자바 기반의 스프링 Web MVC 와 WebFlux 성능 분석

정명교*, 서태원**
*고려대학교 컴퓨터정보통신대학원 소프트웨어공학과
mkyo@korea.ac.kr, suhtw@korea.ac.kr

A Study on Tools for Agent System Development

Myung-Kyo Jung*, Taeweon Suh**
*Graduate School of Computer & Information technology, Korea University
** Graduate School of Computer & Information technology, Korea University

요 약

논블로킹 IO 를 활용한 웹 서비스를 위한 미들웨어 구축 방법은 2009 년 발표된 Node.js 에서 도입된 이후로 여러 언어 및 프레임워크로 전파되기 시작하였다. 자바 진영에서도 Project Reactor 를 통하여 논블로킹 IO 패러다임에 대응하기 시작하였고 이를 스프링 프레임워크로 구현한 WebFlux 가 출시되었다.

본 논문은 자바 기반의 웹서비스 구축 시 스프링 프레임워크를 활용한 블로킹 기법과 논블로킹 기법 간의 차이점을 살펴보고 성능을 분석한다. 이를 통해 가장 효율적인 성능을 발휘할 수 있는 아키텍처 모델을 도출한다.

1. 서론

자바 언어는 비교적 쉬운 문법 및 사용성을 바탕으로 국내에서 수많은 IT 서비스의 사용 언어로 채택되어왔다. 이후 2000 년 초에 스프링 프레임워크가 소개된 뒤 제어 역전과 의존성 주입뿐 아니라 Web MVC 같은 편리한 도구를 통하여 웹 서비스를 쉽고 빠르게 구축하는데 큰 기여를 했다. [1]

스프링의 Web MVC 는 동시 접속 사용자 처리를 위하여 블로킹 IO 모델을 활용한다. 이는 처리 가능한 동시 접속자의 수만큼 스레드를 생성하여 각각의 스레드가 요청이 완료될 때까지 점유 되어있는 상태로 존재하며 클라이언트에게 응답을 돌려줄 때 해당 스레드가 다시 가용한 상태로 전환되는 방식이다.

하지만 클라우드 환경을 기반으로 한 IT 서비스 인프라 환경이 늘어나기 시작하면서 웹 서비스 아키텍처를 구축 시 더 적은 자원으로 더 빠르게 서비스를 제공하려는 수요가 늘어나기 시작하였고, Node.js 에서는 이벤트 루프를 활용한 논블로킹 IO 를 선보이면서 웹 서비스 아키텍처에 새로운 패러다임이 제시되었다. [2]

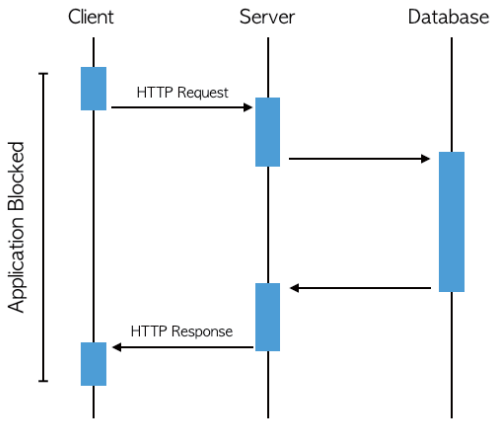
Node.js 의 성공을 통해 자바 진영에서도 논블로킹 IO 모델에 대한 필요성이 꾸준히 제시되기 시작하였고, Java 8 버전부터 함수형 프로그래밍 문법을 일부 지원하기 시작한 이후로 논블로킹 IO 를 자바에서 구현할 수 있는 도구를 제공하기 시작하였다. 스프링 프레임워크 5 버전부터 이를 활용한 웹 프레임워크를 구축하는 도구를 WebFlux 라는 이름으로 지원하기 시작하였다.

본 연구에서는 Web MVC 와 WebFlux 간의 구조적 차이에 대하여 알아보고 실험을 통하여 둘의 성능을 비교하여 그 결과를 바탕으로 웹 어플리케이션 서버 (WAS) 구축 시 최적의 성능을 위한 방법에 대하여 제안한다.

2. 관련연구

2.1 블로킹 IO 방식

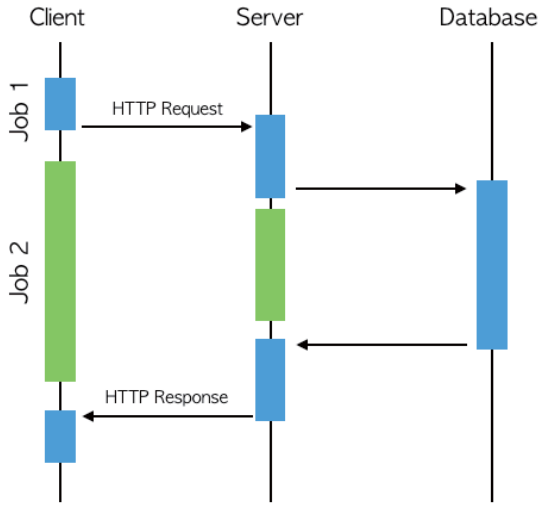
클라이언트로부터 요청이 발생한 시점부터 서버가 이를 처리하고 클라이언트에게 응답을 주기 전까지 대기하는 구조이다. 하나의 스레드가 요청을 처리하는 동안에는 다른 서비스를 수행할 수 없는 블로킹 상태가 된다.



(그림 1) 블로킹 방식의 작업 흐름

2.2 논블로킹 IO 방식

클라이언트가 서버에 요청을 전달한 뒤 대기하지 않고 다른 처리를 수행한다. 처리가 완료된 뒤에는 미리 등록된 콜백 함수를 호출하여 후처리를 진행한다. 이런 방식은 요청을 받고 해당 요청에 대한 결과를 리턴하는 과정에서 다른 태스크를 수행할 수 있기 때문에 다수의 스레드풀을 생성할 필요가 없다.



(그림 2) 논블로킹 방식의 작업 흐름

2.3 스프링 Web MVC

블로킹 방식의 웹 서비스를 구현하기 위해 제공되는 스프링 프레임워크의 라이브러리이다. MVC 는 Model-View-Controller 디자인 패턴의 약어이며 클라이언트로부터의 요청은 Dispatch Servlet 을 통하여 Controller 로 전달되며 이 때 동시 접속을 위한 사용자의 수만큼 스레드가 생성되는 구조이다.

2.4 스프링 WebFlux

논블로킹 IO 처리를 위해 제공되는 스프링 프레임워크의 라이브러리이다. 스프링 프레임워크 5 버전부터 제공되기 시작하였으며 반응형 프로그램을 위한 API 스펙을 구현하였다.

2.5 반응형 프로그래밍

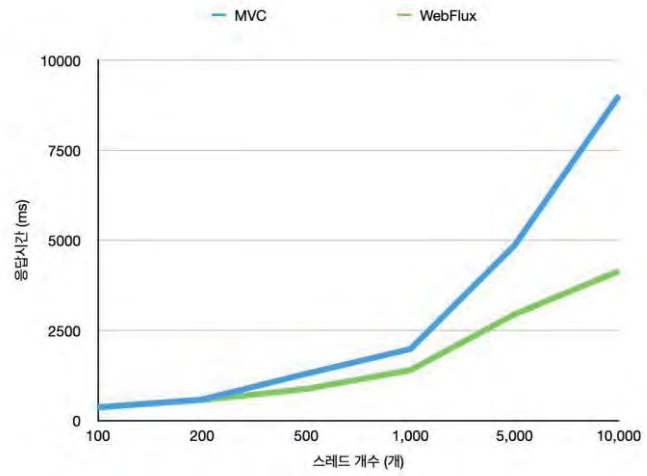
반응형이라는 용어는 IO 이벤트에 반응하는 네트워크 컴포넌트, 마우스 이벤트에 반응하는 UI 컨트롤러 등 이벤트에 반응하는 프로그래밍 모델을 의미한다. 이러한 의미에서 논블로킹 IO 는 반응형이라고 볼 수 있는데, 이는 논블로킹 IO 는 작업이 완료되거나 데이터가 사용 가능한 이벤트에 반응할 수 있는 구조를 갖추기 때문이다. [3]

3. 성능비교

실험은 다음과 같은 방식으로 진행하였다. 스프링 Web MVC, WebFlux 각각을 사용하여 구현한 웹서비스를 구축한다. WebMVC 의 경우 동시접속자는 최대 200 명으로 설정하였다. 테스트 클라이언트에서는 다수의 동시접속자가 접속하는 상황을 가정하여 각 서버에 동시다발적으로 요청을 발생시킨다. 각 서버별로 동시접속자 수에 따른 응답시간을 밀리초 단위로 측정한다. 숫자가 낮을수록 응답속도가 더 빠른 것을 의미한다.

<표 1> 동시접속자 수에 따른 응답시간 비교
(단위: 밀리초)

동시접속자 수	Web MVC	WebFlux
100	363	374
200	583	568
500	1,299	876
1,000	1,987	1,399
5,000	4,867	2,980
10,000	9,018	4,149



(그림 3) 블로킹 방식의 작업 흐름

4. 결과

200 개 이하의 동시 요청에 대해서는 두 서비스간의 응답속도가 동일하지만, 200 을 초과하는 요청부터는 응답속도의 차이가 관찰되었으며 동시접속자의 수가 많을수록 응답속도의 차이가 더 커지는 것을 확인할 수 있다. 이를 통해 다중 사용자에 대한 HTTP 요청 처리시 WebFlux 가 Web MVC 보다 빠르거나 동일한 성능임을 알 수 있다. 즉 구축하는 서비스의 특성상 다량의 동시접속 사용자를 대응해야 하는 경우는 WebFlux 를 사용하여 구축을 하는 것이 성능상으로 더 효과적이며 동시접속 사용자가 적은 서비스의 경우는 Web MVC 로 구축하더라도 동일한 성능을 발휘할 수 있다. 향후에는 실제 웹서비스 운영 환경과 동일한 구조로 데이터베이스를 구성하고 DB 커넥션풀의 논블로킹 여부에 따른 성능 차이를 분석하여 운영 환경에서의 서비스 성능 최적화가 이루어지도록 할 예정이다.

참고문헌

[1] Johnson, Rod, et al. Professional Java development with the Spring framework. John Wiley & Sons, 2009.
[2] Satheesh, Mithun, Bruno Joseph D'mello, and Jason Krol. Web development with MongoDB and NodeJs. Packt

Publishing Ltd, 2015.
[3] Define “Reactive” https://docs.spring.io/spring/docs/
current/spring-framework-reference/web-reactive.html
#webflux-why-reactive