http://dx.doi.org/10.3745/KIPSTC.2011.18C.4.255

IPTV 서비스 검색을 위한 최적화 정보 기반 메타데이터 캡슐화 구조 설계 및 구현

오 봉 진[†] · 백 의 현[††] · 유 관 종[†††]

요 약

TV-Anytime은 XML을 PVR이나 방송서비스에 적용하기 위하여 서비스, 컨텐츠 정보 표현을 위한 스키마와 전송 프로토콜을 정의하여 많은 방송 규격에서 참조 문서로 활용되고 있다. 높은 확장성과 가독성에 비해 텍스트기반으로 정보를 기술하여 문서가 커지는 단점이 존재하며 이를 극복하기 위한 인코딩 알고리즘이 많이 제안되고 있다. 본 논문에서는 TVA 디스크립션을 전송하는 과정에서 문서의 크기를 최소화 할 수 있는 최적화 정보를 서버 단에서 반영하는 효율적인 인코딩 방식과 캡슐화 과정에서 색인 정보에 필요한 정보를 줄이고 정보를 빠르고 직관적으로 수신할 수 있는 인덱싱 방식을 제안한다.

키워드 : 메타데이터, 인코딩, 캡슐화, 인덱싱, XML, IPTV

Design and Implementation of an Optimization information based Metadata Encapsulation Architecture for IPTV Service Discovery

Bong-Jin Oh[†] · Eui-Hyun Paik[††] · Kwan-Jong Yoo[†††]

ABSTRACT

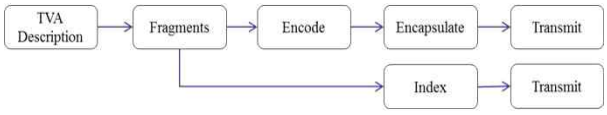
TV-Anytime is an XML-based standard for description of PVR and digital broadcast services. It is referenced by many standards of digital broadcast service for their schemas and delivery protocols of contents guide. Although its readability and extensibility, TV-Anytime has a big problem which generates massive documents due to its text-based description method. Therefore, various encoding algorithms have been proposed to reduce the size of XML documents. This paper proposes efficient metadata encapsulation architecture based on the optimization information generated at server-side to minimize XML documents. Advanced indexing method is also proposed to reduce resources needed to encapsulate indices and to receive metadata with fast and explicit mechanism.

Keywords : Metadata, Encoding, Encapsulation, Indexing, XML, IPTV

1. 서 론

XML (eXtended Markup Language)은 텍스트 기반으로 정보를 기술하는 언어로 높은 가독성과 확장성으로 인하여 다양한 영역에서 서비스나 콘텐츠를 기술하기 위하여 채택되고 있다. XML은 정보를 표현하기 위하여 스키마라는 문법체계를 통해 실제 문서의 기술하는 방법을 정의한 후 그에 따라 문서를 작성하는 방식이다[1,2].

TV-Anytime은 XML을 PVR (Personal Video Recorder)이나 방송서비스에 적용하기 위하여 서비스, 콘텐츠 표현을 위한 스키마와 전송 프로토콜을 정의하여 많은 방송 규격에서 참조 문서로 활용되고 있다. 특히, IPTV는 실시간 방송 외에 VoD, 게임 등의 부가서비스를 포함하고 있으므로 기존의 EPG (Electronic Program Guide)를 위한 DVB-SI (Service Information) 등 테이블 방식으로는 서비스 정보를 표현하는데 제한이 많아 TV-Anytime에서 정의하는 표현 방법을 사용하고 있는 추세다[3,4,5]. TV-Anytime에서 서비스 정보를 XML로 표현한 것을 TVA 디스크립션이라고 하며 (그림 1)과 같은 과정을 거쳐 단말로 전송된다.



(그림 1) TVA 디스크립션 전송 모델

† 정 회 원 : 한국전자통신연구원 책임연구원
†† 정 회 원 : 한국전자통신연구원 소셜컴퓨팅 연구팀장
††† 정 회 원 : 충남대학교 전자계산학과 정교수
논문접수 : 2011년 3월 23일
수 정 일 : 1차 2011년 5월 27일, 2차 2011년 6월 13일
심사완료 : 2011년 6월 14일

XML은 텍스트 기반이기 때문에 문서의 크기가 테이블 방식에 비해 커 전송하기 전에 인코딩을 통해 정보의 양을 줄인 후 블록 단위로 구성하여 전송한다. TV-Anytime에서는 그림1과 같이 독립된 문서로 나누는 단편화 과정을 거쳐 인코딩된 문서(프래그먼트)를 컨테이너라는 캡슐 구조에 넣고, 단말이 쉽게 검색할 수 있도록 색인 정보를 생성하여 전송한다[6,7,8].

인코딩과 색인 과정은 단말이 서비스 정보를 수신하고 관리하는 과정에서 메모리를 적게 사용하고 절차를 생략할 수 있는 중요한 요소로 볼 수 있으며, 다양한 인코딩 알고리즘과 색인 방식이 연구되고 제시되고 있는 상황이다. 기존의 연구는 범용 XML 문서에 대한 인코딩 그리고 PVR 등의 서비스를 고려한 색인정보를 다루기 위한 것이 대부분으로 사업자의 상황에 최적화되지 못함으로써 현실적으로 원하는 성능을 기대하기 힘들다[9,10].

본 논문은 송출되는 서비스 정보의 내용을 학습을 통해 생성된 최적화 정보를 가지고 메타데이터를 인코딩하여 최적화된 문서 크기와 속도를 기대할 수 있도록 하였고, 인덱싱 알고리즘을 방송 서비스에 맞도록 확장하여 색인 정보의 수를 감소시키고 검색 절차를 단순화 시켰다.

본 논문의 구성은 다음과 같다. 제 2장은 본 논문에서 제안하는 최적화 정보 기반 인코딩 방법을 설명하고, 제 3장에서는 컨테이너 기반의 인덱싱 방식에 대해 설명한다. 제 4장에서 상용 IPTV 사업자의 메타데이터를 이용하여 성능 분석 결과를 보이고, 제 5장에서 결론을 맺는다.

2. 최적화 정보 기반 EXI 인코딩 구조

2.1 EXI 인코딩 방식

EXI (Efficient XML Interchange)는 스키마를 반영하여 XML 문서를 인코딩하는 알고리즘 중 가장 성능이 우수한 오토마타 방식의 알고리즘으로 인코딩 스트림은 오토마타의 전이과정에 따라 생성된 이벤트 코드의 연속으로 이루어져 있다[2].

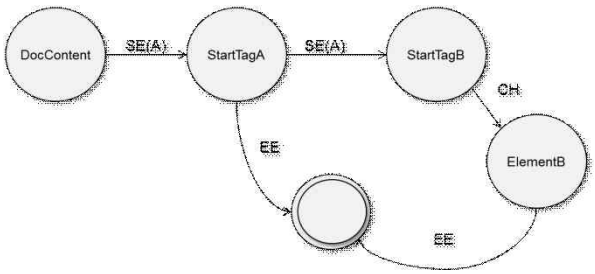
EXI 인코딩 단계는 그래머를 오토마타 기반으로 생성하는 절차와 이를 기반으로 입력 디스크립션을 스캐닝하며 인코딩하는 단계로 나누어진다. 그래머 생성은 프로토 단계, 정규화 단계, 코드 할당 단계 그리고 미정의 구문(Undeclared Product) 추가의 4 단계로 이루어진다.

```
<?xml version="1.0" encoding="UTF-8"?>
<xs: schema xmlns:xs="http://www.w3c.org/2001/XMLSchema"
elementFormDefault="qualified">
    <xs:element name="A">
      <xs:complexType>
        <xs:sequence maxOccurs="1">
          <xs:element name="B" type="xs:string" minOccurs="0"/>
        </xs:sequence>
      </xs:complexType>
    </xs:element>
</xs:schema>
```

(그림 2) 예제 XML 스키마

프로토 단계는 스키마로부터 엘리먼트, 애트리뷰트 별로 단위 오토마타를 추출한다. 추출한 단위 오토마타는 순서대로 연결되며 연결방법은 전 오토마타의 종료 상태를 후 오토마타의 시작 상태로 대체한다. 정규화 과정은 독립적으로 생성된 엘리먼트 이벤트와 속성 이벤트가 그대로 연결되어 생성된 프로토 타입 그래머에서 의미가 없는 분기를 제거한 것을 말한다.

이벤트 코드할당 단계는 생성된 그래머 내의 각 이벤트를 part1~part3로 나누어 이벤트 코드를 부여하는 것으로 part1이 엘리먼트나 속성을 위한 이벤트 코드이고, part2, part3는 프리픽스, URI 혹은 코멘트 등 선택적으로 추가될 수 있는 이벤트로 구성된다. 이벤트 코드 할당 단계에서는 스키마를 통해 분석 가능한 이벤트만 우선 생성된다.



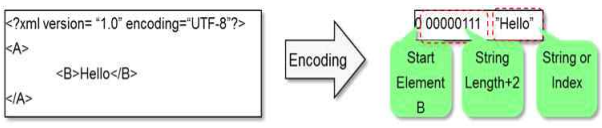
이벤트		이벤트 코드
DocContent	SE(A)	0
StartTagA	SE(B)	0
	EE	1
StartTagB	CH	0
ElementB	EE	0

(그림 3) 오토마타 및 이벤트 테이블 예제

미정의 구문 추가 단계에서 스키마에 표현되지 않은 코멘트나 속성, 엘리먼트를 위한 이벤트를 옵션에 따라 부여하는 단계이다. (그림 3)은 예제 스키마로부터 각 엘리먼트의 전이 과정을 정규화 오토마타로 표현하고 이벤트 코드를 부여한 것이다.

각 노드에서 발생할 수 있는 이벤트 코드의 크기는 노드로부터 전이될 수 있는 목적 노드의 수에 따라 아래와 같이 결정된다.

노드의 이벤트 크기 = $\lceil \log_2(N) \rceil$, N은 전이 가능한 목적 노드 수 (1)



(그림 4) 예제 XML 문서

(그림 4)는 예제와 같은 XML 문서를 (그림 3)의 오토마타를 기반으로 인코딩하여 스트림을 생성한 것을 보여준다. 프리픽스나 A 엘리먼트로의 전이와 같이 각 노드에서 단일 이벤트 발생만 존재하는 경우는 이벤트코드가 삽입되지 않고, StartTagA에서 StartTagB로의 전이와 같이 2개 이상의 이벤트가 존재하는 경우 어떤 이벤트가 발생하였는지 명시하기 위해 인코딩 스트림에 삽입된다. 또한 엘리먼트 값이 존재하는 경우 그 유형에 따라 인코딩된 값이 이벤트 다음에 삽입되는데 본 예제에서는 “Hello”라는 문자열 값이 엘리먼트 B를 위한 값으로써 삽입된 것을 보여준다. 문자열의 경우 최초 발생 시는 스트림에 직접 삽입되고 동일한 문자열이 반복되면 이를 문자열 테이블로 관리하고 인덱스가 삽입되는 방식을 사용한다.

+-+ 각 XML 문서를 EXI로 인코딩한 경우 발생되는 스트림의 크기는 (그림 4)의 문서를 구성하는입력 토큰에 따라 (그림 3)의 그래프를 통해 전이되는 노드를 모두 연결한 경로의 크기로 표현될 수 있으며, 경로를 구성하는 링크의 크기의 총합이 결국 최종 인코딩 스트림 크기가 된다. 각 링크는 전이시 발생되는 이벤트의 크기와 노드의 값을 인코딩한 것을 합한 것이 되므로 다음과 같이 인코딩 스트림의 크기를 표현할 수 있다.

만약, T가 XML 문서를 구성하는 토큰으로 이루어진 집합으로 가정하면 다음과 같이 표현될 수 있다.

$T = \{t_n | 0 \le n \le N\}$, N은 XML 문서를 구성하는 토큰 수 (2)

그리고, P (T)를 T에 포함되는 모든 토큰을 순차적으로 그래프에 적용하여 전이되는 노드로 구성된 경로이고, L은 그래프를 구성하는 노드 집합이며, L(t)는 토큰 t에 의해 전이되는 노드와 직접 연결된 노드의 수로 가정하면, P (T)의 크기는 다음과 같이 정의 할 수 있다.

EXI 헤더 (8bits) + $\sum_{t \in T} \lceil \log_2(L(t)) \rceil$ + $\sum_{t \in T} \lceil encode(Value_t) \rceil$, encode ()는 노드 값 인코딩 함수 (3)

따라서, XML 문서를 이루는 토큰의 수에 따라 선형적으로 증가하고 각 노드의 이벤트 크기 및 노드 값을 인코딩한 결과가 인코딩 크기에 영향을 미치는 주요 요인임을 알 수 있다.

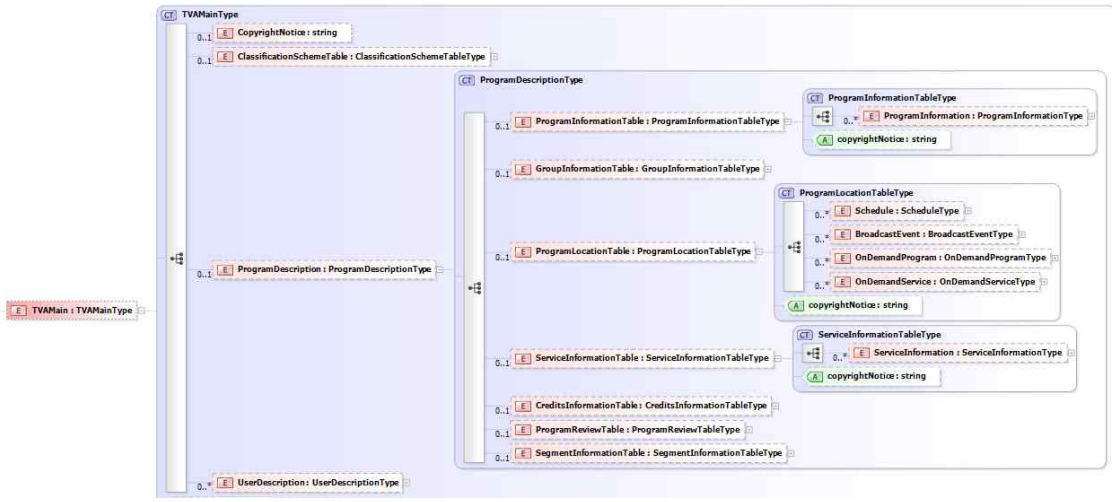
또한 P (T)를 인코딩하기 위한 시간은 T 내의 토큰을 순차적으로 적용하여 노드를 전이하면서 인코딩 절차를 그래머에 따라 진행하는 과정에 소요되는 것으로 볼 수 있으며,

P (T)의 인코딩 시간 $= \left(\sum_{t \in T} \lceil (L(t))(t_c + t_d) \rceil \right)$ + $\sum_{t \in T} t_e$ (4)

여기서 t_c, t_d는 각 노드를 위해 그래머를 생성하고, 제거하기 위해 필요한 시간으로써 전이 가능한 노드 수와 동일하게 생성되며, t_e는 실제 입력 토큰에 대해 수행된 인코딩 절차를 말한다. 입력 토큰에 대해 노드가 이동될 때마다 각 목적 노드로 전이하기 위한 그래머가 동적으로 생성되고 노드 이동시 제거된다. 따라서, 각 노드로부터 전이 가능한 노드의 수에 따라 인코딩 시간이 영향을 받음을 알 수 있다.

2.2 최적화정보 기반 EXI 인코딩 방식

TV-Anytime 디스크립션 작성은 서버의 선택에 따라 선택적으로 필드나 속성을 가지고 정보를 표현할 수 있도록 정의되어 있다. EXI의 경우 정규화된 그래머에서 각 노드에서 발생할 수 있는 이벤트 수를 N이라고 한다면 $\log_2 N$개의 비트로 이벤트코드를 표현할 수 있다. 따라서 TV-Anytime 전체 스키마를 통해 생성한 그래머를 이용한 인코

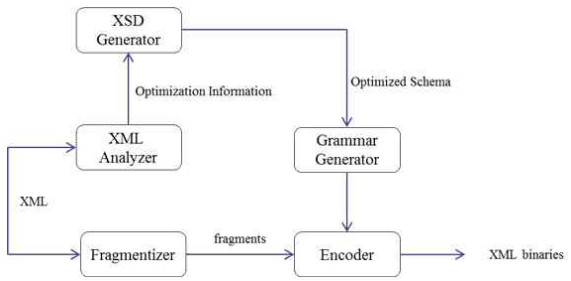


(그림 5) TV-Anytime 스키마 구조

딩보다 서버에 최적화된 스키마를 통해 생성한 그래머를 이용한다면, 전체 노드의 수나 각 노드에서 발생하는 이벤트 수가 적어지고 그에 따른 이벤트 표현을 위한 비트의 크기도 적어지므로 보다 더 효율적인 인코딩을 기대할 수 있다. 특히, 국내 TTA 표준에서 진행 중인 IPTV 콘텐츠 정보 및 전송 규격 기반으로 생성된 샘플을 보면 TV-Anytime 메타데이터에서 주로 사용 되는 엘리먼트 위주로 매우 제한적이며 사용되지 않는 엘리먼트들도 인하여 낭비되는 공간이 많을 것으로 판단된다.

참고로, (그림 5)의 스키마 중 EPG를 위해서 주로 ProgramInformation, ServiceInformation, ProgramLocation 테이블 정도가 사용되며 나머지는 거의 사용되지 않는다.

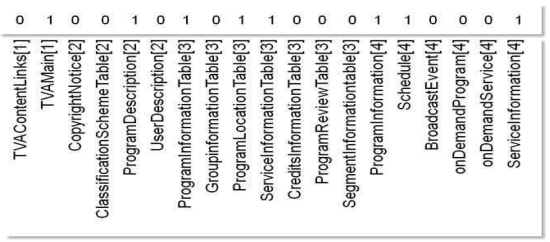
2.2.1 최적화 정보 기반 EXI 인코딩 구조 설계



(그림 6) 최적화 정보 기반 EXI 인코딩 구조

Fragmentatizer가 XML 데이터를 단편화 한 후 모든 프래그먼트를 인코딩 하기 전에 XML Analyzer가 각 프래그먼트 분석을 통해 최종 최적화 정보를 생성한다. 생성된 최적화 정보를 이용해서 XSDGenerator가 최적화된 스키마를 생성한다. 이렇게 생성된 스키마로 Grammar Generator가 그래머를 생성한 후 해당 그래머를 이용해서 해당 프래그먼트를 인코딩 한다. 이렇게 인코딩에 사용된 최적화 정보는 클라이언트로 전송되어 프래그먼트를 디코딩 할 때 사용된다.

메타데이터 최적화 정보는 엘리먼트의 존재유무를 표시하는 비트 플래그(Flag)열로 이루어져 있다. 스키마에 정의된 엘리먼트의 순서대로 해당 엘리먼트가 존재할 경우 해당 비트를 1, 존재 하지 않을 경우 0으로 표시한다. 존재유무를 표시하는 엘리먼트는 메타데이터에서 최적화 깊이를 설정하여 결정하도록 한다. (그림 5)에서 깊이 4를 적용하여 EPG를 위한 3가지 테이블만 사용하는 XML 문서를 위해 생성된 최적화 정보는 다음과 같은 1차원 비트열로 이루어진다.



(그림 7) 예제 문서 기반 최적화 정보

TV-Anytime 스키마를 루트부터 Shallow traverse 하며 상위 노드가 0인 경우는 하위 트리는 탐색하지 않고 비트 플래그 열에서도 제외한다. 깊이 4의 경우 19비트로 최적화 정보가 표현 가능하므로 감소되는 문서의 양에 비해 오버헤드가 상대적으로 크지 않음을 알 수 있다.

2.2.2 최적화 스키마 생성

```
<element name="TVAMain" type="tva:TVAMainType"/>
<complexType name="TVAMainType">
<sequence>
 <element name="CopyrightNotice" type="string" minOccurs="0"/>
 <element name="ClassificationSchemeTable" type="tva:ClassificationSchemeTableType" minOccurs="0"/>
 <element name="ProgramDescription" type="tva:ProgramDescriptionType" minOccurs="0"/>
 <element name="UserDescription" type="tva:UserDescriptionType" minOccurs="0"
maxOccurs="unbounded"/>
</sequence>
</complexType>
```

↓

```
<element name="TVAMain" type="tva:TVAMainType"/>
<complexType name="TVAMainType">
<sequence>
 <element name="ProgramDescription" type="tva:ProgramDescriptionType" minOccurs="0"/>
</sequence>
</complexType>
```

(그림 8) 최적화 정보 기반 스키마 구조 최적화

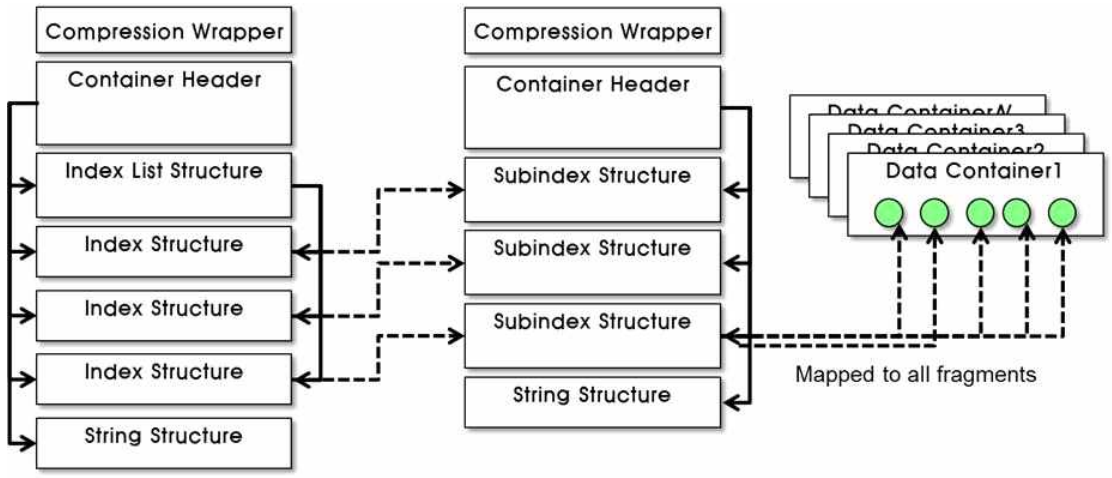
스키마를 최적화 하는 방법은 스키마 구조 최적화와 스키마 정의 최적화로 나눌 수 있다. 구조 최적화는 Sequence, all, choice로 표현된 복합유형 (Complex Type) 엘리먼트 구조 목록을 순차적으로 분석하여, 최적화 플래그에 0으로 설정된 엘리먼트는 삭제하는 방법이다. (그림 5)의 TVA MainType의 하위 엘리먼트 리스트 중 ProgramInformationTable을 제외한 나머지 하위 엘리먼트를 삭제하면 (그림 8)과 같이 엘리먼트 간의 관계가 축소되어 표현된다.

```
<complexType name="UserDescriptionType">
<sequence>
 <element name="UserPreferences" type="mpeg7:UserPreferencesType" minOccurs="0"/>
 <element name="UsageHistory" type="mpeg7:UsageHistoryType" minOccurs="0"/>
</sequence>
</complexType>
<complexType name="ClassificationSchemeTableType">
<sequence>
 <element name="CSAlias" minOccurs="0" maxOccurs="unbounded">
                                                    ...
 </element>
 <element name="ClassificationScheme" minOccurs="0" maxOccurs="unbounded">
                                                    ...
 </element>
</sequence>
</complexType>
<complexType name="ProgramDescriptionType">
<sequence>
 <element name="ProgramInformationTable" type="tva:ProgramInformationTableType" minOccurs="0"/>
                                                    ...
 <element name="SegmentInformationTable" type="tva:SegmentInformationTableType" minOccurs="0"/>
</sequence>
</complexType>
```

↓

```
<complexType name="ProgramDescriptionType">
<sequence>
 <element name="ProgramInformationTable" type="tva:ProgramInformationTableType" minOccurs="0"/>
 <element name="GroupInformationTable" type="tva:GroupInformationTableType" minOccurs="0"/>
 <element name="ProgramLocationTable" type="tva:ProgramLocationTableType" minOccurs="0"/>
 <element name="ServiceInformationTable" type="tva:ServiceInformationTableType" minOccurs="0"/>
 <element name="CreditsInformationTable" type="tva:CreditsInformationTableType" minOccurs="0"/>
 <element name="ProgramReviewTable" type="tva:ProgramReviewTableType" minOccurs="0"/>
 <element name="SegmentInformationTable" type="tva:SegmentInformationTableType" minOccurs="0"/>
</sequence>
</complexType>
```

(그림 9) 최적화 정보 기반 스키마 정의 최적화



(그림 10) 인덱싱 컨테이너 구조

스키마 정의 최적화 방법은 하위 앨리먼트 리스트에서 삭제된 앨리먼트의 선언부를 삭제하는 방법이다. (그림 9)는 ProgramInformationTable에 대한 선언부만 유지하도록 선언부를 최적화 하는 과정을 보여주고 있다.

(그림 9)와 같은 과정을 거쳐 생성된 스키마를 기반으로 최적화된 오토마타를 구성하고 실제 인코딩에 사용될 그래머를 생성한다. 단편화된 프래그먼트는 그래머를 이용하여 인코딩되고 이를 컨테이너라는 캡슐에 저장하여 유니캐스트나 멀티캐스트로 전송된다.

단말에서는 디코딩을 위해 TVAInit 메시지 [8]를 수신하여 디코딩 관련 정보를 획득하게 되는데 인코딩 버전(EncodingVersion)을 EXI를 위해 사용자 영역인 0xF0를 사용한다. 0xF0일 경우 EXI 인코딩 상세정보 기술 부분에 최적화 정보 사용 플래그를 두고 최적화 스키마를 사용할 것인지 결정하고 EXI의 DecoderInit() 영역에 비트 플래그 열을 저장하여 디코딩 시 최적화된 스키마를 최적화 할 때 적용하도록 한다.

3. 컨테이너 위치정보 기반 인덱싱

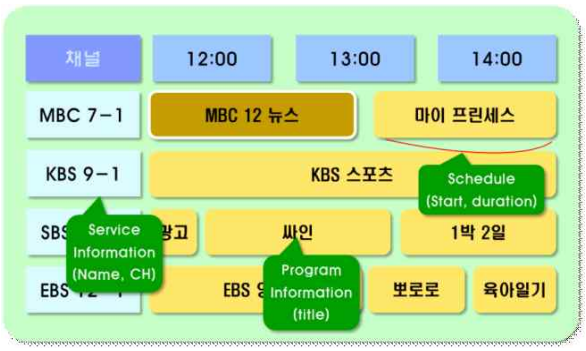
3.1 TV-Anytime 인덱싱 방식

컨테이너 헤더에 색인 정보를 위한 구조로써 인덱스 목록, 인덱스 그리고 서브 인덱스와 색인 구성에 사용되는 문자열을 저장하기 위한 문자열 저장소 구조의 구성 정보가 포함된다.

인덱스 목록 구조는 전체 인덱스 컨테이너에 하나만 존재하고 검색한 메타데이터의 유형과 조건을 포함하고 있다. 인덱스 목록 구조 내의 인덱스는 인덱스 구조와 대응되며 프래그먼트를 검색하기 위한 조건의 범위를 포함하고 있다. 같은 유형의 프래그먼트를 검색 조건의 범위를 분할 하여 다중으로 인덱스 목록 구조에 기술할 수 있고, 그와 대응되어 인덱스 구조가 존재할 수 있다. 일반적으로 인덱스 목록과 인덱스 구조는 그 양이 많지 않으므로 하나의 인덱스 컨테이너에 포함되어 전송된다[8].

TV-Anytime의 인덱싱 알고리즘의 가장 특징은 인덱스 목록 구조가 단일 유형의 프래그먼트를 검색하도록 구성되었다는 것과 함께 서브 인덱스 구조가 인덱스 구조 내에 기술된 검색 조건에 해당하는 모든 프래그먼트에 대한 색인 정보를 갖고 있다는 것이다. 서브 인덱스 구조는 검색하려는 프래그먼트의 수에 비례하여 커지게 되고 일반적으로 인덱스 목록, 인덱스 구조가 포함된 컨테이너와는 별도의 복수개의 컨테이너로 구성되어 전송된다.

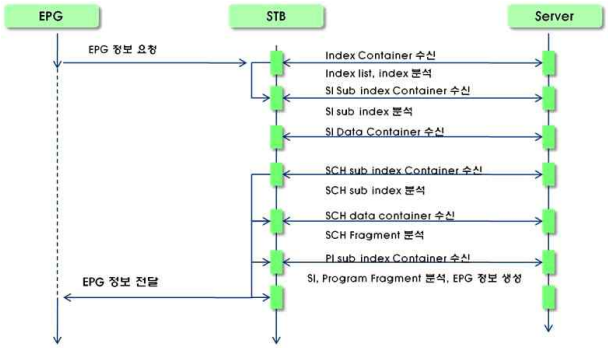
단일 유형의 프래그먼트 검색으로 제한된 TVA 인덱스 알고리즘은 서버의 입장에서도 다양한 조건을 이용하여 데이터 컨테이너를 구성하는 것이 어렵도록 되어 있고, 단말의 입장에서 필요한 정보의 구축을 위하여 여러 번의 검색 절차를 진행하고 또한 불필요한 오버헤드를 감수하도록 요구한다.



(그림 11) EPG 구성에 필요한 메타데이터

(그림 11)은 실시간 방송에서 TVA 메타데이터를 적용하여 컨텐츠 가이드 정보를 구성하고자 할 때 필요한 정보를 보여주는 것이다.

채널 정보는 ServiceInformation 프래그먼트를 이용하여 타이틀, 서비스 식별자 (Service Identifier), 채널 공급자, 로고 그리고 논리적 채널 등의 부가 정보를 전송하고 단말이 서비스 식별자나 채널을 이용하여 EPG 상에 채널 정보를 표현한다. 채널을 기준으로 가로로 기술되는 프로그램의 편성 내역은 Schedule 프래그먼트와 ProgramInformation 프래그먼트를 이용하여 정보를 구성한다. Schedule 프래그먼트는 각 서비스와 Service Id를 이용하여 연관성을 표현하고 시간대별로 방송되는 프로그램 정보를 ScheduleEvent로 정의하여 내부에 포함한다. 프로그램 식별자 (Program ID)를 기반으로 ScheduleEvent와 ProgramInformation 프래그먼트가 대응되며 타이틀, 장르, 시놉시스 등의 정보를 가져오는데 사용된다. 따라서 (그림 12)와 같은 순서대로 메타데이터를 가져오고 EPG를 수행하고 채널 전환 등이 이루어진다[4].



(그림 12) TVA 인덱싱 기반 EPG 수신 절차

현재의 TV-Anytime에서 제공하는 인덱싱 방식은 다음과 같은 문제점이 발생한다.

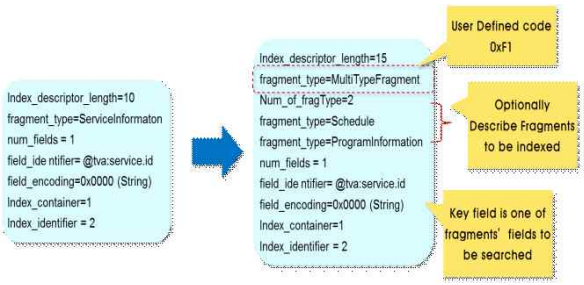
- 필요한 데이터 컨테이너 분석을 위한 색인 정보가 여러 개의 인덱스 컨테이너로 구성되어 수신된다
- 관련된 프래그먼트를 위한 인덱스 컨테이너를 검색하기 위하여 앞서 수신한 프래그먼트를 분석해야만 한다
- 필요한 프래그먼트가 포함된 데이터 컨테이너를 알아내기 위하여 색인정보를 전수조사 해야한다
- 색인정보를 위해 관리해야 할 메모리가 프래그먼트 수에 비례하여 커진다

3.2 컨테이너 위치정보 기반 인덱싱

3.2.1 복수 프래그먼트 유형 지원 인덱스 목록

우선 TVA가 한번에 하나의 프래그먼트 유형을 검색하도록 정의한 인덱스 방식을 복수 유형의 프래그먼트를 동시에 찾을 수 있도록 인덱스 목록 구조 정보를 (그림 13)과 같이 확장한다.

검색할 프래그먼트의 유형을 위한 프래그먼트 식별자를 사용자 영역에서 할당하여 (그림 13에서는 0xF1) 사용하거나 "TVAMain.Composite" 형태로 문자열로 기술한다. 프래그먼트의 유형이 복수 유형 검색 모드일 때 실제 검색되는



(그림 13) 복수 프래그먼트 검색 지원 인덱스

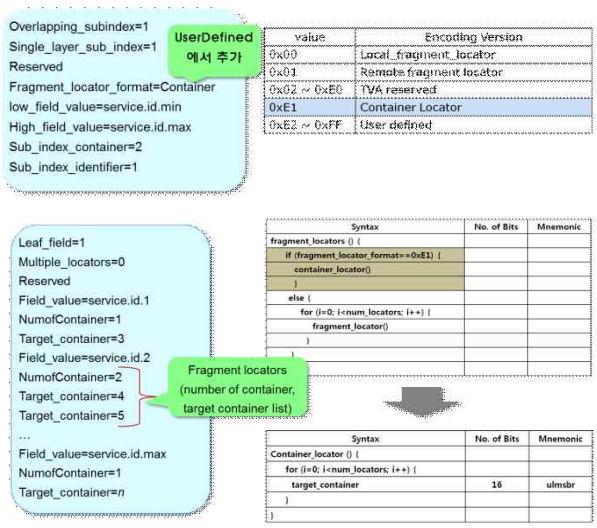
프래그먼트의 유형을 위한 목록을 아래에 정의할 수 있도록 한다. 유형의 수와 그 수만큼의 프래그먼트 식별자 혹은 W3C의 XPATH [1] 방식의 문자열을 기술한다.

그리고 검색 조건은 검색되는 어떤 유형의 프래그먼트 내 필드를 하나 이상 사용하여 기술하도록 한다. 이렇게 하면 자신이 포함하지 않는 필드 값을 가지고 검색될 수 있는 방법이 생겨 서버 측면에서 다양한 데이터 구성을 가지고 송출할 수 있다.

예를 들면, EPG 정보 구성에 있어 StartTime이라는 Schedule 프래그먼트의 필드 값을 중심으로 Schedule, ProgramInformation을 하나로 묶어 송출할 수 있다. ProgramInformation의 경우 StartTime 필드가 없지만 이 시간에 송출되는 프로그램 정보를 기술하는 프래그먼트로써 단말의 입장에서는 수신해야 하고 특히 시간 정보를 기준으로 검색하는 경우가 자주 있다.

3.2.2 컨테이너 위치정보기반 서브인덱스 구조

(그림 14)와 같이 인덱스 구조와 대응되는 서브 인덱스 구조 내의 색인 정보를 각 프래그먼트가 아닌 관련된 프래그먼트의 집합을 가르키도록 한다. 즉, 컨테이너 단위로 위치정보를 가르키도록 인덱스 구조 내에 프래그먼트 위치 정보 유형을 확장하고, 서브 인덱스 구조 내에 각 검색 필드



(그림 14) 컨테이너 위치기반 인덱스 및 서브인덱스

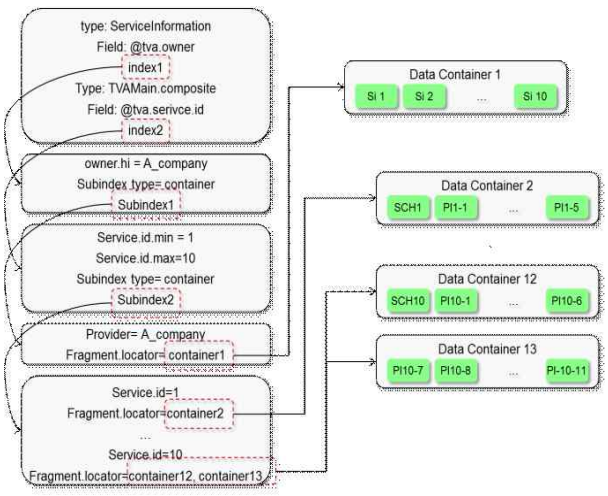
별로 연속된 컨테이너 위치정보를 기술한다.

하나의 컨테이너는 인덱스 조건에 맞춰 구성된 서로 다른 프래그먼트들의 집합이며 복수개의 컨테이너가 될 수 있다. 복수개의 컨테이너로 구성해야 하는 경우는 서브 인덱스를 위한 컨테이너 위치정보를 컨테이너 수만큼 연속하여 기술하면 된다.

컨테이너 위치정보 기반 서브 인덱스는 복수 유형 프래그먼트 인덱스 구조와 결합하여 EPG를 위한 정보 구성과 검색 방법을 직관적이고 효율적으로 지원할 수 있다. 특히 필요한 정보를 검색하기 위해 부가적으로 제공되는 인덱스 구성을 위한 정보의 양을 비약적으로 줄일 수 있다.

(그림 15)는 EPG 정보를 본 논문에서 제안한 인덱싱 방식을 통하여 재구성한 예이다. ServiceInformation 프래그먼트 검색 인덱스와 Schedule과 ProgramInformation을 동시에 검색하기 위한 인덱스 2개가 인덱스 목록 구조에 기술된다.

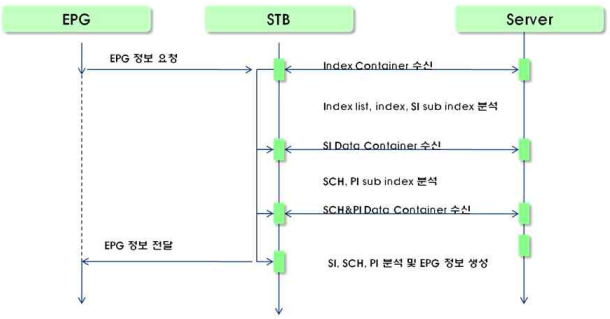
각 채널 정보 표현을 위한 ServiceInformation 검색을 위한 인덱스는 기존의 방식을 그대로 사용한다. 그러나 프로그램의 스케줄과 상세 정보를 위한 Schedule, Program Information 프래그먼트 검색은 복수 유형의 프래그먼트 인덱스로 표현한다. 또한 데이터 컨테이너도 서비스 식별자(Service Identifier)라는 검색 기준으로 Schedule, Program Information 프래그먼트를 취합하여 구성한다. 이때 같은 컨테이너에 포함되는 ProgramInformation 프래그먼트는 Schedule 내의 ScheduleEvent와 대응되는 것들로 구성된다.



(그림 15) EPG용 컨테이너 위치정보 인덱싱 예

(그림 15)와 같은 방식으로 하면 기존의 방식에 비해 다음과 같은 장점이 있다.

- ServiceInformation, Schedule, ProgramInformation 프래그먼트를 위한 인덱스 및 서브 인덱스 구조를 위한 정보가 최소한으로 감소한다.
- 인덱스 컨테이너가 하나로 충분하여 인덱스 컨테이너 관리를 위한 단말 리소스와 분석을 위한 시간 비용이 줄어든다.



(그림 16) 제안된 인덱싱 방법 기반 EPG 구성

- 관련된 프래그먼트가 이미 결합된 상태로 송, 수신되므로 필요한 데이터 컨테이너를 찾아내는 시간적 비용이 줄어든다.

프래그먼트 수신을 위한 인덱스 검색부터 데이터 컨테이너 분석을 위한 절차를 살펴보면 (그림 16)과 같은 데이터 모델을 갖는다. 그림 12의 전통적인 TV-Anytime 인덱싱 방법에 비하여 절차가 단순하고 직관적임을 알 수 있다.

4. 성능분석

〈표 1〉 시험환경

환경		내용
Processor		CPU 1GHz, RAM 512MB
Test Software		EXIfficient 0.6
인코딩	BBC2002 Program Information	17~37KB까지의 공개 문서
인덱싱	ServiceInformation	142 개(142 채널)
	Schedule	512 개(12시간, 3일)
	ProgramInformation	7,750개

4.1 시험환경

인코딩 시험은 PC에서 이루어졌으며 EXI 표준 개발 시 공개된 지멘스의 EXIfficient 0.6 기반으로 스키마를 분석하고 최적화 시키는 모듈을 추가한 이뮬레이터를 이용하여 BBC의 공개 Program Information 문서를 기반으로 이루어졌다. 인덱싱 시험은 국내 IPTV 사업자인 L사의 142채널 3일치(12시간 단위로 구분)에 해당하는 정보를 DVB-SI 스트림으로부터 추출하여 TTA에서 제정중인 표준에 맞게 문서를 생성하여 제안 알고리즘의 성능을 분석하였다. EPG에 필요한 ServiceInformation, Schedule, ProgramInformation을 추출하여 사용했고 각 프래그먼트의 크기는 유형별로 평균 2KB, 1-12KB 그리고 1.5KB의 크기를 갖고 있었다.

4.2 최적화 정보기반 EXI 인코딩 구조

4.2.1 EXI 모델을 이용한 성능 분석

본 절에서는 최적화 정보에 의해 불필요한 엘리먼트가 제거된 스키마를 가지고 인코딩할 때 압축 효율과 인코딩 효율이 향상될 수 있음을 2장의 (3)과 (4)에 기반하여 분석한다.

우선, M을 최적화된 스키마 기반의 노드 집합으로 보면, M ⊆ L이다. 왜냐하면 L 중 T내의토큰에 의하여 전이되지 않는 노드를 제외한 노드의 집합이 M이기 때문이다. 따라서, (3)에서 $\sum_{t\in T}\lceil \log_2(L(t))\rceil$ 를 $\sum_{t\in T}\lceil \log_2(M(t))\rceil$ 로 대체하면 최적화 스키마에 의한 인코딩 스트림 크기를 구할 수 있고, 그에 따른 인코딩 효율에 대한 효과를 정의하면 다음과 같다.

$$e_f = \left(\sum_{t\in T}\lceil \log_2(M(t))\rceil \Big/ \sum_{t\in T}\lceil \log_2(L(t))\rceil \right) \quad (5)$$

T에 포함되는 모든 토큰에 대해 M(t) ≤ L(t)이므로 결국 $e_f \le 1$가 된다. 따라서 스키마를 최적화 하는 것은 압축 효율 향상에 도움을 줄 수 있다고 판단할 수 있다.

인코딩이나 디코딩 시간에 대해서도 식 (4)에서 L(t)를 M(t)로 대체하면 되고, 모든 T내의 토큰에 대해 M(t) ≤ L(t)이므로,

$$e_g = \left(\sum_{t\in T} M(t) \Big/ \sum_{t\in T} L(t) \right) \le 1 \quad (6)$$

로 압축 시간 역시 향상될 수 있는 것으로 판단할 수 있다.

위와 같은 분석에 따라, EXI와 같이 스키마 기반으로 오토마타를 생성하여 인코딩하는 MPEG-7의 BiM (Binary format for Metadata)과 같은 인코딩 방식에서도 동일하게 적용할 수 있으며, 특정 사업자가 사용하는 XML에 대해서도 학습을 통해 미리 최적화 정보를 생성하여 동일하게 적용이 가능함을 알 수 있다.

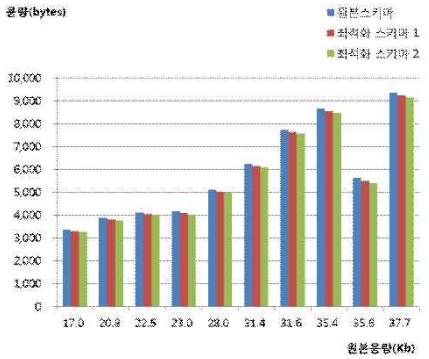
4.2.2 실험을 통한 성능 분석

BBC 문서를 이용하여 기존 EXI 방식에 의한 인코딩 결과 그리고 최적화 정보 기반의 EXI 인코딩 방식에 대해 압축 효율을 비교하였다. 최적화 정보는 깊이 4, 8로 엘리먼트를 최적화 하였고 그래프에서는 각각 스키마1, 스키마2로 표현된다. 그래프 (a)에서 가로축은 원본 메타데이터의 용량을 Kb로 나타내고 세로축은 압축후의 용량을 바이트로 나타낸 것이다.

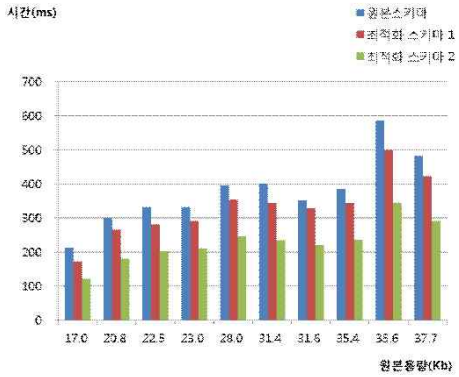
최적화 깊이 8을 이용한 경우가 원본 스키마를 이용한 경우와 비교하여 최대 104.1%, 최소 102.12, 평균 102.9%의 압축효율을 보였다.

프래그먼트 단위로 나누어져 인코딩 되어 문서의 크기가 작고, 엘리먼트 구성 패턴에 따라 스키마 최적화의 효과가 좌우되며 실험결과 예상보다는 압축률의 향상이 적게 나타남을 알 수 있었다.

(b) 그래프는 BBC 프로그램 정보에 대해 EXI와 최적화 정보 기반 EXI 방식에 대한 디코딩 속도를 비교하였다. 디코딩 시간은 최적화 방식이 최대 173.8%, 최소 157.4% 개선되어 평균 165%의 더 빠른 속도를 보였다. 압축률의 향상에 비해 속도의 측면에서 매우 많은 개선이 있음을 보여주는 실험결과이다. 원본 스키마 기반 EXI는 노드간 전이시 불필요한 그래머 처리를 위한 모듈 생성 및 제거를 위한 절차가 포함되어 속도에 대한 비효율성의 존재를 판단할 수 있다.



(a) 압축 크기



(b) 디코딩 속도

(그림 17) 압축 효율 및 디코딩 속도 비교

4.3 컨테이너 위치정보기반 인덱싱

인덱스 컨테이너는 제안된 알고리즘에 의하면 인덱스 목록, 인덱스 그리고 모든 서브 인덱스가 하나의 컨테이너에 포함되어 전송된다. 그러나, TV-Anytime 알고리즘의 경우는 인덱스 1개와 SI (Service Information) 서브인덱스를 위한 컨테이너 1개, SCH (Schedule) 서브 인덱스를 위한 컨테이너 1개 그리고 PI (Program Information) 서브인덱스를 위한 컨테이너 7개로 이루어진다. 인덱스 수의 경우는 제안된 알고리즘이 블록 단위의 위치정보로 색인정보를 표현하므로 SI 1개 그리고 SCH와 PI가 결합된 색인 6개만 서브인덱스로 표현된다.

반면에 TV-Anytime의 색인 정보는 각 프래그먼트 유형별로 데이터 컨테이너를 구성하는 모든 프래그먼트 수만큼 존재함을 알 수 있다. 그러나 제안 알고리즘의 경우 인덱스가 컨테이너 목록으로 표현되므로 하나의 SCH, PI 복합 인덱스의 경우 평균적으로 10개의 컨테이너를 가르키도록 되

〈표 2〉 인덱스 컨테이너 구성(단위: 개)

	제안 알고리즘		TVA 알고리즘	
	Type	Number	Type	Number
사업자 L	Index	1	Index	1
			SI	1
	SI		SCH	1
	SCH, PI		PI	7

〈표 3〉 인덱스 구성(단위: 개)

	제안 알고리즘		TVA 알고리즘	
	Type	Number	Type	Number
사업자 L	SI	1	SI	142
			SCH	512
	SCH, PI	6	PI	7,750

어 있어 하나의 프래그먼트와 맵핑되는 TV-Anytime의 인덱스에 비해 개당 필요한 바이트 수는 크다. 그럼에도 불구하고 전체적으로 비교하였을 때 필요한 메모리 크기는 1/100이하임을 알 수 있다.

5. 결 론

본 논문에서 제안하는 최적화 정보 기반 인코딩과 컨테이너 위치정보 인덱싱 방식으로 기존 인코딩 방식에 비해 평균 2-4%정도의 압축효율이 증가되었고, 디코딩 속도는 상대적으로 50%이상 개선되는 효과가 있었다. 인덱스 관련 정보의 측면에서는 색인에 필요한 정보의 양이 대폭 감소하고 그에 따라 인덱스 정보가 컨테이너 1개로 구성 가능하여, 단말의 리소스 부담과 검색 절차가 간소화 해진다는 장점이 있었고 다양한 조건에 의해 데이터를 송출하고 수신할 수 있었다.

따라서, 스키마 기반의 XML 인코딩 알고리즘과 TVA 인덱싱 방법을 사용하는 IPTV 서비스 플랫폼에 제안 알고리즘을 적용하면, 보다 직관적이고 단말의 리소스를 적게 소비하면서 서비스 정보를 전달하고 분석할 수 있도록 시스템 구축이 가능할 것으로 기대된다.

참 고 문 헌

[1] W3C, "Extensible Markup Language (XML) 1.0 5th Ed," W3C Recommendation, 2006.

[2] W3C, "Efficient XML Interchange (EXI) Format 1.0," W3C Candidate Recommendation, 2009.

[3] ETSI, "Digital Video Broadcasting (DVB), Transport of MPEG-2 TS based DVB Services over IP based Networks," ETSI TS 102 034 V1.4.1, 2009.

[4] TTA, "IPTV 콘텐츠 가이드 정보 및 전송 방식", TTAK.KO-08.0028, 2010.

[5] Hongguang Zhang, Shibao Zheng, "Personalized TV Program Recommendation based on TV-Anytime Metadata," International Symposium on Consumer Electronics, 2005.

[6] 유성재, 최일선, 윤화묵, 안병호, 정희경, "Fast Infoset을 이용한 Binary XML Encoder의 설계 및 구현", 한국해양정보통신학회 춘계종합학술대회, 2006.

[7] TV-Anytime Forum, "TVA Specification S-3 on Metadata, Part A: Metadata Schemas," Ver.1.3, 2002.

[8] TV-Anytime Forum, "TVA Specification S-3 on Metadata, Part B: System Aspects in a Unidirectional Environment," Ver.1.3, 2002.

[9] Young-Guk Ha, Beom-Seok Jang, Bong-Jin Oh, Yu-Seok Bae, Eui-Hyun Baik, "Effective Encoding of TV-Anytime Metadata Using EXI," International Conference of Consumer Electronics, 2011.

[10] 장범석, 하영국, 오봉진, 백의현, "EXI 기반의 효율적인 TV-Anytime 메타데이터 전송 기법", 한국정보처리학회 추계학술대회, 2010.

오 봉 진

e-mail : bjoh@etri.re.kr

1993년 부산대학교 전자계산학과

1995년 부산대학교 전자계산학과(석사)

2008년 충남대학교 컴퓨터공학과 (박사수료)

1995년~1997년 한국전자통신연구원 연구원

1998년~현 재 한국전자통신연구원 책임연구원

관심분야 : 디지털방송, IPTV, 소셜컴퓨팅, 홈네트워크 등

백 의 현

e-mail : ehpaik@etri.re.kr

1984년 숭실대학교 전자계산학과

1987년 숭실대학교 전자계산학과(석사)

1985년 숭실대학교 전자계산학과 (박사수료)

1987년~2002년 한국전자통신연구원 연구원

2003년~현 재 한국전자통신연구원 소셜컴퓨팅연구팀장

관심분야 : CAS, IPTV, 소셜컴퓨팅, 홈네트워크 등

유 관 종

e-mail : kjyoo@cnu.ac.kr

1976년 서울대학교 전자계산학과

1978년 서울대학교 전자계산학과(석사)

1997년 서울대학교 전자계산학과
(박사수료)

1979년~현 재 충남대학교 전자계산학과
정교수

관심분야 : 병렬처리, 클러스터, 스케일러블 코딩, IPTV 등