성능(Performance)이란?

N-계층 서버의 '성능'에 대한 정의와 이해

Version 1.1 2015/02/20

Sunny Kwak

sunykwak@daum.net

Agenda

- **클라이언트/서버 구조 이해**
  - 클라이언트/서버 구조
  - 클라이언트/서버 기능
  - N-계층 구조
  - 3계층 구조
  - 2 계층 vs 3 계층
- **웹 기반 환경에서의 성능**
  - 성능에 대한 질문들
  - 웹 서비스 성능에 대한 통념
  - 다양한 웹 서비스 성능 지표 #1, #2
  - 핵심 성능 지표
  - 송수관 이론
  - 성능 측정 방법 및 도구
  - 성능 측정 항목
  - TPS, PPS, HPS and RPS
  - TPS vs. 평균응답시간
  - 서버의 동시 처리량 이해
  - 동시 사용자
  - 동시 사용자 수 측정
- **용어 정리 및 참고 자료**
  - 용어 정리
  - 참고자료

들어가기에 앞서...

- 본 문서는 '제니퍼 소프트' 이원영 대표님께서 2002년 JCO 컨퍼런스에 강연하신 내용을 학습한 내용을 바탕으로 작성한 것입니다.
- 웹 서비스의 낮은 성능 때문에 고민하거나, 웹 서비스 성능을 측정하는 초보 개발자를 위한 가이드 문서이며 보다 수준 높은 지식은 참조 자료들을 학습해야 합니다.
- 다만, 본 문서의 많은 내용이 이원영 대표님께서 기꺼이 공개하신 지식을 빌린 것이므로 2차 인용 시에도 가급적 본 문서가 참고한 자료들에 대한 링크를 삽입해 주시기를 요청 드립니다.
- 본 문서는 상업적인 목적을 위해 작성된 것이 아닙니다.

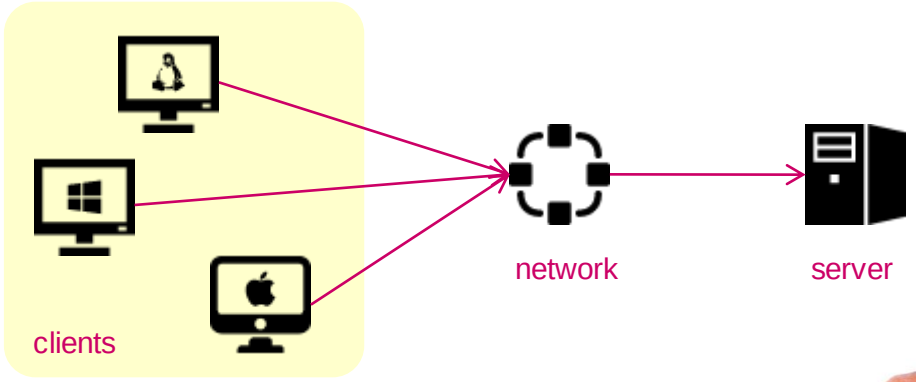
2 계층 구조와 N 계층 구조에 대해서...

클라이언트/서버 구조 이해

클라이언트/서버 구조

- **Client/Server Architecture**
  - 클라이언트 (일반적으로 GUI를 사용하는 어플리케이션)를 서버에서 분리하는 **네트워크 구조**이다. 각각의 클라이언트 소프트웨어 인스턴스는 서버에 요청을 전송할 수 있다.
  - 하나의 서버에 복수의 클라이언트가 접속하게 된다. (일대다 관계)

- **서버 유형**
  - 어플리케이션 서버 (게임, 채팅, 메신저, 증권 거래 서버 등)
  - 파일 및 FTP 서버
  - 터미널 서버
  - 메일, DNS 서버



클라이언트/서버 기능

- **서버 기능 (혹은 특성)**
  - 수동적, 서비스 제공자 (Passive, Slave)
  - 클라이언트 요청을 처리하기 위해 대기.
  - 요청(request)을 처리한 후, 결과를 클라이언트에 회신(reply)

- **클라이언트 기능**
  - 능동적, 의뢰자 (Active, Master)
  - 서버가 수행할 수 있는 요청을 전송함.
  - 회신(응답)이 반환될 때까지 기다림

- **널리 알려진 클라이언트**
  - 가장 보편적인 클라이언트는 인터넷을 통해 웹 서버와 통신하여 웹 페이지를 다운로드하고 사용자에게 보여주는 웹 브라우저이다.

N-계층 구조

- **2 계층 구조 (2-tier architecture)**
  - 클라이언트와 서버 등 2개의 노드(node)로 구성된 구조(architecture)를 2계층 구조라고 부른다.
  - 2 계층 구조에서 서버는 단지 데이터를 저장하는 역할만을 수행하며, 클라이언트가 모든 처리(processing)을 담당한다.

- **2계층 구조의 한계(문제)**
  - 클라이언트(PC 등)의 상대적인 성능이 향상되면서 다양한 처리를 클라이언트로 이전할 수 있으나, 데이터의 무결성(integrity)을 유지(관리)하기가 어렵다.
  - 비즈니스 로직(business logic)을 클라이언트에 두기 어려운 경우도 있다. 예를 들어, 사용자 간의 메시지를 주고 받아야 할 경우 서버는 데이터를 저장하는 역할만 수행하기 때문에 클라이언트 간에 직접 통신을 해야 한다.

- **N 계층 구조 (N-tier architecture)**
  - 2 계층 구조의 한계를 극복하기 위해, 3개 이상의 노드를 네트워크 상에서 구성하는 방식이 N 계층 구조이다. N 계층은 3개, 4개 혹은 더 많은 노드로 구성된다.
  - N 계층 구조가 2 계층 구조를 완전히 대체하는 하는 것은 아니다. FTP, Telnet 서비스 등은 여전히 2계층 구조로 동작한다.

3계층 구조

- **3계층 어플리케이션**
  - 정보, 중간, 클라이언트 등 3개의 계층으로 구성된다.

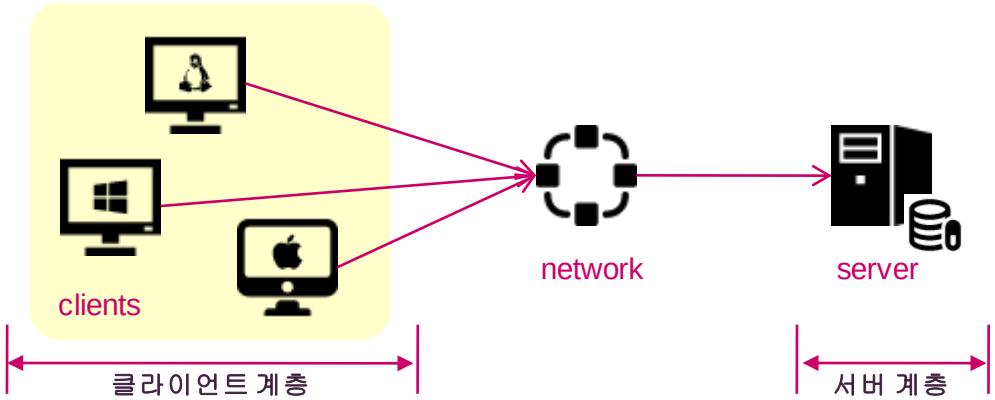
- **정보 계층 (Information tier)**
  - 데이터 계층(data tier) 혹은 최하위 계층(bottom tier)이라 부른다.
  - 어플리케이션을 위한 데이터를 관리한다.
  - 일반적으로 관계형 데이터베이스(Relational Database)를 이용해 데이터를 저장한다.

- **중간 계층 (Middle tier)**
  - 어플케이션 계층(applicatoin tier)으로 부르기도 한다.
  - 비즈니스 로직(business logic) 및 프리젠테이션 로직(presentation logic)을 구현한다.
  - 어플리케이션 클라이언트와 데이터 간의 상호작용을 제어한다.
  - 정보 계층의 데이터와 어플리케이션 클라이언트 간의 매개자(intermediary) 역할을 수행한다.

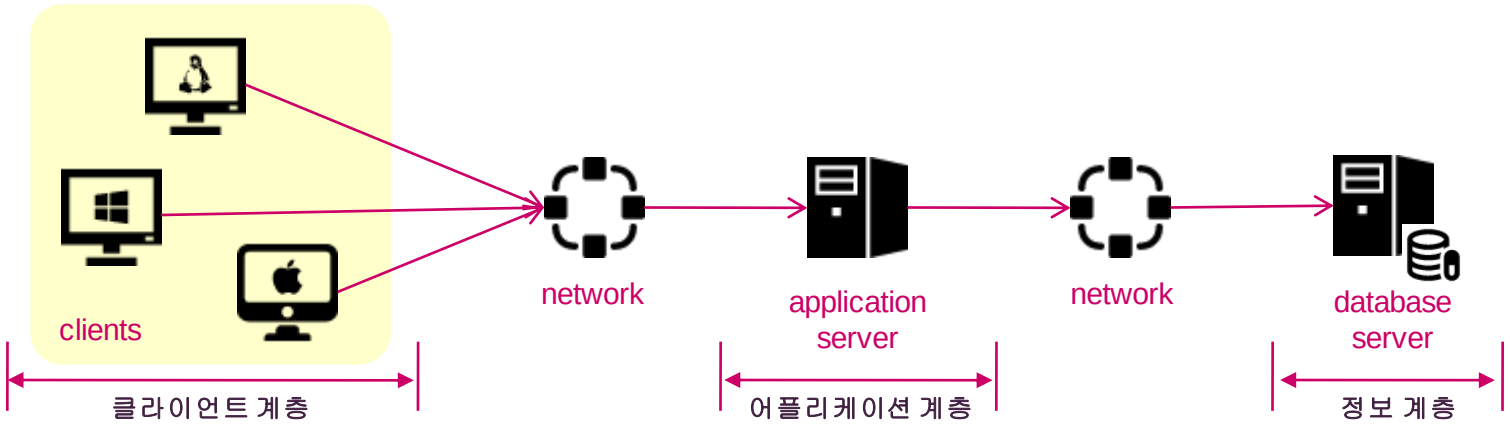
- **클라이언트 계층 (Client tier)**
  - 최상위(top) 계층으로 부르기도 한다.
  - 어플리케이션의 사용자 인터페이스 역할을 수행한다.
  - 중간 계층과 상호작용을 통해 요청을 전달하고 정보 계층에서 데이터를 조회한다.

2 계층 vs 3 계층

2 계층 구조 (2-tier architecture)



3 계층 구조 (2-tier architecture)



웹 서비스 성능 지표 및 성능 측정 방법

웹 기반 환경에서의 성능

성능에 대한 질문들

- **N 계층 구조에서의 성능이란?**
  - 게임, 메신저, 웹, 기업용 솔루션 등 다양한 소프트웨어들이 인터넷을 통해 구동된다. 만일, N 계층 구조에서 서버가 정상적으로 응답하지 못하거나 느려질 경우, 사용자의 어플리케이션 또한 무용지물이 된다.
  - 사용자 어플케이션 성능은 개인용 하드웨어의 성능이 발전함에 따라 거의 문제 되지 않는다.
  - 서버의 성능이 전체 서비스의 품질 및 가용성(availability)를 좌우한다.
  - N 계층 서버의 성능은 클라이언트를 제외한 서버군(server farm) 전체가 상호작용하며 클라이언트의 요청을 처리하는 능력이다.

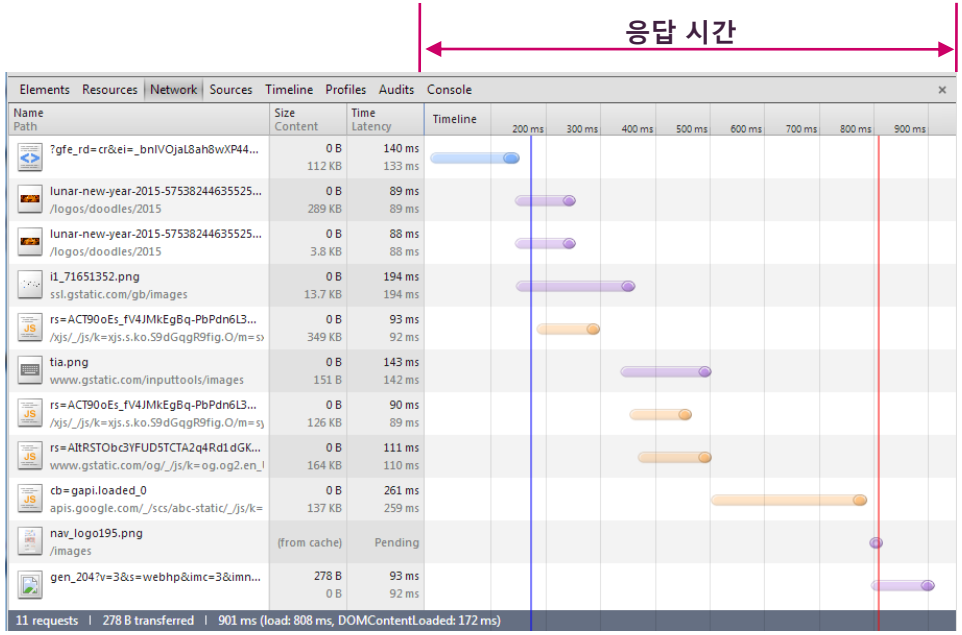
- **어떤 성능을 측정해야 하는가?**
  - 일반적으로 클라이언트 사용자 입장에서 서버에 대해 느끼는 성능은 속도(speed) 혹은 응답 시간(response time)이다.
  - 서버 측 설계자 및 운영 담당자 입장에서는 얼마나 많은 사용자를 감당할 수 있는가? 수용량(capacity)에 더 큰 관심을 가지게 된다.

- **어떤 관점을 가져야 하는가?**
  - 단지, 서버의 하드웨어 자원 (CPU, memory, network)의 용량에 기대어 성능을 높일 수는 없다.
  - 하드웨어 및 소프트웨어의 다양한 한계를 이해해야 하며, 고유의 특성을 이해해야 한다.
  - N 계층에서 서버의 성능 총합은 각 계층 서버 용량의 총합이 아니라, 가장 낮은 성능을 가진 자원 혹은 가장 큰 병목지점(bottle-neck)에 의해 좌우된다.

- **어떻게 튜닝(tunning)할 것인가?**
  - 하드웨어, 소프트웨어 (서버 소프트웨어) 및 운영체제의 메커니즘을 이해해야 한다.
  - 자료구조 및 알고리즘에 따라 성능이 달라진다.
  - 튜닝을 하기 전에 성능 분석을 튜닝 이후에 개선된 결과를 비교하는 것은 필수 작업이다.
  - 튜닝을 수행함에 있어서 늘 가장 좋은 효과(성과)를 얻을 수 있는 방법은 병목 지점을 찾아내고, 그것을 해결하는 것이다. 따라서, 서버 전체의 성능을 분석하는 것 뿐만 아니라, 계층 간, 계층 별 성능을 측정할 수 있는 기법을 학습/연구해야 한다.

웹 서비스 성능에 대한 통념

- **웹 서비스의 성능에 대한 통념**
  - 일반적으로 웹 서비스의 응답 성능이라 하면, 단일 요청에 대한 **응답 시간의 평균 (혹은 체감 응답 속도)** 만을 떠올리기 마련이다.
  - 직관적인 감각에 의존하기도 하고, 스톱워치를 이용해 측정하기도 하며, 성능 분석 툴을 이용해 측정하기도 한다. 응답 시간의 측정 결과에 따라 서비스의 성능이 좋거나 나쁘다고 판단한다.
  - 통상적으로 사용자는 응답 시간이 3초를 넘으면 느리다고 느끼며, 6초를 한계로 본다.



다양한 웹 서비스 성능 지표 #1

- **웹 서비스 요청에 대한 평균 응답 시간**
  - 전체 사용자 수가 증가할수록 응답 시간은 늘어날 수 있다.
    서버가 수용 가능한 임계치(threshold)를 넘을 경우, 응답 속도는 급격히 떨어진다.
  - 동시 요청 수, 사용자의 증가, 사용자의 네트워크 위치 등 각종 조건 및 상황에 따라 가변적이다.
  - 평균 응답 시간 뿐만 아니라, 가장 느린 응답을 지표로 삼기도 한다.

- **단위 시간 당 로그인한 사용자 수**
  - 사용자들이 늘 일정한 횟수(혹은 주기)로 서버에 요청하지 않는다.
  - 웹 서비스는 사용자가 브라우저의 버튼 등을 클릭하지 않으면 서버에 부하를 주지 않는다.
  - 또한, 사용자 마다 서비스 이용 패턴(활동)이 다를 수 있다.
    짧은 시간을 사용하는 사용자도 있고, 오랜 동안 머무는 사용자들도 있어 편차가 크다.

다양한 웹 서비스 성능 지표 #2

- **동시 사용자 수**
  - 로그인 상태를 유지하고 있는 – 세션(session)의 갯수 – 사용자를 측정하는 기법이다.
  - 접속은 했으나, 실제 작업을 수행하지 않는 사용자가 존재할 수 있다.
  - 웹 서비스는 사용자가 로그아웃한 시점을 파악하기 어렵다.
    (로그아웃 버튼을 클릭하지 않고 웹 브라우저를 닫는 사용자도 많다.)

- **특정 시점에 실행 중인 서비스(요청) 갯수**
  - 서버의 자원 상황 (메모리, CPU, 네트워크)에 따라 성능 편차(오차)가 발생한다.
  - 순간 처리량으로 서버의 모든 성능을 평가할 수 없다. 예를 들어, 모든 작업을 메모리를 이용해 처리하면 최대 처리량이 높아 보일 수 있지만, 처리 결과를 디스크 혹은 네트워크로 전송하기 위해서 최대 처리 성능을 보인 이후에 급격히 성능이 떨어지는 현상이 발생한다.
    (이러한 현상은 PC 혹은 클라이언트 어플리케이션에서도 종종 발생한다.)

- **단위 시간(시,분,초)당 처리 가능 요청 건수**
  - 서버가 일정 기간 동안 처리할 수 있는 요청 건수는 최대치를 측정할 수 있다.
  - 서버 관리자 입장에서는 가장 중요한 성능 지표가 될 수 있다.

핵심 성능 지표

- **사용자 입장에서는…**
  - 당연히 서버 응답 시간이 짧을수록 좋다.
  - 게임, 증권거래 서비스 등은 아주 작은 지연이 큰 차이를 초래한다.
  - 문제는 서버가 아무리 빠르더라도 서버와 사용자 사이에는 네트워크 회선이 존재한다. 지연 시간(latency time)이 발생할 수 밖에 없다.

- **서버 설계 및 관리자 입장에서는…**
  - 서버 관리자에게 서버 중단(halt, hang-up)이야 말로 가장 큰 재앙이다.
  - 더불어 서버 용량 산정 및 확장을 위한 비용 책정을 위해서 개별 서버의 처리량(throughput)을 이해하는 것이 중요한 과제(mission)이다.

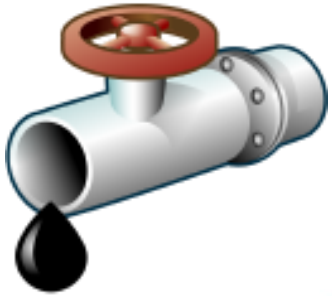
- **결론**
  - 다양한 성능 지표가 존재하나, 가장 중요한 성능 지표 2가지를 선정할 수 있다.
  - 사용자의 최적한 서비스 경험을 위한 **"짧은 응답 시간(short response time)"**, 그리고, 서버의 안정적인 운영을 위한 **"적절한 처리량(proper throughput)".**

송수관 이론

- **송수관 이론 (water pipe theory)**
  - N 계층 서비스 상에서의 서버를 송수관에 비유할 수 있다.
  - 송수관의 성능을 판단하는 가장 이상적인 기준은 관의 굵기(크기), 물의 압력(펌프의 종류)이 아니라, 단위 시간 당 흘려 보낼 수 있는 수량이다.
  - '수량(성능)'은 유속과 송수관 단면의 면적에 비례한다. 유속은 펌프의 종류에 따라 결정되며, 단면의 면적은 파이프 반지름의 제곱에 비례한다. 이를 서버에 적용하면 펌프의 종류는 'CPU의 유형', 단면의 면적은 '메모리 크기와 네트워크 속도'에 비유할 수 있다.
  - 웹 서비스의 성능은 응답 속도만을 판단해서는 안되며, 단위 시간 당 처리량(throughput)을 중요한 지표로 판단해야 한다.

☞ 제니퍼소프트 이원영 대표님 2002년 강의 인용

http://www.javaservice.net/~java/bbs/read.cgi?m=resource&b=jscperf&c=r_p&n=1008701211



성능 측정 방법 및 도구

- **성능 측정 방법** (Performance Estimating)
  - 송수량의 측정은 '수도 계량기'를 이용해 단위 시간당 흘러간 물의 양을 측정하면 되지만, 웹 기반 서비스는 서버에 부하(load)를 유발하고 처리량을 측정해야 한다.
  - 웹 서비스는 최대 처리 가능한 요청 수보다 적은 요청이 들어올 경우, 평균 응답 시간은 짧고 단위 시간 당 처리 건수는 낮게 (당연하지만) 측정된다. 따라서 점진적으로 동시 요청 횟수를 늘려가면서 한계치(max or limit)에 도달할 때까지 성능 테스트를 수행해야 한다.

- **성능 측정 도구**
  - 웹 서비스의 성능을 인간이 수동으로 테스트하는 것은 정밀하지도 않고, 동시 요청 횟수를 늘리는 방법에도 한계가 있기 때문에 동시에 수십/수백 건의 웹 서비스 호출(요청)을 자동으로 수행하고 응답 시간 계측 및 기록을 수행하는 도구를 사용해야 한다.
  - 이러한 도구를 부하 테스트 도구(Stress Test Tool), 혹은 성능 테스트 도구(Performance Test Tool)라고 부른다.

명칭	제조사	상용/오픈	홈페이지
Load Runner	HP	상용	https://saas.hp.com/software/loadrunner
jMeter	Apache	오픈소스	http://jmeter.apache.org/
nGrinder	Naver	오픈소스	http://naver.github.io/ngrinder/

성능 측정 항목

- **성능 측정 항목**
  - 다양한 부하 테스트 도구를 이용한 다양한 성능 지표를 측정할 수 있다.
  - 최소한 (공통적으로) 2가지 항목을 측정할 수 있다.

- **동시 사용자에 따른 평균 응답 시간**
  - Mean response time of active clients
  - 예를 들어, 동시에 100명의 사용자가 웹 서비스(혹은 페이지)를 요청했을 때, 모든 사용자 요청에 대한 응답 시간을 측정한 후, 이에 대한 평균값을 계산하는 것이다.

- **동시 사용자에 따른 단위시간 당 처리 건수**
  - Transaction Per Second of active clients
  - 동시에 100명의 사용자가 웹 서비스를 요청했을 때, 단위 시간(예를 들어 1초) 내에 정상적으로 응답을 완료한 건수를 의미한다.

- **동시 사용자**
  - Active clients or concurrent clients
  - 부하 테스트를 수행할 때 서버에 같은 시점(same time)에 접속하는 사용자를 의미한다.

TPS, PPS, HPS and RPS

- **시간 당 처리량에 관한 4가지 용어**
  - 시간 당 처리량을 의미하는 용어는 TPS, PPS, HPS 및 RPS 등 4가지가 있다.
  - 성능 테스트 도구와 성능 테스트 환경에 따라 같은 의미 혹은 다른 의미로 사용된다.

- **TPS : Transaction Per Second**
  - TPS는 가장 오래 전부터 사용되어 온 단어이며, 클라이언트 서버 환경에서 비롯된 단어이다. 클라이언트/서버 환경에서는 클라이언트(어플리케이션)가 직접 서버(데이터베이스)에 접속하여 쿼리, 트랜잭션을 요청했다. 따라서, 서버(데이터베이스)의 성능을 측정하는 기준으로 초당 몇 건의 트랜잭션을 수행할 수 있느냐가 보편적인 성능 측정 지표였다.

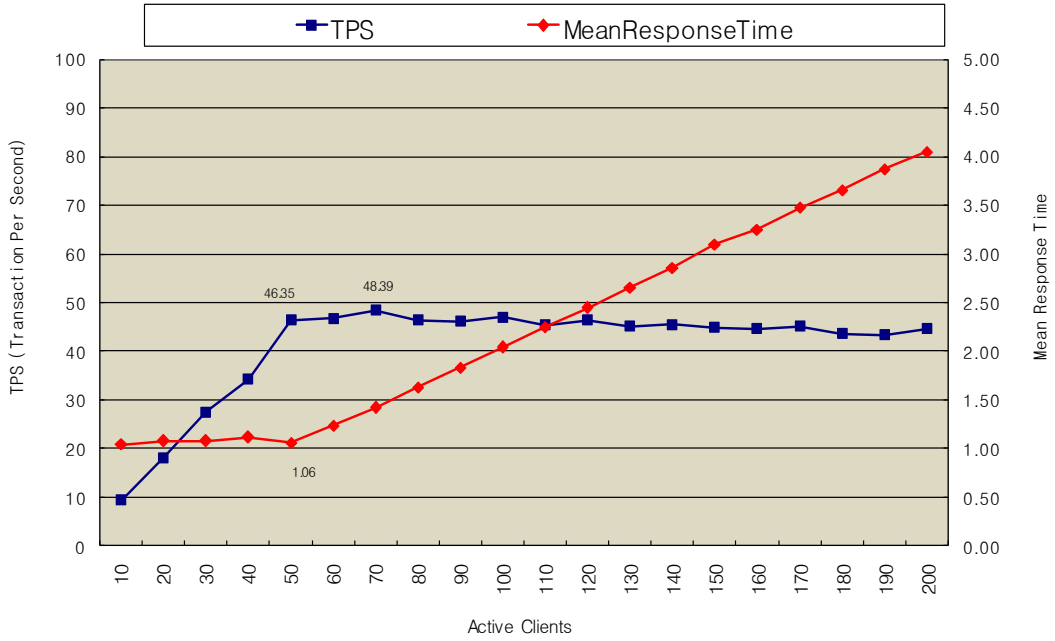
- **PPS : Page Per Second**
  - 웹 서비스에서는 특정 URL을 호출했을 대, 서버 작업(JSP/Servlet 등)의 수행 결과를 화면에 출력하기 위해서 CSS, Image 파일과 같은 정적(static) 컨텐츠도 다운로드 받아야 하기 때문에 Page Per Second라는 성능 측정 기준을 사용하기도 한다.

- **HPS : Hit Per Second, RPS : Request Per Second**
  - HPS, RPS는 일반적으로 정적인 컨텐츠를 제외한 동적 컨텐츠(XML, JSON 등) 요청에 대한 응답시간을 측정하는 기준을 의미한다.

TPS vs. 평균응답시간

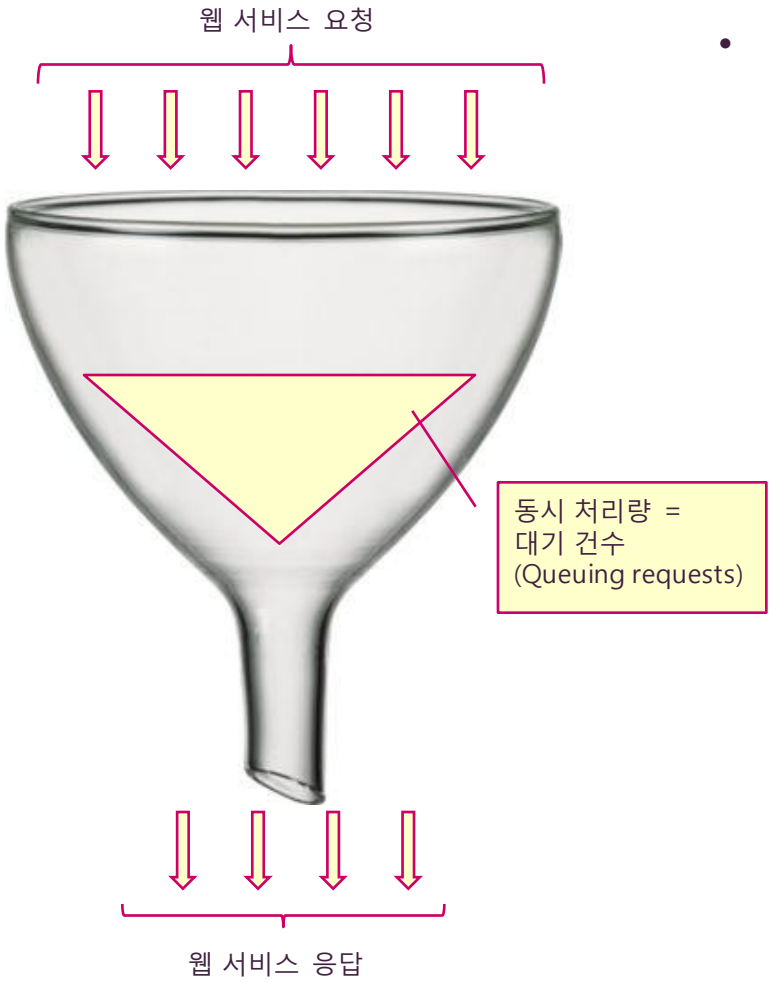
- **평균응답시간**
  - 시스템의 처리능력 한계치보다 낮은 동시 사용자가 접속할 경우에는 일정한 수치를 유지하지만, 한계치를 넘게 되면 사용자 수에 비례해서 느려지게 된다.

- **단위 시간 당 처리 건수**
  - 동시 사용자가 계속 증가해도 최대값 보다 증가하지 않는다. 따라서 서버 성능 측정 기준으로 단위 시간당 최대처리건수를 의미하는 TPS를 사용하는 것이 적합하다.



☞ 그래프 참조 : http://www.javaservice.net/~java/bbs/read.cgi?m=resource&b=jscperf&c=rp&n=1008701211

서버의 동시 처리량 이해



- **깔대기 이론 (Funnel theory)**
  - 서버가 클라이언트의 요청을 처리하는 메커니즘은 깔대기를 이용해 물을 흘려 보내는 행위에 비유할 수 있다.
  - 깔대기로 흘러내리는 물은 웹 서버에 도달하는 요청(request), 아래로 빠져 나가는 물은 응답(response), 깔대기 내에 머물러 있는 물은 처리 중인 작업(processing or job)이다.
  - 머물러 있는 물의 양은 '동시 처리 서비스 개수(concurrent processing service count)'이다.
  - 만일, 점진적으로 요청 건수가 늘어날 경우, 동시 처리량이 늘어날 것이며 서버가 처리할 수 있는 한계를 넘게 되면 오버플로우(overflow) 현상이 발생한다. 이 때, 서버의 처리(혹은 구현) 방식에 따라 일부 요청에 대해서 정상적으로 응답하지 못하거나(does not respond), 서버 자체가 멈추는 현상(hang-up)이 발생한다.

동시 사용자

- **동시 처리량 ≠ 동시 사용자**
  - 동시 처리량 (concurrent service count)는 서버가 특정 시점에 처리하고 있는 요청 건수이며, 동시 사용자 수(concurrent user count)가 아니다.

- **동시 사용자 (concurrent users)**
  - 웹 서비스를 사용하기 위해 클라이언트 어플리케이션을 실행하고 있는 사용자의 수이며, 활성 사용자와 비활성 사용자로 구성된다.
  - 동시 사용자 수 = (활성 사용자 수 + 비활성 사용자 수)
  - 통상 '동시 단말 사용자 수(Concurrent Terminal User)'라고 부른다.

- **활성 사용자 (Active Users or Clients)**
  - 서버로 요청을 보내고 응답을 기다리고 있는 사용자

- **비활성 사용자 (InActive Users or Clients)**
  - 어플리케이션 화면을 조회하고 있거나, 다음 버튼을 누르기 이전 상태인 사용자

동시 사용자 수 측정

	측정 방법	장점	단점
IP 주소 집계	특정 시점 전후에 서버에 접속한 클라이언트 IP 주소를 수집한 후, 중복을 제거.	웹 서버 설정 변경만으로 손쉽게 로그 수집이 가능함.	사용자가 프록시 서버(proxy server), L7 switch 등을 경유해 접속한 경우, 복수의 클라이언트가 동일 IP로 접속함.
Proxy 보정	IP 주소를 집계한 후, 평균 접속 건수 보다 큰 IP 주소를 평균 접속 건수로 나누어 구분함.	IP 주소 집계 방식보다는 정밀함.	과다 접속하는 사용자를 구분할 수 없어서 오차가 발생함.
HTTP 세션 ID 집계	클라이언트가 사용하는 세션 ID를 로그에 남긴 후, 세션 ID를 수집하고 중복을 제거.	가장 정확한 집계 방법	세션 ID를 기록하기 위해 로그 생성 기능을 변경(혹은 구현).

미쳐 설명하지 못한 것들...

용어 정리 및 참고 자료

용어 정리

- Application
  - 어플리케이션이라는 개념은 작은 범주에서는 클라이언트 (PC, mobile 등) 하드웨어에서 동작하는 소프트웨어 프로그램을 의미한다. 넓은 범주에서는 N 계층 구조에서 클라이언트부터 서버까지를 망라하는 전체 시스템을 어플리케이션이라고 부르기도 한다.

- Proxy
  - 프록시 서버(proxy server)는 클라이언트가 자신을 통해서 다른 네트워크 서비스에 간접적으로 접속할 수 있게 해 주는 컴퓨터나 응용 프로그램을 가리킨다. 서버와 클라이언트 사이에서 중계기로서 대리로 통신을 수행하는 기능을 가리켜 '프록시', 그 중계 기능을 하는 것을 프록시 서버라고 부른다.

    http://ko.wikipedia.org/wiki/%ED%94%84%EB%A1%9D%EC%8B%9C_%EC%84%9C%EB%B2%84

- L7 switch
  - OSI 레이어 3~7에 속하는 IP 주소, TCP/UDP 포트 정보 및 패킷 내용까지 참조하여 스위칭하는 네트워크 장비
  - 일반적으로 서버들의 로드밸런싱을 위해 사용됨. 복수개의 웹서버가 있을 때, 임의의 웹서버에 접속을 시도하면, 스위치가 각 서버의 부하를 고려하여 적당한 서버와 연결시켜준다.

    http://freeism.web-bi.net/tc/657

참고 자료

- Lecture 11 client_server_interaction
  - http://www.slideshare.net/Serious_SamSoul/lecture-11-clientserverinteraction-10555127
- Performance concepts (제니퍼 소프트 / 이원영 대표님 강의)
  - http://www.javaservice.net/~java/bbs/read.cgi?m=resource&b=jscperf&c=r_p&n=1008701211
- 이미치 출처
  - http://www.iconarchive.com
  - http://icons8.com
  - http://www.saveur.com/sites/saveur.com/files/images/2011-08/7-BeakerGlassFunnel_400.jpg