Journal of Digital Contents Society Vol. 8 No. 1 Mar. 2007(pp. 41-50)

콘텐츠 적응화 시스템에 SOP(Shared Object Page)를 도입한 개선된 캐싱 기법

장서영*, 정호영**, 강수용***, 차재혁****,

요 약

오늘날 우리는 인터넷에 연결된 PC뿐만 아니라, 이동형 기기인 휴대전화, 무선 인터넷을 이용하는 PDA, 노트북, 그리고 D-TV 등을 통해서도 웹서버에 접근하여 콘텐츠 형태의 정보를 얻고 있다. 본 연구에서는 이러한 환경을 지원하기 위하여 각각의 적응화된 컨텐츠를 저장하는 것을 기반으로 하며, 이를 다양한 기기를 지닌 사용자에게 활용하기 위해 페이지 정보를 지니는 메타 데이터를 적용하는 웹 캐싱 기법(SOP)를 제안한다.

An Enhancing Caching Technique by the SOP(Shared Object Page) for Content Adaptation Systems

Seo-young Jang*

Abstract

People access web contain via PC and many other devices. In other words, not only they access information by a PC connected internet, but also they get information through a mobile phone, a PDA even D-TV. In this article, to resolve the problem, we suppose new web caching mechanism called 'SOP(Shared Object Page)' based on applying of meta data of web page information and storing adapted objects.

Keywords : Contents Adaptation, Content Adaptation System, Content Adaptation Manager

1. 서론

1990년대부터 WWW(World Wide Web)정보서비스는 인터넷 광역망에서 웹 서버에 제공해주는 콘텐츠(text, image, MPEG content 등)를 손쉽게 이용할 수 있게 함으로써 짧은 기간에 수많은 사용자 층을 확보하였다. 익히기 쉬운 웹 브라우저(Web Browser)의 개발 및 발전을 토대로 인터넷의 사용화는 이와 관련된 수많은 기업과 정보서비스 공급자 및 이를 이용하는 사용자의 증가에 따라 계속 가속화되어 왔으며, 오늘날 사용자들은 그들의 다양한 환경(PDA, D-TV, 휴대전화, 무선 네트워크 등)에서 콘텐츠를 요청하며, 원하는 정보를 보기를 원한다. 이러한 사용자들의 요구조건을 충족시켜주기 위해서 사용자가 사용하는 단말기 환경에 적합한 콘텐츠로 변환하여 프리젠테이션 할 수 있는 콘텐츠 적응화(content adaptation)와 이를 수행하는 콘텐츠 적응화 시스템(Content Adaptation System)은 필수적이다. 콘텐츠 적응화 시스템과 관련된 연구들은 콘텐츠의 재활용성을 중점으로 WWW와 관련된 단체를 비롯해 수많은 기업 및 학교에서 활발히 연구 중에 있다[1][2].

기존 적응화 시스템의 연구에서, 적응화 작업은 상대적으로 긴 시간을 요하는 과정이기 때문에 시스템의 성능 저하 및 사용자 기기에 대한 응답속도를 저하시키는 요인이 된다[15]. 이를 개선하는 방안으로 적응화를 요하는 콘텐츠를

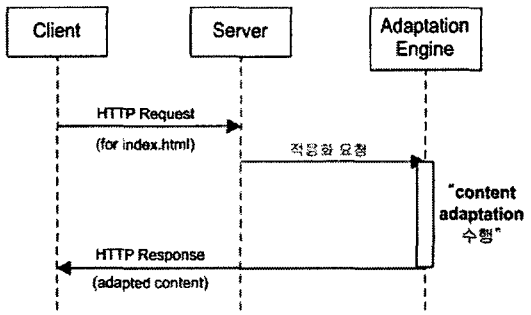
※ 제일저자(First Author) : 장서영
※ 교신저자(First Author) : 차재혁
접수일자:2007년01월11일, 심사완료:2007년02월14일
* 한양대학교 정보통신대학
jang402@hanyang.ac.kr
** 한양대학교 정보통신대학
*** 한양대학교 정보통신대학
**** 한양대학교 정보통신대학
chajh@hanyang.ac.kr

캐시에 저장하였다가 사용자의 재요청이 있을 경우, 적응화 과정을 거치지 않고 저장된 콘텐츠를 즉시 보내주는 웹 캐싱 기법이 필요하다. 따라서 본 연구에서는 적응화 과정의 횟수를 줄여 사용자에게 빠른 응답을 제공하는 새로운 웹 캐싱 기법을 제안한다.

2. 콘텐츠 적응화(Content adaptation)

2.1 콘텐츠의 전송

웹상에서 콘텐츠 전송의 흐름은 다음과 같다.



(그림 1) 일반적인 콘텐츠 전송 형태 1[8]

다양한 기기를 지닌 사용자들이 웹 서버에 콘텐츠를 요청하면, 콘텐츠 적응화 시스템에서 사용자 기기에 맞게 적응화 과정을 수행한 후, 각각의 사용자에게 적합하게 변환된 콘텐츠를 보내준다. 사용자의 콘텐츠 요청에 대해 적응화 과정을 수행할 때, 해당하는 페이지에 포함되어 있는 개별적인 URL들과 이 URL에 해당하는 실제 데이터(Text, Image, MPEG Content 등)를 적합한 형태로 변환시키는 것이다. 이 때, 적응화의 대상은 오브젝트(Object)가 된다[9].

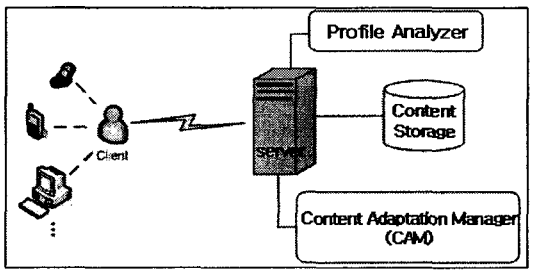
2.2 콘텐츠의 유형

콘텐츠는 다음과 같이 크게 정적인 요소와 동적인 요소 두 가지로 나눌 수 있다.

- 정적인 콘텐츠
  - 정적 콘텐츠 요소인 HTML, XML, GIF, JPEG 등으로 구성되며, 시간이나 사용자에 따라 변하지 않는 콘텐츠이다.
  - 각각의 기기에 맞게 변환된 모든 콘텐츠를 저장하고 있어야 하므로 저장소의 많은 낭비를 초래하는 단점이 있다.
- 동적인 콘텐츠
  - 주식시세, 일기예보, 교통정보 등을 나타내는 콘텐츠로 공간이나 시간에 따라 변하는 콘텐츠이다.
  - 이는 동적으로 변환을 수행해야 하므로 저장소의 낭비는 없지만, 수행하는데 많은 시간을 소요하므로 사용자에게 신속한 서비스를 해줄 수 없는 단점을 지닌다.

2.3 콘텐츠 적응화 시스템(Content Adaptation System)

콘텐츠 적응화 시스템이란 기존의 웹 서버에 존재하는 콘텐츠에 대해 적응화 작업을 수행하여, 각 사용자의 기기에 적합한 콘텐츠를 제공해 주는 시스템이다.



(그림 2) 콘텐츠 적응화 시스템의 전반적인 구조

(그림 2)는 콘텐츠 적응화를 수행하는 시스템의 구조를 간략하게 나타내고 있다[13]. 이는 사용자의 응답을 처리하는 웹 서버, 콘텐츠가 저장되어 있는 콘텐츠 저장소, 적응화를 수행하는 콘텐츠 적응화 관리기(Content Adaptation Manager), 사용자 기기정보를 수집하고 관리하는 플랫폼 프로파일 분석기(Profile Analyzer)로 이루어져 있다.

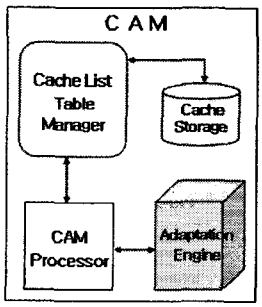
웹 문서에 대한 사용자의 요청이 왔을 때, 콘텐츠 적응화 시스템에서는 플랫폼 프로파일 분석기를 통해 사용자 기기의 정보를 수집한다. 콘텐츠 적응화 관리기는 콘텐츠 저장소의 콘텐츠

를 해당 사용자의 기기에 적합한 콘텐츠로 변환하여 웹 서버를 통해 사용자에게 전달된다.

2.3.1 콘텐츠 적응화 관리기

콘텐츠 적응화 관리기는 콘텐츠 적응화를 수행하는 적응화 엔진(Adaptation Engine)을 중심으로 적응화된 오브젝트를 저장하고 있는 캐시 저장소, 저장소의 검색을 효율적으로 도와주는 캐시 리스트 테이블 관리기(Cache List Table Manager) 그리고, 이들 간의 상호작용을 도와주는 CAM 프로세서로 구성되어, PC기반으로 작성된 콘텐츠를 다양한 기기에 적합하도록 콘텐츠를 변환하는 과정을 담당한다.

본 연구의 콘텐츠 적응화 관리기(CAM)의 구성은 아래의 그림과 같다[13].



(그림 3) 콘텐츠 적응화 관리기 (CAM)

CAM에서의 핵심부분인 적응화 엔진에서는 다음과 같이 두 가지 과정을 수행한다.

- 적응화 규칙 생성(adapted rule generation)[14]
  - 사용자 기기에 적합한 형태로 변환하여 주기 위한 룰을 CC/PP(Composite Capabilities/Preference Profiles)[3]에 근거하여 생성한다.
- 콘텐츠 적응화 과정 수행
  - 생성된 적응화 규칙에 의해 적응화 과정을 수행한다.

2.3.2 웹 캐싱 기법의 필요성

오늘날의 플랫폼 프로파일은 운영체제나 사용하는 장비에 따라 달라진다. 또한 같은 기기라도 모바일 컴퓨터나 PDA 등은 사용자가 쉽게 기기의 특성을 변경할 수 있으므로, 여러 가지 다른 프로파일이 존재할 수 있다. 콘텐츠 적응화 시스템의 적응화 작업은 이 플랫폼 프로파일을 분석하여 수행되므로 같은 콘텐츠라도 플랫폼 프로파일에 따라 적응화된 콘텐츠가 많이 생성된다. 이를 모두 저장하는 것은 낭비가 심하므로 콘텐츠 적응화 시스템에서는 동적으로 변환을 수행할 수 있어야 한다. 그러나 적응화 작업은 긴 수행시간으로 인해 사용자에게 신속한 응답을 해줄 수 없기 때문에 이를 효율적으로 제공하도록 유도하는 웹 캐싱 기법이 필요하다. 그래서 콘텐츠 적응화 관리기에서는 (그림 3)과 같이 웹 캐싱 기법을 적용하고 있다.

2.4 웹 캐싱 기법의 요구사항

콘텐츠 적응화 시스템과 더불어 효율적으로 제공해 주기 위하여, 본 연구에서 고찰한 웹 캐싱 기법의 요구사항은 다음과 같다.

- 다양한 사용 환경을 효과적으로 지원해야 한다.
- 적응화된 콘텐츠들은 URL마다 따로 존재해야 한다.
- 캐시는 적응화 엔진에 근접한 위치에 존재해야 한다.
- 캐시는 웹 서버에 존재하는 콘텐츠가 동적으로 변하는 것을 인지해야 하며, 동시에 적응화 메커니즘의 변화도 동적으로 파악해야 한다.

결과적으로, 사용자에게 신속한 서비스를 제공해줄 수 있어야한다.

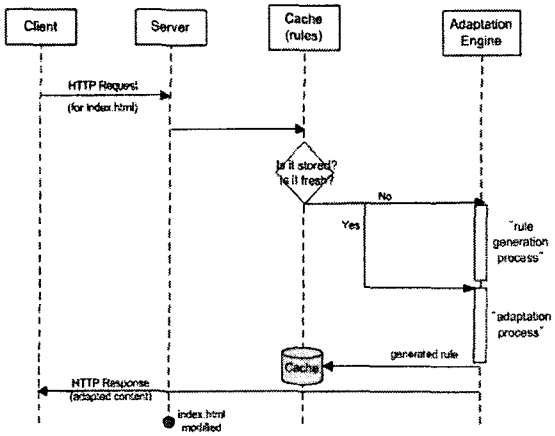
2.5 기존 기법

2.5.1 적응화 규칙을 저장하는 기법

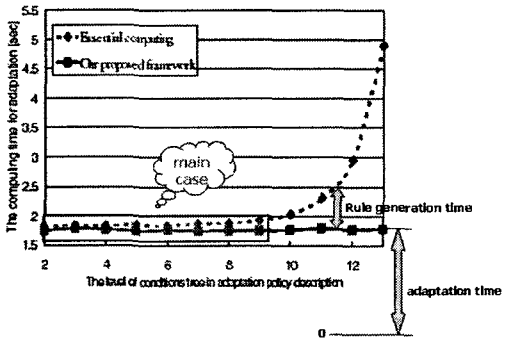
NTT DoCoMo에서 제안한 기법[15]으로, 캐시 저장소에서는 적응화 엔진의 첫 번째 단계(적응화 규칙 생성단계)에서 생성된 적응화 규칙(adapted rules)을 저장하고 있는 기법이다. 이는 규칙 문서의 크기가 오브젝트를 저장하는 것보다 대략 3배가량 작기 때문에 저장 공간을 효율적으로 사용할 수 있다는 장점이 있다.

사용자로부터 콘텐츠에 대한 요청이 왔을 때, 우선 캐시에 이에 해당하는 적응화 규칙이 존재

하는지를 검색하여 존재한다면, 적응화 엔진에서 적응화 규칙을 적용하여 적응화 과정을 수행하여, 바로 사용자에게 응답해 준다. 그렇지 않고 검색에 실패한 경우에는 적응화 엔진에서 적응화 규칙을 생성하는 과정을 거친다. 그리고 적응화 규칙을 적용하여 적응화 과정을 수행하고 사용자에게 적응화된 콘텐츠를 보내준다. 적응화 규칙을 저장함으로써, 적응화 규칙을 생성하는 과정을 생략하기 때문에 사용자에게 신속한 서비스를 제공할 수 있다.



(그림 4) 콘텐츠 전송 형태 2

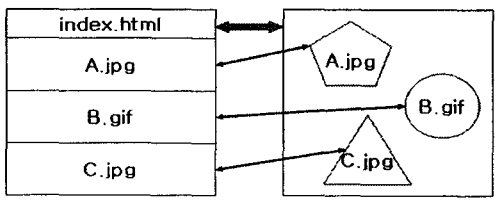


(그림 5) 복잡도에 따른 적응화 규칙 생성시간 및 적응화 수행시간[15]

(그림 5)는 콘텐츠의 복잡도에 따라 적응화 규칙을 생성하는데 걸리는 수행시간과, 콘텐츠 적응화를 수행하는데 걸리는 시간을 보여주고 있다. 이를 보면 적응화 과정을 수행하는데 걸리는 시간은 콘텐츠의 복잡도와 상관없이 일정한 수행시간(≒1.7sec)이 걸린다. 이와 달리 적응화 규칙을 생성하는데 걸리는 수행시간은 콘텐츠의 복잡도가 높음에 따라 적응화 규칙의 복잡도 또한 높아지기 때문에, 콘텐츠의 복잡도가 높을수록 증가하는 모습을 보인다. 이렇듯 이 기법은 콘텐츠의 복잡도가 높은 콘텐츠인 경우에 효율적임을 알 수 있다. 하지만, 일반적인 콘텐츠의 경우에는 콘텐츠의 복잡도가 낮기 때문에 적응화 규칙을 생성하는 시간보다 적응화하는데 걸리는 시간을 줄이는 기법이 필요하다.

2.5.2 일반적인 웹 캐싱 기법을 적용한 경우

대부분의 웹 캐싱 정책들은 (그림 6)과 같이 URL과 오브젝트간에 1:1로 대응하는 것을 기반으로 하고 있다. 서버에서 최근에 보내준 정보(콘텐츠)를 저장하고 있다가 요청시 바로 전송하므로, 동일한 사용자에 대한 오브젝트만 고려되거나, 가장 최근에 접근한 기기에 대한 오브젝트만을 가지고 있기에 대부분의 경우에 적응화 과정이 요구되어야만 한다. 또한 사용자가 일반적인 콘텐츠를 요청한 경우에는 전체 수행시간의 대부분을 차지하는 콘텐츠 적응화 과정을 항상 수행해야 하므로 효율적이지 못하다[16]. 또한, 기기에 따른 오브젝트들이 캐시에서 프로파일별로 관리가 된다 하더라도, 이는 같이 보여줘도 되는 오브젝트(즉, 적응화되어도 똑같은 형태를 가지는)들에 대한 고려를 하고 있지 않아서 오브젝트들을 중복하여 저장하는 현상을 초래한다.



(그림 6) URL과 오브젝트간의 1:1 대응하는 모습

이와 같은 문제점들을 가지고 있기에, 기존의 웹 캐싱 기법은 콘텐츠 적응화 시스템에 적합하지 못하다. 그러므로 콘텐츠 적응화 시스템과 더불어 사용자에게 효율적으로 변환된 콘텐츠를

제공해주기 위해 새로운 웹 캐싱 기법이 요구된다.

3. SOP(Shared Object Page) 캐싱 기법의 설계

3.1 캐시 구조

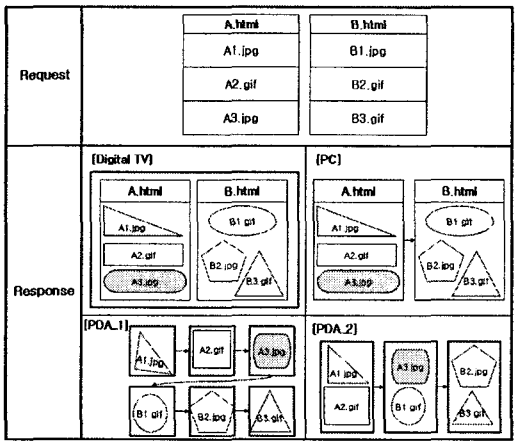
본 연구에서는 콘텐츠 적응화 시스템에 의해 생성된 각각의 적응화된 콘텐츠를 사용자에게 보다 빠르게 지원해주기 위한 웹 캐싱 기법을 제안한다.

캐시의 구조는 크게 두 가지로 구성된다. 하나는 사용자로부터 요청된 콘텐츠가 적응화 과정을 거쳐서 생성된 적응화된 오브젝트들을 저장하고 있는 구조이며, 다른 하나는 이 오브젝트들로 구성된 페이지의 정보를 지니는 XML로 작성된 메타 데이터들을 저장하고 있는 구조이다. 이는 요청된 콘텐츠에 대해 사용자 기기에 적합하게 적응화된 콘텐츠가 캐시에 존재하면 적응화 엔진을 호출하지 않고, 바로 전송해주는 형태이다.

3.1.1 적응화된 오브젝트의 저장

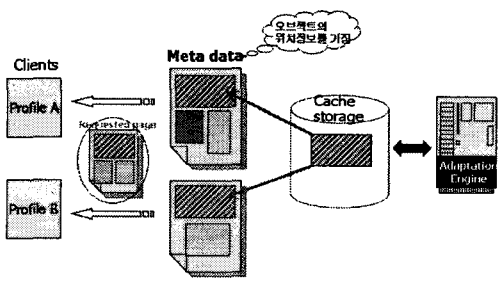
앞서 언급한 기존연구에 의하면, 적응화 엔진에서의 적응화 과정은 콘텐츠의 복잡도와 상관없이 전체 수행시간의 대부분을 차지하고 있기 때문에, 적응화 엔진을 호출하지 않고 캐시에 저장해 두었다가 바로 전송해 줄 수 있는 방법이 필요하다. 그러므로 본 연구에서는 적응화 엔진에서 최종적으로 생성된 적응화된 오브젝트를 저장하는 기본적인 구조를 제안한다.

적응화된 콘텐츠를 저장함에 있어서 중요하게 고려해야 할 사항은 (그림 7)과 같이 URL과 오브젝트간에 M:N 관계가 성립된다는 것이다. URL과 오브젝트간에 M:N 관계가 성립하게 되면, 적응화 엔진에 의해 적응화된 오브젝트 모두를 저장한다하더라도, 사용자 기기환경을 정확하게 알 수 없으면, 사용자에게 적합한 콘텐츠를 보낼 수가 없다. 또한, 콘텐츠 적응화 시스템은 기본적으로 기기의 정보가 다르면 각각에 적합한 적응화된 콘텐츠를 생성한다. 그러나 기기의 정보가 다르다 하더라도, 유사한 경우가 존재하며, 이 경우에는 동일한 형태의 적응화된 콘텐츠가 생성이 될 수 있다[16].



(그림 7) URL과 오브젝트간의 M:N 대응하는 모습

서로 다른 기기에 대한 적응화된 콘텐츠가 동일할 경우, 프로파일의 정보가 조금 다르다고 하여 반복적으로 적응화 과정을 요청한다면, 전체 시스템의 성능저하를 가져오게 되며, 동일한 오브젝트를 중복하여 저장하게 되므로 저장 공간의 낭비도 초래하게 된다. 이를 해결하기 위해서는 기기 정보가 조금 다르더라도 동일하게 사용되는 오브젝트들을 처리할 수 있는 새로운 방안이 모색되어야 한다.



(그림 8) 메타 데이터의 적용 및 활용

3.1.2 메타 데이터의 저장 및 활용

본 연구에서는 캐시에서 동일하게 사용되는 오브젝트를 중복 없이 저장하여 저장소를 효율적으로 관리하고, 매번 적응화 엔진이 호출되는 현상을 방지하여 시스템의 부하를 줄여서 사용

자에게 변환된 콘텐츠를 보다 효율적으로 전송해주기 위해 메타 데이터를 적용한다. (그림 8)은 저장된 오브젝트와 이 오브젝트를 기반으로 구성된 페이지에 대한 정보를 이용한 것으로 페이지 단위의 전송이 가능하다는 것을 보여준다.

(그림 8)을 보면, 프로파일 A와 프로파일 B를 지니는 사용자가 존재하고, 서버에 있는 페이지를 동시에 요청한 경우, 사용자는 그들이 지니고 있는 기기에 맞게 변환된 콘텐츠를 보기를 원한다. 여기에서 (그림 8)의 빗금 친 부분처럼 같이 보여줘도 되는 오브젝트의 경우에 대해, 본 연구에서는 캐시에 저장되어 있는 오브젝트들의 위치를 (그림 9)와 같이 표현하여 페이지에 대한 정보를 가져다주는 메타 데이터를 적용하였다.

```
<PAGE>
  <URL>db.hanyang.ac.kr/index.html</URL>
  <LOCATION>"₩"</LOCATION>
</PAGE>

<OBJECT_1>
  <TYPE>Image</TYPE>
  <URL>db.hanyang.ac.kr/image/A.jpg</URL>
  <LOCATION>"₩"</LOCATION>
</OBJECT_1>

 <OBJECT_1>
  <TYPE>Image</TYPE>
  <URL>db.hanyang.ac.kr/image/B.jpg</URL>
  <LOCATION>"₩"</LOCATION>
</OBJECT_1> ·····
```

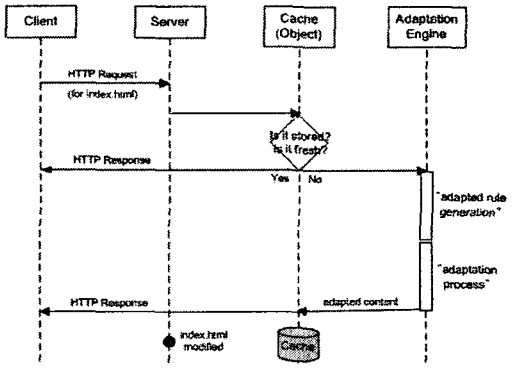
(그림 9) 메타 데이터의 사용 예

(그림 9)와 같은 XML로 생성된 메타 데이터를 이용하면 저장된 적응화된 오브젝트들을 다양한 기기를 지닌 사용자에게 효율적으로 전송할 수 있다. 메타 데이터를 이용하면, 같이 보여줘도 되는 콘텐츠에 대해 중복 저장되는 현상을 제거할 수 있으며, 적응화 엔진이 빈번하게 호출되는 현상도 막아 페이지 단위의 전달을 가능하게 하여 사용자에게 보다 신속한 서비스를 가능하게 할 수 있다. 그러므로 본 연구에서 제안하는 캐싱 기법을 SOP(Shared Object Page)라 지칭한다.

3.2 전체적인 콘텐츠의 흐름

SOP 캐싱 기법을 적용하여 사용자에게 콘텐츠를 전달하는 과정은 다음의 (그림 10)과 같다.

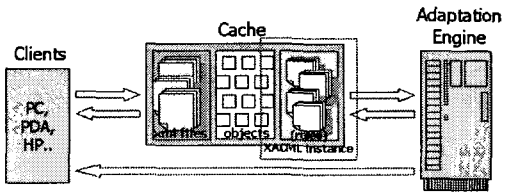
사용자가 웹 서버에 콘텐츠를 요청하면, 사용자 기기에 적합한 콘텐츠가 캐시에 존재하는지를 검색하여, 존재하면 바로 사용자에게 응답하여 준다. 만약 그렇지 않다면, 적응화 엔진을 호출하여 적응화 규칙을 생성하고, 이를 이용하여 적응화 과정을 수행한 후 적응화된 오브젝트들을 사용자에게 응답해줌과 동시에 캐시에 저장한다. 또한 XML을 이용하여 저장된 오브젝트들의 위치를 지니는 메타 데이터를 생성한 후 저장한다.



(그림 10) 콘텐츠 전송 형태 3

3.3 기존기법에 SOP를 도입한 개선된 캐싱 기법

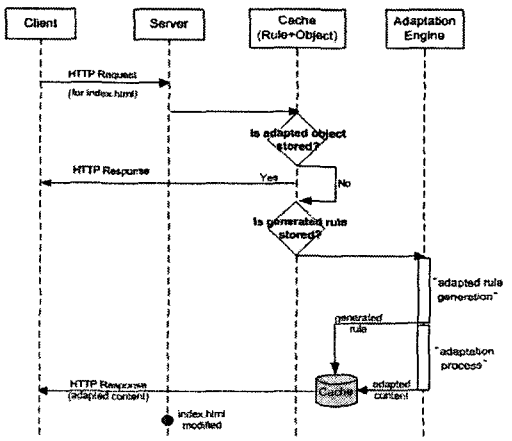
본 연구에서는 오브젝트를 저장하고 남는 여유 공간에 적응화 규칙을 활용하는 방안이 시스템의 성능을 더욱 개선시켜줄 것으로 판단하여, 관련연구에서 제안한 적응화 규칙을 저장하는 기법에 SOP를 적용하였다. 이를 Combined 캐싱 기법이라 지칭하며, 이 기법의 구조는 (그림 11)과 같다.



(그림 11) Combined 캐싱 기법을 적용한 캐시 구조

Combined 캐싱 기법을 적용한 캐시 구조에서는 콘텐츠에 대해 사용자의 요청이 들어오면, 우선 캐시에서 이에 해당하는 XML 파일(메타 데이터)의 존재여부를 검색하여 있는 경우, 적응화된 오브젝트들을 차례로 사용자에게 전송하게

되며, 그렇지 않을 경우 적응화 규칙이 존재하는지에 대해 검색한다. 이때, 해당하는 규칙이 존재하면 적응화 엔진에서는 이를 이용해 콘텐츠 적응화 과정만 수행한 후 사용자에게 보내주며, 그렇지 않은 경우에는 적응화 엔진에서 적응화 규칙의 생성과 이를 적용한 적응화 과정 모두를 수행한 후에 적응화된 콘텐츠를 보내주게 된다. 즉, Combined 캐싱 기법은 오브젝트의 검색에 실패하더라도 미리 저장된 적응화 규칙을 이용하여 성능 향상을 꾀할 수 있다.



(그림 12) 콘텐츠 전송 형태 4

결과적으로, Combined 캐싱 기법은 적응화 규칙을 생성하는데 오래 걸리는 특정 목적을 지닌 콘텐츠를 요청한 경우에 대해서도 효율적인 웹 서비스가 가능하며, 사용자에게 서비스 해주는데 걸리는 시간을 SOP 캐싱 기법보다도 더 단축시킬 수 있었다.

4. 성능평가

4.1 평가 방법

본 연구의 성능 측정은 다음과 같이 사용자 측과 웹 서버 측을 가상으로 설정하고 테스트하였다.

<사용자 측>

- 사용자는 그들이 지니는 각각의 기기를 가지고 콘텐츠를 요청한다.
- 사용자들의 지니는 기기의 종류는 총 20가지로 제한한다.
- 기기에 따른 사용자의 전체 요청회수는 10,000번으로 정한다.
- 일반적인 콘텐츠의 특성상 인기도에 따라 요청되는 회수가 다르므로, 인기도를 반영할 수 있는 Zipf Distribution을 적용한다[17].

<웹 서버 측>

- 웹 서버에 존재하는 오브젝트의 개수는 1000개이다.
- 오브젝트들로 구성되어 있는 페이지의 개수는 50개이다.
- 하나의 페이지 내부에 존재하는 오브젝트의 개수는 한 개 이상 랜덤하게 포함되어 있다.

요청된 콘텐츠를 사용자에게 응답해주는데 걸리는 수행시간을 측정하기 위해서 다음과 같이 총 수행시간에 측정되는 변수를 정하고 테스트한다.

- ⓐ lookup time (캐시를 검색하는데 걸리는 시간)
- ⓑ service time (응답시간)
- ⓒ replacement time (캐시가 꽉 찬 경우 교체하는데 걸리는 시간)
- ⓓ rule generation time (적응화 규칙을 생성하는데 걸리는 시간)
- ⓔ transformation time (콘텐츠 적응화를 수행하는데 걸리는 시간)

<표 1> 총 수행시간 비교

kind of cache (hit/miss) / Variable	적응화 규칙 저장 기법		SOP 기법		Combined approach 기법		
	Hit	Miss	Hit	Miss	Hit	Miss Rule search (hit)	Miss (miss)
ⓐ lookup time	O	O	O	O	O	O	O
ⓑ service time	O	O	O	O	O	O	O
ⓒ replacement time		O		O			O
ⓓ rule generation time		O		O			O
ⓔ transformation time	O	O		O		O	O

<표 1>은 앞서 언급한 총 수행시간에 측정되는 변수들을 가지고 총 수행시간을 비교·분석한 것이며, 마지막으로 캐시의 교체정책으로 가장 대표적으로 사용하고 있는 LRU(Least Recently Used)를 적용하였다[18].

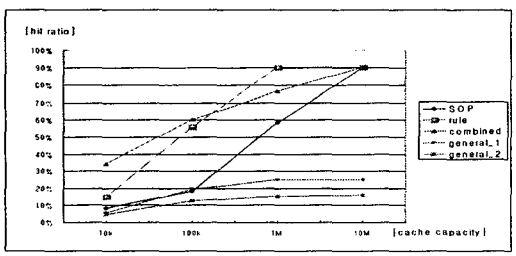
4.2 평가 대상

본 연구에서의 성능 측정은 다음과 같이 5가지 캐싱 기법을 적용하여 테스트하였다.

- 기존 관련연구에서 제안한 적응화 규칙 캐싱 기법*(rule)*
- 본 연구에서 제안하는 SOP 캐싱 기법 *(SOP)*
- 위의 두 가지를 혼합한 캐싱 기법 *(combined)*
- 추가로, 일반적으로 사용되고 있는 캐싱 기법
  - PC 용 콘텐츠만 저장 가능한 캐싱 기법 *(general_1)*
  - 최근에 접근한 기기에 대한 적응화된 오브젝트만을 저장하는 캐싱 기법*(general_2)*

4.3 평가 결과

일반적으로 웹 캐싱 기법의 성능 평가 척도로는 캐시 적중률(cache hit-ratio)을 사용한다. 그러므로 본 성능평가에서는 캐시 크기에 따른 캐시 적중률을 비교해 보며, 성능평가의 목적인 요청된 콘텐츠를 사용자에게 응답해주는데 걸리는 수행시간을 측정한다.



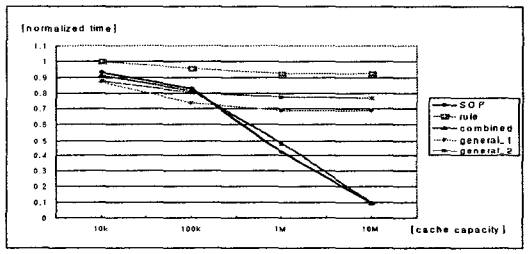
(그림 13) 캐시 크기에 따른 캐시 적중률 비교
(단, zipf parameter값이 0.5 인 경우)

4.3.1 캐시 적중률 비교

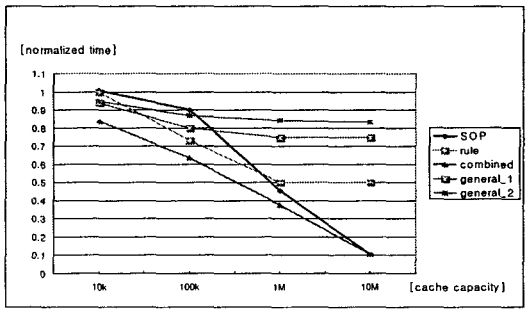
캐시 적중률을 비교한 결과를 나타내는 (그림 13)을 보면, 제한된 캐시 공간에 크기가 작은 적응화 규칙을 많이 저장할 수 있는 적응화 규칙 캐싱 기법이 본 SOP 캐싱 기법에 비해 좋게 나온다. 하지만, 적응화 규칙만 저장하는 경우에는 캐시에서 적중하였다 하더라도, 항상 적응화 엔진에서의 콘텐츠 적응화 과정을 수행해야 하므로 총 수행시간을 비교해 보면 (그림 14)와 (그림 15)에서 보여주듯이, 가장 안 좋은 성능을 보인다. 또한, 두 가지의 캐싱을 혼합한 Combined 캐싱 기법은 적응화 규칙을 함께 저장하고 있기 때문에 SOP 캐싱 기법에 비해 좋은 적중률을 보임을 알 수 있다.

4.3.2 총 수행시간 비교

수행시간은 'Normalized time'으로 적응화 규칙을 저장한 캐싱 기법에서 캐시 크기가 10K인 경우에 걸리는 총 수행시간을 '1'로 정해놓고 비교 측정하였다. 일반적인 콘텐츠의 경우와 특정 목적을 지니는 콘텐츠의 경우 각각에 대해 테스트해 보았다.



(그림 14) 캐시크기에 따른 총 수행시간 비교 1
(단, zipf parameter=0.5, 적응화 규칙을
생성하는 걸리는 시간:적응화를 수행하는데
걸리는 시간=1:9인 경우)



(그림 15) 캐시크기에 따른 총 수행시간 비교 2
(단, zipf parameter=0.5, 적응화 규칙을
생성하는 걸리는 시간:적응화를 수행하는데
걸리는 시간=6:4인 경우)

(관련연구)에서 언급하였듯이 일반적인 콘텐츠의 경우에는 적응화 규칙을 저장하는 것이 큰 의미가 없기 때문에, (그림 14)에서 알 수 있듯이 Combined 기법과 SOP 캐싱 기법이 수행속도가 빠르고, 유사하게 측정됨을 알 수 있으며, 적응화 규칙을 저장하는 캐싱 기법이 가장 안 좋게 나왔다.

또한, 콘텐츠의 복잡도가 높은 콘텐츠의 경우에는 적응화 규칙을 생성하는데 걸리는 수행시간이 오래 걸리기 때문에, (그림 15)와 같이 적응화 규칙을 조금이라도 저장하고 있는 Combined 기법이 SOP 캐싱 기법에 비해 좋게 측정되었으며, 캐시 크기가 클수록 적응화 규칙을 저장하고 있는 캐싱 기법이 여전히 수행시간이 오래 걸림을 사실을 알 수 있다.

5. 결 론

본 연구에서는 적응화 엔진의 부하를 줄여 사용자에게 보다 빠른 서비스를 제공해 주는 새로운 웹 캐싱 기법에 대해 제안하였다.

기존의 관련연구에서 제안한 적응화 규칙만 저장하는 캐싱 기법은 적응화 규칙의 크기가 오브젝트보다 1/3만큼 작기 때문에 작은 저장 공간을 효율적으로 사용할 수 있었고, 특히 콘텐츠의 복잡도가 높을수록 높은 캐시 적중률 및 전체 수행시간의 단축할 수 있다는 점에서 좋은 성능을 보였다. 하지만, 일반적인 콘텐츠의 대부분은 복잡도가 높지 않기 때문에 적응화 규칙을 저장한다는 것에는 큰 의미가 없으므로, 본 연구에서는 일반적인 콘텐츠에 대해 콘텐츠 적응화에 효율적인 오브젝트 기반의 웹 캐싱 기법(SOP)을 제안하였다. 나아가 콘텐츠의 특성에 상관없이 모든 콘텐츠를 고려한 기존 기법에 SOP를 도입한 통합기법을 제안하여, 총 수행 속도를 증가시켜 사용자의 요청에 보다 빠른 응답을 제공해 줄 수 있었다. 또한, 페이지 정보를 지니는 메타 데이터를 활용하여 적응화된 콘텐츠를 사용자에게 서비스가 가능하도록 하고, 캐시 저장소 내에 오브젝트들이 중복되어 저장되는 현상을 방지하여 저장 공간에 대한 높은 효율성 및 페이지 단위의 전송을 가능하게 하여 사용자에게 보다 빠른 서비스를 제공할 수 있었다.

참 고 문 헌

[1] TellaSonera, "Web Content Adaptation", http://www.medialab.sonera.fi"

[2] Tayeb Lemlouma and Nabil Layaida, "Content Adaptation and Generation Principles for Heterogeneous Clients"

[3] http://www.w3c.org/Mobile/CCPP/, 2005/11/8

[4] http://www.w3c.org/2001/di, 2005/11/8

[5] http://www.mpecg.org/MPEG/index.html, 2005/11/8

[6] http://www.i-cap.org/home.html, 2005/11/8

[7] http://www.ietf-opes.org/, 2005/11/8

[8] Markus Hofmann and Leland Beaumont, "Content Networking Architecture, Protocols, and Practice", Wirtschaftsinformatik, 2003

[9] Jinlin Chen, Baoyao Zhou, Jin Shi, Hongjiang Zhang and Qju Fengwu, "Function-Based Object Model Towards Website Adaptation, ACM, 2001

[10] Harini Bharadvaj, Anupam Joshi and Sansanee, "An Active Transcoding Proxy to Support Mobile Web Access", IEEE, 1998

[11] Goh, T., & Kinshuk, D., "Getting Ready For Mobile Learning", World Conference on Educational Multimedia, Hypermedia and Telecommunications (EDMEDIA), 2004

[12] Kinshuk and Tiong-Thye Goh, "Mobile Adaptation with Multiple Representation Approach as Educational Pedagogy", IEEE, 2002

[13] Seoyoung Jang, Byoungchol Chang, Sooyong Kang and Jaehyuk Cha, "An Object Based Web-Caching Mechanism for Content Adaptation", KACE, 2005

[14] Joost Geurts, "Constraints for Multimedia Presentation Generation", CWI, 22nd January 2002

[15] Akira KINNO, Hideki YUKITOMO and Takehiro NAKAYAMA, "An Efficient Caching Mechanism for XML Content Adaptation", IEEE, 2004

[16] Balachander Krishnamurthy Jeffrey C. Mogul David M. kristo, "Key Differences between HTTP/1.0 and HTTP/1.1", Computer Networks: The International Journal of Computer and Telecommunications Networking archive, 1999

[17] http://mathworld.wolfram.com/ZipfDistribution.html, 2005/11/8

[18] Brian D. Davison, "A Web Caching Primer", IEEE,

2001
[19] Sven Buchholz and Alexander Schill, "Adaptation-Aware Web Caching: Caching in the Futhre Pervasive Web", GI/ITG Conference Kommunikation in verteilten Systemen (KiVS), 2003
[20] Bharadvaj, H., Joshi, A., and Auephanwiriyakul, S., "An Active Transcoding Proxy to Support Mobile Web Access", IEEE, 1998
[21] Benyon, D., Stone, D., and Woodroffe, M., "Experience with developing multimedia courseware for the World Wide Web: the need for better tools and clear pedagogy", International Journal of Human-Computer Studies, 1997
[22] Masahiro Hori, Goh Kondoh, Kouichi Ono, Shin-ichi Hirose and Sandeep Singhal, "Annotation-Based Web Content Transcoding", Computer Networks, 2000
[23] Katashi Nagao, Yoshinari Shirai and Kevin Squire, "Semantic Annotation and Transcoding: Making Web Content More Accessible", IEEE, 2001
[24] Ansgar Scherp and Susanne Boll, "Framework support for dynamic personalized multimedia content on mobile systems", Wirtschaftsinformatik, 2004
[25] Olivier Steriger, Touradj Ebrahimi and David Marimon Sanjuan, "MPEG-BASED PERSONALIZED CONTENT DELIVERY", ICIP, 2003
[26] Li Fan, Pci Cao, Wei Lin and Quinn Jacobson, "Web Prefetching Between Low-Bandwidth Clients and Proxies: Potential and Performance", SIGMETRICS, 1999
[27] Dan Duchamp, "PREFETCHING HYPERLINKS", USENIX Association, 1999
[28] B. Krishnamurthy and J. Rexford, "Web Protocols and Caching"
[29] Greg Barish and Katia Obraczka, "World Wide Web Caching: Trends and Techniques", IEEE, 2002

장 서 영

2002년 : 홍익대학교 정보통신학부 (이학사)
2006년 : 한양대학교 정보통신대학 (공학석사)
현 재: (주)메타빌드
관심분야 : OSMU, Database

정 호 영

2004년 : 한양대학교 재료공학과 (학사)
2006년 : 한양대학교 정보통신학 (석사)
2007년 - 현재 : 한양대학교 정보통신학 (박사)

강 수 용

1996년 서울대학교 수학과(학사)
1998년 서울대학교 전산과학과 (석사)
2002년 서울대학교 컴퓨터공학과 (박사)
2000년 - 2002년 : SIGn Co., 선임연구원
2002년 - 2003년 : 서울대학교 Postdoctoral researcher
2003년 - 현 재 : 한양대학교 컴퓨터교육과 조교수

차 재 혁

1987년 : 서울대학교 계산통계학과(학사)
1991년 : 서울대학교 컴퓨터공학과(석사)
1997년 : 서울대학교 컴퓨터공학과(박사)
1997년 - 1998년 : 한국학술진흥재단 부설 첨단학술정보센터선임연구원
1998년 - 2001년 : 한양대학교 사범대학 컴퓨터교육과 조교수
2001년 - 현재 : 한양대학교 정보통신대학 정보통신학부 교수
관심분야 : XML, 데이터베이스, 플래시 메모리 기반 저장시스템, 멀티미디어 콘텐츠 적응화, e-Learning