The Journal of The Institute of Internet, Broadcasting and Communication (IIBC)
Vol. 16, No. 1, pp.69-75, Feb. 29, 2016. pISSN 2289-0238, eISSN 2289-0246

http://dx.doi.org/10.7236/JIIBC.2016.16.1.69
JIIBC 2016-1-9

웹 응용 서비스에서 성능 분석을 위한 실험적인 평가 기법

Experimental Evaluation Method for Performance Analysis in Web Application Services

김남윤*

Namyun Kim*

요 약 대화형 웹 환경에서 응용 프로그램의 성능은 고품질의 서비스를 제공하기 위한 필수적인 이슈이다. 즉, 웹 요청 트랙픽이 일시적으로 급증할 경우, 서버의 CPU 여유 시간이 부족하게 되고 결국 요청의 응답 시간이 증가하기 때문에 클라이언트에게 좋지 않은 경험을 유발하게 된다. 본 논문에서는 J2EE 응용 서버에서 설정 가능한 파라미터인 스레드 풀과 데이터베이스 연결 풀의 크기가 응용 프로그램의 성능에 끼치는 영향을 분석한다. 또한 최적의 파라미터 값을 얻기 위해서 웹 성능 분석을 위한 실험적 평가 기법을 소개한다. 마지막으로 사례 연구를 통해 성능 분석 결과를 제시한다.

Abstract The performance of a web application is an essential issue to provide high quality of the services in interactive web environments. On a sudden increase in traffic in a very short span of time, the servers became CPU starved and would become unresponsive. This would lead to a bad experience for the clients of web service. This paper deals with the effects of two configurable software settings of J2EE application servers: the maximum size of the thread pool and the maximum size of database connection pool. In order to figure out the optimum configuration value, this paper builds experimental evaluation method for web performance analysis. Finally, a case study related with the proper experimental method is presented with performance result.

Key Words : Web Application Server, Performance Evaluation, Thread Pool, DB Connection Pool, Spring Framework

Ⅰ. 서 론

기존의 데스크탑뿐만 아니라 스마트폰 및 태블릿 PC와 같은 모바일 단말기의 증가로 인해 웹 서비스의 성능이 점차 중요해지고 있다[1,2]. 특히 클라우드 환경에서 웹 서비스의 확장성 및 비용 절감을 위해서 서버의 효율적인 운용이 더욱 절실하다고 할 수 있다. 이러한 웹 서비스의 성능은 서버의 설정 파라미터에 많은 영향을 받기 때문에 클라이언트의 부하와 시스템 자원을 고려하여 적절한 파라미터 값을 설정하여야 한다[3].

최근까지 J2EE 응용 서버에서 설정 가능한 파라미터 값이 응용 프로그램의 성능에 끼치는 영향에 대한 연구가 수행되었다. 특히 스레드풀에 존재하는 스레드의 최대 개수, DB 연결 풀에 존재하는 연결의 최대 개수가 시스템의 처리량(throughput)에 미치는 영향을 조사하였다. 그러나 이러한 성능 결과는 웹 시스템의 환경 혹은

*정회원, 한성대학교 정보시스템공학과
접수일자: 2015년 11월 3일, 수정완료: 2015년 12월 26일
게재확정일자: 2016년 2월 5일

Received: 3 November, 2015 / Revised: 26 December, 2015 / Accepted: 5 February, 2016
*Corresponding Author: nykim@hansung.ac.kr
Dept. of Information System Engineering, Hansung University, korea

응용 프로그램의 특성에 매우 가변적으로 나타나는 경향이 있다. 따라서 성능 분석을 위한 체계적인 실험적 모델을 이용하여 응용 환경에 따라 반복 실험하는 것이 무엇보다 중요하다. 본 논문에서는 응용 서버의 동작 흐름과 설정 가능한 파라미터를 분석하고 Apache JMeter[4], JConsole[5]을 이용하여 성능을 분석하는 방법을 제안한다. 즉, JMeter를 사용하여 '오픈 워크로드' 모델하에서 부하를 생성하고 종단간(end-to-end) 응답시간 및 처리량을 도출한다. 그리고 JConsole을 이용하여 시스템 자원(CPU, Memory, Thread 수)를 모니터링하는 과정을 소개한다. 마지막으로 Spring 프레임워크[6]에 기반한 응용 프로그램을 작성하여 웹 응용 프로그램의 성능을 분석하는 사례를 제시하고 성능 분석 결과를 제시한다. 본 실험을 통한 성능 분석 결과는 다음과 같다. 일시적으로 트래픽이 급증할 경우 서버내의 스레드 풀 크기가 작으면 하드웨어 자원을 충분히 활용하지 못하기 때문에 응답 시간이 증가하고 안정적인 상태로 복귀하는데 상당한 시간이 걸린다. 한편, 스레드 풀 크기가 지나치게 크다면 스레드간 문맥 교환과 같은 오버헤드로 인해 CPU 이용률이 증가하기 때문에 응답 시간이 증가하고 불규칙적인 특징을 나타내고 있다. 결국, 일시적인 과부하에 안정적으로 동작할 수 있도록 모니터링을 통해 적절한 값을 선택하는 것이 중요하다고 할 수 있다.

본 논문의 구성은 다음과 같다. 2절에서는 웹 서비스의 성능 분석과 관련한 기존 연구를 소개하고 3절에서는 웹 응용의 성능 분석을 위한 실험적 방법론을 소개한다. 4절에서는 예제 응용 프로그램에 대한 실험 결과를 분석하고 5절에서 결론을 맺는다.

II. 관련 연구

웹 응용 서버의 설정 파라미터가 응용 프로그램의 성능에 미치는 영향에 대해 다양한 연구가 수행되었다[7,8,9]. 기존 연구 [7,8]에서는 클라우드 환경에서 톰캣 서버내 스레드 개수와 DB 연결 수가 시스템에 끼치는 결과는 제시하였다. E-commerce와 같은 대화형 응용에서는 응답시간이 매우 중요하기 때문에 임계치내에(예: 1,2초) 응답 시간을 가지는 요청의 처리량 즉, goodPut을 성능 척도로 분석하였다. 이 연구 결과에 따르면, 스레드 수 및 DB 커넥션 수가 작게 설정될 경우 하드웨어 자원을 충분히 활용하지 못하기 때문에 전체적으로 좋지 않은 성능을 나타낸다. 따라서 하드웨어 자원을 추가적으로 설치하더라도 성능을 향상시킬 수가 없다. 또한 스레드 수 및 DB 커넥션 수가 크다면 스레드간 문맥 교환과 JVM GC(garbage collection) 오버헤드로 인해 응용 프로그램에 가용한 자원이 감소하여 goodPut 처리량이 떨어진다는 사실을 보여주었다. 즉, 시스템 자원이 포화상태에 도달하여 성능이 저하됨을 보여주고 있다. 한편 연구 [9]에서는 응용 서버의 스레드 수와 TCP 연결 큐의 크기에 따른 실험 결과와 함께 큐잉 모델링 기법을 제시하였다. 이 연구에서는 TCP 연결 큐가 클라이언트 수보다 작을 경우, 큐 오버플로로 인해 에러율이 증가함을 보여주었다. 특히 스레드 수가 많을수록 CPU 경쟁으로 에러율이 더욱 증가하고 기존 요청의 응답시간은 감소하는 결과를 제시하였다.

지금까지의 기존 연구도 응용 서버의 설정 파라미터에 따른 성능 결과를 제시하고 있지만, 일시적인 과부하 상태에서 클라이언트 요청의 응답 시간 및 그 분포에 대한 자세한 결과를 제시하지 않았다. 본 논문에서는 '오픈 워크로드' 모델하에서 일시적으로 트래픽을 급격히 발생시켜 응용 프로그램의 반응 정도를 파악하고 그 이유를 제시한다. 또한 응용 프로그램의 성능 분석 도구를 활용한 실험 방법을 제시하여 웹 시스템을 모니터링하고 분석할 수 있는 사례를 제시한다.

III. 성능 분석을 위한 실험적 평가 모델

1. 테스트 베드 환경

웹 서비스는 주로 3-tier 구조로 개발된다. 즉, 웹 서버, 응용 서버, DB 서버와 같이 독립적으로 분리되어 설계된다. 클라이언트 요청은 웹 서버로 전달이 되고, 웹 서버는 요청을 응용 서버로 분배한다. 그리고 응용 서버는 비즈니스 로직을 수행하며 데이터베이스에 질의를 한다. 본 논문에서는 응용 서버의 설정 파라미터가 시스템 성능에 끼치는 영향을 파악하기 위해 응용 서버, DB 서버로 구성된 2-Tier 시스템을 구축하여 실험을 수행하였다.

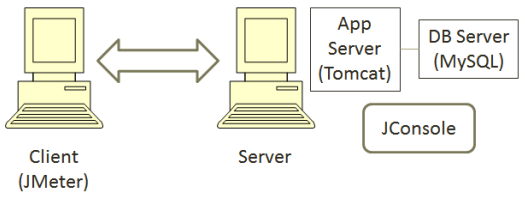


그림 1. 테스트베드의 하드웨어 및 소프트웨어 구성
Fig. 1. The Configuration of Hardware and Software Component in Testbed

그림 1은 클라이언트와 서버의 구성을 보여주고 있는데, 클라이언트 컴퓨터는 Apache JMeter를 사용하여 트래픽을 생성한 후, 종단간 응답 시간 및 처리량을 도시한다. 한편 서버 컴퓨터에서는 톰캣 응용 서버[10]와 MySQL 서버[11]가 동작하며 응용 서버 자원을 모니터링하기 위해 JConsole을 사용하였다. 표 1은 테스트 베드의 하드웨어 및 소프트웨어 구성을 보여주고 있다.

표 1. 웹 응용 서버의 하드웨어 및 소프트웨어 사양
Table 1. The Specification of Hardware and Software of Web Application Server

하드웨어	사양
프로세서	Pentium CPU 3.4 GHz, Dual Core
메모리	2GB
네트워크	100 Mbps
디스크	Intel SSD 330

소프트웨어	사양
운영 체제	CentOS 7
응용 서버	Apache Tomcat 8
DB 서버	MySQL 5.6.26
JDK	jdk_1.8.0

2. 웹 서비스 동작 모델

J2EE 응용 서버인 톰캣은 클라이언트와의 TCP 연결 요청을 수용하는 'acceptor thread'와 실제 작업을 수행하는 'worker thread pool'을 가지고 있다. 클라이언트의 요청시 톰캣의 동작 흐름은 그림 2와 같다.

1) 클라이언트와 서버는 OS 레벨에서 TCP 핸드쉐이크 과정을 거친다. TCP 연결이 완료되었을 경우 '완료 연결 큐(completed connection queue)'에 저장되는데, 톰캣 설정 화일에서 acceptCount는 큐의 크기를 제어하는데 사용된다.
2) 톰캣 'acceptor thread'는 Thread Pool에 있는 'worker thread'가 가용한지 파악한다. 만약 가용한 스레드가 존재할 경우 TCP 연결을 'worker thread'에게 전달한다. 한편 가용한 스레드가 존재하지 않을 경우에는 풀에 있는 스레드의 개수와 설정 파라미터인 maxThreads값을 비교한다. 만약 maxThreads보다 작을 경우 추가적으로 'worker thread'를 생성하고 maxThreads보다 작지 않을 경우에는 'worker thread'가 free할 때까지 대기한다.
3) 'worker thread'는 클라이언트의 요청을 읽고 처리한 후 응답을 전송하는 역할을 담당한다.
4) 'worker thread'가 DB를 접근하는 경우에 DB 연결 생성/삭제에 대한 오버헤드를 줄이기 위해 DB 연결 풀을 이용한다. 만약 DB 연결 풀에 여유가 없을 경우 'worker thread'는 대기한다.

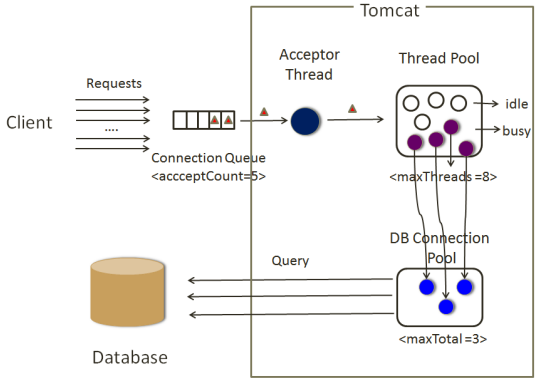


그림 2. 웹 서비스의 동작 흐름
Fig. 2. The Execution Flow of Web Services

그림 2에서 중요한 설정 파라미터는 다음과 같다.

1) 'acceptCount': OS의 '완료 연결 큐'의 크기를 지정하는 값이다. 일시적으로 트래픽이 급격히 증가하는 경우 acceptCount가 시스템에 끼치는 영향은 다음과 같다. 첫째 acceptCount 값이 작아서 큐가 full되는 경우에는 TCP 연결을 저장할 공간이 없기 때문에 'connection timeout error'가 발생할 수 있다. 둘째, acceptCount 값이 클 경우에는 TCP 연결이 큐에 저장될 수 있지만, 시스템이 처리할 수 있는 것 이상으로 클라이언트 요청이 들어올 경우, 모든 'worker thread'가 busy하게 되어 'acceptor thread'는 가용한 'worker thread' 가 존재할 때까지 대기하게 된다. 따라서 클라이언트 요청은 큐에서 장시간 머물게 되기 때문에 'read timeout error'

가 발생할 수 있다.

2) ‘maxThreads’ : ‘worker thread’ 풀에 존재할 수 있는 최대 스레드 개수를 지정하는 값이다. maxThreads 값이 작을 경우에는 멀티 스레딩의 장점을 살리지 못하게 되고 반대로 너무 클 경우에는 스레드 문맥 교환으로 CPU 이용률이 증가하게 되어 실질적인 작업을 수행하지 못하게 된다.
3) ‘maxTotal’: DB 연결 풀에 존재할 수 있는 최대 연결 개수를 지정하는 값이다. 이 파라미터 값은 maxThreads와 밀접한 연관을 가진다. 왜냐하면 충분한 ‘worker thread’가 존재하더라도 maxTotal이 작다면, ‘worker thread’ 는 커넥션을 얻기 위해서 대기하기 때문이다. 클라이언트 요청이 여러 개의 DB 연결을 얻는 경우에는 DB 연결 풀의 크기가 스레드 풀보다 크게 설정해야 하는 반면, 요청 기간동안 하나의 연결만을 유지하는 경우는 동일 값으로 설정한다.

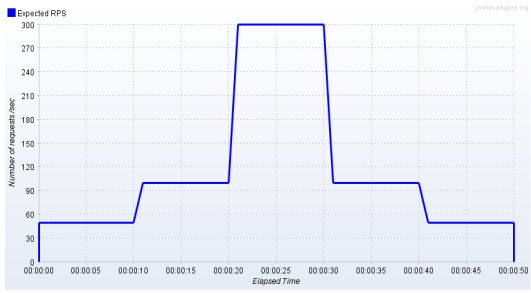
그림 2의 사례에서는 스레드 풀에 존재할 수 있는 최대 스레드 개수는 8개이고 현재 4개 스레드가 수행중임을 보여주고 있다. 한편, DB 연결 풀의 크기가 3이므로 하나의 스레드는 연결이 릴리스될 때까지 대기상태에 있게 된다.

3. 웹 성능 분석 도구

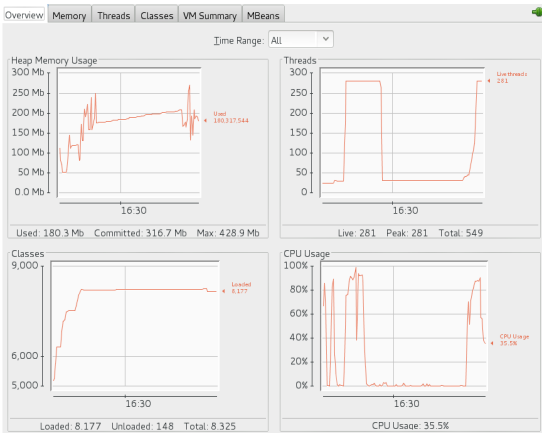
응용 프로그램의 성능 분석 도구로서 Apache JMeter는 성능 테스트용으로, JConsole은 시스템 모니터링을 위해 사용하였다. 여기서 Apache JMeter는 부하(load)를 생성하고 성능을 분석할 수 있는 도구이다. 본 논문에서는 부하 생성을 위한 ‘오픈 워크로드(open workload)’ 모델을 적용한다. 이 모델은 클라이언트 요청 도착이 이전 요청의 종료와는 독립적인 특성을 가지며, JMeter 플러그인 ‘Throughput Shaping Timer’[12]를 활용하였다. 그림 3 (a)와 같이 rps(requests per second) 스케줄을 생성하면 자동적으로 스케줄에 따라 부하를 생성하게 된다. 그림에서는 10초 단위로 50, 100, 300, 100, 50 rps의 부하를 생성하고 있다.

JConsole는 로컬 혹은 원격에서 실행중인 자바 프로그램을 모니터링할 수 있는 GUI 도구로서 CPU 사용량, 메모리 사용량, 스레드 개수 등을 파악할 수 있다. 본 논문에서는 톰캣 서버의 모니터링을 위해 사용한다.

그림 3은 JMeter를 이용한 부하 생성과 JConsole을 이용한 톰캣 서버의 모니터링 그림을 보여주고 있다.



(a) JMeter ‘Throughput Shaping Timer’를 이용한 부하 생성



(b) JConsole을 이용한 서버 자원 모니터링

그림 3. JMeter와 JConsole 수행 화면
Fig. 3. Execution Screen of JMeter and JConsole

4. 응용 프로그램

웹 성능 분석을 위해 Spring MVC에 기반한 응용 프로그램을 제작하였다. 이 프로그램은 고객이 회사에 의견을 제안하는 서비스이며, 제안 추가/삭제/조회 기능을 가지고 있다. 하나의 제안에는 성명, 이메일 주소, 제안 내용으로 이루어져 있다. 그림 4는 고객의 제안 요청에 대한 서비스 동작 흐름을 보여주고 있는데, 응용 프로그램은 컨트롤러, 서비스 계층, 데이터 액세스 계층으로 이루어져 있다. 최종적으로 뷰를 생성한 후, 응답 결과를 클라이언트에게 전송한다.

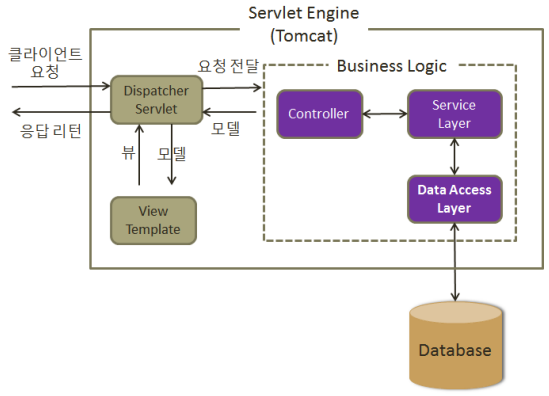


그림 4. 톰캣 서버와 응용 프로그램의 구조
Fig. 4. The Structure of Tomcat Server and Application

Ⅳ. 성능 분석

1. 실험 시나리오

일시적인 과부하에서 웹 서비스의 성능을 파악하기 위해서 그림 3(a)와 같이 부하를 생성한다. 20초~30초 구간에서 300 rps로 증가함으로써 처리량 및 응답 시간 분포를 JMeter를 통해 살펴본다.

한편, 톰캣 서버에서 중요한 파라미터인 maxThreads 값을 변화하면서 시스템 성능을 파악한다. 응용 프로그램에서 클라이언트 요청은 한 번 DB 연결 설정한 후, 한 꺼번에 DB에서 100개의 레코드를 조회하기 때문에 maxThreads와 maxTotal을 동일 크기를 제공하는게 바람직하지만, 동일한 조건에서 비교하기 위해 maxTotal을 100으로 고정시킨 상태에서 maxThreads를 1~300개까지 변화시켰다. 그 외 다양한 파라미터 값은 디폴트 값으로 설정하였다. 표 2는 시스템 파라미터 설정 값을 보여주고 있다.

표 2. 성능 분석을 위한 시스템 파라미터 값
Table 2. System Configuration Parameters for Performance Analysis

실험 파라미터	값
스레드 풀의 최대 스레드 수 (maxThreads)	1, 10, 100, 300
DB 연결 풀의 최대 연결 수 (maxTotal)	100
TCP 연결 큐의 크기 (acceptCount)	100

2. 성능 분석 결과

일시적으로 트래픽이 증가할 경우, 시스템 자원에 여유가 있을 경우에는 톰캣 스레드 풀에 존재하는 스레드 개수가 증가할수록 처리량이 높은 반면, 시스템 자원이 포화 상태에 근접할 경우에는 스레드 개수가 작을수록 처리량이 높은 것으로 알려져 있다. 일시적인 과부하시 톰캣 스레드 풀의 크기(maxThreads)에 따른 응답 시간 및 처리량의 분포는 그림 5, 6과 같다. 실험을 10회 반복 수행한 후, 케이스별 대표적인 그래프를 도시하였다. 성능에 대한 분석 결과는 다음과 같다.

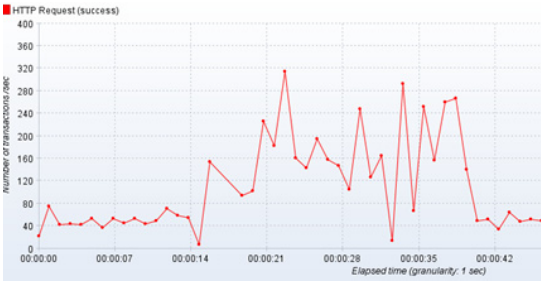
- (Case 1: maxThreads=1) 하나의 스레드만이 수행중이고 나머지 요청은 TCP 연결 완료 규에서 대기하기 때문에 일시적으로 트랙픽이 증가하는 구간(20~30초) 에서 급격히 응답 시간이 증가하는 것을 볼 수 있다. 다른 테스트 케이스와 비교했을 때 응답 시간이 길고 응답시간이 안정 상태로 돌아오는데(42초 지점) 오래 걸리는 것을 볼 수 있다. 스레드 개수가 1인 경우에는 멀티 스레딩을 통한 병행성을 이용하지 못하기 때문에 CPU 이용률이 낮아 상대적으로 좋은 성능을 보여주지 못하고 있다.
- (Case 2: maxThreads=10) 멀티스레딩을 이용함으로써 DB 접근시 유휴 상태인 CPU 를 효율적으로 이용하기 때문에 응답시간이 Case 1보다 좋은 결과를 보여주고 있다. 또한 일시적인 과부하 구간이 지난 후 안정적으로 돌아오는데(38초 지점) 상대적으로 짧게 나타나는 특징을 보여주고 있다.
- (Case 3: maxThreads=100) 스레드 개수가 증가함에 따라 스레드간 문맥 교환과 JVM GC 에 소요되는 시간으로 인해 오버헤드가 증가하게 된다. 이 경우에는 Case 2에 비해 응답시간이 길어지는 단점을 보여주고 있다. 또한 응답 시간 그래프가 부드럽지 못하고 다소 불규칙한 특징을 보여주고 있다.
- (Case 4: maxThreads=300) 스레드 개수가 더욱 증가함에 따라 오버헤드로 인해 CPU가 포화상태가 된다. 따라서 Case 2,3에 비해 응답시간이 증가하는 것을 알 수 있다.

한편, 그림 6의 처리량을 보면 Case 1 에서 다소 낮게 나타나고 그외 경우에는 비슷하게 나타나고 있음을 알 수 있었다.

(a) maxThreads = 1

(b) maxThreads = 10

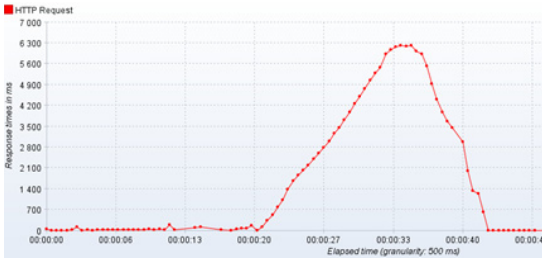
(c) maxThreads = 100



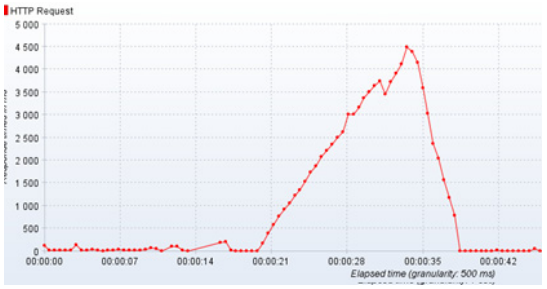
(d) maxThreads = 300

그림 5. 응용 프로그램의 처리량
Fig. 5. The Throughput of an Application Program

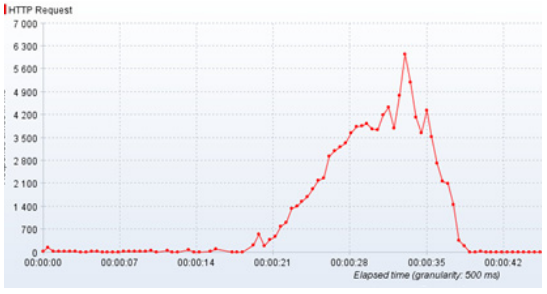
결론적으로, 일시적으로 트래픽이 급증하는 경우에 멀티 스레딩의 장점을 활용하면서 CPU가 포화상태가 되지 않도록 스레드의 개수를 결정하는 것이 중요하다. 따라서 JConsole을 통해 서버의 메모리 및 CPU 이용률을



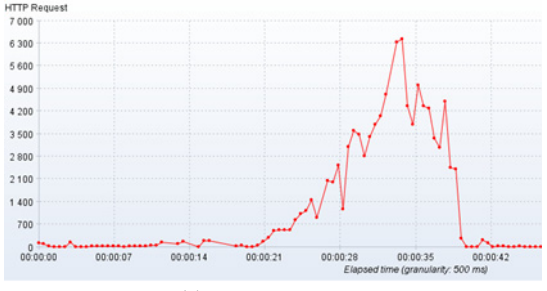
(a) maxThreads = 1



(b) maxThreads = 10



(c) maxThreads = 100



(d) maxThreads = 300

그림 6. 응용 프로그램의 응답 시간
Fig. 6. The Response Time of an Application Program

모니터링하면서 안정적으로 동작할 수 있는 값으로 설정할 필요가 있다. 본 응용에서는 maxThreads가 대략 10~100에서 좋은 성능을 보여주고 있다.

V. 결 론

모바일 단말기의 보급과 클라우드 환경에서는 웹 서비스의 성능 분석이 매우 중요하다. 웹 서비스의 성능은 응용 서버의 시스템 파라미터와 밀접한 관련을 가지는데, 시스템 환경 및 응용 프로그램의 특징에 따라 적절하게 설정할 필요가 있다. 본 논문에서는 성능 분석을 위한 실험적 평가 기법을 제시하고 성능 분석 사례를 제시하였다. 향후에는 실험적 성능 결과를 분석적 모델을 이용하여 해석하고 서버 클러스터링 환경에서 성능에 끼치는 요소를 연구할 계획이다.

References

[1] Hangoo Jeon, Young-Gi Min and Kwang-Kyu Seo, “A Framework of Performance Measurement Monitoring of Cloud Service Infrastructure System for Service Activation,” International Journal of Software Engineering and Its Applications, Vol. 8, No. 5, 2014.

[2] Mansoo Hwang, Kwanwoo Lee, Seonghye Yoon, “Software Development Methodology for SaaS Cloud Service,” The Journal of IIBC, Vol. 14, No. 1, pp.61-67, Feb. 28, 2014.

[3] David Winters, Tomcat Performance Monitoring and Tuning, http://blog.c2b2.co.uk/2014/05/ tomcat-performance-monitoring-and-tuning.html, 2014.

[4] Apache JMeter, http://jmeter.apache.org.

[5] JConsole, http://docs.oracle.com/javase/6/docs/technotes/guides/management/jconsole.html.

[6] Dashrath Mane, Ketaki Chitnis, Namrata Ojha, “The Spring Framework: An Open Source Java Platform for Developing Robust Java Applications,” International Journal of Innovative Technology and Exploring Engineering, Vol. 3 Issue 2, July 2013.

[7] Qingyang Wang, Simon Malkowski, Deepal Jayasinghe, Pengcheng Xiong, Calton Pu, Yasuhiko Kanemasa, Motoyuki Kawaba, and Lilian Harada. “The impact of soft resource allocation on n-tier application scalability,” Proceedings of the 2011 IEEE International Parallel & Distributed Processing Symposium, pp. 1034-1045, Washington, DC, USA, 2011.

[8] Alain Tchana1, Noel De Palma1, Xavier Etchevers, and Daniel Hagimont, “Configuration challenges when migrating applications to a cloud: the JEE use case,” Proceedings of International Conference on Parallel and Distributed Processing Techniques and Applications, 2013.

[9] Gábor Imre, Ágnes Bogárdi-Mészöly, Hassan Charaf, “Measuring and Modeling the Effect of Application Server Tuning Parameters on Performance,” Proceedings of Slovakian-Hungarian Joint Symposium on Applied Machine Intelligence, pp.471-482, 2006.

[10] Tomcat, https://tomcat.apache.org/tomcat-8.0-doc/index.html.

[11] MySQL, http://www.mysql.com.

[12] Throughput Shaping Timer, http://jmeter-plugins.org/wiki/ThroughputShapingTimer.

저자 소개

김 남 윤(정회원)

- 1992년 2월 : 서울대학교 컴퓨터공학과 학사
- 1994년 2월 : 서울대학교 컴퓨터공학과 석사
- 2000년 2월 : 서울대학교 컴퓨터공학과 박사
- 1999년 9월 ~ 2002년 2월 : 삼성전자 무선사업부 책임연구원
- 2002년 ~ 현재 : 한성대학교 정보시스템공학과 교수

<주관심분야 : 웹 서비스, 모바일 서비스, 공개 SW>

※ 본 연구는 한성대학교 교내학술연구비 지원과제임