확장된 서버 사이드 스크립트 기반의 웹 페이지 분석

유일선·양성미[0]
한국성서대학교 정보과학부
{isyou, smyang[0]}@bible.ac.kr

Analysis of Web-pages based on an Extended Server-Side Script

Il-Sun You and Seong-Mi Yang[0]
Dept. of Information Science, Korean Bible University

요 약

CGI 프로그래밍 기법이후에 동적 웹 페이지 기술은 CGI 프로그래밍 기법을 개선하기 위해 활발히 연구되었고, 그 결과 Fast-CGI, 서버 사이드 스크립트 그리고 확장 서버 사이드 스크립트 기법이 제안되었다. 본 논문에서는 이러한 동적 웹 페이지 기술을 고찰함과 동시에 벤치마크 테스트를 통해 확장된 서버 사이드 스크립트 기법과 서버 사이드 스크립트 기법의 성능을 비교분석하였다. 벤치마크 테스트 결과에 의하면 확장 서버 사이드 스크립트 기법이 성능과 프로그램 개발 및 유지보수 비용을 함께 고려할 때 2-계층구조와 3-계층구조에서 우수하다는 것을 알 수 있었다. 특히, 대부분의 웹 시스템이 데이터베이스에 의존하는 전형적인 3-계층구조를 따르고 있기 때문에 웹 환경에서 확장 서버 사이드 스크립트 기법은 다른 기법에 비해 우수한 효율성을 나타내리라 기대된다.

1. 서 론

인터넷이 발전하면서 웹에서 정보를 처리하여 전달하는 기술이 주요 기술로 부각되었다. HTTP(Hypertext transfer protocol)를 이용한 초기 웹 페이지는 정적인(static) 것으로, 웹 서버는 웹 브라우저가 요청한 주소(URL)에서 문서를 찾아 그대로 전송하였다[1-2].

그러나 컴퓨터 시스템과 인터넷 환경이 빠르게 발전하면서 사용자 요청에 따라 문서를 만들어 응답해주는 동적(dynamic) 웹 페이지 기술이 필요하게 되었으며 이를 위해 사용자의 입력을 받을 수 있도록 폼(form) 태그와 여러 가지 입력 태그들이 HTML(Hypertext Markup Language)에 추가되었고 웹 서버와 응용 프로그램 사이의 상호 통신하는 표준인 CGI(Common Gateway Interface)가 제안되었다. 이들 폼 태그와 CGI를 적용하여 웹 서버를 확장하는 기법은 (이하 CGI 프로그래밍 기법) 웹 시스템 개발자가 다양한 동적 웹 페이지 생성 프로그램을 개발할 수 있도록 하여 웹 시스템 개발 패러다임을 정적 웹 페이지에서 동적 웹 페이지로 전환하는 촉매역할을 하였다. 그러나 CGI 프로그래밍 기법은 다음과 같은 두 가지 문제를 갖는다[1]. 첫 번째 문제는 CGI 프로그램 개발 및 유지보수 비용이 크고, 작성한 프로그램에서 오류가 발생할 경우 웹 서버에 치명적인 위험을 가할 수 있다는 것이고, 두 번째 문제는 독립된 프로세스 형태로 개발되었기 때문에 사용자 요청이 있을 때마다 새로운 프로세스를 포크(fork)하여 웹 서버에 성능부담을 초래하는 것이다. 위와 같은 문제점들을 해결하기 위해 서버 사이드 (Server-side) 스크립트 언어가 제안되었다. 서버 사이드 스크립트 언어는 사전에 별도의 번역 과정을 거치지 않고 처리할 때마다 즉시 번역되어 실행되는 인터프리터 언어이기 때문에 웹 페이지 개발과 유지보수 부담을 경감시켰고 실제 명령은 인터프리터가 실행하기 때문에 치명적인 오류로 인한 위협에 대응할 수 있다. 그러나 이러한 장점에도 불구하고 인터프리터 언어의 한계로 인해 다른 컴파일 언어와 비교하여 실행 속도가 현저하게 느린 단점이 있다[1].

최근에 개발자에게는 서버 사이드 스크립트 언어의 장점을 유지하면서 컴파일 기법을 도입하여 처리 속도를 높이는 확장 서버 사이드 스크립트 기법이 제안되었으며 기술의 적용이 빠르게 확산되고 있다[3-4]. 따라서 확장 서버 사이드 스크립트 기법의 효과적인 적용을 위해 기존 동적 웹 페이지 기술과의 체계적인 비교분석이 요구된다. 이를 위해 본 논문에서는 벤치마크 테스트를 적용하여 성능적 측면에서 확장된 서버 사이드 스크립트 기법과 기존의 서버 사이드 스크립트 기법을 비교분석하고자 한다.

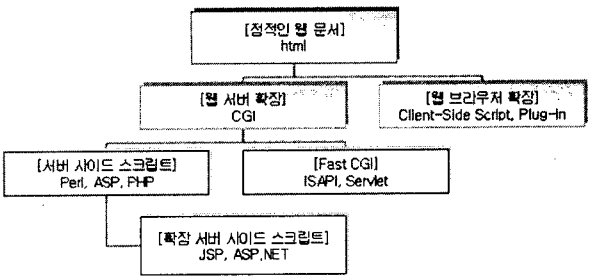
본 논문의 구성은 다음과 같다. 2장에서는 동적 웹 페이지 기술을 분석하고 3장에서는 확장된 서버 사이드 스크립트 기법의 성능분석을 위한 벤치마크 테스트 방법을 기술한 후에 그 결과를 고찰한다. 4장에서는 결론과 함께 향후 연구 과제를 제시한다.

2. 동적 웹 페이지 기술 분석

CGI 프로그래밍 기법 제안이후, 동적 웹 페이지 기술은 1장에서 언급된 CGI 프로그래밍 기법의 두 가지 문제를 해결하기 위해 활발히 연구되었다. 제안된 기술에는 [그림 1]과 같이 Fast-CGI, 서버 사이드 스크립트 그리고 확장 서버 사이드 스크립트 기법이 있다.

2.1 Fast-CGI

앞서 언급된 CGI 프로그래밍 기법의 두 번째 문제인 성능문제에 대한 해결책으로 마이크로소프트의 ISAPI(Internet Server Application Programming Interface)와 자바의 서블릿(Servlet)과 같은 Fast-CGI 프로그래밍 기법이 제안되었다. 이 기술은 CGI가 독립된 프로세스가 아닌 웹 서버의 쓰레드(thread) 형태로 실행되도록 하여 웹 서버의 성능부담을 최소화하는 것을 목



[그림 1] 웹 기술의 변천도

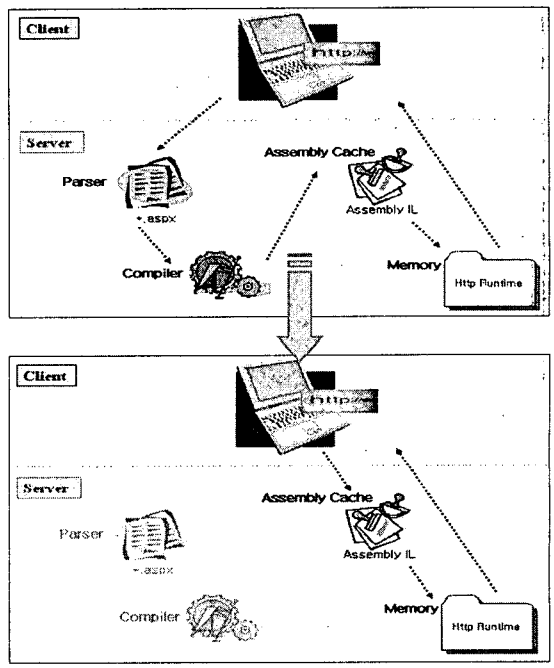
표로 한다. 그러나 이 기법은 CGI의 첫 번째 문제인 높은 개발 및 유지보수 비용문제를 해결하지 못하였고, 특히 웹 서버와 자원을 공유하는 쓰레드 기반의 구조이기 때문에 CGI 프로그램 오류로 인한 웹 서버의 위험성을 증가시켰다. Fast-CGI는 가장 속도가 빠른 동적 웹 페이지 기술이기 때문에 서버 사이드 스크립트 언어의 인터프리터와 같이 다른 동적 웹 페이지 기술을 지원하는 기반 기술로서 사용된다.

2.2 서버 사이드 스크립트

CGI의 두 가지 문제를 동시에 해결하기 위한 방법으로 Perl(Practical Extraction and Reporting Language), ASP(Active Server Page) 그리고 PHP(Hypertext Preprocessor)와 같은 스크립트 중심의 서버 사이드 스크립트 기법이 제안되었다. 서버 사이드 스크립트 기법은 인터프리터 기반의 언어로 CGI 프로그래밍 기법의 첫 번째 문제인 높은 개발 및 유지보수 비용 문제를 해결하였고, 인터프리터를 Fast-CGI 프로그램으로 구현하여 성능문제와 안전성 문제를 해결하였다. 이러한 장점을 토대로 서버 사이드 스크립트 기법은 현재 동적 웹 페이지 기술의 주류를 이룬다. 특히, 마이크로소프트에서 제안한 ASP는 저렴한 개발 및 유지보수 비용과 ODBC(Open Database Connectivity)를 통한 데이터베이스 연동 용이성 그리고 COM(Component Object Model)을 통한 확장성을 가지며 PHP는 플랫폼(platform) 독립성과 공개소스, 무료 소프트웨어라는 장점과 함께 다양한 기능의 함수 제공 및 빠른 속도의 이점을 갖는다.

2.3 확장된 서버 사이드 스크립트

앞서 언급된 장점에도 불구하고 서버 사이드 스크립트 기법은 인터프리터 언어의 한계로 인해 다른 컴파일 언어와 비교하여 실행 속도가 현저하게 느린 단점이 있다. 이러한 문제를 해결하기 위해 개발자에게는 서버 사이드 스크립트 언어의 장점을 유지하면서 컴파일 기법을 도입하여 처리 속도를 높이는 확장 서버 사이드 스크립트 기법이 제안되었다. 확장된 서버 사이드 스크립트 기법에는 JSP(JAVA Server Page)와 ASP.NET이 있으며 이들은 Perl, PHP, ASP 등 기존의 웹 프로그래밍 언어들이 가지고 있었던 처리 속도 문제를 개선하였다. [그림 2]는 ASP.NET의 동적 웹 페이지 처리 과정을 보여준다. [그림 2]에서처럼 확장된 서버 사이드 스크립트 기법은 초기에 한번 웹 페이지를 컴파일하여 실행화일을 캐쉬에 저장한 후, 이후의 요청은 캐쉬에 있는 실행화일을 통해 처리하기 때문에 컴파일 언어 수준의 빠른 속도를 제공할 수 있다. 이후의 웹 페이지 수정이 발생하면 수정이후 첫 요청에서 컴파일이 다시 수행되는데 이러한 과정이 개발자에게 투명하기 때문에 개발자에게는 스크립트 언어 수준의 개발용이성을 제공할 수 있다.



[그림 2] ASP.NET의 동적 웹 페이지 처리과정

3. 확장 서버 사이드 스크립트 기법의 성능 분석

본 절에서는 벤치마크 프로그램을 이용하여 확장 서버 사이드 스크립트 기법의 성능을 분석한다. 이를 위해 ASP.NET과 ASP를 각각 확장 서버 사이드 스크립트 기법과 서버 사이드 스크립트 기법으로 선택하고 동일한 기능을 수행하는 벤치마크 프로그램을 ASP.NET과 ASP 언어로 작성하여 작업처리 시간을 비교하였다.

```
void doWork(int count)
{
  int i, j;
  int sum;


  sum = 0;
  for(i=0;i<count;i++)
    for(j=0;j<1000000;j++)
      sum = sum + j;
}
```

[그림 3] 벤치마크 프로그램 구조

벤치마크 프로그램은 [그림 3]과 같이 입력된 count 값에 따라 1,000,000회 반복하여 덧셈을 수행하는 구조이다.

확장 서버 사이드 스크립트 기법의 성능 테스트는 일반적인 웹 시스템 구조인 웹 서버에서 직접 연산을 처리하는 2-계층구조와 웹 서버 외에 별도의 응용 서버에서 연산을 처리하는 3-계층구조로 분리되어 수행되었다.

[표 1] 2-계층구조 실험결과

비교항목	count=20 (초)	count=30 (초)
ASP	8	13
ASP.NET	0.03125	0.046875
ASP with COM	0.0143	0.0188

ASP program

```
<%
...
Response.write now & "<br>"
for i=0 to 20
  sum = 0
  for j=0 to 1000000
    sum = sum + j
  next
next
Response.write now & "<br>"
%>
```

ASP.NET with C# program

```
<script language="C#" runat="server">
int sum=0;
public void Page_Load( Object Src, EventArgs e )
{
 int i, j;
 ...
 for(i=0; i<20; i++)
 {
  sum = 0;
  for(j=0;j<1000000;j++)
    sum = sum + j;
 }
 ...
 Response.Write(r_time.ToString());
}
</script>
```

ASP with COM program

```
<%
...
Dim kissCOM
set kissCOM=Server.CreateObject("test1.testBigSum")
kissCom.testCalc3 20,rtime,rcnt,rsum,diff,clk1,clk2
set kissCOM = nothing
Response.Write rtime
%>
```

test1.testBigSum COM program

```
STDMETHODIMP CtestBigSum::testCalc3(...)
{
 ...
 for(i=0; i<count; i++)
 {
  sum = 0;
  for(j=0;j<1000000;j++)
    sum = sum + j;
 }
 ...
}
```

[그림 4] 2-계층구조 벤치마크 프로그램 (count=20)

3.1 2-계층구조 결과 분석

웹 브라우저로부터 요청된 연산이 직접 웹 서버에서 수행되는 2-계층구조의 성능 테스트를 위해 벤치마크 프로그램을 [그림 4]와 같이 각각 ASP와 ASP.NET(with C#) 그리고 COM 기반의 ASP로 개발하였다. 프로그램 테스트는 Windows 2003 Server의 IIS 환경에서 count 값을 각각 20과 30을 설정하여 평균 작업처리시간을 측정하는 방식으로 수행되었으며 프로그램의 수행결과는 [표 1]과 같다. [표 1]에 의하면 스크립트 언어만으로 개발된 ASP 프로그램과 컴파일된 실행코드를 사용하는 ASP.NET 및 COM 기반의 ASP 프로그램과의 처리시간이 현저하게 차이가 나는 것을 볼 수 있다. 컴파일 된 실행코드를 사용하는 ASP.NET과 COM 기반의 ASP 프로그램과의 실행시간을 비교해보면 COM 기반의 ASP 프로그램이 ASP.NET에 비해 빠르다는 것을 알 수 있는데 이는 실행속도가 가장 빠른 C 언어로 구현된 COM 모듈에서 대부분의 연산이 수행되기 때문에 COM 기반의 ASP가 우수한 것으로 분석된다. 즉 실험 결과에 의하면 COM 기반의 ASP를 적용하는 것이 실행속도의 관점에서 가장 유리할 수 있다. 그러나 COM 기반의 ASP의 경우 별도로 COM 모듈을 개발해야 하는 개발 및 유지보수 비용이 크다는 한계가 있다. 특히, 이러한 COM 모듈이 ASP 프로그램의 대부분을 차지한다면 Fast-CGI 기법과 큰 차이가 나지 않는다. 따라서 실행속도와 함께 프로그램 개발 및 유지보수 비용을 고려한다면 ASP.NET으로 개발된 확장 서버 사이드 스크립트 기법이 우수하다는 것을 알 수 있다.

3.2 3-계층구조 결과 분석

웹 서버 외에 별도의 응용 서버에서 연산을 처리하는 3-계층구조의 성능 테스트를 위해 [그림 3]의 벤치마크 기능을 별도로 수행하는 응용 서버 (App. Server)를 두고 응용 서버에 접속하여 연산결과를 가져오는 ASP.NET 프로그램과 응용 서버에 접속하여 연산결과를 가져오는 COM을 사용하는 ASP 프로그램의 작업처리시간을 비교하였다. 프로그램 테스트는 Windows 2003 Server의 IIS 환경에서 count 값을 각각 100과 1000, 5000, 7000을 설정하여 평균 작업처리시간을 측정하는 방식으로 수행되었으며 프로그램의 수행결과는 [표 2]와 같다. [표 2]에 의하면 ASP.NET 프로그램과 COM 기반의 ASP 프로그램과의 처리시간이 거의 차이가 나지 않는 것을 볼 수 있다. 이러한 현상은 ASP.NET 프로그램과 COM 기반의 ASP 프로그램이 실행 시간의 대부분을 응용 서버에 의존하기 때문이다. 이처럼 전체 작업처리 시간 중 중요한 부분이 다른 응용 서버의 작업처리 시간에 의존적일 때 COM 기반의 ASP 프로그램보다 개발비용 및 유지보수 비용에서 강점을 보이는 ASP.NET이 유리하다는 것을 알 수 있다.

[표 2] 3-계층구조 실험결과

비교항목	App. Server	ASP.NET	ASP with COM
count=100 (초)	0.0625	0.078125	0.078
count=1000 (초)	0.640625	0.657	0.65625
count=5000 (초)	3.21875	3.234375	3.235
count=7000 (초)	4.515625	4.563	4.531

3. 결론

본 논문에서는 기존의 동적 웹 페이지 기술을 분석하고 벤치마크 테스트를 통해 확장 서버 사이드 스크립트 기법의 성능을 분석하였다. 벤치마크 테스트 결과에 의하면 확장 서버 사이드 스크립트 기법이 성능과 프로그램 개발 및 유지보수 비용을 함께 고려할 때 2-계층구조와 3-계층구조에서 우수하다는 것을 알 수 있었다. 특히, 대부분의 웹 시스템이 데이터베이스에 의존하는 전형적인 3-계층구조를 따르고 있기 때문에 웹 환경에서 확장 서버 사이드 스크립트 기법은 다른 기법에 비해 우수한 효율성을 나타내리라 기대된다. 향후 연구과제로는 대표적인 확장 서버 사이드 스크립트 언어인 JSP와 ASP.NET 두 방법의 비교분석이 요구된다.

6. 참고문헌

[1] 김연진, "알기쉬운 ASP.NET," 정보문화사, 2002
[2] "HTTP Made Really Easy - A Practical Guide to Writing Clients and Servers," http://www.jmarshall.com/easy/http/
[3] MSDN, http://msdn.microsoft.com/library/kor/
[4] "Taeyo's ASP & .NET," http://www.taeyo.pe.kr/