Reactor Pattern

Agenda

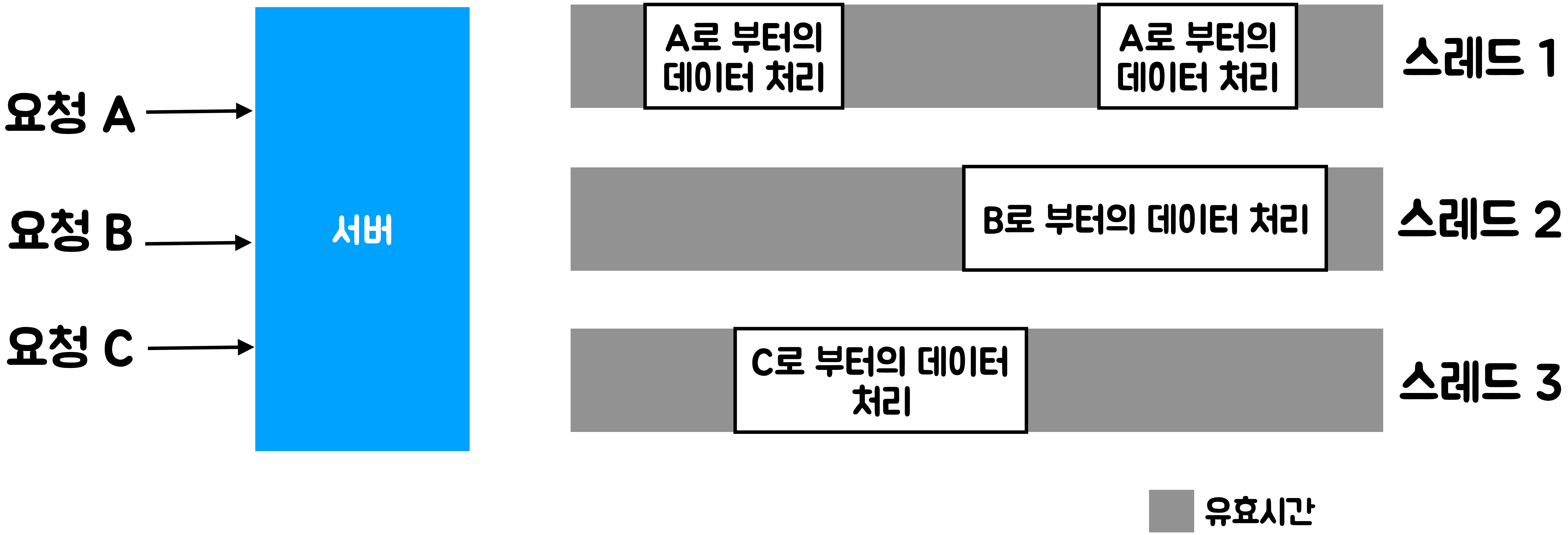
- Blocking, Non-Blocking I/O Model

- Event Demultiplexing

- Reactor Pattern

Blocking I/O Model

data = socket.read(); // 데이터를 사용할 수 있을 때 까지 블록됨



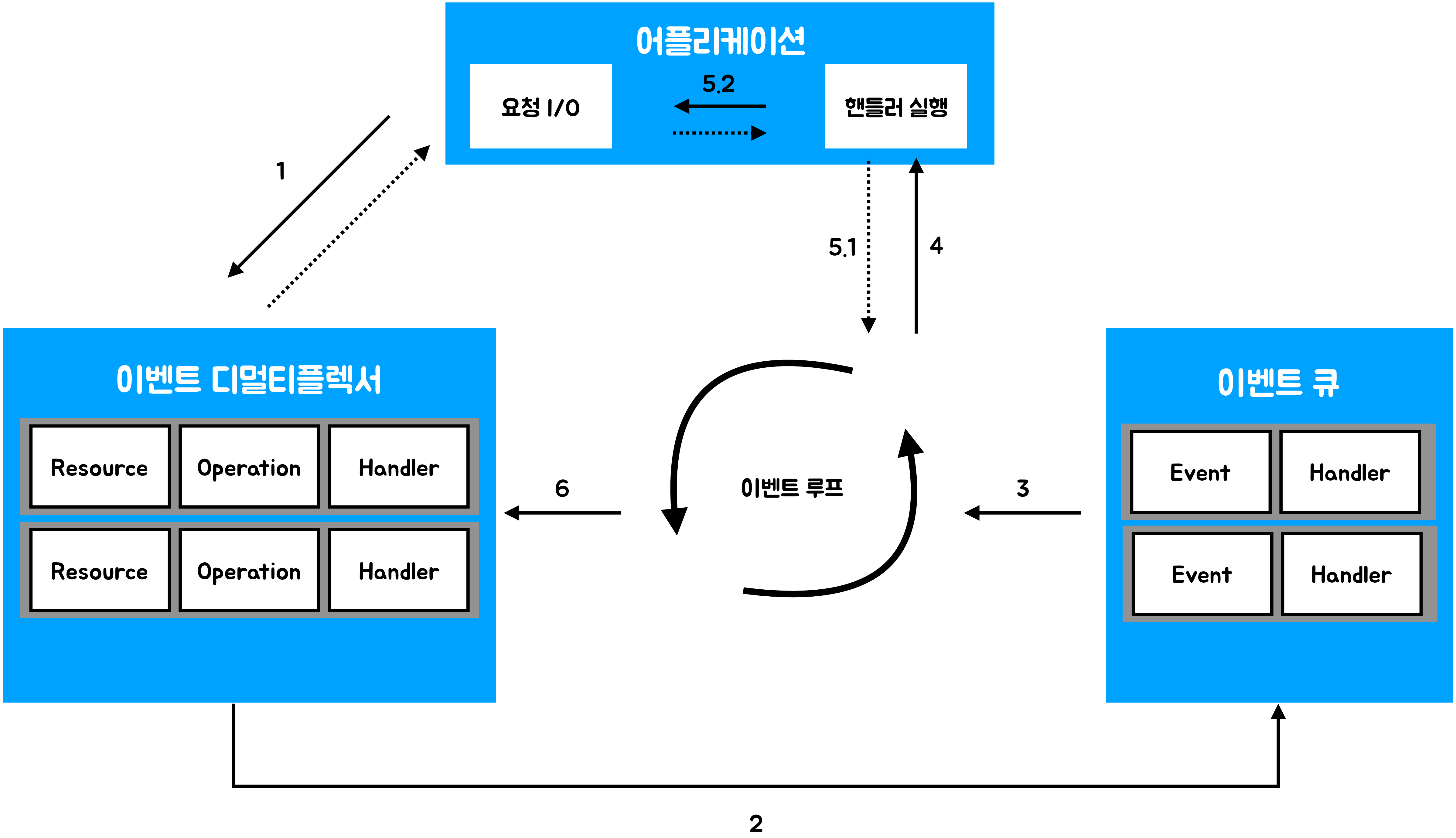
Non-Blocking I/O Model

```
resources = [socketA, socketB, pipeA]
while(!resources.isEmpty()) {
    for(i=0;i<resources.length;i++){
        resource = resouces[i]
        let data = resouces.read()
        if (data == NO_DATA_AVAILABLE) continue
        if (data == RESOURCE_CLOSED) resouces.remove(i)
        else consumeData(data)
    }
}
```

Event Demultiplexing

```
socketA, pipeB
watchedList.add(socketA, FOR_READ)
while(events = demultiplexer.watch(watchedList)) {
    foreaceh(event in events) {
        data = event.resource.read()
        if (data == RESOURCE_CLOSED)
            demultiplexer.unwatch(event.resource)
        else consumeData(data)
    }
}
```

Reactor Pattern


어플리케이션
요청 I/O
5.2
핸들러 실행
1
5.1
4
이벤트 디멀티플렉서
Resource
Operation
Handler
Resource
Operation
Handler
6
이벤트 루프
3
이벤트 큐
Event
Handler
Event
Handler
2