동적 확장성을 제공하는 주문형 비즈니스 시스템

김정숙, 조성재, 송준화°
한국과학기술원 전자전산학과
{sun, stanleylab, junesong°}@nclab.kaist.ac.kr
임민열, 박형우
한국과학기술 정보연구원 그리드 연구실
mylim@hpcnet.ne.kr
hwpark@kisti.re.kr

An on demand e-Business system providing dynamic scalability

Jungsook Kim, Sung-Jae Jo, Junehwa Song°
EECS, KAIST
Hyungwoo Park, Minyeol Lim
KISTI GRID research team

요 약

인터넷 비즈니스(e-Business)가 보편화되면서 다양하고 복잡한 서비스들이 제공되고 있다. 이러한 서비스들은 주로 정적 컨텐트를 사용했던 종전의 서비스와는 달리, 동적 컨텐트를 사용하여 급변하는 사용자의 요구에 부응하고자 한다. 하지만 동적 컨텐트는 정적 컨텐트에 비해 많은 시스템 자원을 필요로 하여 전체 시스템의 부하를 높이는 주요 원인으로 지적되고 있다. 한편, 서비스의 수요는 그 예측이 매우 어렵기 때문에 순간적으로 급증하는 수요에도 안정적인 서비스를 제공하는 방법이 시급히 요구되고 있다. 따라서 본 논문에서는 동적 확장성(dynamic scalability)을 제공하는 WADN 시스템을 제안한다. WADN은 서버의 부하를 검사하여, 이를 바탕으로 전체 서버의 수를 탄력적으로 조정한다. 그리고 요청된 애플리케이션을 자동으로 수정하는 기능을 구현하여 새로운 서버에 전송되면 바로 실행 될 수 있도록 하였다. WADN을 이용하면 인터넷에 있는 유휴 자원들을 효과적으로 이용할 수 있고, 시스템 관리자가 적은 비용으로 안정적인 서비스를 제공할 수 있다.

1. 서 론

인터넷과 웹 기술 발전을 기반으로 한 e-Business는 기업의 영업활동에서 필수적인 수단으로 각광 받고 있다. 그러나 e-Business는 전통적인 비즈니스와 비교되는 확연한 특징을 가진다. 첫째, e-Business 서비스에 대한 요청(request)은 일시적으로 집중되는 특징을 보이며, 이러한 일시적인 집중현상은 예측하기 어렵다. 왜냐하면 사용자들의 관심사는 매우 비슷하면서 유동적인 경향이 있는데, 인터넷의 높은 접근성(accessibility)이 이러한 경향을 더욱 두드러지게 하기 때문이다. 둘째, 고객의 요구에 능동적으로 대응하기 위해, 대부분의 진보된 형태의 e-Business는 동적 컨텐츠를 기반으로 서비스된다. 전자 상거래(e-Commerce), 맞춤형 정보 제공서비스(personalized information presentation)등이 대표적인 예라 할 수 있다.

위 두 가지 특징은 e-Business 시스템을 쉽게 과부하 상태에 놓이게 할 수 있다. 급작스럽게 증가한 동적 컨텐츠에 대한 수요를 처리하기 위해, 시스템은 정적 컨텐츠를 서비스하는 시스템보다 월등히 많은 컴퓨팅 자원을 필요로 하게 된다. 서버의 과부하 상태는 서비스 제공자와 서비스 요청자 모두에게 심각한 문제를 야기한다. 만약 서비스 요청 율(request rate)이 서버의 용량을 초과한다면, 서버의 응답시간과 접속 에러 율(connection error rate)은 기하급수적으로 증가할 것이고, 어느 서비스 요청자도 제대로 된 서비스를 받을 수 없는 치명적인 상황에 직면하게 된다. 이와 같은 상황에서는 유명 웹 사이트 조차도 대다수의 고객을 잃을 수 있다. Zona Research의 연구 리포트는 전자 상거래 사이트의 경우 응답시간이 7초보다 작을 경우 7%의 고객이 사이트를 떠나지만, 이 비율은 기하급수적으로 증가하여 응답시간이 12초보다 커지면 70%나 되는 고객이 그 사이트를 버리고 떠난다는 것을 보여주었다[1].

본 논문은 e-Business 시스템의 일시적인 과부하 문제를 효율적으로 풀기위하여 WADN(Web Application Delivery Network)이라는 새로운 시스템을 제안한다. WADN은 인터넷 환경에서 고객 수요의 폭발적인 증가에도 유연하게 대처할 수 있는 동적 확장성을 제공해줌으로서 기업이 지속적이고 안정적인 서비스를 제공하는 것을 가능하게 해준다. 이를 위해 WADN은 필요에 따라 서버의 개수를 탄력적으로 조절하여 시스템이 항상 최적의 서버개수를 유지하도록 해준다. WADN은 주기적으로 시스템의 부하 상태를 검사하여 시스템이 과부하 되었는지를 판단한다. 시스템의 과부하 여부는 CPU 부하, IO 부하, 네트워크 커넥션 개수 등 여러 파라미터를 이용하여 측정할 수 있다. 시스템에 과부하가 발생할 것이 예측되면, WADN은 실시간으로 새로운 서버(이하 유동서버라 칭함)를 시스템에 추가하여 시스템의 용량을 증가시킨다. 이와 같은 방법으로 WADN은 개별 서버의 부하를 일정수준 이하로 유지할 수 있다.

WADN은 위에서 언급한 일련의 작업을 자동으로 수행한다. 시스템 관리자는 시스템을 설치하는 초기에 WADN의 작동에 필요한 설정만 해주면 된다. 다시 말해서, 유동 서버 풀에서 적당한 유동서버를 선택하는 작업과, 선택된 유동 서버에 애플리케이션을 전송•탑재시키는 과정이 모두 시스템 관리자의 개

입 없이 이루어진다. 한편 애플리케이션의 전송은 그 애플리케이션의 실행 환경이 바뀜을 의미한다. WADN은 이러한 실행 환경 변화에도 애플리케이션이 영향 받지 않고 동작할 수 있도록 환경 변화에 영향 받는 애플리케이션의 설정을 자동으로 수정해준다.

WADN을 이용하면 다음과 같은 여러 가지 이점을 얻을 수 있다. 첫째, WADN은 최소의 비용으로 고객 수요의 변동에 영향 받지 않는 안정적인 서비스를 제공할 수 있다. 둘째, WADN은 인터넷 자원을 효율적으로 사용한다. 즉, 기업은 IT 종량제를 통해 자신이 유동 서버를 사용한 만큼만 요금을 지불함으로서 소유비용을 줄이고, 유동서버 제공자는 유동서버를 시간대에 따라 서로 다른 웹 사이트들과 공유함으로서 자원을 효율적으로 이용할 수 있다. 셋째, WADN은 자동화된 관리를 통해 기업의 조직원들이 복잡한 장비나 시스템의 운영, 관리로부터 자유롭게 되어 자기 본연의 역할과 임무에 집중할 수 있도록 해준다.

마지막으로 본 논문의 나머지 부분은 다음과 같이 구성된다. 2장에서 우리는 웹 애플리케이션 성능을 높이기 위한 기존의 연구에 대해서 소개하고, 3장에서는 동적 확장성을 제공하는 시스템인 WADN에 대해서 소개하겠다. 마지막으로 4장에서 결론에 대해 논하면서 마치겠다.

2. 관련연구

동적 컨텐츠를 서비스해주는 시스템의 성능을 높이기 이한 많은 연구들이 있었다. 이 연구들은 대부분 시스템의 용량을 증가시키는 방법을 취하였다.

동적 컨텐츠 캐싱[2, 3, 4]은 정적 컨텐츠 캐싱처럼 웹 애플리케이션 서버로 향하는 요청을 서버를 대신해서 캐시가 서비스해주는 방법이다. 이 방법은 서버의 부하를 줄일 수 있을 뿐 아니라 네트워크 트래픽을 줄일 수 있다는 장점을 가진다.

동적 컨텐츠 캐싱을 위해서 Canda et al.은 리버스 프락시(reverse proxy)에서 sniffer와 invalidator를 이용한 지능형 캐싱 기술을 제안하였다[5]. 이 기술은 캐시된 컨텐츠 중 최신 버전이 아닌 것을 선택한 후 무효화(invalidation)시키는 방법이다. 이를 위해 sniffer는 쿼리(query)와 그 쿼리에 의해서 생성된 웹 컨텐츠 사이의 관계를 기록한다. 데이터베이스에 업데이트가 발생할 경우 invalidator는 sniffer가 갖고 있는 쿼리 정보에 근거해서 그 쿼리에 의해 영향 받는 웹 컨텐츠를 무효화시켜 더 이상 서비스되지 못하도록 한다. 그러나 이 방법은 캐시된 컨텐츠에 대응되는 쿼리와 똑 같은 쿼리에 대해서만 동작하기 때문에 효과적이지 못하다. 게다가 복잡한 무효화 작업은 서버에 추가적인 부하를 야기 시키는 문제점이 있다.

이 연구에 더해 Luo et al.[6]는 프락시에서 데이터베이스 서버를 대신해서 단순한 쿼리를 처리할 수 있는 쿼리 애플릿(query applet)을 제안하였다. 쿼리 애플릿은 URL에 근거해서 쿼리와 그에 대응하는 결과간의 관계를 보유하고 있다. 쿼리 애플릿은 요청 파라미터에서 쿼리를 추출한 후 자신이 캐시하고 있는지 검사한다. 만약 추출된 쿼리에 대응하는 결과를 자신이 캐시하고 있다면 쿼리 애플릿은 그 결과를 즉각적으로 돌려준다. 뿐만 아니라 쿼리 애플릿은 XML 형태로 결과를 저장하기 때문에, 캐시된 데이터의 적절한 조합으로 결과를 만들 수 있는 쿼리도 서비스해줄 수 있다. 그러나 이 방법은 매 쿼리 요청마다 기존 결과의 조합으로 새로운 쿼리를 서비스해줄 수 있는지 검사하는 오버헤드가 크다는 단점이 있다.

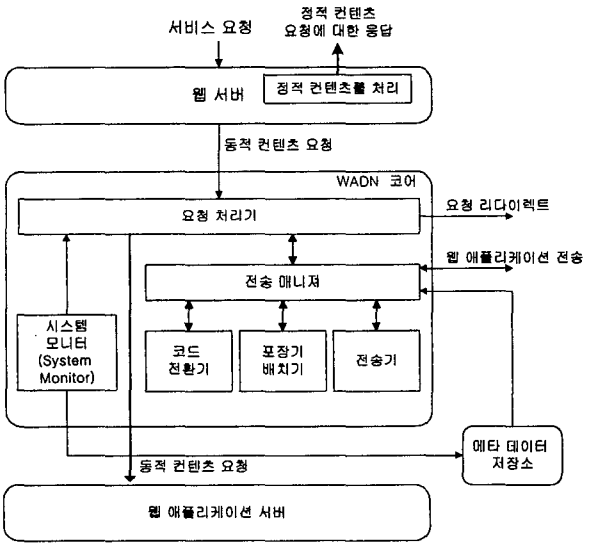
서버의 용량을 증가시키기 위해 Adro[7], Akamai[8], Digital Island[9]같은 다양한 Content delivery networks (CDN)이 연구 중이다. 이 연구는 네트워크의 에지(edge) 존재하는 CDN 서버에 컨텐츠를 복사해서 네트워크 트래픽을 줄이고 서버의 부하를 줄이고자 하는 방법이다. CDN은 각각의 CDN 서버가 서로 연결된 분산 구조를 갖는다. Paul and Feri[10]은 이러한 구조를 갖는 캐시가 지속적으로 높은 hit ratio와 빠른 응답시간을 가지며 효과적으로 부하 분산을 할 수 있다는 것을 밝혔다. 이러한 CDN의 장점에도 불구하고 아직까지 CDN은 동적 컨텐츠를 서비스하기에는 관련 기술이 성숙하지 못했다.

3. 디자인 및 구현

본 장에서는 WADN 시스템의 전체적인 구조에 대해서 살펴보고, 각 구성 요소에 대해서 설명하고자 한다.

3.1 WADN의 구조

WADN 시스템은 그림 1에서 보는 바와 같이 웹 서버, 웹 애플리케이션 서버, WADN 코어(core)로 구성된다. WADN 코어는 WADN 시스템의 핵심 기능을 담당하는 부분으로 요청 처리기, 시스템 모니터(System monitor), 전송 매니져, 코드 전환기, 포장기와 배치기, 전송기로 구성되어 있다. 각 구성요소에 대한 자세한 설명은 다음 절에서 기술하고, 여기에선 간단히 주요 기능만을 언급하겠다. 요청 처리기는 근원 웹 애플리케이션 서버의 부하상황을 바탕으로 서비스 요청의 리다이렉션(redirection) 여부를 결정한다. 시스템 모니터는 근원 웹 애플리케이션 서버의 자원을 감시하여 그 부하 정보를 메타 데이터 저장소와 요청 처리기에 전달한다. 전송 매니져는 메타 데이터를 바탕으로 가장 적합한 유동 서버를 선택한 후, 해당 웹 애플리케이션의 전송 과정을 관리한다. 코드 전환기는 웹 애플리케이션이 유동 웹 애플리케이션 서버에 전송되어 바로 실행 될 수 있도록 코드를 수정하는 기능을 담당한다. 포장기는 전송할 웹 애플리케이션의 모든 구성 요소를 하나의 패키지로 만들고 배치기는 전송된 패키지를 풀어서 유동 서버에 배치하는 역할을 한다. 마지막으로, 전송기는 패키지화된 웹 애플리케이션을 선택된 유동 서버에 전송하는 역할을 한다.



<그림 1. WADN System의 전체 구조>

WADN 코어는 하나의 모듈로 웹 서버와 웹 애플리케이션 서버의 중간에 위치함으로서 다음과 같은 장점들을 갖는다. 첫째, 기존에 있는 웹 서버나 웹 애플리케이션 서버를 특별히 고치지 않고도 동적 확장성을 갖는 WADN 시스템을 만들 수 있다. 간단히 웹 서버와 웹 애플리케이션 서버 사이에 위치한

커넥터 모듈을 WADN 코어로 교체하면 WADN 시스템으로 전환된다. 둘째, WADN 코어가 시스템에 추가되어서 생기는 오버헤드를 줄 일 수 있다. 근원 웹 애플리케이션 서버로 요청을 보내기 전에 WADN 코어에서 미리 선별하여 리다이렉트할 요청에 대한 소켓 커넥션은 만들지 않기 때문에, 불필요한 시스템 자원의 낭비를 막을 수 있다.

그림 2에서 화살표 (1) ~ (6)은 WADN 시스템이 서비스 요청을 받았을 때의 일반적인 작동 흐름을 보여주고 있다.

● 웹 서버는 고객으로부터 서비스 요청을 받는다.(1)

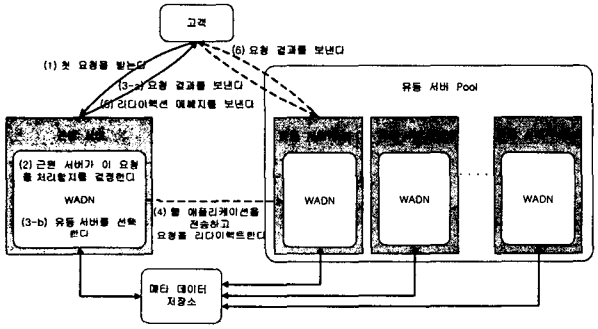
● 웹 서버는 이 요청을 WADN 코어의 요청처리기로 전달한다. 요청 처리기는 근원 서버의 부하(load)를 검사하여 이 요청의 전송 여부를 결정한다.(2)

● 근원 서버의 부하가 크지 않으면, 요청처리기는 근원 웹 애플리케이션 서버에 요청을 전달한다. (3-a) 그렇지 않은 경우, 전송 매니저를 호출한다. (3-b)

● 근원 서버에 있는 전송 매니저는 요청 받은 웹 애플리케이션을 유동 서버로 전송한다. (4)

● 웹 애플리케이션 전송 및 설치가 완료되면, 근원 서버에 있는 요청 처리기는 클라이언트에게 유동 서버로 요청을 보내라는 메시지를 보낸다.(5)

● 클라이언트는 요청 결과를 유동 서버에게 받는다.(6)



<그림 2. WADN의 일반적인 작동 흐름>

3.2 WADN의 구성요소

3.2.1 요청 처리기(Request Processor)

요청 처리기는 서버의 과부하 상황을 탐지하여 서버가 안정적인 웹 서비스를 제공하는 것을 보장한다. 근원 서버에 과부하가 생기면, 요청 처리기는 채택하고 있는 전송 정책과 시스템 모니터로부터 받은 시스템의 자원 정보를 가지고 동적 확장 여부를 결정한다.

3.2.2 시스템 모니터(System Monitor)

시스템 모니터는 웹 서비스를 지원하는 시스템의 자원(CPU, I/O, 메모리, network resource등)에 발생하는 부하를 검사한다. 이렇게 검사한 정보는 요청 처리기와 메타 데이터 저장소에 저장된다. 또한, 다양한 로드 메트릭(load metric)을 사용할 수 있게 하여 여러 자원들을 서비스 제공자 각각의 상황에 맞게 효과적으로 검사할 수 있다.

3.2.3 전송 매니저(Delivery Manager)

전송 매니저는 유동서버를 선택하고 전송하는 기능을 담당한다. 전송 매니저는 서버 선택 정책에 따라 서버 풀(server pool)에서 적합한 유동 서버를 선택한다. 서버는 주로 부하 정도와 유동서버의 위치, 그리고 전송할 웹 애플리케이션을 해당 서버가 가지고 있는 지의 여부로 선택한다.

3.2.4 코드 전환기(Source Converter)

코드 전환기는 전송할 웹 애플리케이션이 가지고 있는 특정 웹 애플리케이션 서버에 종속적인 코드를 유동 웹 애플리케이션 서버에 적합한 코드로 변환 시킨다. 이렇게 함으로써 웹 애플리케이션이 해당 유동서버로 전송 된 후 바로 실행이 가능하도록 하였다.

3.2.5 포장기와 배치기(Packer & UnPacker)

일반적으로, 하나의 웹 애플리케이션은 여러 가지 구성 요소로 이루어져 있다. 포장기는 이러한 웹 애플리케이션이 작동하는데 필요한 모든 구성 요소들을 모아서 하나의 패키지로 압축한다. 배치기는 전송된 웹 애플리케이션을 받아서 압축을 풀고 유동 서버에 배치하는 역할을 담당한다.

3.2.6 전송기(Deliverer)

전송기는 패키지화된 웹 애플리케이션을 선택된 유동 서버로 전송한다. 전송기가 웹 애플리케이션의 전송을 끝마치면, 전송 매니저는 요청 처리기에게 전송의 완료를 알린다. 이후 요청 처리기는 유동 서버에게 요청을 리다이렉트한다.

4.결론

본 논문은 인터넷상에서 동적 컨텐츠를 서비스해주는 시스템이 효율적로 안정적인 서비스를 할 수 있도록 하기위하여 WADN이라는 새로운 시스템을 제안하였다. WADN은 서비스 요청의 수에 따라 동적으로 서버를 확장하여 시스템의 전체 용량을 증가시키는 방법이다. 이를 위해 우리는 유동서버를 관리하는 방법과 유동서버에 웹 애플리케이션을 자동으로 전송, 배치 할 수 있는 방법에 대해서 연구하였다. 기존의 웹 서비스 제공자는 미리 고객의 수요를 예측하기 힘들기 때문에 서버를 효율적으로 사용하기 어려웠다. 우리가 제안한 WADN을 이용할 경우 서버 개수를 동적으로 변화시켜 인터넷의 자원 활용도를 높일 수 있게 되고, 서비스 제공자는 총 소유비용(Total Cost of Ownership, TCO)을 획기적으로 줄이면서 안정적인 서비스를 할 수 있게 된다.

5.참고자료

[1] www.zonaresearch.com

[2] V. Holmedahl, B. Smith, and T. Yang.; "Cooperative Caching of Dynamic Content on a Distributed Web Server"; Proc. of 7th IEEE International Symposium on High Performance Distributed Computing (HPDC-7), Chicago, July 28-31, 1998.

[3] Florescu, Daniela, Issarny, Valerie, Valduriez, Patrick, Yagoub, Khaled; "Caching Strategies for Data-Intensive Web Sites"; Proc. of the Int. WWW Conf. ,Amsterdam , May 15-19, 2000

[4] Karen Witting, Cameron Ferstat, Paul Reed; "A Publishing System for Efficiently Creating Dynamic Web Content"; Proc. of IEEE INFOCOM, 2000

[5] Candan, Li, Luo, Hsiung, and Argrawal; "Enabling Dynamic Content Caching for Database-driven Web sites"; ACM SIGMOD, May 2001

[6] Qiong Luo and Jeffrey F. Naughton and Rajasekar Krishnamurthy and Pei Cao and Yunrui Li; "Active Query Caching for Database Web Servers"; WebDB (Informal Proceedings) 2000

[7] Adero Inc. http://www.adero.com

[8] Akamai Technology. http://www.akamai.com

[9] Digital Island, Ltd. http://www.digitalisland.com

[10] S.Paul and Z.Fei; "Distributed caching with centralized control"; 5th Int.Web caching and Content delivery Workshop, May 2000