발송 환경에서 데이터 방송을 위한 오브젝트 캐릭터 캐싱

및 효율적인 캐싱 기법 제안

김세창† 원재훈 고상원 정재민 김경선

한양대학교 컴퓨터공학과

vbcchang@naver.com, jhwon@cse.hanyang.ac.kr, funkcode@gmail.com, jkijim@naver.com

jekim@cse.hanyang.ac.kr

Object Carousel Manager And Caching Strategy For Efficiency System

In Data Broadcasting System

KIM, SECHANG†, WON, JAEHOON, KO, SANGWON, JEON, JEMIN, Jungsun Kim

HPC&OT Lab., Dept. of Computer Science, Hanyang University

요 약

본 논문에서는 데이터 방송 환경에서의 DSM-CC 관련 Object Carousel Manager를 구현하고, 성능향상

을 위한 캐싱 기법을 제안한다. Object Carousel Manager 구현은 다음과 같은 순서로 이루어진다. 먼저,

방송스트림에서 해당 PID를로 얻어온 모듈을 키시에 저장한 후, 모듈에서 BIOP 메시지 오브젝트별로 분

석한 바 페어 한다. 그 후, BIOP 페어의 BIOP 메시지 오브젝트를 해당 서비스 게이트웨이, 디렉터리, 파일로

변환해준다. 마지막으로, 변경된 파일시스템에서 해당하는 Xlet Application을 실행시켜 주게 된다. 시스템

상태를 구현하여 저장하는 캐싱 기법은 다음과 같다. Memory Caching Priority Descriptor를 이용하여,

Object들을 포함하고 있는 각각의 모듈에 대해서 Version Number를 Checking하는 주기적인

Transparency_level을 부여하고, Application 실행 시 보여 주는 해당 순서대로 필요한 모듈에 대해 우선순위

위로 높여주어, 캐시에 업데이트 시 이를 반영하여, 시스템 전체의 성능을 향상시킨다.

키워드 : Object Carousel Manager, Data Broadcasting, Caching Strategy, Caching Priority Descriptor

그리고, 이에 막힌 객체를 스트림 파일에서 얻기 위해

서는 스트림 파일의 Carousel 주기를 기다려 주는 파일을 검색하게 되므로 그만큼의 시스템 Delay가 생

기게 된다. 그로 세 순서기에는 이 객체 파일들을 원하는 시점에 사용할 수 있도록 미리 저장해두고 사용

할 수 있는 캐시가 필요하게 된다.

고비용의 서비스의 캐시를 사용하기 위해서는 시스템 성능을 최대화 시켜줄 수 있는 약간은 캐시 전략이 필요

하다. 그러므로 데이터를 빠르게 캐시에 고정시키기 위해서는 العم도 방식의 시간, 오브젝트 관리

하고 관리하기 위한 Object Carousel Manager 및 Caching Strategy를 제안하고 구현한다. Object Carousel Manager 시스템에서는

방송 스토리의 시작 또는 끝의 Module을 저장하고, 여기에

서 업데이트되는 BIOP 메시지 객체를 모두 저장하게 된다.

Carousel 구조의 방송 스토리 특성상 발생하게 되는 시

1. 서 론

방송 시스템에서, 데이터 방송이란 고대역의 서비스를 이용하는 디지털방송에 부가적으로 Xlet Application Class를 개발하여 방송하는 시스템을 지칭한다. 방송 환경에서는 수신기의 저작권해와 송신과정에서의 오류를 고려하여, Carousel이라는 구조로 방송하는 방식으로 페어링과 페어링을 반복하게 된다. 

즉, 데이터 방송 환경에서는 DSM-CC의 Object Carousel구조는 구조를 이용하여, 서비스에 Xlet Application과 관련된 대역 스토리와 파일 시스템 구조를 객체 형태의 메시지로 만들어서 보여주게 된다.carousel 구조에서는 Object Carousel 구조에서 원하는 Xlet Application과 파일을 실행하기 위해서는 Object Carousel을 관리하기 위한 Manager가 필요하다.
스템 Delay와 Load를 줄이기 위한 DSM-CC 관련 Object를 개선 하고, 모니터링 하는 방법을 제안한다.

본 논문은 다음과 같이 구성된다. 2장에서는 관련 연구를 소개한다. 3장에서는 Object Carousel Manager를 이용하여 원하는 Module, BIOP 메시지를 추출하고, 이로부터 Xlet Application 프로그램을 생성하는 방법을 설명한다. 4장에서는 얻어진 Module을 효율적으로 캐싱하는 기법과 모니터링 방법을 설명한다. 5장에서는 결론 및 향후 파학을 기술한다.

2. 관련 연구

2.1 방송환경의 스크립트를 수신하는 수신기에서의 MHP 미들웨어 구조

그림 1. MHP 미들웨어 Architecture

그림1에서 보는 것과 같이 방송환경의 스크립트를 전송받는 수신기에서의 MHP 미들웨어 구조는 시스템 소프트웨어의 어플리케이션 사이에 위치하여 해당 어플리케이션 프로그램이 제대로 동작할수 있도록 시스템 소프트웨어를 통해 리소스를 사용하게 해준다. 즉, 이런 미들웨어 구조 안에서 방송환경과 관련된 시스템들이 상호작용하게 되는 것이다.[5]

2.2 방송 파일 시스템의 시스템 구조

그림 2. 방송 파일 시스템의 시스템 구조

그림2에서 보는 것과 같이 수신기에서의 소프트웨어 구조는 다음과 같다.[6] 3가지 층으로 구성되어 System Layer는 해당 Application을 동작하도록 JVM위에 DSM-CC 관련 미들웨어 API를 제공해 주고 있다. 즉, DSM-CC와 관련된 방송 파일 시스템의 구조는 보는 바와 같이 3가지 층으로 구성하여 동작하게 된다.

2.3 Object Carousel

방송 환경에서 해당 Xlet Application과 관련 파일들을 파일 시스템 구조로 방송 스트림 안에 보내기 위해서는 BIOP 객체 메시지 형태로 담아서 보내주게 된다.[7] 즉, 파일 시스템의 루트에 해당하는 BIOP::ServiceGateway Message, 디렉터리인 BIOP::DirectoryMessage, 해당 파일을 담아서 보내게 되는 BIOP::FileMessage형으로 구성된다. 이렇게 구성된 BIOP 메시지 객체들을 메모리의 ModuleID를 가지는 Module에 여러 BIOP 메시지를 담아서 반복해서 Carousel구조로 방송 스트림 파일에 담여서 보내주게 된다. 수신부에서는 이렇게 전송된 BIOP Module을 BIOP 메시지 Syntax에 따라 분석하고, 해당 파일 시스템을 구성하여, 원하는 Xlet Application 프로그램을 실행할 수 있게 된다.

그리고, 모듈을 구성하기 위해서 필요한 DSI, DII 메시지 안에 Object Carousel을 위해서 루트 디렉터리 정보를 나타내기 위한 ServiceGatewayInfo() syntax와 모듈 정보를 나타내는 BIOP::ModuleInfoMessage syntax를 추가적으로 전송하게 된다.[7][8]

방송 환경에서는 이런 식으로 해당 객체들을 메시지 형태로 보낸 후, 모듈에 담아서 수신기 전송 오류나 저장 공간의 제약사항을 고려하며 반복해서 보내주게 되는 것이다.

2.4 DSM-CC 관련 Object 캐싱


2.4.1 Transparent Caching

DII 메시지를 처음 캐시에 저장한 후, 최소한 500ms동안에는 저장된 데이터를 신뢰할 수 있는 데이터로 간주한다. 500ms시간 후에 캐시에 저장된 데이터를 다시 전송받은 후, DII 메시지를 업데이트 하게 된다.

Transparent Caching기법과 관련하여, 모니터링 형태의 수행방법에 따라 다음과 같은 두 가지 방법이 존재한다.

2.4.1.1 Active Caching

463
DII 메시지에 대해 전용의 Section Filter를 사용하여, 계속해서 모니터링 하는 방법이다. 캐싱된 내용물에 대해 바로 유 효성을 알게 하며, 내용이 변화 되었을 때, 바로 모니터의 업데이트를 알려준다. 하지만, DII 메시지가 매우 높은 비율로 반복 되는 경우에는 DII 메시지를 받은 후, 500ms동안에는 Section Filter사용을 멈출 수 있게 된다. 이런 경우, 전체 시스템 성능 면에서 시스템 Load가 줄어들게 된다.

2.4.1.2 Passive Caching
전용의 Section Filter를 사용하는 경우에는 캐싱 되어 있을 수 있는 내용들의 양이 제한되며 전체 시스템의 load가 증가할 수 있다. 그래서 데이터가 필요해서 요청이 되는 경우, 그 시점에 DII 메시지 데이터의 모니터링을 시작하여 개시 안에 저장된 데이터의 유 효성을 체크하는 방법이다. 이 방법은 사용하였을 경우에는 원하는 데이터를 요청했을 때, Delay가 발생할 수 있다.

2.4.2 Semi-transparent Caching
마찬가지로 Semi-transparent Caching 방법은 최소 30s 동안에는 캐시에 저장된 DII 메시지를 유의한 데이터로 간주한다. 즉, 이 기간 후에 데이터가 재전송 되지 않으면, 캐싱된 DII 메시지의 상태를 신뢰할 수 없게 된다.

2.4.3 Static Caching
앞의 두 방식과는 달리, 처음 캐시에 저장된 DII 메시지를 해당 Application의 Lifetime 동안 내내 사용하게 된다. 처음 캐시에 저장된 후에는 DII 메시지의 Version Number를 체크할 필요가 없게 된다.

고비용의 저장량 캐시의 사용과 시스템 전체의 성능의 극대화를 위해서 이런 캐싱방법을 다각적으로 고려하여 시스템의 설계가 되어져야 하는 것이다.

3. Module과 BIOP 메시지를 사용하여 Object Carousel Manager 설계 및 구현

방송 시스템 환경의 스트림에서 원하는 Xlet Application Class를 실행하기 위해서는 Object Carousel Manger가 필요하게 되는데, 다음과 같이 Object Carousel Manger시스템을 설계하고 구현하였다.

3.1 시스템 전체 아키텍처
시스템 전체 환경을 그림 3에서 나타내었다. 먼저 전송되어오는 방송 스트림(Transport Stream)에서 DSM-CC Section을 통해 원하는 DSI, DII, DDB Message를 검색하게 된다. 이때, Table Id와 Table Id

그림 3. Object Manager 전체 시스템 환경도

Extension을 통해서 각 메시지를 구분해준다. 검색된 DSI, DII, DDB 메시지를 통해 Module안을 생성해낸다. 생성된 Module안에는 BIOP::ServiceGatewayMessage, BIOP::DirectoryMessage, BIOP::FileMessage등이 존재한다. Module안에서 BIOP 메시지 시작을 알리는 Magic 넘버(4byte) 0x42494F50을 통해 BIOP 메시지 시작점을 찾게 된다. BIOP 메시지 공통 syntax인 ObjectKind_data 필드를 통해 BIOP 메시지를 구분한다. ObjectKind_data값에 해당하는 BIOP 메시지를 구분한 값을 표1에 제시하였다.[7]

표1. BIOP Message의 ObjectKind_data

<table>
<thead>
<tr>
<th>BIOP Message type</th>
<th>ObjectKind_data</th>
<th>Alias_type_id</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOP::ServiceGateway</td>
<td>0x737E5700</td>
<td>&quot;src&quot;</td>
</tr>
<tr>
<td>BIOP::Directory</td>
<td>0x64697200</td>
<td>&quot;dir&quot;</td>
</tr>
<tr>
<td>BIOP::File</td>
<td>0x66696C00</td>
<td>&quot;fil&quot;</td>
</tr>
</tbody>
</table>

생성된 BIOP::ServiceGateway 메시지를 가지고 CarouselId에 해당하는 Root 디렉터리를 생성한다. Root 디렉터리의 하부 구조는 BIOP::Directory 메시지를 통해서 하위 디렉터리를 구성한다. 실제 데이터 파일은 BIOP::File 메시지의 Content_data_byte을 통해서 실제 파일 내용을 얻어 와서 해당 디렉터리 공간 안에 만든다. 디렉터리안의 Xlet Application 프로그램을 Application Manger 를 통해서 실행을 하게 된다.

3.2 알고리즘
Object Carousel Manager 시스템의 알고리즘을 간략하게 서술하면 다음과 같다.
procedure insert(aModule)
//반들어진 모듈을 Object Carousel Manager에게 넘겨서 관리는 함수
Object Carousel 파일 시스템을 만들어주는 함수
//aModule: 방송 스트림에서 얻어온 모듈 데이터
BIOP::ServiceGateWay, BIOP::Directory, BIOP::File 메시지를 모두 Manager 시스템에 저장하여, 각 메시지가 필요한 경우 바로 접근하여 사용할 수 있는데 있다. 그래서 방송 시스템 스트림 파일의 Carousel 구조를 통해 나타날 수 있는 시스템 Delay를 줄일 수 있는 장점이 있다.

storeBIOPMessage(biopmessage);
//반들어진 BIOP 메시지를 저장하는 함수
makeObjectCarouselFilesystem(biopmessage);

end of insert

procedure makeBIOPMessage(aModule)
for (i=0; i<aModuleSize; i++)
if (objectKindData == 0x73726770) //BIOP::ServiceGateWay Message
    parsingBIOPServiceGateWay(aModule);
end if

if (objectKindData == 0x64697200) //BIOP::Directory Message
    parsingBIOPDirectory(aModule);
end if

if (objectKindData == 0x666966C00) //BIOP::File Message
    parsingBIOPFile(aModule);
end if

end for

end of makeBIOPMessage

procedure makeObjectCarouselFileSystem(biopmessage)
//biopmessage: 저장된 BIOP 메시지 데이터
if (biopmessage == ServiceGateway)
    makeRootDirectory();
end if

if (biopmessage == Directory)
    makeDirectory();
end if

if (biopmessage == File)
    makeFile();
end if

end of

4. Module을 효율적으로 캐싱하고 모니터링 하는 방법


표 2. Caching Priority Descriptor Syntax

<table>
<thead>
<tr>
<th>Syntax</th>
<th>bits</th>
<th>Type</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>cachingPriorityDescriptor()</td>
<td></td>
<td>uimsbf</td>
<td>0x71</td>
</tr>
<tr>
<td>descriptor_tag</td>
<td>8</td>
<td>uimsbf</td>
<td></td>
</tr>
<tr>
<td>descriptor_length</td>
<td>8</td>
<td>uimsbf</td>
<td></td>
</tr>
<tr>
<td>priority_value</td>
<td>8</td>
<td>uimsbf</td>
<td></td>
</tr>
<tr>
<td>transparency_level</td>
<td>16</td>
<td>uimsbf</td>
<td></td>
</tr>
</tbody>
</table>

5. 결론 및 향후연구

본 논문에서 방송 환경의 데이터 방송에 대해서 해당 Xlet Application 프로그램을 분석하고 관리하며 실행할 수 있도록 만들어주는 Object Carousel Manager 시스템을 설계하고 구현하였다. 시스템의 가장 큰 특징으로는 방송 스트림에서 Manager 시스템을 통해 얻게 된 DSI, DII, DDB, Module, BIOP::ServiceGateWay, BIOP::Directory, BIOP::File 메시지를 저장하여 필요한
시정에 접근이 가능하도록 하여 Carousel 구조를 통해 나 타나게 되는 시스템 Delay를 줄일 수 있는 것이다. 보다 효율적인 캐싱 기법을 위해 DII의 Caching Priority Descriptor를 사용하여 시스템 전체 성능을 향상 시킬 수 있는 방법을 제안하였다.

향후, 제안한 캐싱 기법을 방송 수신기 시스템에 도입 하여 성능 향상을 시킬 예정이다.

참고 문헌