2007년도 대한전기학회 하계학술대회 논문집 2007. 7. 18 - 20

Web CGI 모듈의 통합운영 방식에 의한 웹 서버 성능개선

한소희, 민수홍, 조동섭
이화여자대학교 컴퓨터학과

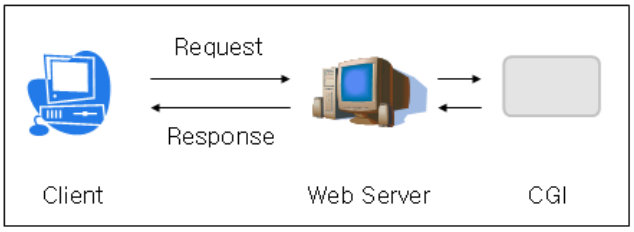
Performance Enhancement of Web Server Through Integration of Web CGI Modules

So-hee Han, Su-hong Min, Dong-sub Cho
Computer Science and Engineering, Ewha Womans University

Abstract - WWW(World Wide Web)의 개발 이래 내 컴퓨터 안에만 존재하는 워드와 한글 등의 문서들은 네트워크를 타고 지구 어딘가에 있는 누군가의 모니터에 디스플레이되고 있다. 그러나 컴퓨터 기술은 이와 같은 단순한 HTML파일을 보여주기보다 방명록이나 동영상처럼 동적으로, 실시간으로 정보를 보여주는 수준으로 진화해야 했다. 이러한 기술을 처음으로 가능하게 한 것이 CGI 기술이다. 초기의 CGI 기술은 성능상의 문제로 개발과 발전을 거듭해 현재의 FastCGI나 mod_perl과 같은 기술이 제안되었다. 그러나 FastCGI나 mod_perl도 아직은 동적 정보를 제공하는 완벽한 기술은 아니며 각각 장단점을 지니고 있다. 본 논문에서는 각 기술의 장단점을 살펴보고 이들의 장점을 통합한 개선된 CGI기술을 제안한다.

1. 서 론

최근의 인터넷 서비스는 단지 HTML문서를 보여주는데 그치지 않는다. 사용자가 원하는 서비스를 요청시마다 동적으로 생성해주어야 한다. 따라서 웹 서버가 데이터베이스에 저장된 문서들을 참조하는 것만으로는 동적 정보를 생성해내기 힘들다. CGI 기술은 이러한 웹 서버의 역할을 확장해준다. 즉 CGI 기술은 애플리케이션으로써, 웹 서버가 클라이언트로부터 요청을 받아서 CGI 프로그램에게 실행을 부탁하면 CGI 프로그램은 웹 서버로부터의 기본 정보를 토대로 요청된 정보를 바로 생성해준다. 다음 웹 서버에게 생성된 정보를 전달하고 다시 웹 서버는 클라이언트에게 마지막 결과를 HTML 형태로 전달한다. 이 과정을 다음 그림과 같이 표현 할 수 있다.



<그림 1> CGI 기술

최근의 대형 사이트들은 방대한 양의 데이터 처리를 위해 동적 정보를 생성 해주는 웹 애플리케이션 서버를 따로 두기도 하지만, 데이터베이스의 규모가 작고 적은 수의 사용자에게 단순한 서비스를 제공하는 경우나 임베디드 시스템을 위한 개발 프로그램에 있어서는 웹 서버 내에서 동적으로 정보를 생성해주는 CGI 기술이 더 적합하다. 그러나 초창기에 개발된 CGI 기술은 성능상의 문제점을 앉고 있었다. 바로 프로세스 생성에 의한 시간지연과 자원 오버헤드다. 이에 따라 성능 개선과 수행 속도 개선을 앞세운 FasgCGI와 mod_perl과 같은 기술이 제안되었다. 위의 기술들은 적합한 환경에 따라 선호, 사용되고 있으며 초창기 CGI 기술과 마찬가지로 각각의 장단점을 지니고 있다. 따라서 본 논문에서는 이미 제안된 위 기술들을 살펴보고 각각의 장점을 통합한 개선된 CGI 기술을 제안한다.

본 논문의 구성은 다음과 같다. 2장 본론에서 최초 CGI 기술 이후에 제안되어온 Web Server API와 FasgCGI, mod_perl에 대해 알아봄으로써 현제까지의 CGI 기술의 상황에 대해 정리한다. 그리고 본 논문에서 제안한 새로운 통합 CGI 기술에 대해 설명하고 마지막 3장에서는 결론과 향후 과제로 본 논문을 맺는다.

2. 본 론

2.1 초창기 CGI 기술의 문제점

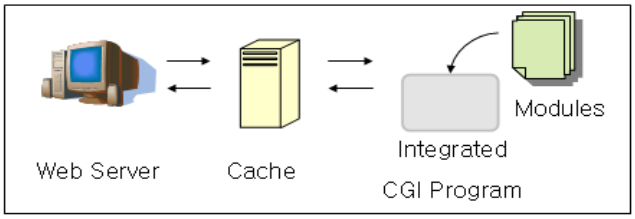
초창기 개발된 CGI 기술의 문제점으로 지적된 것은 앞서 언급한 성능상의 문제다. 즉 웹 서버는 요청된 정보를 생성하기 위해 CGI 프로그램을 수행시키는 데, 문제는 모든 요청마다 fork과정을 통해 프로세스를 생성한다는 것이다. 또한 프로세스 생성과 함께 환경 설정과 데이터베이스 커넥션도 매번 새로 해주어야 한다. 이렇게 하면 요청 처리 시간이 길어지는 한편 모든 프로세스가 동시에 자원을 요구하므로 자원 오버헤드가 심각해진다.

2.2 Web Server API, FastCGI, mod_perl

위와 같은 초기 CGI의 문제를 해결하기 위해 제시된 첫 번째 방안은 Web Server API이다. Web Server API는 CGI 프로그램이 웹 서버 프로세스 내에서 하나의 프로세스로 수행이 되면서 클라이언트의 요청을 처리한다. 따라서 fork과정이 생략되므로 수행 처리 속도가 빨라진다. 그러나 웹 서버 내에서 수행이 되므로 버그가 담긴 프로세스가 수행될 경우 웹 서버 전체에 영향을 줄 수 있고 vendor마다 적용 API가 달라 표준으로써 사용 할 수가 없다. Web Server API 보다 개선된 FastCGI는 웹 서버 프로세스 내에서 수행되는 것이 아니고 웹 서버 프로세스와 독립적으로 수행된다. 따라서 Web Server API에 비해 보안상 안전하다. 또한 FastCGI는 OS의 환경을 이용하는 대신 TCP/IP 소켓 방식을 택함으로써 요청이 들어오면 해당 요청을 수행하고 다음 요청을 기다린다. 이렇게 하나의 프로세스만 수행됨으로써 초기 CGI 기술과 같은 fork 과정이 없고 자원 커넥션도 지속적이다. 다음으로 제안된 mod_perl은 전 세계 웹 서버의 60%이상을 장악하고 있는 아파치 웹서버를 위한 CGI 프로그램이다. Mod_perl은 애플리케이션 프로그램이 모듈단위로 구성되어 웹 서버 프로세스에 필요시마다 적재된다. 현재 최고의 처리 속도를 자랑하지만 단점은 아파치 웹 서버에서만 사용가능하고 또한 초기 CGI나 FastCGI와 달리 사용자가 아파치 웹 서버 API에 새로이 익숙해져야 한다는 것이다.

2.3 개선된 통합 CGI 기술

본 논문에서는 앞서 살펴본 이미 제안된 CGI 기술의 장점들을 통합한 CGI 기술을 제안한다. 먼저 초기 CGI기술이나 FastCGI 처럼 요청에 따른 프로세스를 웹 서버와 독립적으로 수행한다. 이 프로세스는 경량의 기본 독립 프로세스로써 계속 수행된다. 따라서 모든 자원에 커넥션을 지속적으로 유지할 수 있다. 동적으로 생성되는 웹 페이지를 보면 한 페이지 내에도 일부는 페이지의 정적인 정보를 포함하고 있다. 따라서 페이지 전체를 매번 새로 생성해내는 것은 시간과 자원 낭비다. 이에 페이지 내의 기능 적인 부분들을 mod_perl과 같이 모듈화 하여 구성한 후 앞서 설명한 기본 독립 프로세스에 기능별로 적재, 통합하여 실행한다. 또한 자주 참조, 수행되는 모듈들은 미리 통합시켜 cache에 저장한다. 이 과정을 다음 그림과 같이 표현 할 수 있다.



<그림 2> 제안된 통합 CGI 기술

위와 같이 수행했을 경우 FastCGI에 의한 독립프로세스 운영과 mod_perl에 의한 모듈 구성으로부터 성능과 속도 향상, 두 가지 모두를 실현할 수 있게 된다. 먼저 기본 독립프로세스로 운영함으로써 fork과정의 프로세스 생성이 생략 되므로 속도가 개선되는 한편, 웹 서버에도 영

향을 주지 않는다. 그리고 모듈 단위로 구성함으로써 불필요한 모듈에 대해서는 수행을 하지 않으므로 수행속도를 단축할 수 있다. 또한 추가적인 기능이 필요 할 경우 모듈을 작성, 추가함으로써 기능을 확장 할 수 있다. 마지막으로 자주 참조, 실행되는 통합된 모듈들을 미리 cache에 저장함으로써 모듈을 통합하는 시간마저 단축되어 수행속도를 높일 수 있다.

3. 결 론

본 논문에서는 최근의 웹 서버 애플리케이션의 경향과 달리, 작은 규모의 데이터베이스를 사용하거나 적은 수의 사용자가 간단한 동적 데이터를 제공 받고자 할 때, 그리고 임베디드 시스템 개발에 있어서는 여전히 CGI 기술이 더 적합하다는 판단 아래 기존의 CGI기술의 문제점과 이 문제를 해결하기 위해 선행 연구된 FastCGI와 mod_perl에 대해서 살펴봤다. 그리고 본 논문에서는 FastCGI와 mod_perl의 장점을 통합, 개선시킨 통합 CGI기술을 제안하였다. 제안한 통합 CGI기술의 핵심요소 3가지는 기본 독립프로세스 운영, 모듈 단위 구성, cache정책사용이며 제안한 통합 CGI 기술은 이 요소들로부터 성능과 속도 모두를 향상 시킬 수 있다. 앞으로 진행 할 과제는 독립 프로세스나 모듈구성과 같은 성능 개선 차원의 새로운 CGI 기술에 대해 연구하는 것이다.

[참 고 문 헌]

[1] 전광일 외 10명, “운영체제 내부구조 및 설계원리”, 2006

[2] Larry Wall, Tom Christinasen & Ranal L. Schwarts, “Programming Perl”, 1996

[3] Ganesh Venkitachalam, Tzi-cker Chiueh “High Performance Common Gateway Interface Invocation” IEEE Workshop on Internet Applications, P,4, 1999

[4] Alexandros Labrinidis, Nick Roussopoulos “Generating Dynamic content at database-backed web servers: cgi-bin vs mod_perl” ACM SIGMOND Record, Volume 29 Issue P,26~31, 2000

[5] Anindya Datta, Kaushik Dutta, Helen Thomas, and Debra VanderMeer, Krithi Ramamritham “Accelerating Dynamic Web Content Generation” Internet Computing, IEEE, Volume 6, Issue 5, P,27-36, 2002