밍 구조

Netty

Bootstrap

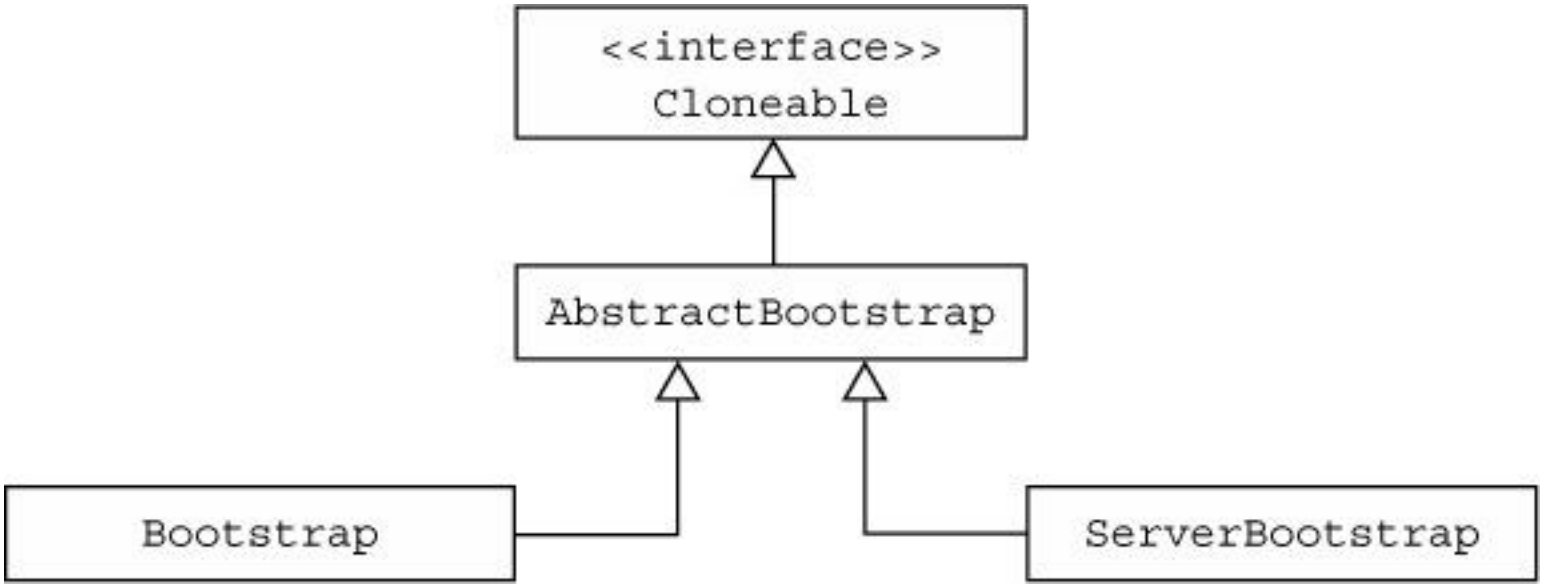
EventLoop

ChannelHandler
ChannelPipeline

Bootstrap

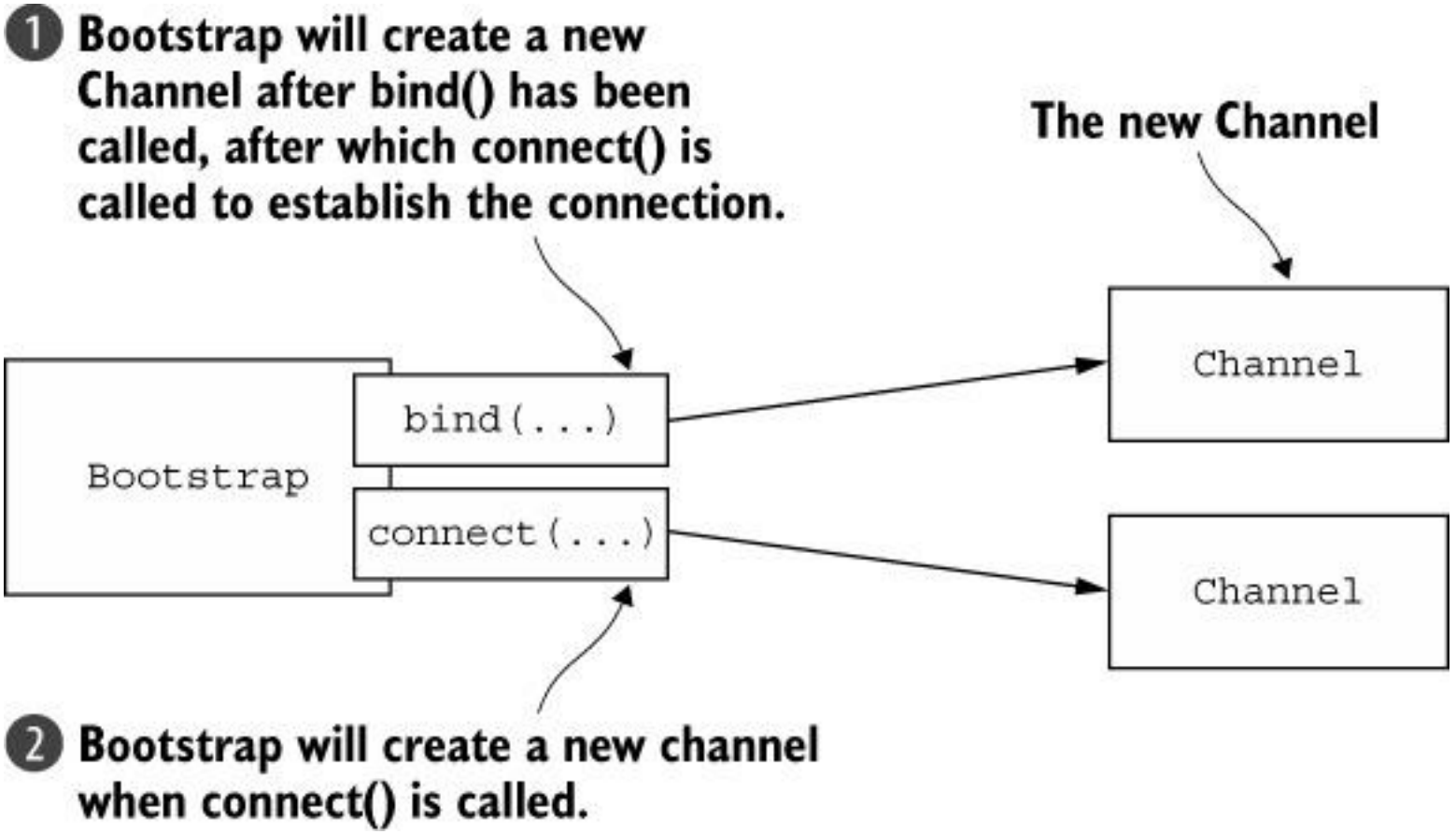
Netty

Bootstrap



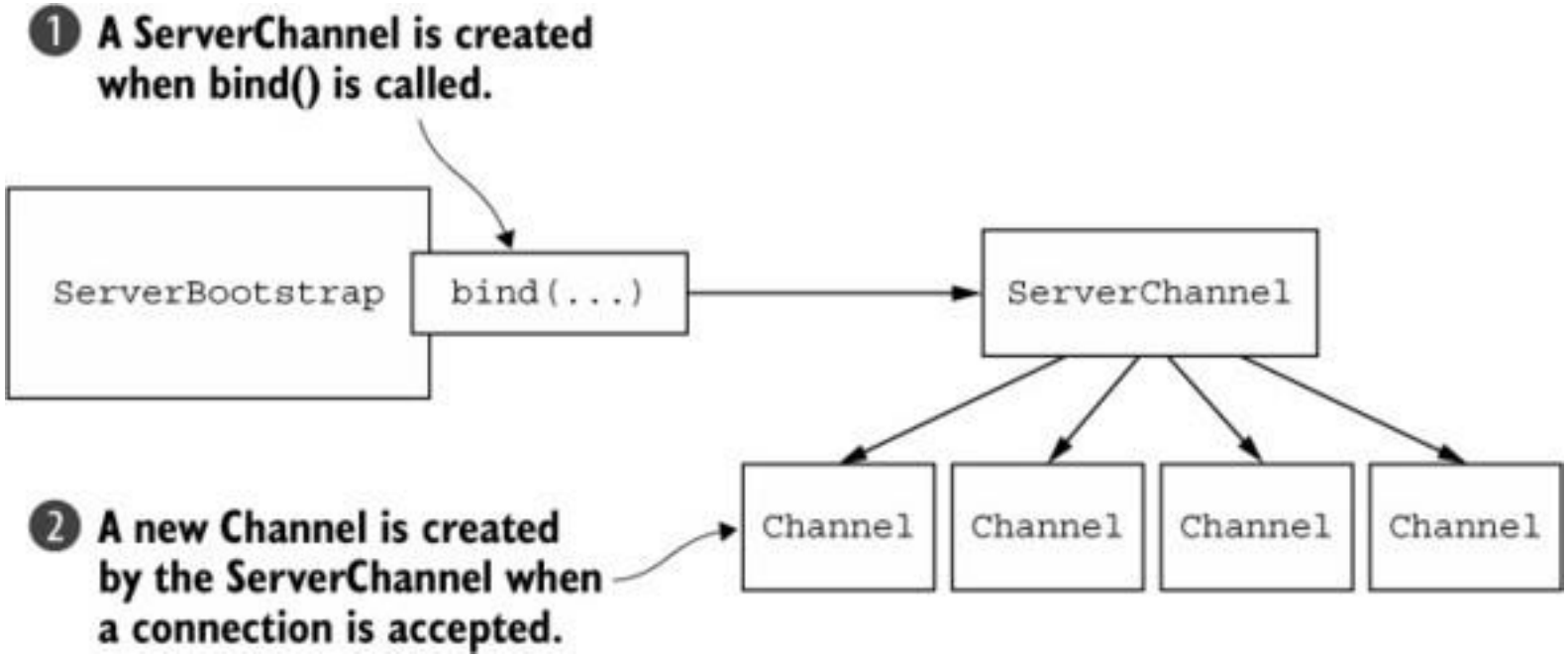
Netty

Bootstrap



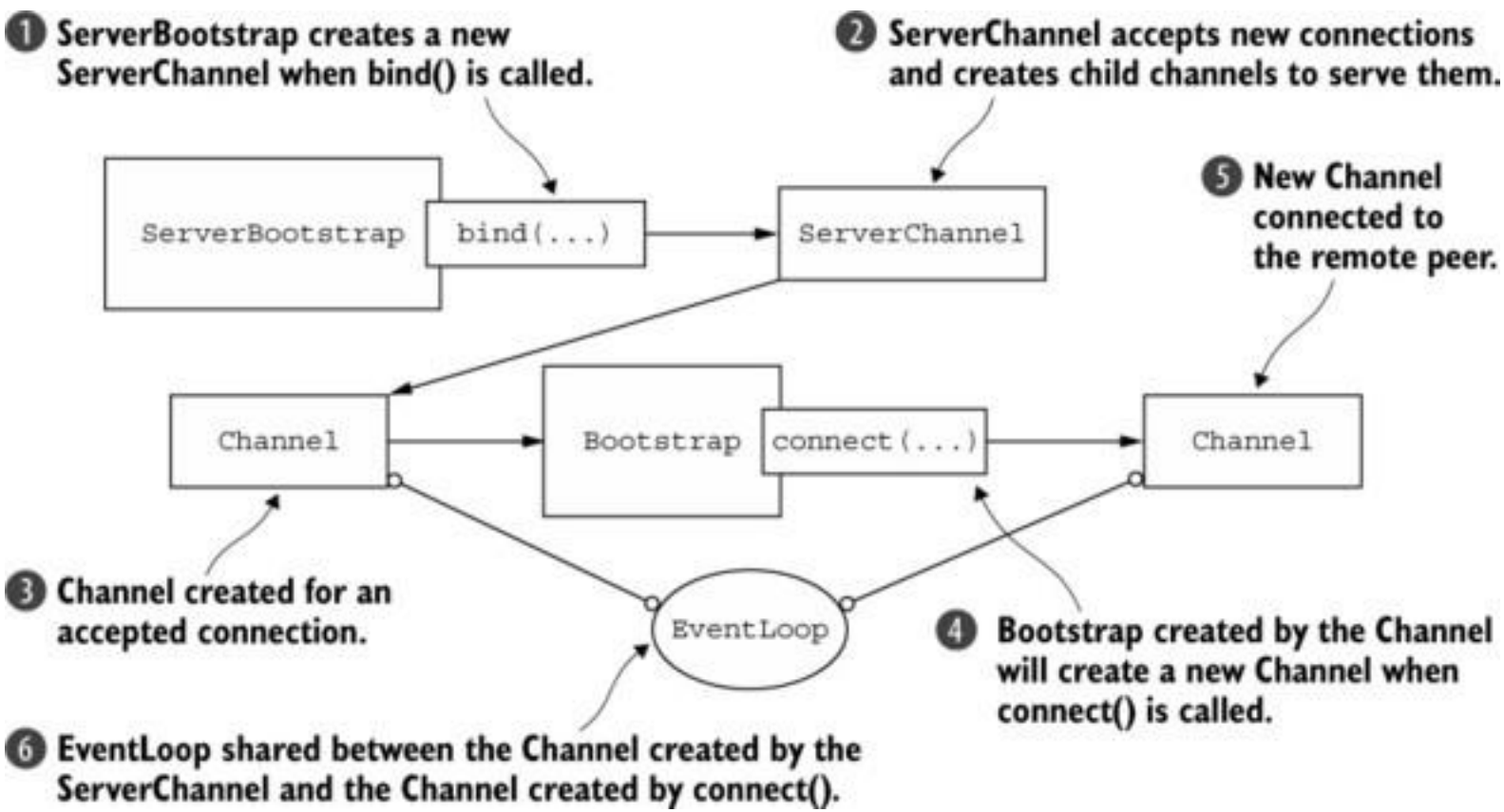
Netty

Bootstrap



Netty

Bootstrap



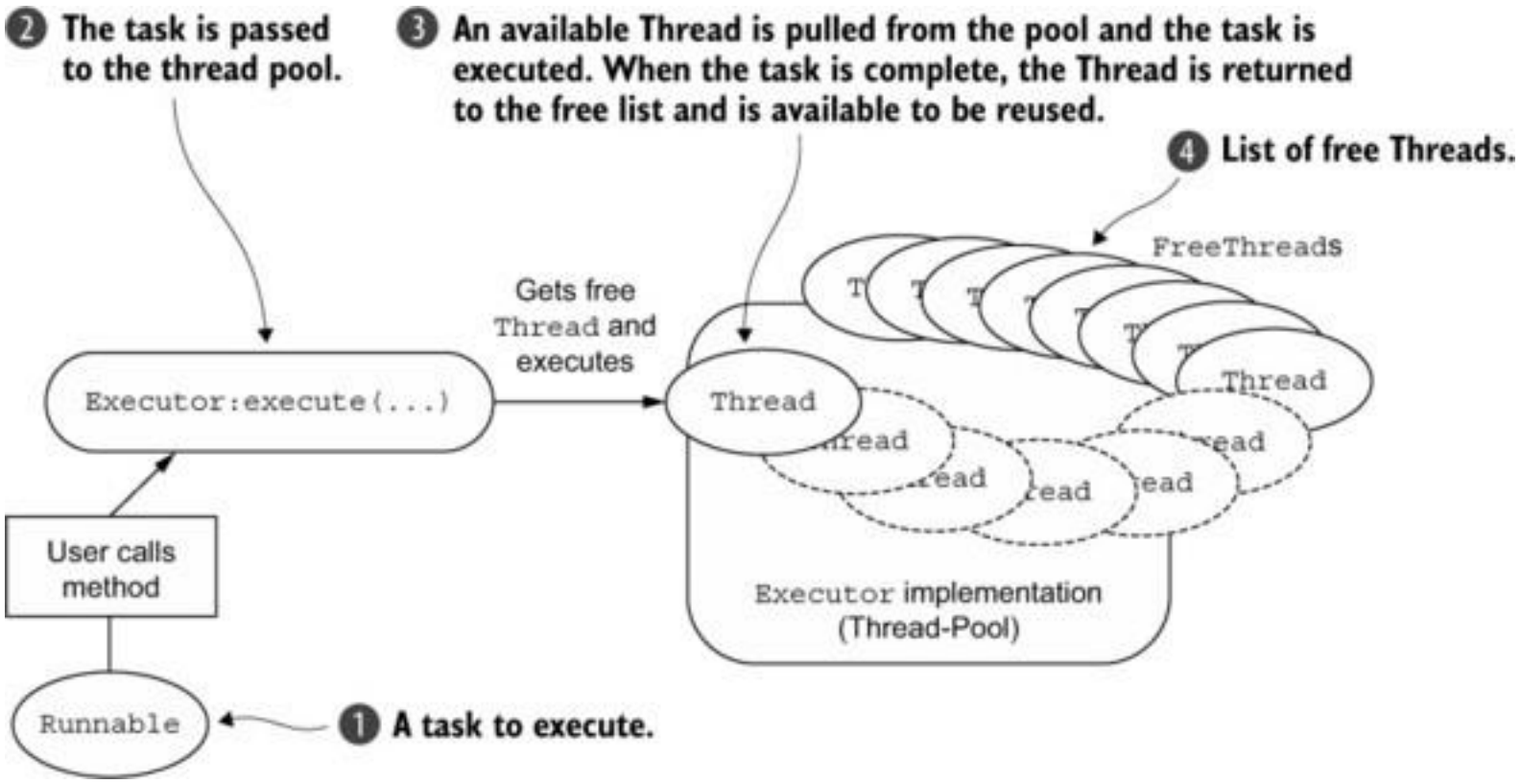
EventLoop

EventLoop

- Netty는 **event loop** 기반 프레임워크 : Event Queue + Event Loop
- **single thread event loop** 기반 프레임워크 : **node.js**
- **multi thread event loop** 기반 프레임워크 : **Netty, Vert.x**

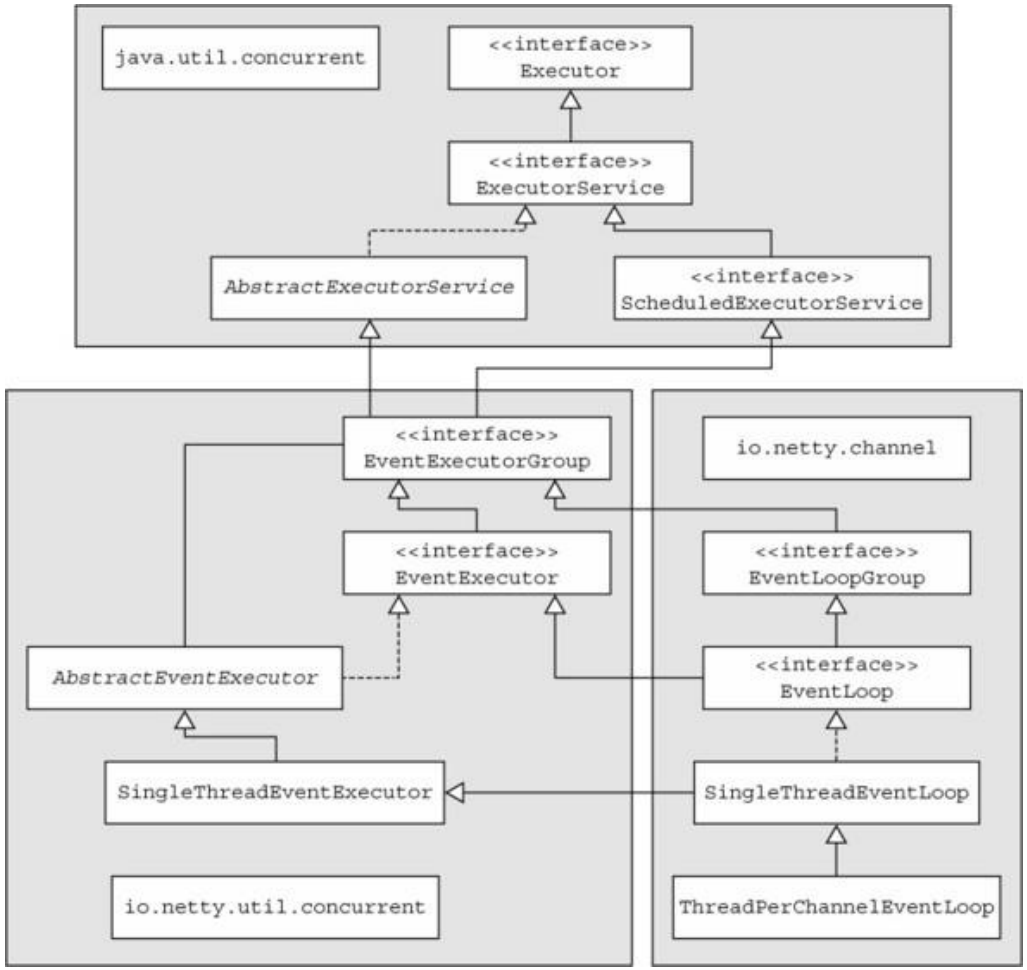
Netty

EventLoop



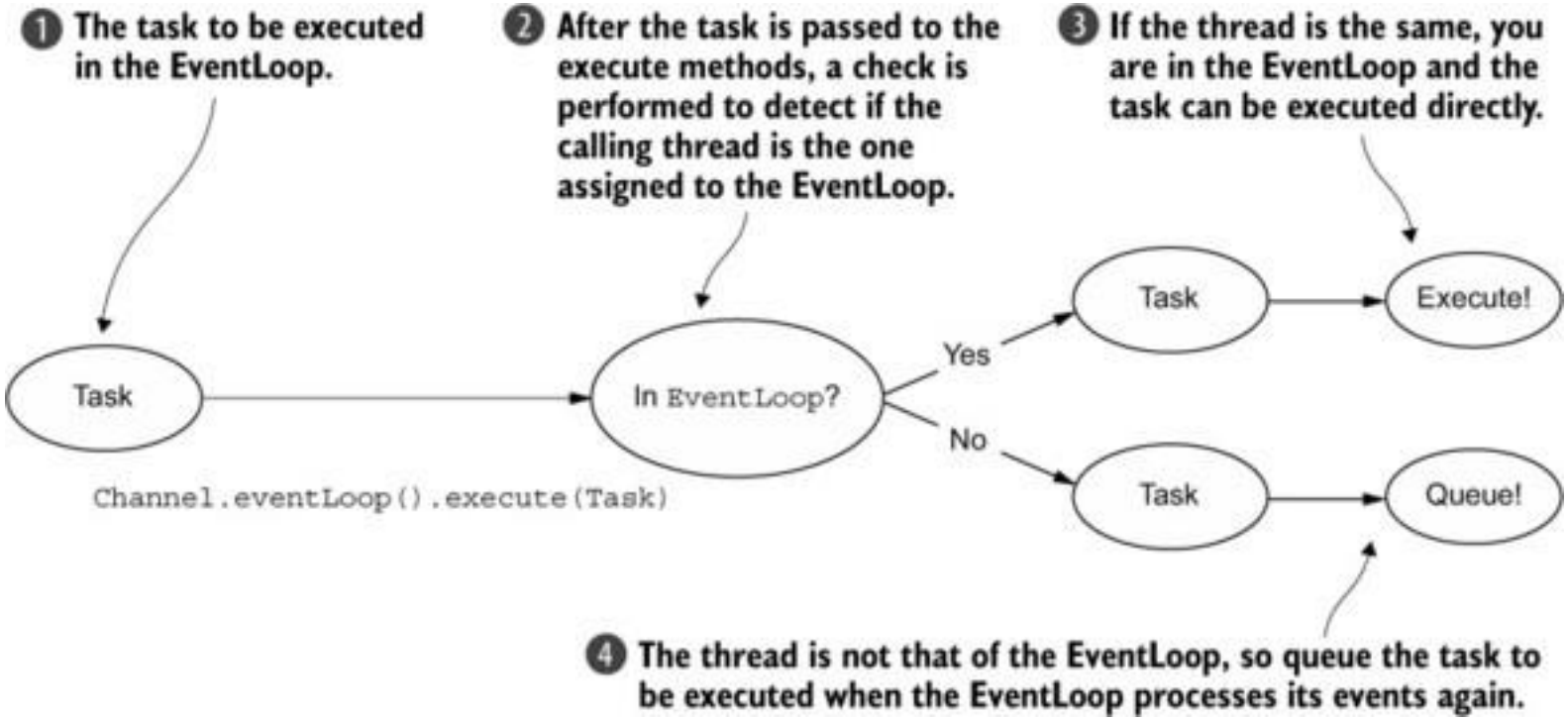
Netty

EventLoop



Netty

EventLoop



EventLoop

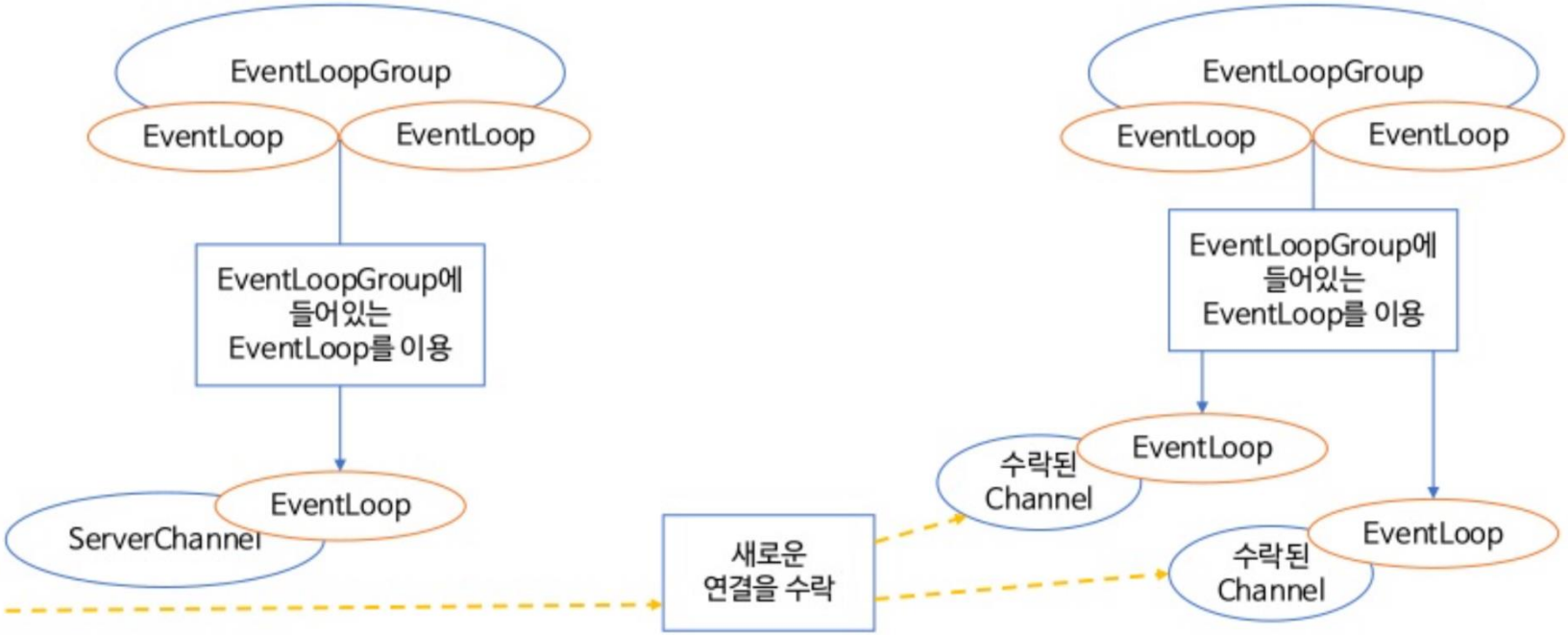
Channel, Thread, EventLoop, EventLoopGroup 간의 관계

- 하나의 EventLoopGroup은 하나 이상의 EventLoop를 포함
- EventLoopGroup은 Channel에 EventLoop를 할당
- 하나의 EventLoop는 수명주기 동안 하나의 Thread로 바인딩
- 하나의 Channel은 수명주기 동안 하나의 EventLoop에 등록
- 하나의 EventLoop를 하나 이상의 Channel로 할당 가능
- EventLoop가 생성 및 할당 되는 방법은 전송의 구현에 따라 다름 (Blocking 또는 Non-Blocking)

EventLoop

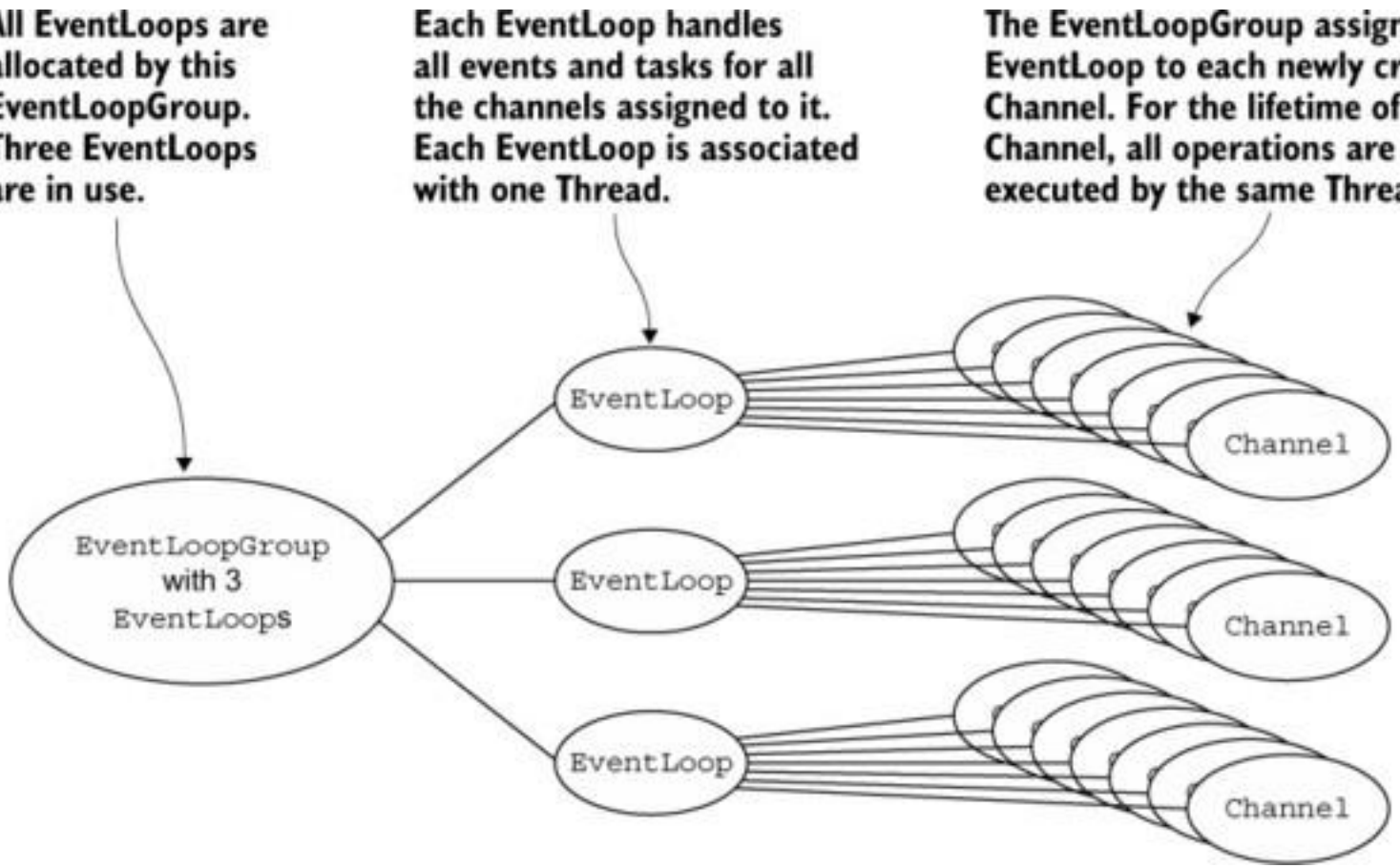
ServerBootstrap은 각기 다른 Channel의 두 집합을 필요로 함

1. 로컬 포트로 바인딩된 서버 자체의 수신 소켓을 나타내는 ServerChannel
2. 서버가 수락한 연결마다 하나씩 들어오는 클라이언트 연결을 처리하기 위해 생성된 모든 Channel



Netty

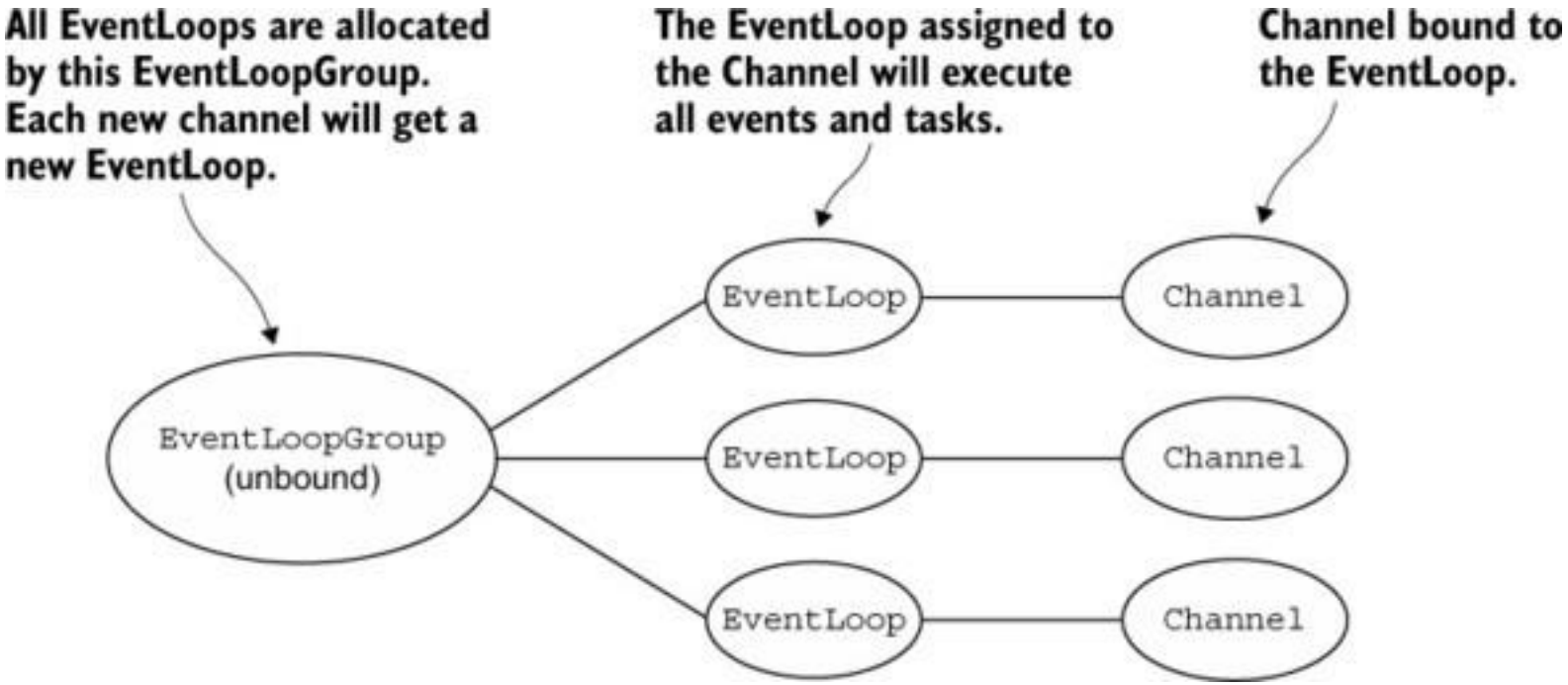
EventLoop



Non-Blocking

Netty

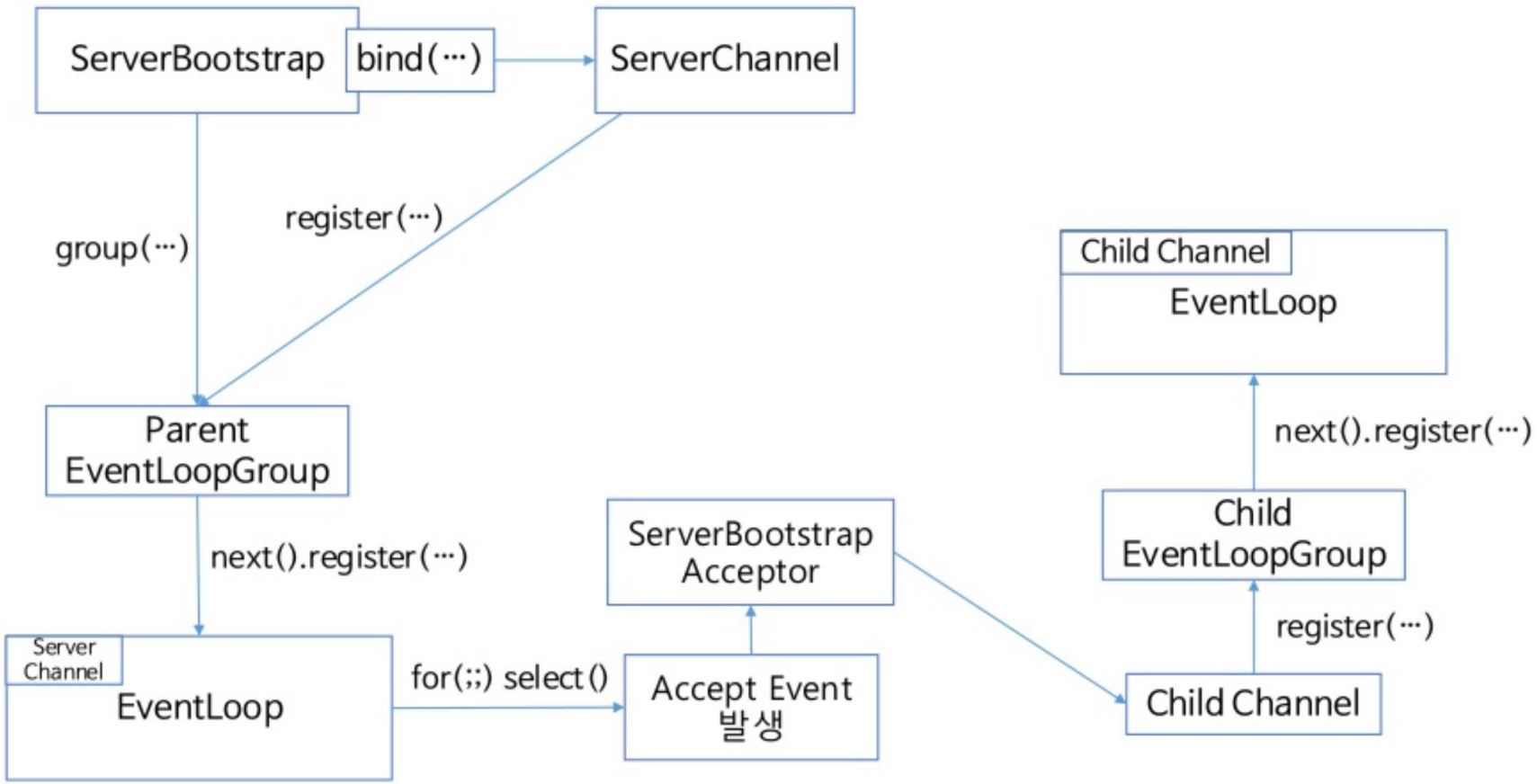
EventLoop



Blocking

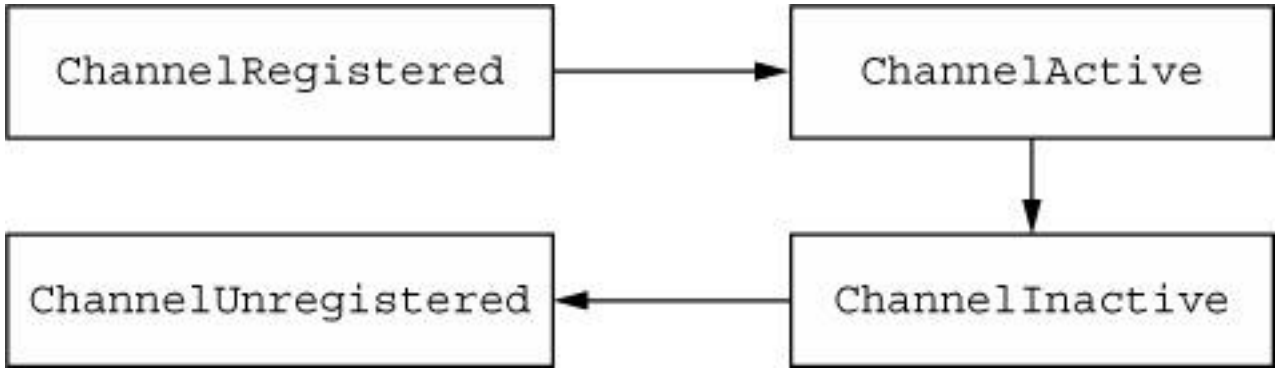
Netty

EventLoop



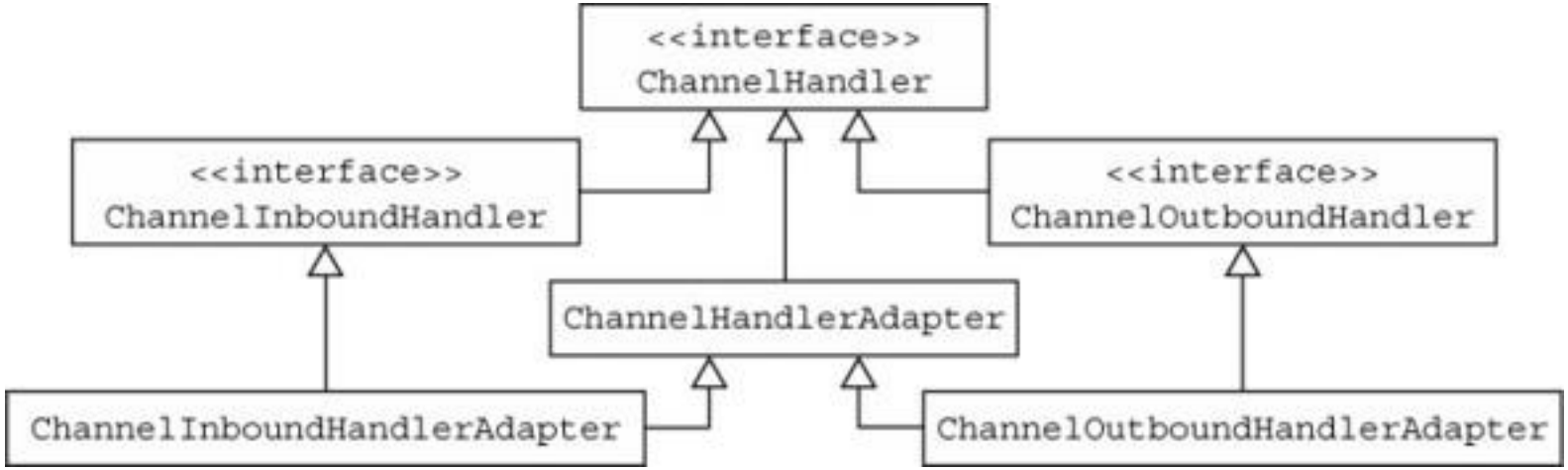
ChannelHandler
ChannelPipeline

ChannelHandler, ChannelPipeline



상태	기술
ChannelUnregistered	채널이 생성되었지만 EventLoop에 등록되지 않았습니다.
ChannelRegistered	채널이 EventLoop에 등록됩니다.
ChannelActive	채널이 활성 상태입니다 (원격 피어에 연결됨). 이제 데이터를 받고 보낼 수 있습니다.
ChannelInactive	채널이 원격 피어에 연결되어 있지 않습니다.

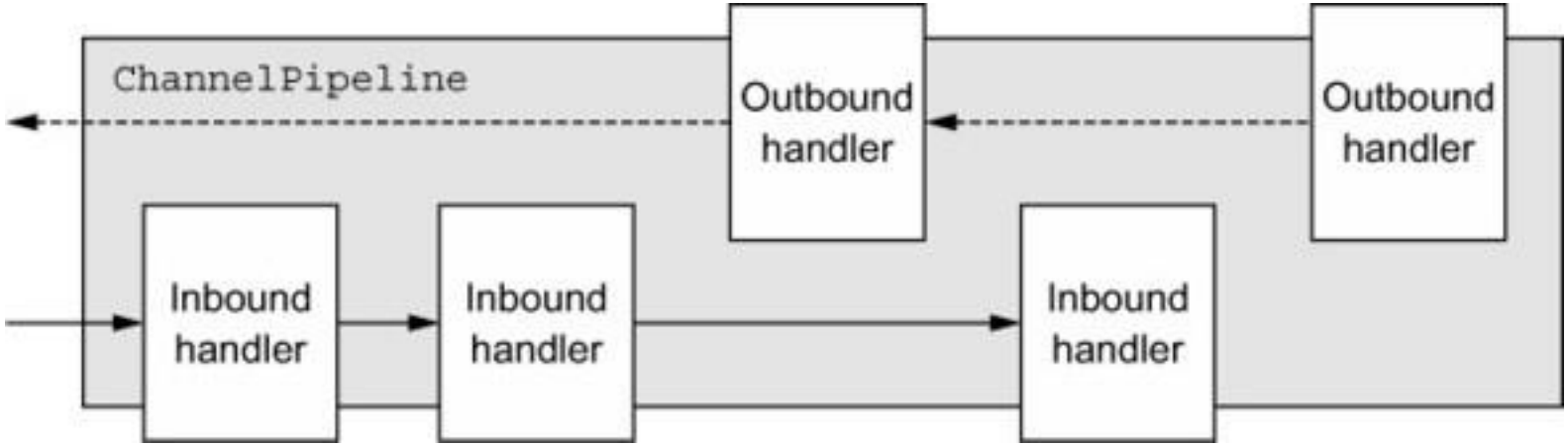
ChannelHandler, ChannelPipeline



유형	기술
handlerAdded	ChannelHandler가 ChannelPipeline에 추가 될 때 호출됩니다.
handlerRemoved	ChannelPipeline에서 ChannelHandler가 제거 될 때 호출됩니다.
exceptionCaught	처리 중 ChannelPipeline에서 오류가 발생하면 호출됩니다.

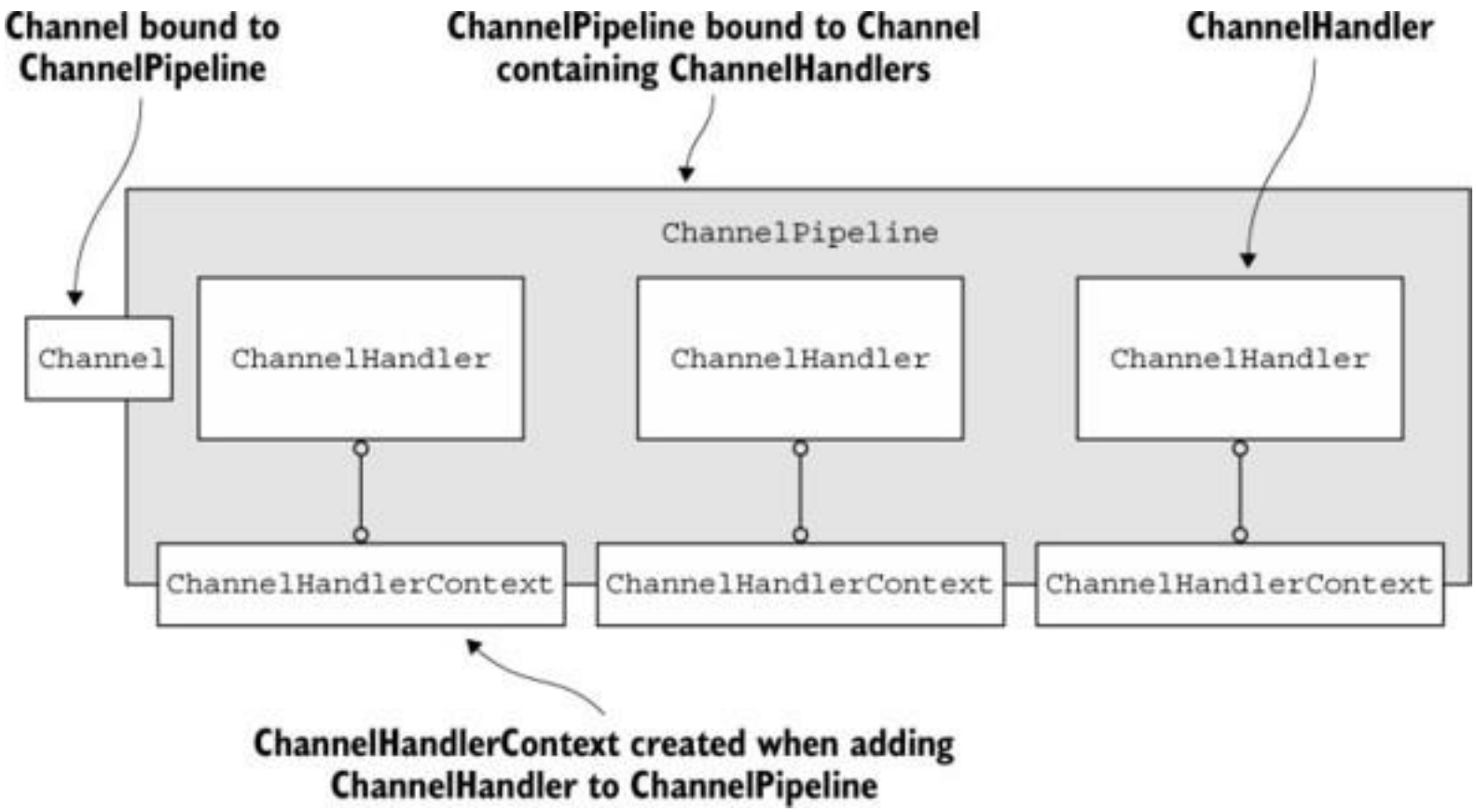
Netty

ChannelHandler, ChannelPipeline



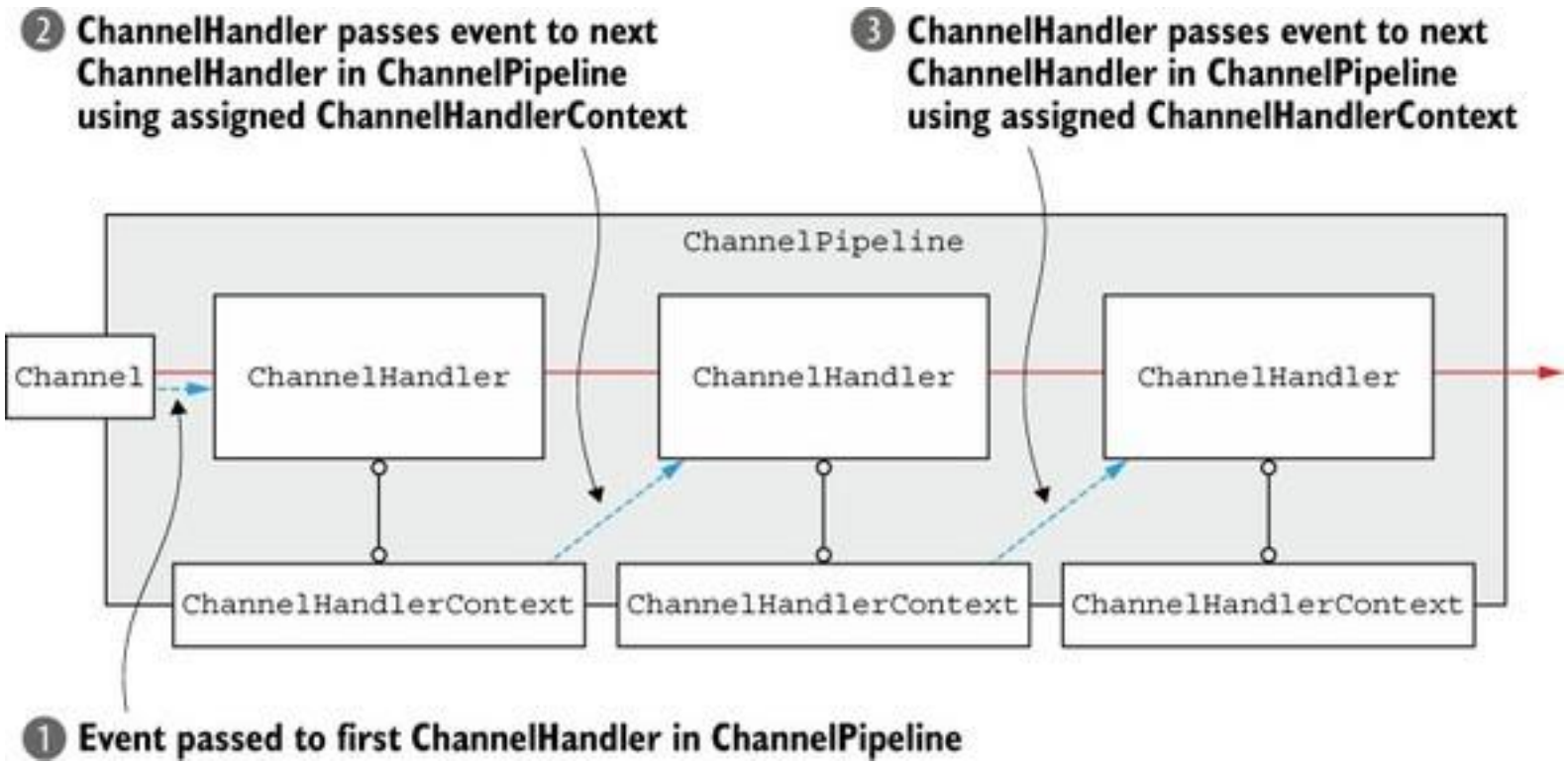
Netty

ChannelHandler, ChannelPipeline



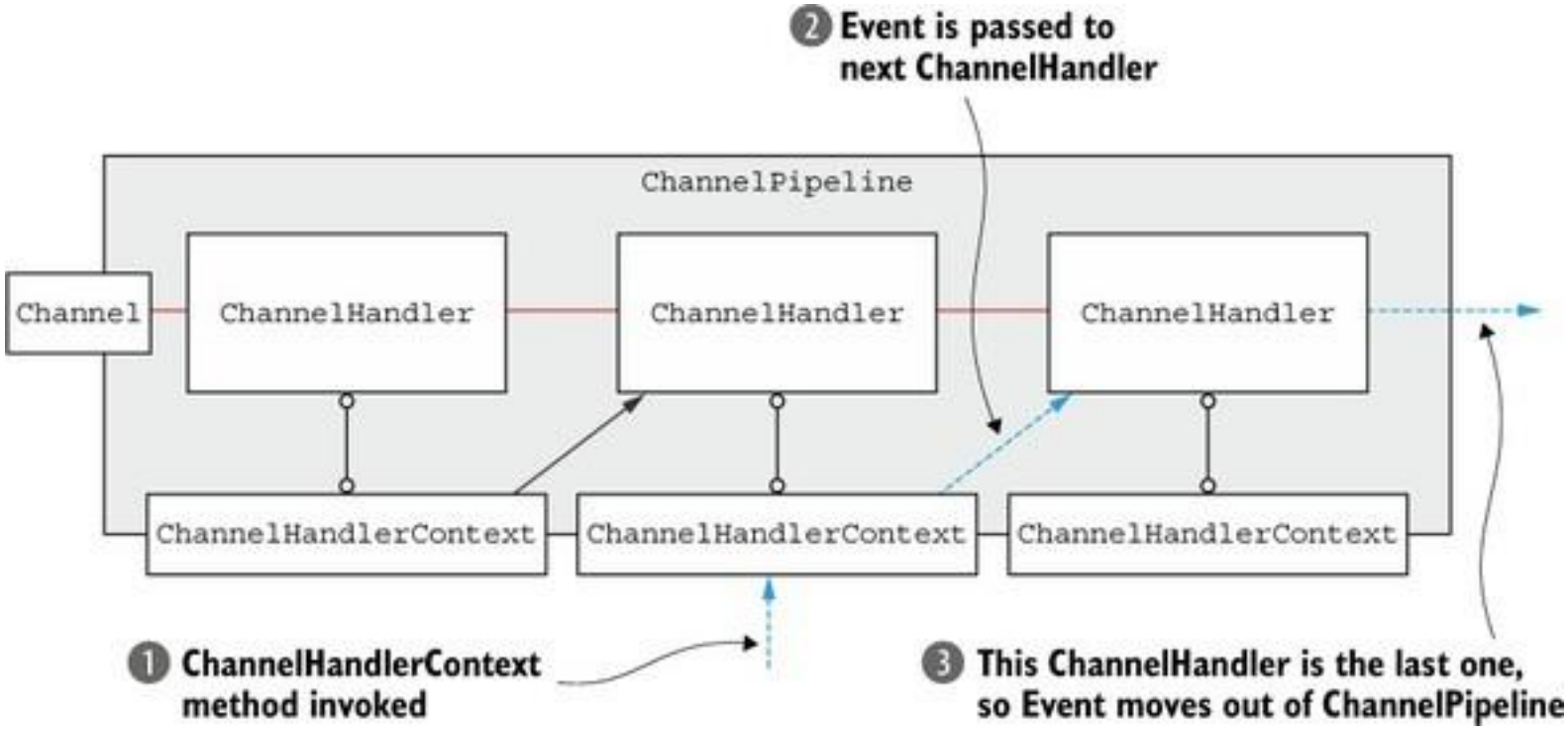
Netty

ChannelHandler, ChannelPipeline



Netty

ChannelHandler, ChannelPipeline



Netty

ChannelHandler, ChannelPipeline