동적 공유객체 서블릿을 사용한 웹 서버 어플리케이션 기술

김대중° 곽덕훈
한국방송통신대학교 평생대학원 정보과학과
daejung@sysdeveloper.net° dhkwak@knou.ac.kr

A Web-server Application Technology using Dynamic Shared Object Servlet

Daejung Kim° Dukhoon Kwak
Department of Computer Science, Korea National Open University

요 약

웹 어플리케이션 개발에 있어서 동적인 컨텐츠를 생성하는 기술은 매우 다양하다. 본 논문은 동적인 컨텐츠를 생성하는 어플리케이션을 위하여 동적공유객체를 사용하는 기술인 DHE(DCL HTTP Server Extension)에 대하여 다룬다. DHE는 DHE 서블릿과 DHE 서블릿 컨테이너로 이루어져 있다. 서블릿(Servlet)은 HTTP 요청에 대한 구체적인 처리를 하도록 개발된 DSO이고 서블릿 컨테이너에 의하여 실행된다. 서블릿 컨테이너는 이미 널리 사용되고 있는 웹 서버 소프트웨어의 플러그인(plug-in) 형태로 개발되며 서블릿에게 웹 서버 소프트웨어간에 이식이 가능하도록 하는 독립된 환경을 제공한다. DHE의 유효성을 검증하기 위해 동일한 알고리즘이 적용되어 작성된 ASP, PHP, JSP 어플리케이션과 성능비교 실험을 실시 하였다. 실험의 결과는 DHE가 가장 적은 VM(Virtual Memory)를 사용하고 있었고 200라인 이상(27.8KB)의 문자열을 생성하는 실험에서 단위 시간당 처리할 수 있는 HTTP 요청의 개수가 JSP에 비하여 3배 이상, ASP, PHP에 비하여 6배 이상의 결과를 얻었다.

1. 서론

동적인 컨텐츠를 생성하는 웹 서버 어플리케이션 주요기술은 ASP[1], PHP[2]와 같이 스크립트를 위한 인터프리터를 웹 서버의 프로세스에 플러그인 형태로 상주시키는 방법과 JSP, 자바 서블릿과 같은 JAVA기술[3]을 적용한 형태로 발전하고 있다. 기존의 이러한 방식들의 공통적인 단점은 각각의 실행환경에 의한 특징 때문에 시스템 자원에 대한 효율을 높이는데 있어서 한계점을 가진다는 것이다. ASP, PHP와 같은 인터프리터 방식은 HTTP 요청마다 스크립트가 번역되어 실행되고, JAVA 기술을 적용한 경우는 실행환경이 JVM(Java Virtual Machine)에 제한되며 대체적으로 JVM이 시스템의 메모리 자원을 많이 필요로 하는 단점이 있다.

웹 서버 어플리케이션의 실행 속도를 높이기 위한 최선의 방법은 웹 서버의 API를 사용하여 서버의 플러그인 형태로 어플리케이션을 개발하는 것이다. 대부분의 웹 서버 소프트웨어들은 서버의 기능을 확장하기 위한 방법으로 C언어 기반의 API를 제공한다. 그러나 웹 서버의 API는 서버 소프트웨어에 종속되기 때문에 서로 다른 서버 소프트웨어간의 이식이 거의 불가능한 단점이 있다.

본 논문에서 제안하는 웹 서버 어플리케이션 기술은 DHE 서블릿 인터페이스를 구현한 운영체제의 동적공유객체(DSO: Dynamic Shared Objects)와 이에 대한 실행환경을 제공하는 웹 서버의 플러그인 모듈인 DHE 서블릿 컨테이너에 관한 것이다. DHE 서블릿은 C/C++와 같은 컴파일러 언어로 작성되어 네이티브(native) 바이너리로 만들어지기 때문에 ASP, PHP, JSP와 같은 기존의 방법들과 비교하여 최소의 시스템자원 환경에서 최대의 실행속도를 기대할 수 있다. 그러나, 이것은 웹 서버 프로세스에 적재(load)된 후 실행기회가 주어지면 어떠한 방법으로든 통제되지 않기 때문에 본 연구는 서블릿의 실행환경뿐만 아니라 서블릿의 디버깅을 위한 개발환경에 많은 부분이 할애 되었다.

2. 관련연구 및 배경지식

2.1 Apache HTTP Server

전통적인 아파치 서버의 프로세스 구조는 단일의 부모 프로세스와 재사용되는 자식 프로세스로 구성된다. 부모 프로세스는 서버가 시작될 때 구성(configuration)을 읽어 들이고 자식 프로세스의 풀(pool)을 관리하며 자식 프로세스는 단일의 소켓 연결을 사용하여 HTTP 요청을 처리한다. 전통적인 프로세스 모델은 fork 시스템호출을 제공하는 UNIX와 같은 프로세스 기반의 운영체제에 적당하며 Windows에서는 스레드 기반으로 구성된다. 아파치 2.0부터 POSIX 스레드 라이브러리(pthread)를 사용할 수 있는 UNIX 운영체제에서는 혼합형 다중프로세스 다중스레드(hybrid multi-process multi-threaded)구조가 사용될 수 있다[4][5].

아파치 웹 서버는 모듈을 통하여 기능을 확장할 수 있다. 모듈은 서버를 컴파일 할 때 httpd 실행파일에 정적으로 링크하거나 DSO로 분리하여 컴파일 할 수 있다. 본 연구의 DHE 서블릿 컨테이너는 아파치 웹 서버의 확장모듈로 구현된다.

2.2 Microsoft IIS

Windows NT 기반의 운영체제에서 인터넷 서비스를 위한 구성요소인 IIS(Internet Information Services)는 응용프로그램 보호기능을 위하여 다중 프로세스 구조를 가질 수 있다. IIS 5.0 응용프로그램은 웹 사이트의 디렉터리(directory)경계로 구분되며 웹 서버(inetinfo.exe)와는 다른 별도의 프로세스(dllhost.exe)에서 격리되어 실행될 수 있다. ISAPI 확장모듈은 IIS에서 실행되는 실제 어플리케이션으로 윈도우의 DLLs로 구현되며 IIS에 의해 제어되는 프로세스에 적재된다. ISAPI 확장

모듈은 해당 서비스에 대하여 HTTP 요청이 있는 경우 적재되고 처리가 완료되면 제거된다. 만약 응용프로그램이 캐시 되도록 설정되어 있으면 서버가 종료될 때까지 적재된 상태를 유지한다[1][6].

2.3 Dynamic Shared Objects

현대의 많은 운영체제는 특별한 형식의 실행코드 조각을 담고 있는 파일인 동적공유객체(DSO: Dynamic Shared Objects)를 응용프로그램의 실행시간에 적재하여 사용할 수 있도록 하는 기능을 제공한다. 이러한 기능은 운영체제마다 약간의 차이점은 있지만 기본적인 기능은 거의 비슷하며 크게 두 가지로 구분될 수 있다.

첫 번째는 공유라이브러리(shared library)라고 부르며 일반적으로 UNIX에서는 libfoo.so, libfoo.so.1.2와 같이 lib로 시작한다. 윈도우에서는 .dll 확장자를 갖으며 사용하고자 하는 DLL의 임포트(import)라이브러리에 대한 정적인 링크가 필요하다. 이들 공유 라이브러리는 어플리케이션(실행파일이나 DSO)이 실행되기 위해 적재될 때 운영체제의 런타임 링커가 모듈의 의존관계를 분석하여 참조된 공유 라이브러리를 반복적으로 연결하여 필요한 심볼을 찾고 재배치 한다. 두 번째는 공유객체(shared objects)라고 부르며 파일의 확장자는 자유롭다. 공유객체는 운영체제가 제공하는 DSO관련 함수를 사용하여 적재하고 필요한 함수나 데이터의 주소를 직접 찾아서 사용한다[7][8]. 본 연구의 서블릿이 여기에 해당된다.

2.4 DCL 프로젝트

DCL(Daejung's Class Library) 프로젝트는 GNU/Linux를 비롯한 UNIX 운영체제와 Microsoft의 Windows 운영체제에서 소스레벨 이식성을 전제로 하여 시스템 프로그래밍을 위한 C++ 라이브러리 시스템 구축을 위한 개인 프로젝트로 운영체제 객체, 콜렉션(collection)클래스, 데이터베이스 드라이버와 이에 접근을 위한 클래스 등 다양한 클래스들의 패키지로 이루어져 있다[9]. 본 논문에서 다루고 있는 DHE는 DCL 프로젝트의 서브프로젝트이다.

3. DCL HTTP Server Extension

3.1 기본 구성요소와 인터페이스 모델

DHE의 실행환경을 이루는 기본 구성요소는 HTTP 서버, 서블릿 컨테이너 그리고 서블릿의 세 요소로 이루어져 있다. DHE는 독립적인 서버로 동작하지 않고 기존에 널리 사용되는 HTTP 서버의 환경에서 동작한다. 서블릿은 HTTP 요청을 분석하고 이에 대한 실질적인 서비스를 담당하는 부분으로 DSO로 작성되며 파일의 확장자는 .dhe를 갖는다. 서블릿 컨테이너는 HTTP 서버가 제공하는 API의 프로세스 환경 내에서 동작하며 HTTP 서버의 확장모듈로 구현되어 서블릿 모듈에 대한 적재와 제거, 그리고 서블릿에게 실행 기회를 제공한다.

DHE 인터페이스의 역할은 HTTP 서버, 서블릿 컨테이너 그리고 서블릿 간의 컴파일타임 의존관계를 제거하고 각 구성요소 사이에서 메시지를 전달하는 것이다. 특히, 컴파일타임 의존관계를 제거하는 역할은 매우 중요하다. 이것은 HTTP 서버의 API가 C언어만을 사용할 수 있을 경우 C++를 비롯한 다른 언어를 사용할 수 없는 경우가 있는데 Apache 서버의 경우가 이에 해당된다. 인터페이스 명세는 표준 C언어로 기술되어 있기 때문에 서블릿 개발을 위해 반드시 C/C++를 사용할 필요는 없으며 표준 C언어 함수 호출규약을 제공할 수 있고 DSO를 개발할 수 있는 언어는 무엇이든 가능하다. 다만 서블릿 모듈의 DSO 진입점(entry point)은 반드시 C언어 네임 맹글링(name-mangling)을 사용하여야 한다.

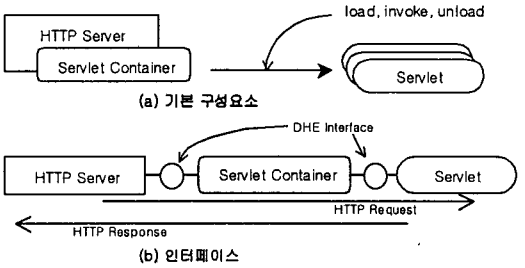

그림 1 DCL HTTP Server Extension

DHE 인터페이스를 사용하여 서블릿을 개발할 때에는 인터페이스 명세 외의 다른 어떠한 임포트(import) 라이브러리로 필요 없다. 서블릿 컨테이너가 서블릿을 호출할 때에는 DHE 인터페이스를 통해 서버의 API에 접근할 수 있는 함수와 데이터를 함께 제공하기 때문이다.

3.2 서블릿 컨테이너

DHE.INI는 서블릿 컨테이너의 설정과 서블릿 서비스에 관한 정책을 저장하고 있는 파일로 기본적인 구현은 단일의 텍스트 파일로 이루어져 있다. 만약, HTTP 서버가 다중 프로세스 구조일 경우, 서블릿 컨테이너는 관리서버로부터 초기화되고 통제된다. DHE.INI의 SERVLET섹션은 서버의 실행시간에 특정 서블릿 모듈에 대하여 사용가능(enable)과 캐시(cache)설정에 대한 수정이 가능하고 이는 실시간으로 반영된다. enable 속성은 서버의 실행시간에 서블릿 모듈의 파일을 수정할 수 있도록 하는 방법을 제공한다. 이는 서블릿 모듈은 DSO 이기 때문에 서블릿이 사용되고 있는 중에는 모듈의 파일이 수정될 수 없기 때문이다. cache 속성은 자주 호출되는 서블릿에 대하여 DSO의 반복적인 런타임링킹 과정을 제거한다.

3.3 서블릿 개발과 디버깅 환경

서블릿을 개발하는데 있어서 다양한 언어를 사용할 수 있지만 서블릿 컨테이너가 HTTP에 관한 모든 부분을 처리해 주지 않기 때문에 상당 부분의 HTTP 관련 코드가 필요하다. DCL 프로젝트의 DCLNet 라이브러리는 C++로 작성되어 있으며 서블릿을 구현하는데 필요한 관련 HTTP와 HTML 클래스 그리고 서블릿 클래스를 사용한 프레임워크를 제공한다.

DCLNet을 사용하여 서블릿을 개발할 때에는 HttpServlet클래스로부터 파생된 단 하나의 클래스가 있어야 하고 실행시간에는 이 클래스의 인스턴스(instance)가 단 한 개 있어야 한다. 일반적인 서블릿 모듈을 개발하고자 할 때에는 HttpServletEx 클래스를 사용한다. 이 클래스는 HttpServletContextEx 클래스와 함께 사용되며 서블릿의 서비스 요청에 동반한 Cookie, QUERY_STRING, POST 데이터 등에 대한 처리를 자동화 하고 이에 대한 컬렉션 객체를 유지한다. 이 클래스의 중요한 기능은 디버그 버전에서 DCL의 실행시간 디버깅 기능[10]을 사용할 수 있도록 하게 하고 이의 결과를 웹 브라우저를 통하여

보고하는 것이다. DCL의 실행시간 디버깅의 기능은 다중 스레드 환경에서 스레드별 ASSERT, TRACE를 가능하게 하고 잘못 사용된 동적메모리 객체에 대한 정보를 표시한다.

4. 성능평가를 위한 실험

4.1 실험환경

실험에 사용된 서버는 Intel의 Pentium III 850Mhz/512MB, Pentium 4 1500Mhz/512MB의 하드웨어에 GNU/Linux와 Microsoft Windows 2000 Server를 각각 사용하였고, 클라이언트는 Pentium 4 1400Mhz/512MB에 Windows 2000을 설치하여 이들을 100Mbps 스위치 허브로 연결하였다.

서버 어플리케이션 기술들 간의 비교를 위하여 ASP, PHP, JSP 스크립트와 DHE 서블릿을 작성하였다. 각각은 그것을 작성하는 언어적 특성이 있기 때문에 동일하게 작성할 수는 없으나 문자열 연산을 중심으로 동일한 알고리즘이 되도록 하여 정해진 개수의 문자열을 생성하도록 하였다. 따라서 동일한 값을 지정하면 ASP, PHP, JSP, DHE가 생성하는 HTML의 크기는 동일하다. 다만, HTTP 서버마다 HTTP Header-Field를 동일하게 생성하도록 하는 것은 임의로 제어할 수 없기 때문에 전송되는 전체 바이트의 개수는 서버마다 다를 수 있다.

웹 서버의 실질적인 성능은 단위 시간당 얼마나 많은 요청을 처리할 수 있는가에 달려 있다고 볼 수 있는데 이를 위하여 사용된 측정 도구로 ApacheBench[4]를 사용하였다.

4.2 실험결과

실험은 전체 요청의 개수와 동시요청의 개수를 각각 640, 64개로 하고 문자열의 개수를 지정하는 line 파라미터를 20, 200, 2000으로 하여 각각을 측정 하였다. 그림에서 dhe(c)로 표기된 것은 서블릿을 캐시(cache)되도록 설정한 것이고 그렇지않은 것은 기본(default)상태 이다. 기본캐시상태는 다중 스레드 방식의 HTTP 서버에 동시 접속이 있는 경우 서블릿은 하나 이상의 스레드에 의해 사용될 수 있으며 Linux의 apache 1.3.23의 경우 다중 프로세스 방식으로만 서비스 하기 때문에 각각의 요청마다 서블릿의 적재와 제거가 반복된다.

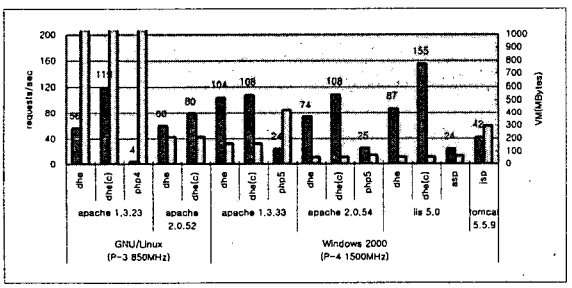

그림 2 ApacheBench 실험결과: line=200, 28525bytes

기본캐시상태에서 "line=20" 지정한 실험은 다른 서버 어플리케이션 기술에 비하여 두드러진 성능을 보이지 않고 IIS 5.0에서는 ASP보다 떨어졌다. 이것은 각 요청에 있어서 서블릿의 실행 시간에 비하여 서블릿의 링킹시간이 더 많이 포함되어 있기 때문이다. 그러나 서블릿의 실행시간이 상대적으로 길어지게 되면 DHE의 성능은 두드러지게 드러난다.

국내의 대표적인 검색엔진과 쇼핑몰이 생성한 HTML의 크기가 대부분 50KBytes이상이고 300KBytes에 이르는 것을 감안하면 "line=200"과 "line=2000"의 실험결과가 유효하다고 볼 수 있다.

5. 결론

본 논문은 네이티브 바이너리로 컴파일된 운영체제의 DSO를 사용하여 웹 서버 어플리케이션을 개발할 수 있도록 하는 DHE라 불리는 기술을 제안한다. DHE의 기본 구성요소는 서블릿과 서블릿 컨테이너가 있으며 서블릿 컨테이너에 의하여 서블릿이 실행된다. 서블릿 컨테이너는 기존에 널리 사용되는 웹 서버의 플러그인 형태로 구현되어 동작하며 Windows와 Linux에서 IIS 5.0과 Apache 1.3.x, 2.x 버전을 위한 컨테이너를 개발하였다. 서블릿 컨테이너는 서로 다른 웹 서버간의 차이점을 흡수하여 같은 운영체제 하에서는 서로 다른 웹 서버라 할지라도 같은 서블릿 모듈을 사용하는 것이 가능하다.

DHE 서블릿 개발을 위하여 HTTP, HTML관련 클래스 및 서블릿 관련 클래스가 DCL에 추가되었고, 다중 스레드 환경에서의 실행시간 디버깅을 위한 DCL의 핵심 코드가 재작성 되었다. 디버그 버전으로 컴파일된 서블릿은 ASSERT와 TRACE를 가능하게 하고 잘못 사용된 동적 메모리 객체 사용에 관한 정보를 표시한다.

기존의 널리 사용되고 있는 동적 컨텐츠 기술과의 성능비교를 위한 실험에서 DHE는 가장 적은 VM을 사용하고 있었고 200라인(27.8kb)이상의 문자열을 생성하는 서버 어플리케이션에서 단위 시간당 처리할 수 있는 HTTP 요청의 개수가 ASP, PHP와 비교하여 6배 이상, JSP와는 3배 이상의 결과를 얻었다. 이러한 결과는 DHE를 사용하여 웹 서비스를 하게 될 경우 기존 기술을 사용하는 3~6개 이상의 시스템을 단 한 개의 시스템으로 대체할 수 있고 서버 어플리케이션의 빠른 응답은 보다 다양한 웹 어플리케이션 개발에 대한 가능성을 의미한다.

참고문헌

[1] Microsoft Corporation, "IIS Web Development SDK", http://msdn.microsoft.com/
[2] The PHP Group, "PHP Hypertext Preprocessor", http://www.php.net/
[3] Sun Microsystems, "Java Technology", http://java.sun.com/
[4] Apache Software Foundation, "The Apache HTTP Server Project", http://httpd.apache.org/
[5] Ibrahim Haddad, "Apache 2.0: The Internals of the New, Improved", 2001.08, http://www.linuxjournal.com/
[6] Microsoft Corporation, "Internet Information Services 설명서"
[7] Sun Microsystems, "Linker and Libraries Guide", 2002, http://docs.sun.com/
[8] Microsoft Corporation, "Dynamic-Link Libraries", http://msdn.microsoft.com/
[9] 김대중, "DCL Project", http://www.sysdeveloper.net/dcl/
[10] 김대중, "DCL Runtime Debugging Support", 2005.06
[11] 김대중, "DCL HTTP Server Extension Version 2", 2005.06