FBMF Standard

미래방송미디어포럼표준(국문표준)
FBMF-STD-032 제정일: 2025. 12. 05.

방송 클라우드 시스템 – 상호작용 콘텐츠

Broadcast Cloud System – Interaction Content

(앞 표지

FBMF
미래방송미디어표준포럼
Future Broadcast and Media Standards Forum

표준초안 검토 위원회 방송 클라우드 분과위원회

표준안 심의 위원회 운영위원회

	성명	소 속	직위	위원회 및 직위
표준(과제) 제안	박경모	CAST.ERA	CTO	방송 클라우드 분과장
표준 초안 에디터	곽진석	카이미디어	이사	방송 클라우드 분과 간사
	박재형	한시간컴	팀장	방송 클라우드 분과 위원
	강전호	한시간컴	선임연구원	방송 클라우드 분과 위원
사무국 담당	함상진	KBS	수석	미래방송미디어표준포럼 사무총장

본 표준 발간 이전에 접수된 지식재산권 확약서 정보는 본 표준의 '부록(지식재산권 확약서 정보)'에 명시하고 있으며, 이후 접수된 지식재산권 확약서는 미래방송미디어표준포럼 웹사이트에서 확인할 수 있습니다.
본 표준과 관련하여 접수된 확약서 외의 지식재산권이 존재할 수 있습니다.

발행인 : 미래방송미디어표준포럼 의장
발행처 : 미래방송미디어표준포럼
06130, 서울특별시 강남구 테헤란로 7길 22 신관 1108호
Tel : 02-568-3556, Fax : 02-568-3557
발행일 : 2025.12

목차

방송 클라우드 시스템 - 상호작용 콘텐츠
Broadcast Cloud System - Interactive Content

1 적용 범위

1.1 Introduction and Background

이 문서에서는 대화형 콘텐츠에서 지원하는 ATSC 3.0 Receiver 에서 향상된 뷰어 환경을 제공하기 위해 사용할 수 있는 환경 및 인터페이스에 대해 설명합니다.

1.2 Organization

이 문서는 다음과 같이 구성됩니다.

- Section 1 - 이 사양의 범위, 소개 및 배경
- Section 2 - 규범적이고 유익한 참조
- Section 3 - 규정 준수 표기법, 용어 정의 및 약어
- Section 4 - 시스템 수준에서 대화형 콘텐츠 환경 개요
- Section 5 - 참조 Receiver 모델 사양
- Section 6 - Broadcaster Application 을 관리하는 방
- Section 7 - 이 표준에서 지원하는 다양한 미디어 플레이어에 대한 세부 정보
- Section 8 - 수신기에서 지원하는 WebSocket 인터페이스 개요
- Section 9 - WebSocket 인터페이스의 지원되는 메서드
- Annex A - 유익한 애플리케이션 수명 주기 시퀀스 다이어그램
- Annex B - 이 표준에서 사용하는 JSON-RPC 2.0 사양의 전체 복사본

2 인용 표준

[1] ATSC: "ATSC Standard: System Discovery and Signaling," Doc. A/321:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.
[2] ATSC: "ATSC Standard: Link Layer Protocol," Doc. A/330:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.
[3] ATSC: "ATSC Standard: Signaling, Delivery, Synchronization, and Error Protection," Doc. A/331:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.
[4] ATSC: "ATSC Standard: Service Announcement," Doc.

A/332:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.

[5] ATSC: "ATSC Standard: Content Recovery in Redistribution Scenarios," Doc. A/336:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.

[6] ATSC: "ATSC Standard: Application Event Delivery," Doc. A/337:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.

[7] ATSC: "ATSC Standard: Captions and Subtitles," Doc A/343:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.

[8] ATSC: "ATSC Standard: ATSC 3.0 Security and Service Protection," Doc. A/360:2024-04, Advanced Television Systems Committee, Washington, DC, 3 April 2024.

[9] [9] CTA: "CTA Specification: Web Application Video Ecosystem – Web Media API Snapshot", Doc. CTA-5000-G, Consumer Technology Association, Arlington, VA, October 2024.

[10] CTA: "CTA Bulletin: Recommendations for User Overrides for Closed Caption Decoders", Doc. CTA-CEB35, December 2019.

[11] IEEE: "Use of the International Systems of Units (SI): The Modern Metric System," Doc. SI 10, Institute of Electrical and Electronics Engineers, New York, NY.

[12] IETF: "Augmented BNF for Syntax Specifications: ABNF," Doc. RFC 5234, Internet Engineering Task Force, January 2008. https://tools.ietf.org/htm/rfc5234

[13] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Authentication," Doc. RFC 7235, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7235

[14] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Caching," Doc. RFC 7234, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7234

[15] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Conditional Requests," Doc. RFC 7232, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7232

[16] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Message Syntax and Routing," Doc. RFC 7230, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7230

[17] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Range

Requests," Doc. RFC 7233, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7233

[18] IETF: "Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content," Doc. RFC 7231, Internet Engineering Task Force, June 2014. https://tools.ietf.org/html/rfc7231

[19] IETF Internet-Draft: "JSON Schema: A Media Type for Describing JSON Documents", September 16, 2019. Work in Progress. https://tools.ietf.org/html/draft-handrews-json-schema-02

[20] IETF: "Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies," RFC 2045, Internet Engineering Task Force, November 1996. https://tools.ietf.org/html/rfc2045

[21] IETF: BCP 47, "Tags for Identifying Languages," Internet Engineering Task Force, Reston, VA, September 2009. https://tools.ietf.org/html/bcp47

[22] IETF: "The JavaScript Object Notation (JSON) Data Interchange Format," RFC 7159, Internet Engineering Task Force, March 2014. https://tools.ietf.org/html/rfc7159

[23] IETF: "The Web Origin Concept," RFC 6454, Internet Engineering Task Force, December 2011. https://tools.ietf.org/html/rfc6454

[24] IETF: "The WebSocket Protocol," RFC 6455, Internet Engineering Task Force, December 2011. https://tools.ietf.org/html/rfc6455

[25] IETF: "Uniform Resource Identifier (URI): Generic Syntax," RFC 3986, Internet Engineering Task Force, January 2005. https://tools.ietf.org/html/rfc3986

[26] IETF: "A Universally Unique IDentifier (UUID) URN Namespace," Doc. RFC 4122, Internet Engineering Task Force, July 2005. https://tools.ietf.org/html/rfc4122

[27] IETF: "The Base16, Base32, and Base64 Data Encodings," RFC 4648, Internet Engineering Task Force, October 2006. https://tools.ietf.org/html/rfc4648

[28] IETF: "The "data" URL scheme," RFC 2397, Internet Engineering Task Force, August 1998. https://tools.ietf.org/html/rfc2397

[29] ISO/IEC: ISO/IEC 23009-1:2014, "Information technology — Dynamic adaptive streaming over HTTP (DASH) — Part 1: Media presentation description and segment formats,"

International Organization for Standardization, 15 May 2014.
[30] ISO/IEC: “Information technology – High efficiency coding and media delivery in heterogeneous environments – Part 1: MPEG media transport (MMT),” Doc. ISO/IEC 23008-1:2017(E), International Organization for Standardization / International Electrotechnical Commission, Geneva, Switzerland.
[31] W3C: "Encrypted Media Extensions," W3C Recommendation, World Wide Web Consortium, 18 September 2017. http://www.w3.org/TR/encrypted-media/
[32] W3C: "UI Events KeyboardEvent key Values," Section 3.18, Media Controller Keys, W3C Candidate Recommendation, 1 June 2017, World Wide Web Consortium. https://www.w3.org/TR/DOM-Level-3-Events-key/#keys-media-controller
[33] W3C: "Media Source Extensions," W3C Recommendation, World Wide Web Consortium, 17 November 2016. https://www.w3.org/TR/media-source/
[34] W3C: "Mixed Content," W3C Candidate Recommendation, Worldwide Web Consortium, 2 August 2016. (work in process). http://www.w3.org/TR/mixed-content/
[35] W3C: "XML Schema Part 2: Datatypes Second Edition," W3C Recommendation, Worldwide Web Consortium, 28 October 2004. https://www.w3.org/TR/xmlschema-2/
[36] WHATWG: "Living Standard", "Fetch Commit Snapshot", 30 November 2020: https://fetch.spec.whatwg.org/commit-snapshots/eda41525e3b462ce2035dd3cfc4a6ec1fc093c1d/
[37] ATSC: "ATSC Standard: Companion Device, " Doc. A/338:2024-04, Advanced Television Systems Committee, 3 April 2024.
[38] ATSC: "ATSC Recommended Practice: Techniques for Signaling, Delivery and Synchronization," Doc. A/351:2024-04, Advanced Television Systems Committee, 3 April 2024.
[39] ATSC: "ATSC Recommended Practice: Digital Rights Management (DRM)," Doc.A/362:2024-04, Advanced Television Systems Committee, 3 April 2024.
[40] CTA: "Recommended Practice for ATSC Television Sets, Application Runtime Environment (CTA-CEB32.8-C)," December 2024, https://shop.cta.tech/collections/standards/products/reco

mmended-practice-for-atsc-3-0-television-sets-application-runtime-environment-cta-ceb32-8-c

[41] DASH-IF: "Guidelines for Implementation: DASH-IF Interoperability Point for ATSC 3.0," Version 1.1, DASH Industry Forum, 12 June 2018. https://dashif.org/docs/DASH-IF-IOP-for-ATSC3-0-v1.1.pdf

[42] DASH-IF: "Implementation Guidelines: DASH events and timed metadata tracks timing and processing model and client reference model," DASH Industry Forum. https://dashif.org/docs/EventTimedMetadataProcessing-v1.0.2.pdf

[43] DASH-IF: "Protection System-Specific Identifiers," DASH Industry Forum. https://dashif.org/identifiers/content_protection/

[44] IANA Registry: Uniform Resource Names (URN) Namespaces. https://www.iana.org/assignments/urn-namespaces/urn-namespaces.xml

[45] IEEE: IEEE Registration Authority. https://regauth.standards.ieee.org/standards-ra-web/pub/view.html

[46] JSON-RPC: "JSON-RPC 2.0 Specification," JSON-RPC Working Group. http://www.jsonrpc.org/specification

[47] JSON Schema: "JSON Schema: A Media Type for Describing JSON Documents," Internet Engineering Task Force, JSON-Schema Working Group, 17 September 2019. http://json-schema.org/latest/json-schema-core.html (work in progress)

[48] W3C: "TTML Profiles for Internet Media Subtitles and Captions 1.0.1 (IMSC1)," W3C Recommendation, Worldwide Web Consortium. http://www.w3.org/TR/ttml-imsc1.0.1

[49] W3C: "XML Linking Language (XLink)," Recommendation Version 1.1, Worldwide Web Consortium, 6 May 2010. http://www.w3.org/TR/xlink11/

[50] WHATWG: "HTML Living Standard," Section 9.3 "Web sockets," Web Hypertext Application Technology Working Group. https://html.spec.whatwg.org/multipage/web-sockets.html

[51] J. Keiser, D. Lemire, "Validating UTF-8 In Less Than One Instruction Per Byte," Software: Practice and Experience, Vol. 51, No. 5, October 2020. https://arxiv.org/abs/2010.03090

[52] Android Media TV onSignalStrengthUpdated https://developer.android.com/reference/android/media/tv/TvView.TvInputCallback#onSignalStrengthUpdated(java.lang.String,%20int) ATSC: "ATSC Standard: Interactive Content A/344:2025-06

3 용어 정의 및 약어

용어, 약어 및 단위의 정의와 관련하여 IEEE(Institute of Electrical and Electronics Engineers)가 발표한 표준에 설명된 대로 IEEE(Institute of Electrical and Electronics Engineers)의 관행을 사용해야 합니다. 약어가 IEEE 관행에서 다루지 않거나 업계 관행이 IEEE 관행과 다른 경우 해당 약어는 이 문서의 섹션 3.3 에 설명되어 있습니다.

3.1 약어

ABNF	Augmented Backus-Naur Form
AEA	Advanced Emergency Alert
AEAT	Advanced Emergency Alert Table [3]
AMP	Application Media Player
API	Application Programming Interface
ATSC	Advanced Television Systems Committee
A/V	Audio/Video
BA	Broadcaster Application
BCP	Best Current Practice
CD	Companion Device [37]
CDM	Content Decryption Module
CORS	Cross-Origin Resource Sharing
CSS	Cascading Style Sheets
CTA	Consumer Technology Association
DASH	Dynamic Adaptive Streaming over HTTP [29]
dB	Decibels
DOM	Document Object Model
DRM	Digital Rights Management
DWD	Distribution Window Description [3]
EFDT	Extended File Delivery Table
EME	W3C Encrypted Media Extensions [31]
ESG	Electronic Service Guide [4]
FDT	File Delivery Table
FLUTE	File Delivery over Unidirectional Transport
GIF	Graphics Interchange Format

HDMI	High Definition Multimedia Interface
HELD	HTML Entry pages Location Description [3]
HTML5	HyperText Markup Language, Fifth Version
HTTP	HyperText Transfer Protocol
HTTPS	HyperText Transfer Protocol Secure
IANA	Internet Assigned Numbers Authority
ID	Identifier
IMSC1	Internet Media Subtitles and Captions 1.0 [48]
IP	Internet Protocol
JPEG	Joint Photographic Experts Group
JSON	JavaScript Object Notation [22]
JSON-RPC	JSON Remote Procedure Call (Annex A)
LCT	Layered Coding Transport
LLS	Low-Level Signaling [3]
MIME	Multipurpose Internet Mail Extensions
MMT	MPEG Media Transport
MPD	Media Presentation Description [41]
MPEG	Moving Pictures Experts Group
MPU	Media Processing Unit
MSE	W3C Media Source Extensions [33]
msec	Milliseconds
NRT	Non-Real Time
OSN	On Screen message Notification [3]
PD	Primary Device [37]
PNG	Portable Network Graphics
PTP	Precision Time Protocol
RDT	Recovery Data Table [5]
RFC	Request For Comment
RMP	Receiver Media Player
ROUTE	Real-Time Object Delivery over Unidirectional Transport [3]
RPC	Remote Procedure Call
RRM	Reference Receiver Model
SHA1	Secure Hash Algorithm 1
SLS	Service-Level Signaling [3]
SLT	Service List Table [3]
S/MIME	Secure/Multipurpose Internet Mail Extensions [8]
sRGB	Standard Red Green Blue
STB	Set-Top Box
TV	Television

UI	User Interface
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Universal Resource Name
UTC	Universal Time Coordinated
UTF-8	Unicode Transformation Format - 8-bit
UUID	Universally Unique Identifier
VDS	Video Description Service
W3C	Worldwide Web Consortium
WHATWG	Web Hypertext Application Technology Working Group
WS	WebSocket [50]
XHR	XMLHttpRequest
XLink	XML Linking Language
XML	eXtensible Markup Language

3.2 용어

Application Context Cache: **Application Context Cache** 는 **Receiver Web Server** 를 통해 검색하기 위해 브로드캐스트의 정보가 수집되는 개념적 저장 영역입니다. 이 문서에서는 편의를 위한 것이지만 실제 스토리지로 구현된 것처럼 **Application Context Cache** 를 참조합니다. **Application Context Cache** 는 각 **Broadcaster Application** 과 연결된 **Application Context Identifier** 에 해당합니다. ROUTE 를 통해 전달되는 파일에는 연관된 **Application Context Cache** 를 결정하는 속성이 포함되어 있습니다.

Application Context Identifier: **Application Context Identifier** 는 **Receiver** 가 연결된 **Broadcaster Application** 에 제공하는 리소스를 결정하는 고유한 URI 입니다. 리소스는 여러 **Application Context Identifier** 와 연결될 수 있지만 **Broadcaster Application** 은 단일 **Application Context Identifier** 와만 연결됩니다. **Application Context Identifier** 구문의 세부사항은 HELD 에 정의되어 있음 [3].

Base URI: RFC 3986 [25]에 정의된 대로 **Base URI** 는 **Broadcaster Application** 이 **Application Context Cache** 내의 파일에 액세스하는 데 사용하는 URL 의 초기 부분을 지정합니다. **Base URI** 는 **Application Context Cache** 내에서 파일의 전체 URL 을 가져오기 위해 파일의 상대 URI 경로 앞에 추가됩니다. **Base URI** 는 **Broadcaster Application** 에 대해 정의된 **Application Context Identifier** 를 기반으로 **Receiver** 에

의해 고유하게 생성됩니다.

Broadcaster Application: 본 문서에서는 **Broadcaster Application**이 **Entry Page**로 알려진 HTML5 문서 및 해당 문서에 의해 직간접적으로 참조되는 기타 HTML5, CSS, JavaScript, 이미지 및 멀티미디어 리소스로 구성된 파일 모음에 구현된 기능을 지칭하기 위해 사용되며, 모두 ATSC 3.0 서비스에서 브로드캐스터에 의해 제공된다. **Broadcaster Application**은 대화형 서비스를 제공하는 광범위한 웹 애플리케이션의 클라이언트 측 기능을 나타냅니다. 브로드캐스터는 클라이언트 측 문서와 코드만 전송하기 때문에 구별됩니다. 이 광범위한 **Web Application**의 서버 측은 ATSC 3.0 **Receiver**에 의해 구현되며 모든 애플리케이션에 대해 표준화된 API를 가지고 있습니다. 브로드캐스터는 서버 측 애플리케이션 코드를 제공할 수 없습니다. 브로드캐스터는 **Broadcaster Application**을 통해 **Broadcaster Application**과 함께 작동하는 웹 기반 문서 및 코드를 제공하여 **Broadcaster Application**을진정한 **Web Application**으로 만들 수 있습니다. **Broadcaster Application**을 구성하는 파일 모음은 표준 방식으로 웹을 통해 전달되거나 ROUTE 프로토콜을 통해 패키지로 방송망을 통해 전달될 수 있습니다.

Entry Package: **Entry Package**에는 **Broadcaster Application**의 기능을 구성하는 하나 이상의 파일이 포함되어 있습니다. **Entry Package**에는 **Entry Page**와 JavaScript, CSS, 이미지 파일 및 기타 콘텐츠를 포함한 추가 지원 파일이 포함됩니다.

Entry Page: **Entry Page**는 먼저 **User Agent**에 로드해야 하는 애플리케이션 시그널링에서 참조하는 초기 HTML5 문서입니다. **Entry Page**는 **Entry Package**의 파일 중 하나입니다.

Event Stream: **Event Stream**은 DASH 시그널링, MMT 시그널링 또는 동적 메시지의 일련의 메시지로, 미디어 세그먼트 내에 정의된 메시지에 포함됩니다. **Event Stream**에 포함된 이벤트는 **Broadcaster Application** 측에서 대화형 작업을 시작할 수 있습니다.

Filter Code: A/331[3]의 정의를 참조하십시오. 파일에 정의된 **Filter Code**를 **Application Context Cache**에 설정된 **Filter Code Instance**와 비교하여 파일을 캐시에 저장할 수 있는지 여부를 결정합니다.

Filter Code Instance: **Filter Code Instance**는 **Broadcaster Application**이 **Receiver**가 연결된 **Application Context Cache**에

저장하는 NRT 데이터를 제어하는 방법을 제공합니다. Filter Code Instance는 Filter Code 값 및 연결된 만료 시간이 있는 Application Context Cache와 연결된 데이터 구조입니다. Filter Code Instance는 만료 시간이 충족될 때까지 Application Context Cache와 함께 유지됩니다. Filter Code Instance 만료는 명시적으로 설정하거나 그렇지 않은 경우 Broadcaster Application의 수명으로 기본값으로 설정할 수 있습니다.

MMT Asset: MMT Asset 또는 Asset은 'mmpu' 상자에 제공된 동일한 Asset ID를 가진 하나 이상의 MPU 모음입니다. [30].

MMT-Asset File: MMT-Asset File은 일련의 MMTP 패킷으로 구성되며, 각 패킷에는 다양한 미디어 구성 요소에 대한 MPU를 구성하는 헤더와 페이로드가 포함되어 있습니다. 각 구성 요소(비디오, 오디오, 자막 등)는 인코딩되고 바이너리 데이터 구조로 패킷화된 다음 MMTP 패킷 내에서 함께 다중화됩니다 [30].

Receiver: 이 문서에서의 Receiver는 Reference Receiver Model의 기능을 구현하는 엔터티를 나타냅니다.

Receiver Web Server: Receiver Web Server는 User Agent가 개념적으로 Application Context Cache에 상주하는 ROUTE를 통해 전달된 파일에 액세스할 수 있는 수단을 제공하는 수신기의 개념적 구성 요소입니다.

Receiver WebSocket Server: Receiver WebSocket Server는 User Agent가 Receiver에 대한 정보에 액세스하고 Receiver가 제공하는 다양한 기능을 제어할 수 있는 수단을 제공합니다.

Redistribution: ATSC 3.0 서비스가 ATSC 3.0 이외의 프로토콜을 통해 Receiver에게 전달되는 사용 사례; 예를 들어 HDMI.

Reference Receiver Model: 이 문서에 지정된 API 및 동작을 실행할 수 있는 개념적 수신기 디바이스입니다. 이 문서는 실제 수신기 구현을 알리기 위한 모델의 규범적 속성을 지정합니다.

Reserved: 나중에 표준에서 사용할 수 있도록 따로 보관합니다.

User Agent: 웹 콘텐츠를 검색하고 렌더링하는 수신자가 제공하는 소프트웨어입니다. User Agent는 HTML5, CSS 및 JavaScript를

해석하고, 미디어, 텍스트 및 그래픽을 렌더링하고, 사용자 상호 작용 대화 상자를 만들 수 있습니다.

Web Application: **Web Application** 은 URL 을 사용하여 웹을 통해 액세스되는 클라이언트/서버 프로그램입니다. 클라이언트 측 소프트웨어는 **User Agent** 에 의해 실행됩니다

4 OVERVIEW

4.1 Application Runtime Environment

이 표준은 **Broadcaster Application** 을 실행하는 데 필요한 환경의 세부 정보를 정의합니다. 브로드캐스트 환경에서 **Broadcaster Application** 과 연결된 파일은 개념적 캐시 영역으로 압축을 푼 ROUTE 패키지로 전달됩니다. 그러면 **Broadcaster Application** 의 페이지와 리소스가 **Receiver** 와 연결된 **User Agent** 에서 사용할 수 있게 됩니다. 광대역 환경에서 **Broadcaster Application** 을 시작하는 것은 **Receiver** 의 특수한 동작이나 개입 없이 일반 웹 환경에서와 동일한 방식으로 작동합니다.

Broadcaster Application 은 W3C 호환 **User Agent** 내에서 실행되어 **Receiver** 의 일부 그래픽 요소에 액세스하여 사용자 인터페이스를 렌더링하거나 **Receiver** 가 제공하는 일부 리소스 또는 정보에 액세스합니다. **Broadcaster Application** 이 **Receiver** 에게 알려진 정보와 같은 리소스에 액세스해야 하거나 **Broadcaster Application** 이 브라우저에서 널리 구현되는 표준 W3C **User Agent** API 에 의해 정의되지 않은 특정 작업을 수신자에게 수행하도록 요구하는 경우, **Broadcaster Application** 은 이 사양에 정의된 JSON-RPC 메시지 세트를 사용하여 **Receiver WebSocket Server** 에 요청을 보냅니다.

이 표준에 정의된 JSON-RPC 메시지는 **Broadcaster Application** 이 다른 방법으로는 연결할 수 없는 리소스에 액세스하는 데 필요한 API 를 제공합니다. 이러한 JSON-RPC 메시지를 통해 **Broadcaster Application** 은 **Receiver** 에서 수집되거나 수집된 정보를 쿼리하고, 브로드캐스트 신호를 통해 알림을 받고, 표준 JavaScript API 를 통해 사용할 수 없는 작업 수행을 요청할 수 있습니다.

일반 웹 환경에 배포된 HTML5 응용 프로그램과 ATSC 3.0 브로드캐스트 환경에 배포된 응용 프로그램 간에는 주목할만한 차이점이 있습니다. ATSC 3.0 브로드캐스트 환경에서 **Broadcaster Application** 은 다음을 수행할 수 있습니다.

- 브로드캐스트 또는 브로드밴드에서 리소스에 접근할 수 있음.
- 다음과 같이 JavaScript API 를 통해 사용할 수 없는 특정 기능을 수행하도록 **Receiver** 에게 요청합니다.
  - **Receiver** 에서 제공하는 미디어 플레이어(이하 **Receiver Media Player**)를 활용하여 다음을 수행합니다.

- ◆ 브로드캐스트 시그널링 전달 메커니즘을 통해 미디어 콘텐츠 스트리밍
- ◆ 브로드밴드 전송 메커니즘을 통해 미디어 콘텐츠(즉, 유니캐스트) 스트리밍
- ◆ 브로드캐스트 또는 광대역 전달 메커니즘을 통해 다운로드한 미디어 콘텐츠 재생

■ MSE 및 EME 를 활용하여 방송 또는 광대역을 통해 스트리밍되는 미디어 콘텐츠를 재생합니다.

- TV 서비스의 수신에 특정한 정보, 예를 들어, 자막 표시 상태 및 언어 참조를 조회하고, 이 정보의 변경 사항에 대한 통지를 수신하는 단계;
- 미디어 콘텐츠 또는 신호에 포함된 "스트림 이벤트"에 대한 알림을 수신합니다.

두 모델의 또 다른 주목할만한 차이점은 일반 웹 환경에서 뷰어가 원하는 웹 사이트의 URL 을 지정하여 HTML5 응용 프로그램 실행을 직접 제어할 수 있다는 것입니다. ATSC 3.0 환경에서는 사용자가 서비스를 선택하여 작업을 시작하지만 실제 애플리케이션 URL 은 뷰어가 명시적으로 선택하지 않고 대신 브로드캐스트 시그널링을 통해 제공됩니다. 이 경우, 브로드캐스트 시그널링에 제공된 URL 에 의해 참조되는 Broadcaster Application 을 시작하거나 종료하는 것은 사용자 에이전트를 사용하는 수신자의 책임입니다.

Broadcaster Application 은 User Agent 를 통해 제공되는 일련의 기능에 의존합니다. Broadcaster Application 의 페이지가 User Agent 에 제공되는 방법을 설명하는 것은 이 사양의 범위를 벗어나지만 페이지를 제공하기 위해 표준 웹 기술을 사용하는 것이 좋습니다.

표 4.1-1 은 브로드캐스터 제공 애플리케이션이 Receiver 가 제공하는 기능에 액세스하는 데 사용하는 API 유형을 보여줍니다.

<표 4.1-1> Application Actions and APIs
[출처: A344]

애플리케이션에서 요청한 조치	애플리케이션에서 사용하는 API
브로드밴드에서 미디어 파일 다운로드 요청	Receiver WebSocket Server API, 본 표준의 섹션 9.2 에 설명
언어, 접근성 옵션, 폐쇄 자막 설정을 포함하여, 사용자 디스플레이 및 표시 환경 관련 선호 정보를 조회	섹션 9.2 및 9.6.2 의 이 사양에 설명된 푸시 또는 풀 모델을 통해 설명
방송으로부터 다운로드된 미디어 파일의 스트리밍을 요청	섹션 9.2 및 9.6.2 의 이 사양에 설명된 푸시 또는 풀 모델을 통해 설명
브로드밴드를 통해 전송되는 미디어 스트림을 재생하도록 Receiver Media Player 요청	Receiver WebSocket Server API, 본 표준의 섹션 9.7.3 에 설명

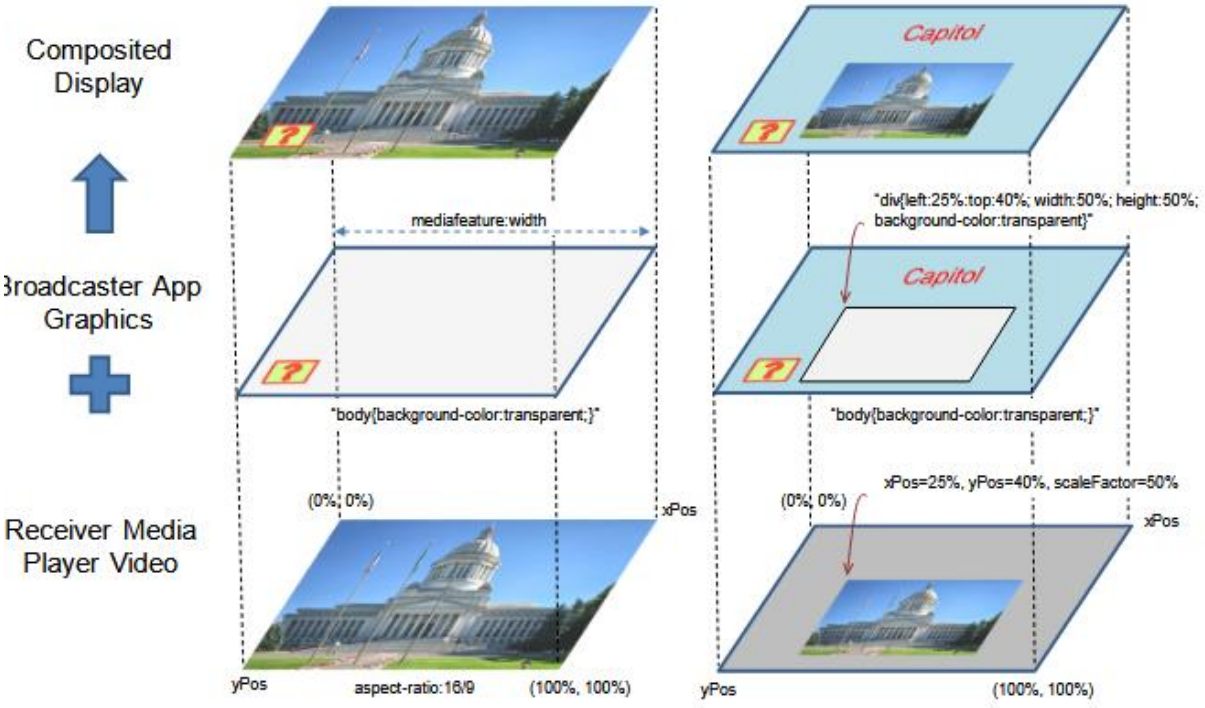
방송망을 통해 전송되는 스트림 이벤트 알림에 대한 구독(또는 구독 취소)	Receiver WebSocket Server API, 본 표준의 섹션 9.6.1 과 9.6.2 에 설명
방송망을 통해 전송되는 스트림 이벤트 알림 수신	Receiver WebSocket Server API, 본 표준의 섹션 9.6.3 에 설명
현재 선택한 브로드캐스트 서비스의 ID 를 학습하기 위해 Receiver 를 쿼리합니다.	Receiver WebSocket Server API, 본 표준의 섹션 9.2.3 에 설명
사용자 표시 및 프레젠테이션 기본 설정에 대한 변경 사항에 대한 알림 수신	Receiver WebSocket Server API, 본 표준의 섹션 9.3.6 에 설명
수신자에게 새 브로드캐스트 서비스 선택 요청	Receiver WebSocket Server API, 본 표준의 섹션 9.7.1 에 설명

4.2 Receiver Media Player Display

4.2.1 Rendering Model

RMP 는 Broadcaster Application 의 보이는 출력 뒤에 비디오 출력을 표시합니다. 그림 4.2.1-1 은 수신기에서 수행되는 관계 및 구성 기능을 보여줍니다.

그림 4.2.1-1 은 두 가지 예를 보여줍니다. 왼쪽 예제에서 Broadcaster Application 의 그래픽 출력은 Receiver Media Player 에서 렌더링되는 전체 화면 비디오에 오버레이됩니다. 응용 프로그램 향상이 있는 선형 A/V 서비스의 경우, Broadcaster Application 은 그래픽에 더 많은 영역을 사용하기를 원할 수 있으므로 Receiver Media Player 에 비디오 크기를 조정하도록 지시할 수 있습니다. 섹션 9.7.2 에 설명된 JSON-RPC 메시지는 RMP 가 렌더링하는 비디오의 크기를 조정하고 배치하도록 지시하는 데 사용됩니다. 이 시나리오는 그림 오른쪽에 표시된 예에 설명되어 있습니다. Broadcaster Application 은 비디오 삽입을 둘러싼 화면의 모양을 정의할 수 있습니다. RMP 비디오가 배치되는 직사각형 영역이 투명으로 지정되는 방식으로 배경을 정의하여 이를 수행할 수 있습니다.



(그림 4.2.1-1) RMP 를 사용한 애플리케이션 향상을 위한 렌더링 모델.
[출처: A344]

Broadcaster Application 은 User Agent 의 그래픽 창([0,0]에서 두 축 모두에서 전체 100%까지)이 전체 차원에서 RMP 논리 비디오 디스플레이 창에 직접 매핑될 것으로 예상할 수 있습니다. 대부분의 Receiver 의 사용자 인터페이스는 스크롤 막대 조작을 편리하게 사용하도록 설정하지 않을 수 있으므로 Broadcaster Application 은 대부분의 상황에서 표준 W3C 메커니즘을 사용하여 스크롤 막대를 사용하지 않도록 설정하는 것을 고려해야 합니다.

Receiver 는 일부 사용자 상호 작용 또는 기타 유사한 이벤트로 인해 Broadcaster Application 위에 자체 네이티브 애플리케이션을 렌더링하도록 선택할 수 있습니다. 예를 들어, 이는 시청자가 Broadcaster Application 활성화된 동안 Receiver 설정을 구성하도록 선택할 때 발생할 수 있습니다.

Receiver 가 자체 네이티브 애플리케이션을 제공할 때 Receiver 는 표준 W3C 알림 메서드를 통해 Broadcaster Application 에 더 이상 포커스(이벤트)가 없음을 알려야 합니다. Broadcaster Application 은 자신을 숨기거나 현재 표시를 유지하도록 선택할 수 있습니다. 이 동작은 각 Broadcaster Application 의 설계에 달려 있습니다.

또한 수신기는 비디오 스케일링 및 본격적인 Broadcaster Application 과 관련된 문제를 방지하기 위해 실행된 Broadcaster Application 을 숨기도록 선택할 수 있습니다. Broadcaster Application 이 숨겨져 있는지 여부에 대한 동작은 Receiver 에 맡겨져 있지만, 연결된 서비스가 선택된 상태로 유지되고 애플리케이션 시그널링이 다른 Broadcaster Application 을 선택하지 않는 한 수신기는 Broadcaster Application 을종료할 것으로 예상되지 않습니다.

Broadcaster Application 이 숨겨져 있는지 또는 Receiver 네이티브 애플리케이션 뒤에 있는지 여부에 관계없이 Broadcaster Application은 표준 W3C 알림 방법을 통해 포커스(이벤트)를 잃었다는 알림을 받습니다.

4.2.2 Closed Captioning

자막은 모든 비디오 및 Broadcaster Application 콘텐츠 위에 렌더링될 수 있다.

최악의 경우 표시되는 캡션이 불투명할 수 있으며 종료 버튼과 같은 Broadcaster Application의 중요한 요소를 다룰 수 있습니다. Broadcaster Application 이 캡션에 의해 가려지는 것을 완화할 수 있도록 세 가지 API가 제공됩니다. 이것들은:

- Query Display Components API (Section 9.1.12)
- Video Scaling and Positioning API (Section 9.6.2)
- Graphics Display Regions API (Section 9.6.8)

Query Display Components API 는 두 가지 기능을 제공합니다. 첫째, Receiver가 비디오 및 캡션 스케일링과 관련된 수신기의 기능을 Broadcaster Application 에 알리는 데 사용할 수 있습니다. 예를 들어 L-막대 레이아웃 시나리오를 생각해 보십시오. Receiver가 비디오 및 캡션 크기 조정을 지원하는 경우 캡션은 비디오 상단에만 존재해야 하며 L-막대는 가려지지 않습니다. 반면에 비디오 및 캡션 크기 조정이 지원되지 않는 경우 Receiver은 캡션이 있는 경우 캡션에 의해 가려지지 않도록 조치를 취해야 할 수 있습니다.

Query Display Components API의 두 번째 기능은 수신기가 캡션에 사용 중인 현재 영역을 Broadcaster Application에 알리는 데 사용할 수 있다는 것입니다. Broadcaster Application은 이 정보를 사용하여 캡션이 차지하는 디스플레이 영역을 피할 수 있습니다. 이 기능은 자막이 활성화된 경우 클릭 유도 문구 프롬프트가 표시되기 직전에 사용하기 위한 것입니다. (사용자가 아무런 조치도 취하지 않고도 Broadcaster Application의 존재를 사용자에게 알리기 위해 제한된 시간 동안 클릭 유도 문안 프롬프트를 Broadcaster Application 에 의해 표시될 수 있습니다.) 캡션 영역의 크기와 위치로 인해 표시 영역 충돌이 발생하거나 Broadcaster Application 그래픽의 레이아웃이 중단될 수 있으므로 Broadcaster Application 을 정상적이고 장기적으로 프레젠테이션하는 동안 사용할 수 없습니다.

The Video Scaling and Positioning API (section 9.6.2)에는 Receiver 에서 지원하는 최소 배율에 대한 선택적 정보가 응답에 포함되어 있습니다. Receiver는 캡션이 너무 작아서 합리적으로 읽을 수 없도록 하기 위해 캡션이 있을 때 상대적으로 큰 최소 배율 인수를 설정할 수 있습니다. API가 오류를 반환하는 경우 Broadcaster Application은 배율 인수와 x 및 y 위치가 범위 내에 있는지 확인하고 배율을 다시 시도할 수 있습니다.

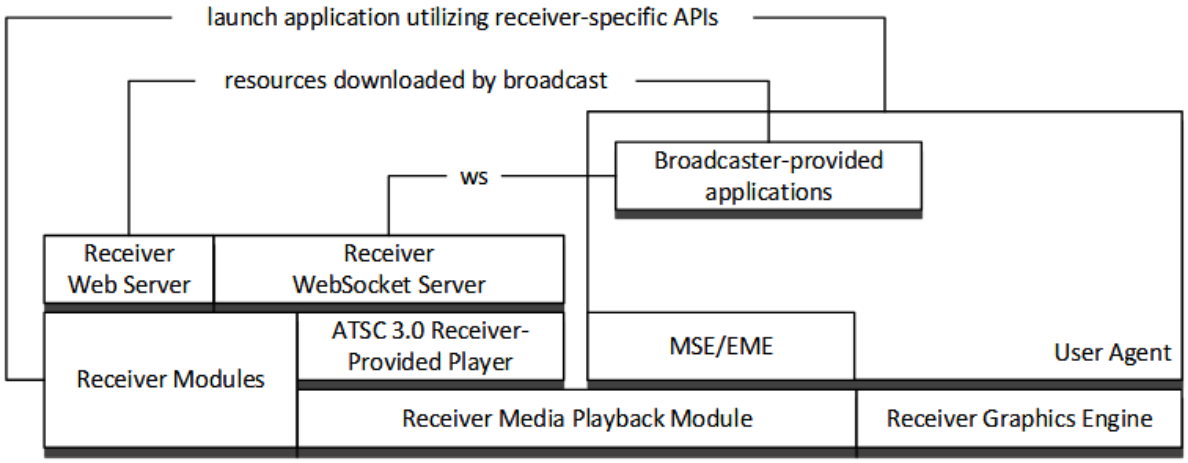
Graphics Display Regions API 는 Broadcaster Application 에서

수신자에게 Broadcaster Application의 그래픽 컨텐츠가 포함된 디스플레이 영역을 알리는 데 사용할 수 있습니다. Receiver는 잠재적으로 이 정보를 사용하여 Broadcaster Application 그래픽을 가리지 않도록 캡션의 위치를 변경할 수 있습니다.

5 ATSC REFERENCE RECEIVER MODEL

5.1 Introduction

ATSC 3.0 Reference Receiver Model은 이 섹션에 설명된 여러 논리적 구성 요소로 구성될 수 있습니다. 실제로 주어진 논리 구성 요소 중 여러 개를 하나의 구성 요소로 결합하거나 하나의 논리 구성 요소를 여러 구성 요소로 나눌 수 있습니다. 그림 5.1-1은 ATSC 3.0 Reference Receiver Model의 논리적 구성 요소를 보여줍니다. 소프트웨어 스택은 계층화 아키텍처를 보여주지만, 이 사양에 지정된 API를 준수하는 Receiver가 제공하는 User Agent 구현에서 실행되는 Broadcaster Application을 제외하고 한 모듈이 시스템의 다른 모듈에 액세스하기 위해 아래 계층을 사용해야 한다는 의미는 아닙니다.



(그림 5.1-1) ATSC 3.0 Reference Receiver Model 논리적 구성 요소.

[출처: A344]

5.2 User Agent Definition

Receiver는 CTA Web Media API Snapshot (CTA-5000-G)[9]에 지정된 모든 규범 요구 사항을 준수하는 HTML5 사용자 에이전트를 구현해야 합니다. 또한 다음 섹션에 설명된 기능이 지원될 것으로 예상됩니다

5.2.1 HTTP Protocols

User Agent는 RFC 7230에서 RFC 7235까지 지정된 HTTP 프로토콜을 구현해야 하며, 참조 [13], [14], [15], [16], [17] 및 [18]을 참조합니다. User Agent는 Web Origin Concept 표준[23]과 HTTP State Management

Mechanism 표준(Cookies)([9] 섹션 4.2)도 구현해야 합니다.

5.2.2 XMLHttpRequest (XHR)

User Agent는 [9]의 [XHR] 참조의 XMLHttpRequest 및 관련 인터페이스를 지원해야 합니다. 요청 URL이 브로드캐스트 리소스를 식별하는 XHR 요청의 경우, 요청은 인터넷 웹 서버가 아닌 Receiver Web Server로 전달됩니다.

5.2.3 Cross-Origin Resource Sharing (CORS)

User Agent는 WHATWG Fetch [36]에 정의된 대로 Cross-Origin Resource Sharing을 지원해야 합니다.

5.2.4 Mixed Content

User Agent는 W3C Mixed Content 표준[34]에 따라 암호화 및 인증된 문서의 컨텍스트에서 암호화되지 않거나 인증되지 않은 연결을 통해 콘텐츠 가져오기를 처리해야 하지만 Broadcaster Application은 신뢰할 수 있는 콘텐츠만 참조하는 것이 좋습니다. Application Context Cache 내의 파일에 대한 참조(아래 섹션 5.3 참조)는 W3C 혼합 콘텐츠 용어에서 " priori authenticated"으로 간주됩니다. Application Context Cache에서 액세스한 모든 리소스는 보안 컨텍스트 내에서 액세스된 것으로 간주됩니다.

5.2.5 Transparency

User Agent의 그리기 창의 배경은 기본적으로 투명할 수 있습니다. 그럼에도 불구하고 Broadcaster Applications은 Receivers 간에 일관성을 유지하기 위해 불투명하거나 투명해야 하는 영역을 명시적으로 지정하는 것이 좋습니다. 따라서 예를 들어 웹 페이지의 요소(예: 테이블 셀)에 CSS 스타일 속성 "background-color: transparent"가 포함되어 있고 해당 영역이 불투명 요소가 있는 다른 레이어로 덮여 있지 않은 경우 Receiver Media Player에서 제공하는 비디오 콘텐츠(섹션 4.2 참조)가 해당 영역에 표시될 수 있습니다. 특정 영역은 투명으로 지정하고 다른 영역은 불투명으로 지정할 수 있습니다.

5.2.6 Full Screen

Section 4.2에서 언급한 바와 같이, 수신기는 User Agent의 그래픽 창 [0,0]을 전체 치수의 RMP 논리적 비디오 디스플레이 창에 직접 양쪽 축에서 전체 100%로 매핑해야 합니다. CSS MediaQueries [9]의 "너비" 미디어 기능은 RMP 논리 비디오 디스플레이 창의 너비와 일치해야 합니다. 대부분의 보기 조건에서 RMP 논리적 비디오 표시 창은 전체 화면을 채울 것으로 예상됩니다.

5.2.7 Visibility and Focus

Receiver 는 CTA-5000-G[9]에서 요구하는 대로 W3C Page Visibility Level 2 API 를 사용하여 디스플레이 출력이 표시되는지 여부를 Broadcaster Application 에 알려야 합니다. Receiver 는 Receiver 기본 설정 대화 상자, 콘텐츠 차단 또는 기타 Receiver 정보 표시를 포함하되 이에 국한되지 않는 다양한 이유로 Broadcaster Application 의 디스플레이 출력을 가리거나 음소거하도록 선택할 수 있습니다. 마찬가지로, 수신기는 CTA-5000-G [UIEvents] [9]에서 요구하는 W3C Focus Events 와 표준 DOM activeElement 속성을 제공하여 Broadcaster Application 이 사용자 입력을 수신할 수 있는지 여부를 결정할 수 있도록 해야 합니다.

5.3 Application Context Identifier, Base URI and Cache Path

5.3.1 Application Context Identifier

Broadband 을 통해 전달되는 각 파일에는 일반적인 절대 URL 이 연결되어 있습니다. 방송망을 통해 전달되는 각 파일에는 브로드캐스트에서 신호를 보내는 상대 URL 참조가 연결되며, 브로드캐스트에서 신호를 보내는 하나 이상의 Application Context Identifiers 도 연결됩니다. 아래에 지정된 대로 Receivers 는 각 브로드캐스트 파일에 상대 URL 참조를 하나 이상의 절대 URL 로 변환하는 기본 URI 를 할당하고 해당 Application Context Identifiers 를 고려합니다.

Application Context Identifiers (appContextId)는 Receivers 가 연결된 실행 중인 Broadcaster Application 에 제공하는 리소스를 결정하는 고유한 URI 입니다. Broadcaster Application 에 바인딩될 애플리케이션 컨텍스트 식별자는 HELD [6]에서 신호를 받습니다. Application Context Identifiers 는 많은 Broadcaster Application 과 연결될 수 있으며, 동일한 Broadcaster Application 은 많은 Application Context Identifier 와 연결될 수 있습니다. 그러나 실행 중인 각 Broadcaster Applications 은 단일 Application Context Identifier 와 연결되어야 합니다. 각 Application Context Identifier 는 수신기가 연결된 Broadcaster Application 에서 사용할 리소스를 결합해야 하는 고유한 개념 환경을 형성합니다. 이 고유한 개념 환경을 여기서는 Application Context Cache 라고 합니다. 따라서 각 Application Context Identifier 는 Application Context Cache 를 고유하게 식별합니다.

Broadcaster Application 은 쿠키 및 기타 로컬 User Agent 스토리지 요소를 설정하기 위한 로컬 네임스페이스를 관리해야 합니다.

현재 Application Context Identifier 가 동일하게 유지되면 Entry Page 가 변경될 수 있더라도 연결된 모든 리소스는 Receiver Web Server 를 통해 Application Context Cache 내에서 계속 사용할 수 있습니다. Entry Page 변경은 섹션 6.3 에 설명된 대로 애플리케이션 시그널링에 의해 신호화됩니다.

Application Context Identifier 가 변경되면 Receiver 는 해당 Application Context Identifier 에 대해 이전에 생성된 Application Context Cache 를 재사용하거나 새 캐시를 생성할 수 있습니다. Receiver 는 이전 Application Context Cache 가 곧 필요할 수 있다는 가정 하에 다른 Application Context Identifier 를 가진 Broadcaster Applications 에 알려지지 않았지만 이전 Application Context Cache 를 유지하도록 선택할 수 있습니다. 또는 Receiver 는 Application Context Cache 와 연결된 리소스를 해제할 수 있습니다. 파일이 캐시되지 않은 경우 Receiver Web Server 는 파일의 다음 배달을 기다리거나 오류 코드로 요청에 응답할 수 있습니다. 파일 캐싱 결정은 전적으로 Receiver 구현에 맡겨집니다. 그러나 Application Context Cache 파일과 관련된 속성은 Receiver 캐싱 메커니즘에 우선 순위 정보를 제공하기 위한 것입니다(Section 6.2 참조).

appContextId 는 인터넷에서 확인할 수 있을 필요가 없습니다. 권한의 도메인 이름 루트 부분은 Broadcast Application 의 서명자 중 한 명(저자 또는 방송사/배포자)이 등록하고 통제해야 합니다.

appContextId 의 생성은 전역적으로 고유해야 하지만 Receivers 는 이를 불투명 문자열로 처리하고 모든 문자열 구문을 허용해야 합니다.

appContextId URI 의 예는 다음과 같습니다.

```
urn:uuid:f81d4fae-7dec-11d0-a765-00a0c91e6bf6
http://kids.pbs.org/app1
urn:tv:nbc.com
```

5.3.2 Origin Considerations

웹 리소스의 출처(origin)는 RFC 6454 에서 정의되어 있다. 기술적 설명은 다양한 엣지 케이스를 다루기 위해 복잡하지만, 생성된 URI 는 일반적으로 “스킴(scheme)” 부분(예: “http”)과 “권한(authority)”, 일반적으로 IP 주소 또는 호스트명(예: 10.2.12.45:8080)으로 구성되는 형태여야 한다.
방송 파일의 출처를 결정하는 URI 의 일부를 생성하기 위해 ATSC 3.0 Receiver 가 사용하는 알고리즘은 아래에 명시된 제약사항을 준수해야 한다
또한 참고로, 브로드밴드 소스(broadband source) 로부터 제공되는 리소스는 해당 리소스를 호스팅하는 웹 서버가 그 출처(origin)를 정의한다.

5.3.3 Base URI

Application Context Cache 의 기본 URI 는 9.3.14 절의 Cache Request APIs 를 통해 액세스되는 브로드캐스트 리소스, Broadcaster Application 리소스 및 광대역 리소스에만 사용됩니다. HTML5 *fetch()* [36]의 광대역 리소스

캐싱은 별개이며 **Receiver**의 **Web Server User Agent** 캐싱 정책의 적용을 받습니다.

Receiver는 전 세계적으로 통계적으로 고유한 방식으로 **Base URI**를 생성할 때 난독화 기능(예: SHA1)을 구현해야 합니다. 난독 처리 함수는 *appContextId*를 일부 디바이스별 정보와 결합하여 난독 처리 함수에 대한 지식만으로는 브로드캐스터 애플리케이션 또는 외부 엔터티가 다음 중 하나를 수행할 수 있도록 충분하지 않습니다.

- Base URI에서 *appContextId* 복구
- *appContextId*에서 동일한 Base URI 만들기
- 한 **Receiver**의 **Base URI**를 사용하여 다른 **Receiver**의 동일한 리소스에 대한 액세스를 제공합니다.

ROUTE EFDT 신호를 사용하여 **Broadcast** 리소스를 여러 캐시에서 공유할 수 있습니다. 위의 구성을 통해 브로드캐스트 또는 광대역 제공 **Broadcaster Application**은 여러 서비스를 통해 동일한 로컬 스토리지 정보에 액세스할 수 있습니다.

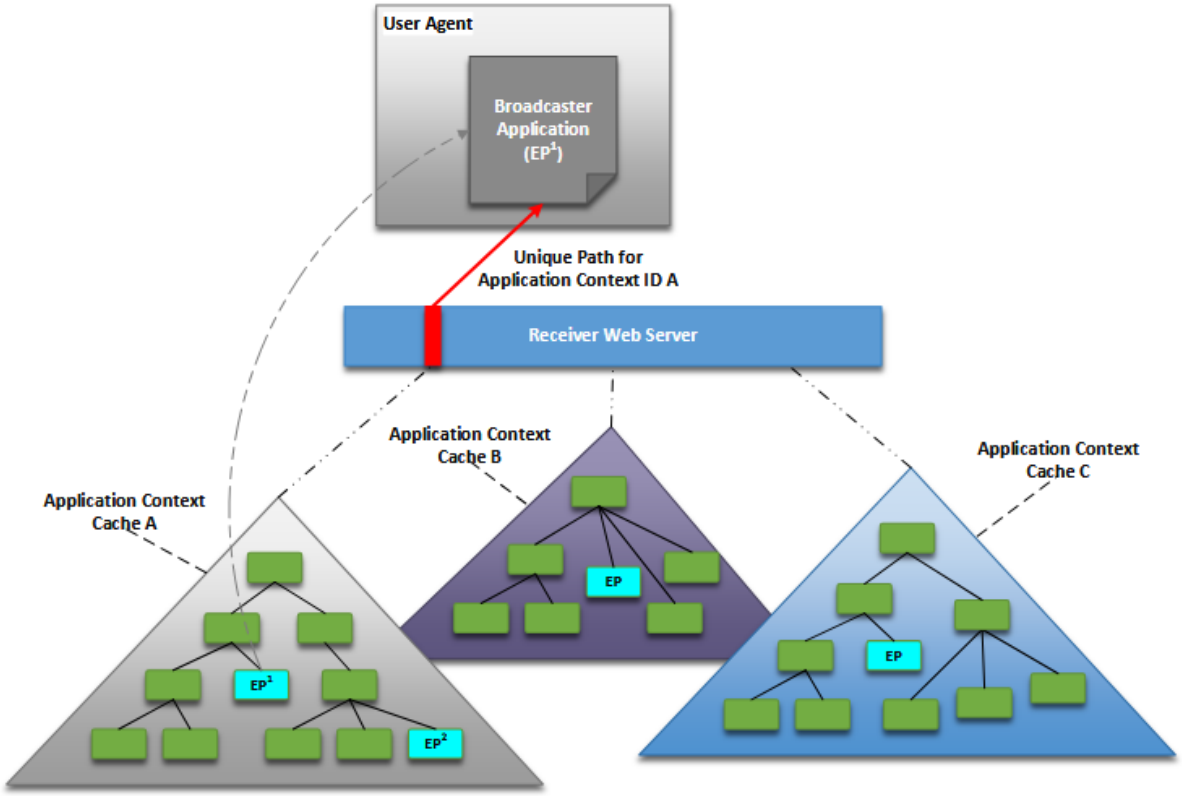
예를 들어 멀티파트 패키지에서는 다음을 수행합니다.

Service 1 contains:
EFDT.FDT-Instance.File@Content-Location="package1"
EFDT.FDT-Instance@afdt:appContextIdList="http://kids.pbs.org/app1"
다중 파트 경계가 있는 리소스가 있는 경우, "Content-Location: folder1/file1.txt".

Service 2 contains:
EFDT.FDT-Instance.File@Content-Location="package2"
EFDT.FDT-Instance@afdt:appContextIdList="http://kids.pbs.org/app2"
다중 파트 경계가 있는 리소스가 있는 경우, "Content-Location: folder1/file2.txt".

Service 3 contains:
EFDT.FDT-Instance.File@Content-Location="package3"
EFDT.FDT-Instance@afdt:appContextIdList="http://kids.pbs.org/app1 http://kids.pbs.org/app2"
다중 파트 경계가 있는 리소스가 있는 경우, "Content-Location: folder2/file3.txt".

그런 다음 app1에 대한 캐시에는 folder1/file1.txt와 folder2/file3.txt가 모두 포함되고 app2에 대한 캐시에는 folder1/file2.txt와 folder2/file3.txt



(그림 5.3.3-1) Application Context Identifier Conceptual Model.
[출처: A344]

그림 5.3.3-1 는 Application Context Identifiers 가 Broadcaster Application 및 브로드캐스트 파일과 어떻게 관련되는지에 대한 개념 모델을 제공합니다. 다이어그램은 지정된 Application Context Identifier 에 고유한 URI 를 사용하여 Receiver Web Server 를 통해 Broadcaster Application 에서 리소스(파일 및 디렉터리)를 사용할 수 있는 방법의 예를 제공합니다. 그림에서 Broadcaster Application 은 Entry Page, EP^1 을 사용하여 실행된 User Agent 에서 작동하는 것을 보여줍니다. EP^1 이 활성화된 동안 애플리케이션 시그널링은 EP^2 로 지정된 Entry Page 를 시작할 수 있습니다. 이 경우 Application Context Cache A 와 이에 대한 액세스는 EP^2 와 함께 로드된 User Agent 와 일정하게 유지됩니다. Receiver 는 다른 Application Context Identifier 에 해당하는 다른 Application Context Cache 에 대한 액세스를 제공할 수도 있고 제공하지 않을 수도 있습니다. Broadcaster Applications 은 Receiver 가 제공하는 자체 Application Context Cache 내의 리소스 또는 boradband 을 사용할 수 있는 경우 인터넷에 대한 액세스를 제한해야 합니다.

여러 브로드캐스트에 걸쳐 있는 서비스에서 제공되는 Broadcaster Applications 은 동일한 Application Context Identifier 를 가질 수 있으므로 확장된 캐싱 기능이 있는 Receiver 가 튜닝 이벤트 전반에 걸쳐 리소스를 유지할 수 있습니다. 이를 통해 관련 Broadcaster Applications 을 위해 여러 브로드캐스트에서 리소스를 제공할 수 있는 광범위한 유연성이 가능합니다.

6 BROADCASTER APPLICATION MANAGEMENT

6.1 Introduction

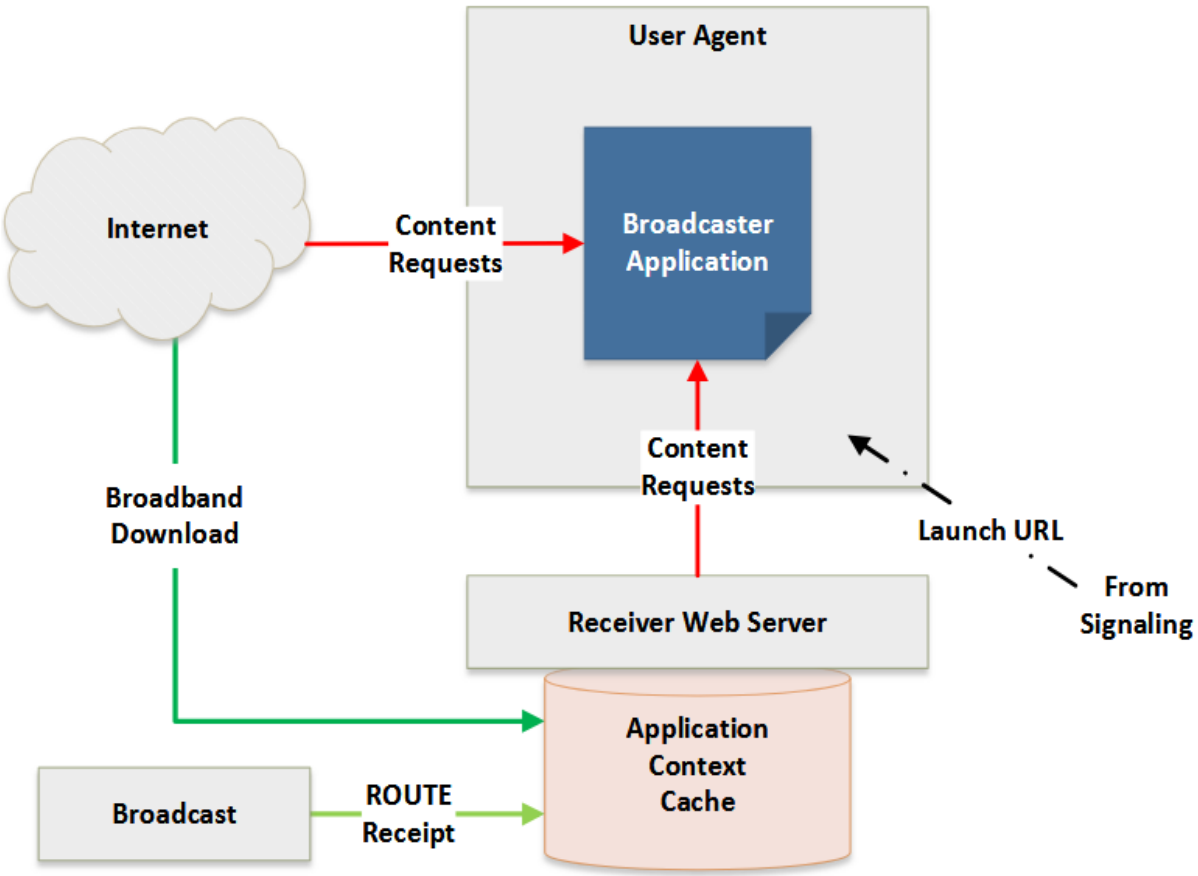
Broadcaster Application 은 하나 이상의 패키지 내에서 별도로 또는 함께 전달될 수 있는 HTML5, JavaScript, CSS, XML, 이미지 및 멀티미디어 파일로 구성된 문서 세트입니다.

이 섹션에서는 Broadcaster Application 패키지가 어떻게 구성되어 있는지 설명합니다.

- Downloaded,
- Signaled,
- Launched, 그리고
- Managed.

또한 Broadcaster Application 이 Receiver 에서 사용할 수 있는 리소스에 액세스하는 방법에 대해 설명합니다.

그림 6.1-1 은 일반화된 참조 Receiver 아키텍처 내의 다양한 개념 간의 관계(즉, Receiver Web Server 가 User Agent 와 별도의 물리적 장치에 있는지 여부)를 도식화합니다. 특정 Receiver 구현을 정의하기 위한 것이 아니라 이 섹션에서 설명하는 다양한 요소 간의 관계를 보여주기 위한 것입니다.



(그림 6.1-1) Receiver Conceptual Architecture.

[출처: A344]

Broadcaster Application 은 Receiver 가 애플리케이션 시그널링 정보(아래 section 6.3 참조)를 수신한 후 시작된 다음 실행 URL 을 User Agent 에 전달하고, 사용자 에이전트는 URL 에서 Broadcaster Application 의 Entry Page 를 로드합니다. URL 은 서비스 애플리케이션 시그널링에서 형식이 지정되는 방식, 특히 HELD [6]의 HTMLEntryPackage@bcastEntryPageUrl 또는 HTMLEntryPackage@bbandEntryPageUrl 속성에 따라 인터넷 서버 또는 Receiver Web Server 를 가리킬 수 있습니다. Broadcaster Application 항목 URL 을 User Agent 에 전달하는 구체적인 메커니즘은 Receiver 구현 세부 정보입니다. 그러나 항목 URL 에는 섹션 8.2 에 설명된 대로 제공해야 하는 특정 인수가 있습니다.

기본 Broadcaster Application Entry Page 가 로드되면 다양한 로컬 또는 외부 URL 에서 콘텐츠를 요청하기 시작할 수 있습니다. 이는 W3C 호환 방식으로 JavaScript 또는 표준 HTML5 *href* 요청을 통해 수행할 수 있습니다. ROUTE 파일 전달을 통해 방송망을 통해 수신된 모든 콘텐츠는 Application Context Cache 를 통해 사용할 수 있으며 Receiver Web Server 를 사용하여 액세스할 수 있다고 가정합니다. 이 표준은 이것이 수행되는 방법이나 캐시 또는 스토리지가 구현되는 방법에 대해 어설션하지 않습니다. 그러나 Broadcaster Application 이 Receiver Web Server 에 대한 HTTP 요청을 사용하여 리소스에 액세스할 수 있는 방법을 설명합니다.

User Agent 는 section 5.2 에 따라 다양한 로컬 W3C 스토리지 메커니즘을 지원합니다. User Agent 는 콘텐츠의 내부 캐싱을 수행할 수도 있습니다. User Agent 내에서 구현된 내부 W3C 호환 스토리지 메커니즘을 그림 6.1 에 별도로 표시된 Application Context Cache 와 혼동해서는 안 됩니다. Broadcaster Application 은 표준 W3C 인터페이스를 사용하여 다양한 User Agent 스토리지 기능을 검색하고 사용할 수 있습니다.

6.2 Application Context Cache Management

6.2.1 Signaling Intent for File Caching

방송망을 통해 Application Context Cache 로 전달되는 모든 파일은 A/331 [3]에 설명되어 있고 이 표준의 섹션 6.5.3 에서 요구하는 대로 멀티파트/서명된 패키지로 전달됩니다. ROUTE 관점에서 각 패키지는 불투명 파일 개체이므로 *EFDT* 의 *FDT-Instance* 요소 내의 후속 *File* 요소는 멀티파트/서명된 패키지 개체만 설명합니다.

기본 헤더는 패키지 내의 파일을 설명하는 필수 경계 텍스트를 포함하여 다양한 매개 변수를 설명하는 멀티파트/서명된 패키지 내에 정의됩니다. 파일 데이터는 이러한 경계로 분리된 블록 내에 상주하며, 이 블록은 차례로 개별 파일 데이터 앞에 있는 개별 파일 데이터 앞에 있는 헤더 요소들을 포함하며, 이는 패키지 블록 내의 파일에 특정한 메타데이터를 제공할 수 있다.

패키지 내에 포함된 파일에 대한 매니페스트를 제공하려면 A/331 Section 7.1.6에 정의된 대로 *metadataEnvelope*가 패키지의 첫 번째 개체로 포함되어야 합니다. 콘텐츠는 *metadataEnvelope*에 포함되어서는 안 됩니다. "referenced" 모드를 사용해야 합니다.

metadataEnvelope 조각 내에서 metadataEnvelope.item 요소는 패키지 내의 각 파일에 해당하는 존재해야 합니다. metadataEnvelope.item 속성은 다음과 같이 해석됩니다.

- 요구되는 @metadataURI 속성은 이 metadataEnvelope.item 이 참조하는 파일의 상대 경로(relative path)를 제공하여야 한다. 해당 URI 값은 참조된 파일의 경계 헤더(boundary header)의 일부로 포함된 Content-Location 매개변수에 제공된 상대 경로와 일치하여야 한다. 이는 **Application Context Cache** 내에서 파일의 상대 경로를 제공하기 위한 것으로 예상된다. 다중 파트(multipart) 리소스인 경우에는, 일반적으로 비(非) 다중 파트 리소스의 상대 경로를 설정하는 데 사용되는 EFDT.FDT-Instance.File@Content-Location 값은 무시하여야 한다.
- @version 속성은 패키지 내에서 참조된 파일의 새로운 버전이 제공될 때마다 증가한다. **Receiver**는 파일 버전 변화를 감지할 때 @validFrom 속성에 의존하여야 하며, @version 속성은 무시하여도 무방하다.
- @validFrom 속성은 필수이며, 참조된 파일이 마지막으로 수정된 시점을 나타내야 한다. 파일의 새로운 버전은 해당 파일과 연관된 metadataEnvelope.item 내의 이 타임스탬프를 갱신함으로써 신호화되어야 한다. 이 값은 **Receiver Web Server**를 통해 파일에 접근할 때 제공되는 Last-Modified HTTP 헤더 매개변수로 제공되는 것이 원칙이다. 단, Section 6.2.1.1에 기술된 바와 같이 경계 헤더(boundary header)에 ATSC-HTTP-Attributes 매개변수가 포함되어 있고, 해당 매개변수가 이 기본값을 재정의하는 경우에는 예외로 한다. 또한 이 값은 파일의 유효 기간(age)을 계산할 때 사용되는 것이 일반적이다.
- 선택적 속성인 **@validUntil**에 제공되는 날짜 및 시간 값은 파일이 더 이상 필요하지 않으며 **Application Context Cache**에서 해제될 수 있는 시점을 나타내는 데 사용되는 것이 일반적이다. **Broadcaster Application**은 **Receiver Web Server**를 통해 파일에 접근할 때 제공되는 Expires HTTP 매개변수를 사용하여 파일의 만료 시간을 확인할 수 있다. 단, Section 6.2.1.1에 기술된 바와 같이 경계 헤더(boundary header)에 ATSC-HTTP-Attributes 매개변수가 포함되어 있고, 해당 매개변수가 이 기본값을 재정의하는 경우에는 예외로 한다.
- The required @contentType attribute shall provide the MIME type of the referenced file. This attribute shall match the Content-Type value defined as part of the boundary header within the package, if provided. Note that the EFDT.FDT-Instance.File also contains a Content-Type definition, but this

should always be multipart/signed indicative of the referenced package object. This value may be accessed through the Content-Type HTTP header parameter supplied when accessing the file through the Receiver Web Server.

이 표준은 A/331 [3]에 정의된 *metadataEnvelope.item* 표준을 확장하는 추가 속성을 정의합니다. 표 6.2.1-1 은 Application Context Cache 를 대상으로 하는 서명된 패키지 내에 포함될 때 ATSC 확장 속성에 대한 유익한 정의를 제공합니다. 속성의 규범적 의미 체계는 표 아래에 제공됩니다.

<표 6.2.1-1> ATSC-Defined Extension to the metadataEnvelope.item Elemen

Attribute Name	Cardinality	Data Type	Description
@contentLength	0..1	long	Provides the length in bytes of the referenced file. This value may be accessed through the Content-Length HTTP attribute in a response to a User Agent request.

@contentLength – 선택적 @contentLength 속성은 패키지 내에서 참조된 파일의 길이(바이트)를 정의해야 합니다. 정의된 경우 길이 값은 참조된 파일에 액세스할 때 **Receiver Web Server** 에서 제공하는 Content-Length HTTP 헤더 요소의 일부로 사용할 수 있어야 합니다.

6.2.1.1 Boundary Header HTTP Attribute Definition

경계 헤더 요소인 ATSC-HTTP-Attributes 는 선택적으로 멀티파트/서명된 패키지 내의 파일 경계 헤더에 제공될 수 있습니다. 이 요소는 **Receiver Web Server** 를 통해 연관된 파일이 요청될 때마다 제공될 것으로 예상되는 HTTP 헤더 요소 목록을 제공합니다. 이 속성의 구문은 다음과 같이 정의되어야 합니다.

```
attributes := "ATSC-HTTP-Attributes" ":" parameter[";" parameter]*
```

각 매개변수는 RFC 7230 [16]에 정의된 HTTP 헤더 필드와 동일하나, HTTP 헤더 필드에서 사용하는 콜론 “:”은 멀티파트 표준 RFC 2045 [20]에서 요구하는 제약을 준수하기 위해 등호 “=”로 대체되어야 한다.

6.2.2 Application Context Cache Hierarchy Definition

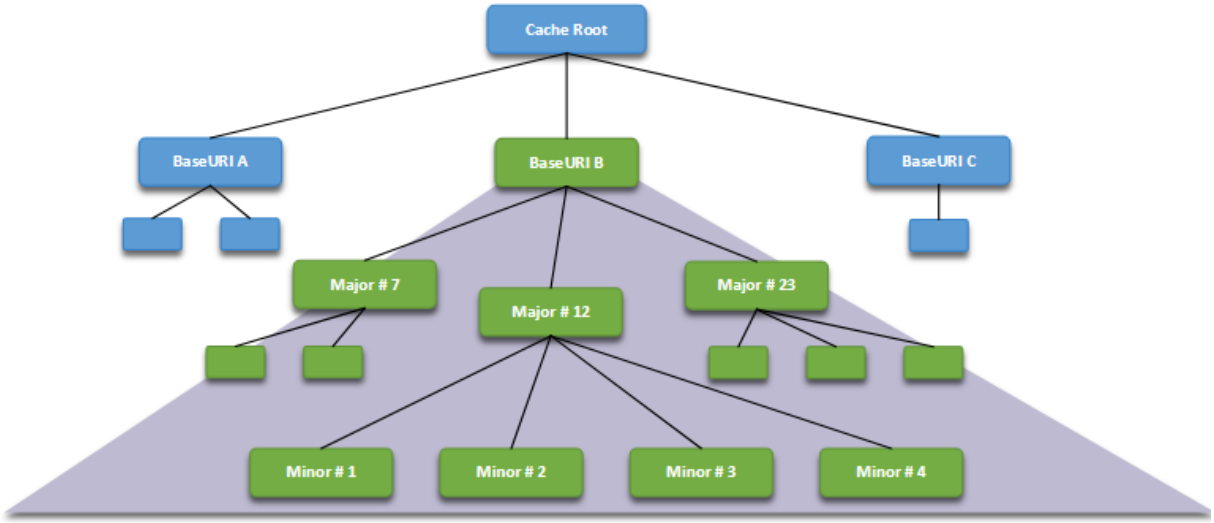
모든 interactive content 는 서명된 파일 패키지로 전달되며 ROUTE 를 통해 전송됩니다(섹션 6.5). 애플리케이션 시그널링이 특정 **Application Context Identifier** 를 나타내는 서명된 모든 패키지는 고유한 **Base URI** 를 사용하여 **Broadcast Application** 에서 액세스할 수 있는 단일 계층 구조로 제공되어야 합니다. **Base URI** 개념과 해당 의미 체계는 섹션 5.3 에 설명되어 있습니다.

패키지 또는 파일이 ROUTE 파일 스트림 내에서 수신 시 **Application Context Cache** 에 즉시 저장되는지 또는 **Receiver** 가 특정 브로드캐스트 데이터 요소가 참조될 때까지 스토리지를 연기하도록 선택하는지 여부는 수신기 구현 결정입니다. 그러나 기본 **Receiver Web Server** 가 요청된 콘텐츠를 제공할 수 없는 경우 400-series *Client Error* 또는 500-series *Server Error* 내의 HTTP 상태 코드가 반환되어 일부 오류 조건이 발생했음을 나타냅니다[18]. 일반적으로 *404 Not Found*, *408 Request Timeout* 또는 *504 Gateway Timeout* 오류입니다. 그러나 **Broadcaster Applications** 은 **Entry Package** 에 포함되지 않은 리소스를 참조할 때 HTTP 상태 코드를 처리하도록 구성되어야 합니다.

마찬가지로, 본 문서는 **Broadcaster Application** 에 필요한 패키지가 전송되는 빈도 또는 애플리케이션 신호 메타데이터가 전송되는 빈도를 지정하지 않습니다. 이러한 결정은 콘텐츠 시청 중에 **Broadcaster Application** 기능이 실제로 필요한 시간, 서비스를 선택한 후 **Receiver** 가 **Broadcaster Application** 에 액세스해야 하는 속도, **Broadcaster Application** 리소스를 전달하는 데 필요한 전체 대역폭 등 여러 요인에 따라 달라집니다. 이러한 요소와 기타 요소는 전반적인 목적에 따라 모든 **Broadcaster Application** 에 대해 다를 수 있습니다. Application Signaling 표준인 A/331[3]에서는 HELD 가 **Broadcaster Application** 에 신호를 보낼 때 브로드캐스트 패키지를 사용할 수 있어야 합니다. 또한, A/331 은 또한 **Broadcaster Application** 이 HELD 에서 신호를 보내는 시점이 아닌 다른 시간에 패키지가 브로드캐스트에서 사용할 수 있을 때를 알리는 메커니즘을 제공하는 DWD(Distribution Window Description Fragment)를 정의한다.

브로드캐스터는 위의 섹션 6.2.1 에 설명된 디렉토리 및 경로 메커니즘을 사용하여 **Application Context Cache** 루트 디렉토리 아래의 계층 구조를 정의하고 관리할 책임이 있습니다. **Application Context Cache Base URI** 수준 아래의 모든 계층은 브로드캐스터가 정의해야 합니다.

이러한 계층 구조를 정의하는 방법에 대한 예는 아래에 설명되어 있습니다. 그림 6.2.2-2 는 **BaseURI B** 에 대한 이러한 계층 구조의 예를 보여줍니다.



(그림 6.2.2-2) Example Application Context Cache Hierarchy.

[출처: A344]

예를 들어, 브로드캐스터가 ROUTE 데이터를 전송하여 **Broadcaster Application** 파일이 그림 6-2 에 표시된 계층 구조의 "Minor #2" 디렉토리에 배치된다고 가정합니다. 또한 **Broadcaster Application Entry Page** 를 "main.html"이라고 가정합니다. 브로드캐스터 애플리케이션을 시작하기 위해 이 예제의 애플리케이션 시그널링은 HELD [6]의 HTMLEntryPackage@bcastEntryPageUrl 속성에 "12/2/main.html"의 상대 URI 를 제공합니다. 로컬 경로는 브로드캐스터가 정의한 규칙이며 "red/green"이라고 쉽게 부를 수 있습니다. **Broadcaster Application Entry Page** 의 실제 URL 은 다음과 같습니다.

<BaseURI>/12/2/main.html

URL 의 **Receiver**-created **<BaseURI>** 부분은 굵게 강조 표시됩니다. **Broadcaster Application** 은 그림 6.2.2-2 의 "*BaseURI B*" 상자 아래의 녹색 상자로 지정된 *AppContextID* 계층 구조에 해당하는 모든 디렉터리를 참조할 수 있습니다.

Receivers 는 **Receiver WebSocket Server API** 를 통해 현재 **Application Context Identifier** 와 연결된 **Base URI** 정보를 **Broadcaster Application** 에 제공합니다(섹션 9.2.7 참조). 이 **Base URI** 는 **Broadcaster Application** 이 브로드캐스트에서 소싱된 경우 일반 W3C 문서 개체 모델(DOM)을 통해 사용할 수 있습니다. 이 경우 **Broadcaster Application** 은 상대 URL 을 사용하거나 *Document.location* 을 사용하여 전체 URL 을 구성하여 콘텐츠에 액세스하기만 하면 됩니다. **WebSocket API** 는 광대역에서 호스팅되는 **Broadcaster Application** 에 가장 적합하며 브로드캐스트를 통해 수신된 리소스에 액세스할 수 있습니다.

6.2.3 Active Service Application Context Cache Priority

Broadcaster Application(섹션 6.3 참조)의 **Entry Package** 로 알려진 패키지된 리소스가 획득되어 **Application Context Cache** 에 배치되면 **Broadcaster Application** 은 **Entry Package** 리소스, 즉 단일 패키지에서 **Broadcaster Application Entry Page** 와 함께 전달된 파일이 **Broadcaster Application** 이 **User Agent** 에 로드되는 한 계속 사용할 수 있다고 가정할 수 있습니다. **Receiver** 가 전체 **Entry Package** 를 **Broadcaster Application** 에 안정적으로 제공할 수 없는 경우 **Broadcaster Application** 이 시작되지 않을 것으로 예상됩니다. 이를 통해 브로드캐스터는 **Broadcaster Application** 에 필요한 파일을 결정하고 모두 하나의 패키지로 전송할 수 있으므로 **Broadcaster Application** 이 실행된 경우 해당 파일을 사용할 수 있습니다. 또한 **Receiver** 는 현재 서비스 항목 패키지와 관련된 **Broadcaster Application(s)**에 대해 특정

서비스와 함께 전달된 파일을 캐시하기 위해 모든 노력을 기울여야 합니다.

Receiver는 현재 활성 계층 구조의 파일을 수용하기 위해 활성화되지 않은 캐시의 다른 부분을 해제해야 할 수 있습니다. 실제로, Receiver가 현재 활성 서비스를 수용하기 위해 전체 캐시를 제거해야 할 수도 있다. 사용자가 현재 서비스를 적극적으로 선택했기 때문에 수신자는 현재 서비스 콘텐츠가 사용자 관점에서 다른 모든 콘텐츠를 선점한다고 가정합니다.

Broadcaster Application은 필터 코드 메커니즘을 사용하여 필요한 캐시의 양을 제한할 수 있습니다. Filter Codes는 애플리케이션 콘텐츠를 구별하는 방법을 제공하여 Broadcaster Application이 Application Context Cache에 배치할 패키지와 무시할 패키지를 표시할 수 있도록 합니다. Broadcaster Application이 Filter Code를 관리할 수 있도록 WebSocket API 세트가 제공됩니다(Section 9.9 참조).

Broadcaster Application은 콘텐츠를 사용되지 않는 것으로 표시하여 리소스가 더 이상 필요하지 않음을 수신자에게 알릴 수 있습니다. 파일 또는 전체 디렉토리 계층을 사용되지 않는 것으로 표시할 수 있는 WebSocket API가 제공됩니다(Section 9.7 참조). 사용되지 않는 것으로 표시된 리소스에 대한 후속 액세스로 인해 오류가 발생할 것으로 예상됩니다. 또한 HELD(A/331 [3])는 지정된 날짜 및 시간 이전에 생성된 Application Context Cache의 모든 파일을 제거해야 함을 나타내는 Broadcaster Application과 연결된 속성 *@clearAppContextCacheDate*를 제공합니다. 이렇게 하면 Application Context Cache에 있는 이전 파일을 더 이상 사용되지 않는 것으로 표시할 수 있습니다.

6.2.4 Cache Expiration Time

Section 6.2.1에 정의된 파일의 *@validUntil* 속성은 브로드캐스터가 Broadcaster Application이 현재 User Agent에 로드되어 있는지 여부에 관계없이 만료 시간에 도달할 때까지 파일이 Application Context Cache에 남아 있을 것으로 예상한다는 것을 나타냅니다. 그러나 로드된 Broadcaster Application 션의 스토리지 요구 사항은 *@validUntil* 속성보다 우선하므로 Receiver는 현재 서비스에 스토리지를 제공하기 위해 *@validUntil* 시간 전에 다른 서비스에서 파일을 해제해야 할 수 있습니다. Broadcaster Application은 리소스를 사용할 수 없을 가능성에 대처할 준비가 되어 있어야 합니다. Receiver는 다른 캐시 관리 기준에 따라 *@validUntil* 날짜 및 시간이 경과한 후 편리할 때마다 Application Context Cache 내에 리소스를 남겨두거나 제거할 수 있습니다.

Entry Package의 일부인 리소스의 경우, Broadcaster Application은 시작 시 해당 파일을 사용할 수 있다고 가정할 수 있으며, Section 6.2.3에 설명된 대로 Broadcaster Application이 *@validUntil* 시간에 관계없이 실행 중인 경우 계속 사용할 수 있습니다.

6.2.5 Advanced Emergency Alert Enhancement Content Considerations

AEAT는 URL을 통해 AEA 향상 콘텐츠를 참조할 수 있습니다. 이 콘텐츠는 브로드캐스트에서 별도의 ROUTE 스트림으로 전달되며, 이 경우 참조 URL은 상대적이거나 광대역을 통해 정규화된 URL이 제공됩니다.

브로드캐스트의 경우 브로드캐스터는 AEA 향상 콘텐츠가 포함된 서명된 패키지를 설명하는 EFDT 조각의 일부로 AEA 향상 콘텐츠에 액세스해야 하는 모든 **Broadcaster Application**의 **Application Context ID**를 제공할 책임이 있습니다. **Receiver**는 나열된 **Application Context ID**에 해당하는 **Application Context Cache**에서 사용할 수 있도록 하여 이 컨텐츠를 일반 ROUTE 전달 컨텐츠와 동일하게 처리해야 합니다. AEA 향상 콘텐츠는 다른 ROUTE 전달 데이터와 동일한 계층을 차지할 것으로 예상되므로 브로드캐스터는 해당 파일 계층을 정의할 때 충돌이 발생하지 않도록 해야 합니다.

마찬가지로, **Broadcaster Application**에서 수행하는 모든 캐시 관리는 *@clearAppContextCacheDate* 신호를 보내거나 **Mark Unused API** (Section 9.7 참조)를 사용하여 콘텐츠가 더 이상 필요하지 않음을 나타낼 때 AEA 이벤트의 수명을 고려해야 합니다. AEA 이벤트와 연결된 모든 브로드캐스트 NRT 데이터에는 AEA 이벤트 지속 시간 이상으로 정의된 *@validUntil* 속성이 있어야 합니다.

6.3 Broadcaster Application Lifecycle

이 Section에서는 **Broadcaster Application** 수명 주기를 규정합니다. HELD의 구문 및 의미론은 A/331 [3] section 7.1.8을 참조하십시오. **Broadcaster Application** 수명 주기에 대한 자세한 설명은 부록 A를 참조하십시오. 이 라이프사이클은 모든 *SLT.Service@serviceCategory* 값에 적용된다.

다양한 형태의 "로드"라는 용어는 실행에 필요한 리소스가 **Receiver** 메모리로 가져와 실행이 시작되거나 계속되었음을 의미합니다. *serviceCategory*에 대한 종속성은 없습니다.

Broadcaster Application 수명 주기는 처음에 유효한 HELD 및 그 안의 항목의 유무로 표시됩니다. **Broadcaster Application**이 서비스의 일부인 경우, HELD는 적어도 HELD가 유효할 때마다 해당 서비스에 대한 SLS의 모든 인스턴스에 존재해야 합니다. HELD 구문이 일치하고 현재 UTC 시간이 *@validFrom* 이전(있는 경우)이 아니며 HELD의 하나 이상의 항목에 대해 *@validUntil* 후(있는 경우)가 아닌 경우 "유효"합니다. ATSC A/336에 설명된 콘텐츠 복구 방법을 사용하여 얻은 HELD의 경우, *@validFrom* 및 *@validUntil* 평가하기 위해 현재 UTC 시간 대신 **Recovery Media Timeline** [5]의 현재 프레젠테이션 시간이 사용될 것으로 예상됩니다.

HELD를 받으면 **Receiver**는 HELD에 하나 이상의 항목으로 표시된 대로 후보 **Broadcaster Application** 목록을 작성해야 합니다. **Receiver**는 과거의

*@validUntil*을 나타내는 모든 항목을 제외하고 미래의 *@validFrom*를 나타내는 모든 항목을 제외해야 합니다. 마찬가지로, 수신자는 수신기의 기능에 의해 충족되지 않는 *@requiredCapabilities* 나타내는 모든 항목을 제외해야 합니다. 이 필터링된 목록을 사용하여 수신기는 실행할 Broadcaster Application 을 별하는 단일 HELD 항목을 식별해야 합니다.

- 목록에 현재 로드된 Broadcaster Application 과 동일한 *@appId* 및 *@appContextId* 항목을 가진 항목이 있는 경우 해당 항목이 선택되고 현재 Broadcaster Application 이 계속 실행될 수 있습니다("다시 로드"가 필요하지 않음).
- 목록에 사용자가 선택한 *@appId* 및 *@appContextId* 와 일치하는 항목이 있는 경우 해당 항목이 선택되고 실행을 계속할 수 있습니다("다시 로드"가 필요하지 않음).
- 목록에 정확히 하나의 항목이 있거나 정확히 하나의 항목이 *@default=True* 로 표시되고 앞의 조건이 적용되지 않는 경우 해당 항목이 로드할 후보입니다.
- 이 항목 목록이 비어 있는 경우 Broadcaster Application 이 현재 로드된 경우 수신기는 현재 Entry Page 를 언로드해야 합니다.
- 이 항목 목록에 둘 이상의 후보가 있고 그 중 어느 것도 기본값으로 표시되지 않은 경우 동작이 정의되지 않습니다.

Broadcaster Application 은 Entry Package 가 Application Context Cache 내에서 완전히 사용할 수 있을 때 "로드"됩니다. 전체 URL 이 제공되는 외부 광대역 참조의 경우 HELD 의 Broadcaster Application section 에 대한 *HTMLEntryPackage@bbandEntryUrl* 속성에 제공된 URL 이 직접 시작될 것으로 예상됩니다.

Broadcaster Application 이 현재 로드되고 서비스 변경이 시작되는 경우(Receiver 또는 Broadcaster Application 에 의해), Receiver 는 Receiver 가 Service Change API 를 사용하여 채널 변경 처리를 시작하기 전에 Broadcaster Application 에서 특정 리소스를 해제하도록 요청할 수 있습니다(섹션 9.16 참조).

Broadcaster Application 이 현재 로드되고 후보 항목의 *@appContextId* 및 *@appId* 값이 동일한 경우 현재 로드된 Broadcaster Application 은 후보 항목과 동일하고 실행 중인 Broadcaster Application 은 유지되며 서비스 변경 알림을 요청한 경우 수신합니다.

마찬가지로, Broadcaster Application 이 현재 로드되고 있고 해당 Broadcaster Application 의 *@validUntil* 속성에 도달한 경우, Receiver 는 현재 Broadcaster Application 을 언로드하고 아래와 같이 진행해야 합니다.

Broadcaster Application 이 현재 로드되고 Broadcaster Application 이 후보 항목과 다른 경우, Receiver 는 현재 Entry Page 를 언로드하고 후보 엔트리

Broadcaster Application 의 Entry Page 를 로드해야 합니다.

현재 로드된 Broadcaster Application 이 없는 경우 Receiver 는 후보 항목 Broadcaster Application 의 Entry Page 를 로드해야 합니다.

주어진 Entry Page 내의 논리는 위에서 설명한 대로 하위 페이지를 로드하고 언로드할 수 있습니다. 이러한 작업은 Broadcaster Application 에 따라 다르며 HELD 의 적용을 받지 않습니다.

HELD 가 SLS 에 더 이상 없거나 더 이상 유효하지 않은 경우 수신자는 현재 Entry Page 를 언로드해야 합니다.

Broadcaster Application 이 실행 중일 때 사용자 또는 Broadcaster Application 이 공통 *@appContextId* 를 사용하여 다른 서비스를 요청하고 있는 경우 *@appId* 수신자는 *"requested"="true"*와 함께 서비스 변경 알림(섹션 9.3.3 참조) 이벤트를 발행해야 합니다. 이 경우 Broadcaster Application 은 즉시 다음을 수행해야 합니다.

1. 모든 AMP 활동을 중단하고 모든 AMP 관련 리소스를 해제합니다.
2. RMP URL API 설정(section 9.6.3 참조)이 RMP 를 적극적으로 사용하는 경우 *"operation"="stopRmp"*를 사용하여 RMP URL API 설정을 실행합니다.
3. 모든 방송 프로그램 관련 정보를 지우고,
4. 비디오 크기 조정 및 위치 지정을 재설정하고(section 9.6.2 참조)
5. Application Context Cache 에서 필요한 리소스 요청을 시작합니다.

요청된 서비스를 획득한 후 요청된 서비스에 공통 *@appContextId* 가 있고 있는 경우 *@appId* 있는 경우 Receiver 는 *"requested" = "false"*로 서비스 변경 알림(섹션 9.3.3 참조)을 발행할 것으로 예상됩니다.

사용자 또는 Broadcaster Application 이 다른 *@appContextId* 다른 서비스를 요청하고 있는 경우 *@appId* 서비스 변경 알림 이벤트가 발생하지 않고 Broadcaster Application 이 종료됩니다.

HELD 는 여러 방송사 애플리케이션(Broadcaster Application)을 시그널링할 수 있으므로, 서비스 간 방송사 애플리케이션 변경을 관리하는 것 외에도, 방송사 애플리케이션이 Launch Broadcaster Application API(section 9.6.6 참조)를 사용하여 **동일한 서비스 내에서 새로운 방송사 애플리케이션으로 전환**을 시작할 수 있다. 이것은 동일하거나 다른 Application Contexts 에 있을 수 있습니다.

새 Broadcaster Application 이 유효하고 해당 리소스를 사용할 수 있는 경우 호출 Broadcaster Application 이 종료되고 새 Broadcaster Application 이 로드됩니다. 호출 Broadcaster Application 은 Launch Broadcaster Application API 를 호출하기 전에 알림, 구독 및 기타 모든 리소스를 해제해야 합니다. *appContextId* 가 Broadcaster Application 간에 다른 경우 시작 Broadcaster Application 은 호출 Broadcaster Application 리소스에 액세스할 수 없습니다.

6.4 Broadcaster Application Events (Static / Dynamic)

Broadcaster Application이 수행할 작업은 브로드캐스트 또는 broadband을 통해 전달되거나 재배포 설정에서 워터마크를 통해 전달되는 알림에 의해 시작될 수 있습니다. A/337 [6]은 이러한 통지에 대해 "Events"이라는 용어를 사용한다.

Events의 방송망 전송은 ROUTE/DASH 기반 서비스 및 MMT 기반 서비스에 대한 전달을 포함하여 A/337[6]의 Section 4.1에 정의되어 있습니다.

Events의 broadband망 전송은 ROUTE/DASH 기반 서비스 및 MMT 기반 서비스에 대한 전달을 포함하여 A/337[6]의 Section 4.2 및 4.5에 정의되어 있습니다.

A/337[6]에 정의된 events의 방송망 및 broadband 전송은 모두 배치(Static) 및 증분(Dynamic) 전달을 지원합니다.

재배포 설정에서는 A/337[6]의 Section 4.3에 설명된 대로 비디오 및 오디오 워터마크를 통해 이벤트를 전달할 수도 있습니다.

Event 스트림 알림을 등록하고 수신하는 데 사용되는 **WebSocket API**의 자세한 표준은 section 9.5에 나와 있습니다.

6.5 Broadcaster Application Delivery

A/331[3]에 설명된 ROUTE의 파일 전달 메커니즘은 ATSC 3.0 브로드캐스트를 통해 파일 모음을 별도로 또는 패키지로 전달하는 수단을 제공합니다. ROUTE 전달 파일은 section 6.2에 설명된 대로 **Receiver Web Server**를 통해 **User Agent**에서 사용할 수 있습니다. **Receiver**가 액세스할 수 있는 웹 서버에 게시하여 동일한 파일 컬렉션을 broadband 전송에 사용할 수 있습니다. 또한 애플리케이션 시그널링[6]은 **Broadcaster Application Entry Page** 소스와 브로드캐스트를 통해 전달되는 다른 파일 및 패키지의 위치를 결정합니다.

1. 상대 **Entry Page** URI는 **Broadcaster Application Entry Page**의 소스가 브로드캐스트 ROUTE 데이터임을 나타냅니다.
2. 절대 **Entry Page** URL은 **Broadcaster Application Entry Page**가 broadband망에서 제공되어야 함을 나타냅니다.
3. 브로드캐스트를 통해 파일 또는 패키지가 전달되는 경우 애플리케이션 신호는 파일 및/또는 패키지를 전달하는 데 사용되는 LCT 채널을 식별합니다.

초기 **Entry Page**는 HELD의 HTMLEntryPackge@bbandEntryPageUrl 또는 HTMLEntryPackage@bcastEntryPageUrl 속성에 따라 broadband망 또는 방송망에서 제공될 수 있지만 **Broadcaster Application** 자체 내에서 방송망 또는 broadband망 리소스를 사용하는 데 대한 제약은 없습니다. **Broadcaster Application** 파일의 하이브리드 전달이 허용되고 예상됩니다.

6.5.1 Broadcaster Application Packages

Broadcaster Application을 구성하는 파일 구성 요소는 ROUTE를 사용하여 하나 이상의 다중 파트 MIME 패키지 내에서 브로드캐스트를 통해 전달되거나 HTTPS를 사용하여 광대역을 통해 개별 파일로 전달되어야 합니다. Receiver Web Server를 통해 제공되는 모든 파일은 A/331[1]에 설명된 대로 서명된 패키지로 Receiver 에게 전달되어야 합니다.

Broadcaster Application에서 사용하는 모든 리소스가 방송망을 통해 전달될 때 단일 ROUTE 패키지로 전달될 필요는 없습니다. 브로드캐스터는 서명된 Entry Package에 상대적으로 작은 Entry Page를 전송하도록 선택할 수 있으며, 이 페이지는 부트스트래핑 작업을 수행하여 브로드캐스트 전달 경로를 통해 전달되었거나 액세스할 수 있는 다른 리소스와 광대역 요청을 사용하여 가져와야 하는 리소스를 결정합니다. Entry Page 가 포함된 Entry Package 는 Broadcaster Application을 시작하기 전에(section 6.2.3 에 따라) 전체를 수신해야 하기 때문에 Broadcaster Application은 기본 리소스에 대한 확인을 생략하고 초기 시작 시간을 단축할 수 있습니다. Broadcaster Application은 Receiver의 리소스 가용성에 따라 증분 기능을 가질 수 있다고 생각할 수 있습니다. 즉, Broadcaster Application은 더 많은 리소스를 사용할 수 있으므로 기능을 추가할 수 있습니다. 사용할 수 있도록 할 특정 서명된 패키지를 선택하기 위한 제어는 Filter Codes API를 사용하여 제공됩니다(section 9.9 참조).

또한, Broadcaster Application은 Broadcaster Application을 전통적인 의미의 진정한 Web Application으로 만들기 위해 광대역 웹 서버에서 리소스 및 콘텐츠를 요청하거나 다른 활동을 수행할 수 있습니다. Broadcaster Application은 모든 수신기에 필요한 모든 리소스를 위한 충분한 스토리지가 포함되어 있지 않을 수 있으며 broadband 망 연결이 없을 수 있음을 알고 있어야 합니다

6.5.2 Broadcaster Application Packages

Broadcaster Application 리소스 파일 및 패키지는 언제든지 업데이트될 수 있습니다. 새 파일 또는 패키지가 전달되고 있는지 확인하는 메커니즘은 ROUTE[3]에서 사용하는 기본 표준인 FLUTE 에 정의되어 있습니다. 브로드캐스터는 Event Stream 알림을 보내 Broadcaster Application 에 변경되었음을 알릴 수 있습니다. Broadcaster Application은 Event Stream 알림에 따라 이러한 변경 사항을 해결하는 방법을 결정합니다

6.5.3 Broadcaster Application Packages

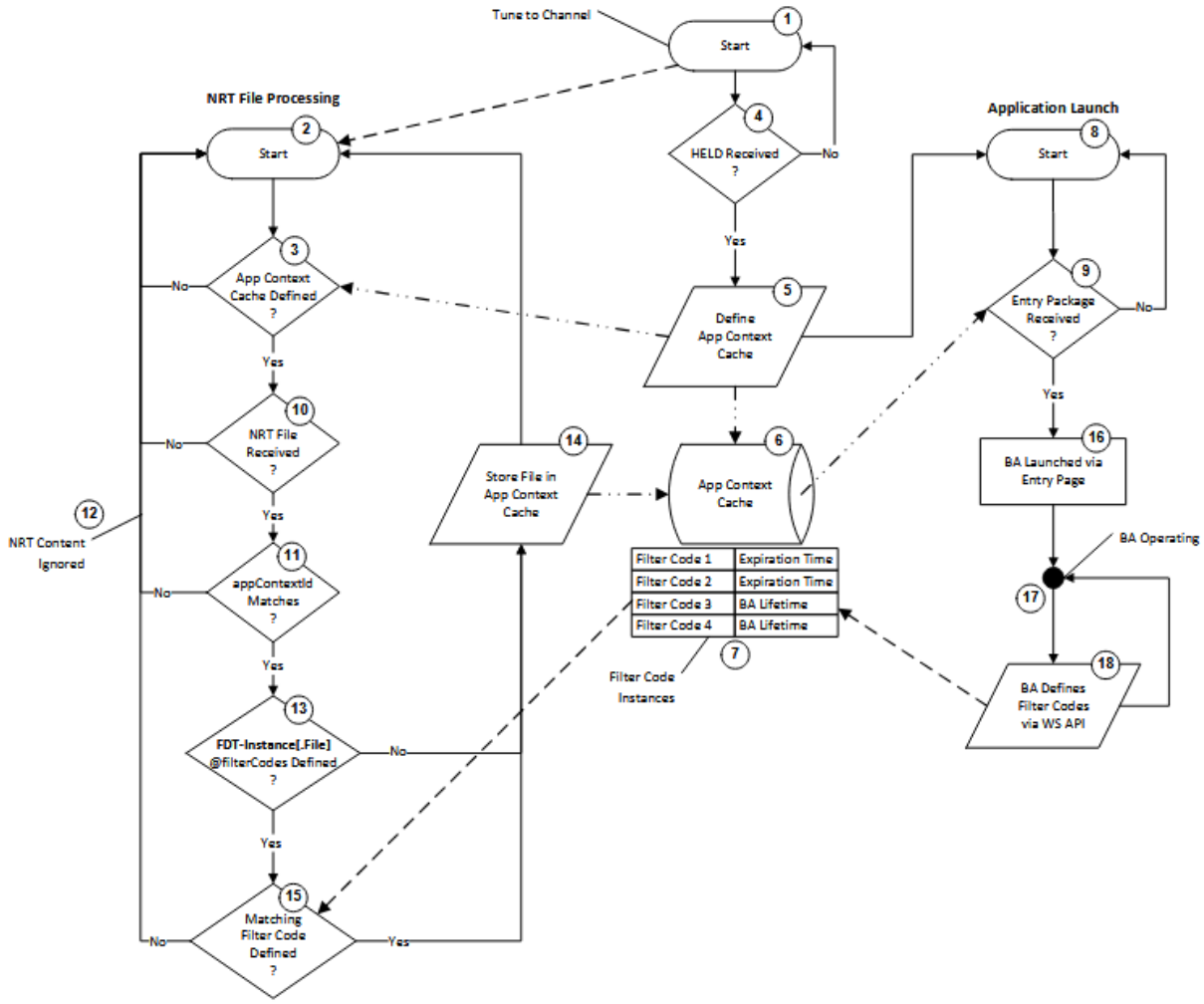
Filter Codes는 Broadcaster Application이 연결된 Application Context Cache 에 저장되는 NRT 데이터를 제어할 수 있는 방법을 제공합니다. 브로드캐스트는 다양한 수신기에서 실행되는 Broadcaster Application 의

가능한 모든 인스턴스에 대한 모든 NRT 데이터를 포함해야 합니다. 그러나 개별 Broadcaster Application은 서비스, 사용자, Receiver 인스턴스 또는 기타 제약 조건에 따라 특정 NRT 데이터만 필요할 수 있습니다.

Filter Code Instance는 Application Context Cache와 연결된 값 및 선택적 만료 시간입니다. Filter Code Instance는 만료 시간이 초과될 때까지 또는 만료 시간이 지정되지 않은 경우 Broadcaster Application의 수명 동안 Application Context Cache와 함께 유지됩니다. Receiver가 Application Context Cache의 리소스를 회수하는 경우 만료되지 않은 연결된 Filter Code Instance도 회수될 것으로 예상됩니다. 회수되면 Filter Code Instance 만료 시간에 관계없이 Application Context Cache 또는 연관된 Filter Code Instance가 정의되지 않습니다.

Filter Codes를 사용하여 NRT 데이터 스토리지를 제어하려면 브로드캐스트 및 Broadcaster Application을 가장 효과적인 방식으로 엔지니어링할 수 있도록 Receiver가 들어오는 신호 및 NRT 데이터를 Application Context Cache로 처리하는 방법을 명확히 해야 합니다. 그림 6-3은 Receiver가 NRT 데이터 및 Filter Code Instance를 기반으로 Application Context Cache에 콘텐츠를 배치하는 데 사용할 것으로 예상되는 개념적 프로세스의 순서도를 제공합니다. Broadcaster Application의 실행 상태도 표시되며, 그 시작은 필터링되지 않은 NRT 데이터 패키지의 수신에 의존하고 Broadcaster Application이 정의할 때까지 Filter Code Instance가 존재하지 않을 수 있습니다.

이 논의의 목적을 위해 Broadcaster Application은 로컬 Application Context Cache에서 실행되도록 의도되었으며 모든 데이터는 broadband망 소스가 아닌 broadcast망 내의 ROUTE 스트림을 통해 수신된다고 가정합니다. Filter Code 처리는 여전히 브로드캐스트 스트림에서 선택한 NRT 데이터를 필터링하는 데 사용될 수 있지만 Broadcaster Application과 필요한 데이터 모두의 broadband망 소스는 그림 6-3의 순서도에 표시된 결정 중 일부를 생략할 수 있습니다.



(그림 6.5.3-1) Filter Code Processing Flowchart.
[출처: A344]

다음 설명은 앞의 그림 6.5.3-1 에 표시된 단계별 주석(annotation)에 기반한다.

처리는 (1) 수신기(Receiver)가 특정 채널에 튜닝될 때 시작된다. 초기 검색(Discovery) 과정의 일부로, 선택된 서비스와 관련된 서비스 레벨 시그널링(Service Level Signaling, SLS) 은 하나 이상의 ROUTE 세션(ROUTE session) 이 존재하며, 이 세션들이 NRT 데이터를 전송하고 있음을 나타낼 수 있다.

ROUTE NRT 데이터 스트림이 존재하면, 그림 6.3 의 왼쪽에 나타난 NRT 파일 처리(NRT File Processing) 절차가 (2) 단계에서 시작된다.

Application Context Cache 로 전달될 NRT 콘텐츠의 경우, 콘텐츠가 저장되기 전에 해당 캐시가 먼저 정의되어야 한다 **(3)**.

Application Context Cache 는 **Receiver** 가 리소스를 회수(reclaim)하지 않았다면 이미 존재할 수도 있으나, 이는 채널 전환 간의 시간, 사용 가능한 리소스의 양 등 여러 요인에 따라 달라질 수 있다.

또한, 다른 목적을 위한 NRT 데이터가 수신될 수도 있으나, 이러한 사용 사례는 본 논의의 범위를 벗어난다.

SLS 처리가 진행되는 과정에서, 서비스 시그널링에는 선택된 서비스와 연관된 **Broadcaster Application** 이 존재함을 나타내는 HELD table [3] 이 포함될 수 있다. HELD 가 수신되면 (4), 이는 *HELD.HTMLEntryPackage@appContextId* [3]에 정의된 **Application Context Identifier**를 확인하기 위해 처리된다. 이후 **Application Context Cache (6)** 가 **(5)** 단계에서 할당되며, 여기에 관련된 NRT data 가 저장될 수 있다. 이때, 이전에 동일한 **Application Context Identifier** 를 사용한 **Broadcaster Application** 이 존재하고 수신기가 해당 리소스를 아직 회수하지 않은 경우, **Application Context Cache** 가 이미 존재할 수도 있다.

마찬가지로, **Filter Code Instances** 역시 이전 **Broadcaster Application** 인스턴스에 의해 설정된 경우, 만료 시간이 아직 지나지 않았다면 **Application Context Cache** 와 연계되어 있을 수 있다. 이러한 **Filter Code Instances**는 **그림 6-3** 에서 "Filter Code 1"과 "Filter Code 2"로 표시되어 있으며, 각각의 만료 시간(Expiration Time)은 (7) 로 주석(annotation)되어 있다. 도식 내의 "Filter Codes 3"과 "4"에 해당하는 **Filter Code Instances** 는 이후에 설명된다.

HELD table 은 또한 *HELD.HTMLEntryPackage@bcastEntryPageUrl* [3]을 통해 **Application Launch** process 를 시작하게 한다 (8). 이후 **Receiver**는 **Entry Package** 가 수신되어 **Application Context Cache**에서 사용 가능해질 때까지 대기해야 한다 (9). 만약 **Application Context Cache** 가 여전히 존재하며 적절한 **Entry Package**를 포함하고 있는 경우, **Broadcaster Application** 은 즉시 시작될 수 있다.

Receiver 가 **Entry Package** 가 사용 가능해지기를 기다리는 동안 (9), **NRT File Processing logic** 은 파일이 수신되기를 기다리고 있다 (10). 파일을 수신하는 과정은 복잡하며, ROUTE stream 내의 시그널링에 의존한다는 점에 유의해야 한다. 파일의 각 부분이 도식에 표시된 테스트에 따라 개별적으로 검사될 가능성이 높지만, 단순화를 위해 흐름도(flowchart)는 파일 단위의 처리 과정을 보여준다.

파일이 수신되면, 먼저 (11) 해당 파일이 **Application Context Cache**에 저장되어야 하는지를 확인한다. 이 여부는 *FDT-Instance@appContextIdList* 또는 *FDT-Instance.File@appContextIdList* [3] 내에 일치하는 *appContextId* 가 존재하는지에 따라 결정된다. 만약 일치하는 *appContextId* 가 없다면, 해당 파일은 무시되고 처리 과정은 (12) 로 이어진다.

appContextId 가 일치하는 경우, 수신기는 파일에 **Filter Code** 가 존재하는지를 확인한다 (13). **Filter Code** 는 *FDT-Instance@filterCodes* 리스트 속성에서 전체 파일에 대해 정의되거나, *FDT-Instance.File@filterCodes* [3] 리스트 속성에서 특정 파일에 대해 개별적으로 정의될 수 있다. **Filter Code** 리스트 속성을 처리한 후에도 파일에 할당된 **Filter Code** 가 없으면, 해당 파일은 (14) **Application Context Cache** (6)에 저장된다. 참고로, *File@filterCodes* 리스트는 *FDT-Instance@filterCodes*

리스트보다 우선 적용된다. 파일에 하나 이상의 Filter Code 가 정의된 경우, 처리 과정은 설정된 Filter Code Instance 가 존재하는지를 확인하는 단계 (15) 로 이동한다.

초기 Entry Package 는 ROUTE 시그널링 내에 어떠한 Filter Code 도 정의되어 있지 않을 것으로 예상된다. 이로 인해 Entry Package 는 수신되는 즉시 항상 Application Context Cache 에 저장된다. Application Launch process 는 Entry Package 가 수신되기를 (8) 기다린 후, *HELD.HTMLEntryPackage@bcastEntryPageUrl* [3]을 사용하여 Broadcaster Application 을 실행 (16) 한다.

Broadcaster Application 은 실행을 시작한 (17) 후, 자신이 필요로 하는 NRT file 을 정의하기 위해 Set Filter Code Instances API (Section 9.7.1)를 호출한다 (18). 그림 6-3 에서, Broadcaster Application 에 의해 새로 정의된 Filter Code Instances 가 (7) 로 표시되어 있다. 이 중 "Filter Code 3" 과 "Filter Code 4"를 포함한 Filter Code Instances 는 "BA Lifetime"으로 표시된 만료 시간을 가지며, 이는 해당 인스턴스들이 Broadcaster Application 이 종료될 때 만료됨을 의미한다.

하나 이상의 Filter Code Instance 가 정의되면 (7), NRT 파일에 시그널링된 Filter Code 와 이미 설정된 Filter Code Instance 간의 일치 여부를 확인하는 검사 (15) 가 수행된다. 이 검사가 성공하면, 해당 파일은 (14) Application Context Cache (6)에 저장된다.

구체적인 검사 로직은 다음과 같다:

- *FDT-Instance.File@filterCodes* 가 존재하고, Filter Code Instance 값들과 *FDT-Instance.File@filterCodes* 리스트 간의 교집합이 비어 있지 않다면, 해당 파일을 Application Context Cache 에 저장한다.
- 그렇지 않고, *FDT-Instance@filterCodes* 가 존재하며, Filter Code Instance 값들과 *FDT-Instance@filterCodes* 리스트 간의 교집합이 비어 있지 않다면, 해당 파일을 Application Context Cache 에 저장한다.
- 위의 두 조건이 모두 만족되지 않으면, 해당 파일은 무시된다.

참고로, *FDT-Instance.File@filterCodes* 가 우선적으로 검사된다. 이는 A/331 표준에서, 전체 FDT-Instance 수준에서 설정된 Filter Code (*FDT-Instance@filterCodes*) 보다 개별 파일 수준의 Filter Code (*FDT-Instance.File@filterCodes*) 가 우선 적용되어야 한다고 명시하고 있기 때문이다 (참조: A/331 Section A.3.3.2.3 [3]). 따라서, 특정 파일에서 전체 *FDT-Instance@filterCodes* 로 정의된 Filter Code 를 제거하고자 할 경우, 해당 파일에 빈 *FDT-Instance.File@filterCodes* 속성을 지정함으로써 이를 무효화할 수 있다.

Filter Code Instances 및 NRT File Processing 가 계속 작동하는 동안 새 Filter Codes 가 NRT 파일과 함께 도착할 수 있고, Filter Code Instances 가 만료될 수 있으며, Broadcaster Application 이 새 Filter Code Instances 를 정의할 수 있습니다. NRT File Processing 루프는 Filter Code 신호 없이 해당

파일뿐만 아니라 하나 이상의 **Filter Code Instance** 값과 일치하는 하나 이상의 **Filter Code** 집합이 있는 파일뿐만 아니라 현재 *appContextId* 와 일치하는 수신된 파일에 대해 지속적으로 작동합니다. 다른 모든 파일은 무시되어야 합니다.

Receiver는 **Filter Code**를 기반으로 저장된 **Application Context Cache**의 파일을 추적할 필요가 없습니다. 즉, **Broadcaster Application** 이 **Filter Code Instance**를 변경하는 경우 **Receiver**는 현재 정의된 **Filter Code Instance**와 더 이상 일치하지 않는 파일을 캐시에서 제거하지 않을 것으로 예상됩니다. **Broadcaster Application**에서 이러한 파일이 더 이상 필요하지 않은 경우 **Mark Unused API** (Section 9.7)를 사용하여 파일을 불필요한 것으로 지정할 수 있습니다.

6.6 Security Considerations

모든 **Broadcaster Application** 파일은 방송망을 통해 전송되는 경우, A/331 Section A.3.3.5 [3] 및 A/360 Section 5.2 [8]에 기술된 **ROUTE Signed Package** 메커니즘을 사용하여 전송되어야 한다. A/331 은 **Broadcaster Application** 파일을 MIME multipart 패키지로 캡슐화하는 방식을 정의하고 있으며, A/360 은 이러한 MIME multipart 패키지를 S/MIME 래퍼로 캡슐화하여 **Broadcaster Application** 파일의 보안을 보장하는 방법을 규정하고 있다. 파일은 **Receiver Web Server** 인터페이스를 통해 **Application Context Cache** 내에서 접근할 수 있는 경우, 해당 **Broadcaster Application**의 일부로 간주된다. 서명 기능을 지원하는 **Receiver**의 경우, 올바르게 서명된 패키지의 콘텐츠만 **Receiver Web Server** 인터페이스를 통해 접근할 수 있도록 제공하는 것이 요구된다.

방송사 애플리케이션 파일들은 원하는 만큼의 서명된 패키지를 통해 방송망으로 전송될 수 있으며, 파일을 여러 서명 패키지로 분할하는 데에는 어떠한 제약도 없습니다. 실제로는 방송사 애플리케이션의 핵심 기능은 **Entry Package**를 통해 제공되고, 확장된 콘텐츠 및 기능은 별도의 패키지를 통해 제공되는 것이 일반적일 수 있습니다. 또한, **Filter Codes** 메커니즘(Section 9.9 참조)을 사용하여 사용자 선호도나 기타 선택 기준이 파악될 때, 다양한 패키지를 선택적으로 사용할 수도 있습니다. 파일이 여러 개의 별도 패키지로 분할되어 있더라도, 이러한 모든 패키지는 앞 문단에서 명시된 요구사항에 따라 반드시 서명되어야 합니다.

브로드밴드를 통해 전송되는 **Broadcaster Application** 파일은 표준 W3C 보안 메커니즘을 사용하여 보호되어야 합니다. 브로드밴드 서버와의 모든 연결은 A/360 Section 5.1 [8]에서 기술된 바와 같이 보안 연결을 사용해야 합니다. 보안 연결을 통해 브로드밴드로 수신된 콘텐츠는, 방송망을 통해 서명된 패키지로 수신된 콘텐츠와 동등하게 신뢰할 수 있는 것으로 간주됩니다.

6.7 Security Considerations

ATSC 3.0 Companion Device 표준(A/338 [37])은 보조 기기(Companion

Device, CD)라 불리는 별도의 장치가 **Receiver**, 즉 기본 기기(Primary Device, PD)와 상호작용하는 방법을 정의합니다. A/338 표준은 본 문서의 Section 8 과 Section 9 장에서 정의된 API 를 확장하여, **Broadcaster Application** 이 보조 기기에서 실행되는 CD 애플리케이션을 검색(discover)하고 실행(launch 할 수 있도록 하는 CD Manager API 를 제공합니다. 이 CD Manager API 는 또한 CD 애플리케이션과 **Broadcaster Application** 간의 애플리케이션-대-애플리케이션 통신을 가능하게 하는 **WebSocket** service end points 를 얻는 기능도 제공합니다. **Broadcaster Application** 은 여러 end point 를 요청함으로써 복수의 연결을 지원할 수 있습니다. **CD Manager API** 를 사용하기 위해 방송사 애플리케이션은 Section 8.2.1 절에 정의된 메커니즘을 통해 **WebSocket** URL 을 얻을 수 있습니다.

보조 기기 애플리케이션의 탐색 및 실행 절차, 애플리케이션 간 **WebSocket 통신 메커니즘**, 그리고 CD 애플리케이션이 사용할 수 있는 수신기 **WebSocket API** 에 대한 완전한 세부 내용은 ATSC 3.0 Companion Device 표준(A/338 [37])에 기술되어 있습니다.

7 MEDIA PLAYER

ATSC 3.0 **Receiver** 환경에서는 broadcast, broadband 망, 또는 재배포를 통해 전달된 미디어 콘텐츠를 재생할 수 있는 두 가지 소프트웨어 구성 요소가 존재합니다. 본 명세서에서는 이 두 논리적 구성 요소를 각각 Application Media Player(AMP)와 Receiver Media Player(RMP)라고 지칭하며, 본 Section 에서 자세히 설명합니다. **AMP** 는 HTML5 **Broadcaster Application** 의 일부로 포함되는 JavaScript 코드(e.g., DASH.js)입니다. **RMP** 는 수신기별 구현체(Receiver-specific implementation)입니다. **AMP** 는 HTML5 <video> 태그와 MSE(Media Source Extensions)를 사용하여, 콘텐츠의 출처(origin)나 전달 경로에 상관없이 미디어 콘텐츠를 재생할 수 있습니다. RMP 의 설계 및 구현 세부 사항은 본 명세서의 범위에 포함되지 않으며, 본 명세서에 제공되는 RMP 설계 설명은 단지 informative reference 입니다. AMP 또는 RMP 를 사용하여 미디어 콘텐츠를 재생할 경우, 여러 사용 사례(use case)가 존재합니다.

- Broadcast 또는 Hybrid Broadband / Broadcast Live Streaming - 콘텐츠 세그먼트가 통신망(Broadband) 또는 방송망(Broadcast)을 통해 전달됨.
- Broadband Media Streaming - 통신망을 통한 미디어 콘텐츠 스트리밍(On-demand 또는 Linear 서비스).
- Downloaded Media Content - 방송망 또는 통신망을 통해 사전에 다운로드된 미디어 콘텐츠. 방송망 또는 통신망을 통한 미디어 다운로드 방식의 세부 사항은 본 표준 Section 9.1.14 에 설명됨.

재생되는 미디어 스트림의 유형은 라이브 방송 스트림의 MPD 시그널링 또는 특정 Broadcaster Application 로직에 따라 결정됨.

DASH Client 명세서[41]는 이러한 플레이어의 동작 기대 사항을 제공하며, 본 문서에서는 추가 설명하지 않음.

재배포 시나리오에서는 ATSC 3.0 프로토콜을 사용하지 않는 방식(e.g., HDMI)으로 미디어 콘텐츠가 전달됨. 재배포 서비스는 RMP 를 통해 제공됨.

7.1 Utilizing RMP

RMP 는 방송망을 통해 스트리밍되는 미디어 콘텐츠를 재생하도록 Receiver 로직에 의해 트리거되거나, Broadcaster Application 의 명시적 요청에 의해 트리거될 수 있음. 이러한 구분 사항은 본 섹션에서 자세히 설명됨

7.1.1 Broadcast or Hybrid Broadband and Broadcast Live Streaming

새로운 서비스로 채널을 전환할 때, RMP 는 미디어 스트림을 즉시 재생할지, 아니면 Broadcaster Application 이 재생 여부를 결정할 때까지 대기할지 판단함. A/331 [3]에서 지정된 HELD 은 어떤 미디어 플레이어(AMP 또는 RMP)가 미디어 스트림을 재생할 것인지 신호를 제공하지만, Receiver 로직은 HELD 에 명시된 신호와 관계없이 미디어 스트림을 재생하도록 선택할 수 있음. ROUTE/DASH 의 MPD 또는 MMT 의 MP Table 에 포함된 정보가 미디어 스트림 세그먼트가 방송으로부터 재생될지, 혹은 통신망과 방송을 결합하여 재생될지 결정함.

7.1.2 Broadband Media Streaming

RMP 는 서비스 시그널링에서 broadband 망을 통해 MPD URL 을 제공하거나, Broadcaster Application 이 RMP 에게 스트림 재생을 요청할 경우, 브로드밴드 미디어 스트리밍으로 전달된 서비스를 재생할 수 있음. 이 문서에서는 라이브 브로드밴드 스트리밍 재생과 주문형(on-demand) 브로드밴드 재생 간의 구분은 두지 않음. 두 사용 사례에 대해 MPD 가 어떻게 구성되는지는 DASH 클라이언트 명세 [41]에서 추가로 설명됨.

7.1.3 Downloaded Media Content

Broadcaster Application 의 요청에 따라 RMP 는 broadband 망 또는 방송망을 통해 전달된 다운로드한 미디어 콘텐츠를 재생할 수 있습니다. Broadcaster Application 은 section 9.6.3 에 설명된 대로 RMP URL WebSocket API 설정을 사용하여 이러한 요청을 할 수 있습니다.

7.1.4 Redistribution

대화형 콘텐츠는 재배포 시나리오에서 지원될 수 있습니다. **Receiver**는 재배포 소스로부터 ATSC 3.0 서비스를 획득하고, RMP를 사용하여 서비스를 제공하고, A/336[5]에 설명된 대로 서비스 및 애플리케이션 시그널링을 획득하고, section 6.5에 설명된 대로 broadband망을 통해 **Broadcaster Application** 패키지를 획득할 수 있습니다.

7.2 Utilizing AMP

7.2.1 Broadcast or Hybrid Broadband and Broadcast Live Streaming

비록 방송 또는 하이브리드 라이브 미디어 스트리밍은 일반적으로 RMP에 의해 재생되지만, AMP가 콘텐츠 재생을 요청할 수도 있음. HELD 내 플래그는 RMP가 미디어 콘텐츠를 즉시 재생할 수 있는지, 아니면 서비스가 AMP가 라이브 미디어 스트리밍을 재생할 것으로 예상하는지를 나타냄. **Receiver**는 이 시그널링된 expectation을 무시할 수 있으며, 이 경우 RMP는 라이브 미디어 스트리밍을 즉시 재생할 수 있음. AMP가 미디어를 재생할 수 있는 여러 방법은 다음 섹션에서 설명됨.

7.2.2 Broadband Media Streaming

Broadband망을 통해 전달되는 미디어 스트림을 재생할 때는 DASH 클라이언트 표준 [41]에서 제공하는 내용 외에 특별한 고려사항이 없음. 기타 미디어 유형은 **Query Device Info API**(section 9.11 참조)를 통해 제공되는 장치 기능 정보에 따라 **Receiver**에서 지원될 수 있음.

7.2.3 Downloaded Media Content

AMP는 NRT를 사용하여 방송망에서 다운로드한 미디어 콘텐츠를 재생하거나 **Cache Request APIs**를 사용하여 broadband 망에서 재생할 수 있습니다(section 9.2.14 참조). 두 경우 모두 결과 미디어 콘텐츠는 **Application Context Cache**에 배치됩니다. 그런 다음 **Broadcaster Application**은 MSE 또는 EME와 함께 HTML5<video> 태그를 사용하여 playout을 시작할 수 있습니다.

7.2.4 AMP Utilizing the Pushed Media WebSocket Interface

이 메커니즘을 통해 AMP는 방송망을 통해 전달된 콘텐츠를 재생할 수 있습니다. **Broadcaster Application**은 section 8.2.1에 지정된 바이너리 **WebSocket** 연결을 엽니다. 이러한 **WebSocket** 연결을 여는 것은 수신자가 미디어 콘텐츠의 초기화 및 미디어 세그먼트를 검색하고 이러한 연결을 통해 **Broadcaster**

Application 에 전달하라는 암시적 요청입니다. 그런 다음 Broadcaster Application 은 MSE 또는 EME 와 함께 HTML5<비디오> 태그를 사용하여 디코딩 및 프레젠테이션을 위해 미디어 콘텐츠를 수신기의 디코더에 전달합니다. 의 경우 미디어는 ROUTE 클라이언트를 통해 브로드캐스트에서 검색되고 WebSocket 서버를 통해 AMP 로 푸시됩니다. Broadband 망의 경우 미디어는 HTTP 클라이언트를 통해 원격 HTTP 서버에서 검색되고 WebSocket 서버를 통해 AMP 로 푸시됩니다

8 ATSC 3.0 WEBSOCKET INTERFACE

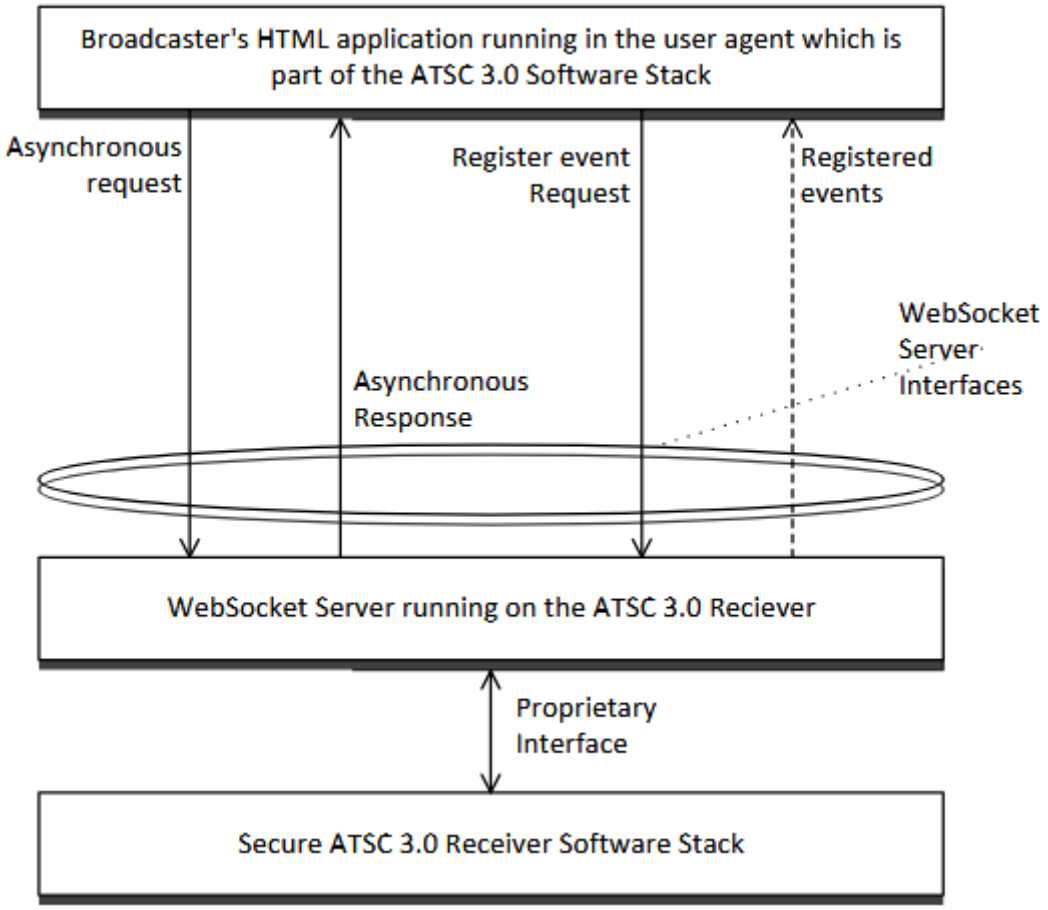
8.1 Introduction

Receiver 의 Broadcaster Application 은 Receiver 플랫폼과 정보를 교환하여 다음을 수행할 수 있습니다.

- 사용자 설정 검색
- Receiver 에서 Broadcaster Application 으로 이벤트 수신
  - 사용자 설정 변경 통지
  - DASH 스타일 또는 MPU 스타일 Event Stream 이벤트(방송망으로부터)
- Receiver 작업 요청

이러한 기능을 지원하기 위해 Receiver 는 웹 서버를 포함하고 WebSocket RPC 호출 집합을 노출합니다. 이러한 RPC 호출은 수신기와 수신기 플랫폼에서 실행되는 Broadcaster Application 간에 정보를 교환하는 데 사용할 수 있습니다. 그림 8.1-1 은 이러한 구성 요소 간의 상호 작용을 보여줍니다.

중앙 집중식 Receiver 아키텍처의 경우, Receiver Web Server 는 일반적으로 User Agent 의 Broadcaster Application 에 의해 Receiver 내에서만 액세스할 수 있습니다.



(그림 8.1-1) Communication with ATSC 3.0 Receiver.

[출처: A344]

하나 이상의 ATSC 3.0 WebSocket 인터페이스가 Receiver 에 의해 노출됩니다. 모든 수신기는 명령 및 제어에 사용되는 WebSocket 인터페이스를 지원합니다. 일부 수신기는 비디오, 오디오 및 캡션 바이너리 데이터에 대해 각각 하나씩 세 개의 추가 WebSocket 인터페이스도 지원합니다. Broadcaster Application 또는 컴패니언 디바이스는 명령 및 제어 인터페이스에 연결하여 수신기에서 상태 및 설정을 검색하고 채널 변경과 같은 작업을 수행할 수 있습니다.

8.2 Interface Binding

여기에 설명된 API는 WebSocket 인터페이스를 활용하기 때문에 Broadcaster Application 은 표준 브라우저 기능에 의존하여 연결을 열 수 있으며 Broadcaster Application 에 특정 기능이 있을 필요가 없습니다.

Receiver 가 제공하는 WebSocket 서버와 통신하기 위해서는 Broadcaster Application 이 WebSocket 서버의 URL을 알아야 합니다. WebSocket 서버 위치는 네트워크 토폴로지(예: 통합 아키텍처 대 분산 아키텍처)에 따라 다르거나 Receiver 구현에 따라 다를 수 있습니다. Broadcaster Application 에서 이러한 차이점을 숨기기 위해 Broadcaster Application Entry Page URL은 Receiver WebSocket 서버의 위치에 대한 정보를 제공하는 쿼리 용어 매개변수와 함께

시작됩니다.

Broadcaster Application 의 Entry Page 가 User Agent 에 로드되면 URL 에는 Receiver 가 지원하는 ATSC 3.0 WebSocket Server Interface 의 Base URI 를 제공하는 쿼리 용어가 포함되어야 합니다. 마찬가지로 수신기는 이 표준의 릴리스 날짜를 포함하는 다른 쿼리 용어를 제공하여 지원되는 WebSocket API 의 현재 버전을 보고해야 합니다. 추가 선택적 쿼리 용어는 *callerIdQuery* 입니다. 이 용어는 Broadcaster Application 이 다른 Broadcaster Application 에 의해 시작되었을 때 존재할 것으로 예상됩니다.

RFC 5234 [12]에 정의된 ABNF 구문을 사용하여 쿼리 구성 요소는 다음과 같이 정의되어야 합니다.

```
query = ((wsQuery "&" revQuery) / (revQuery "&" wsQuery)) [callerIdQuery]
wsQuery = "wsURL=" ws-url
revQuery = "rev=" yyyymmdd
callerIdQuery = "&callerId=" appId
```

ws-url 은 기본 WebSocket URI 이며 RFC 6455 [24]에 정의된 대로여야 합니다. *yyyymmdd* 값에는 현재 표준이 발표된 연도(*yyyy*), 월(*mm*) 및 일(*dd*)이 포함되어야 합니다. 예를 들어, 이 표준의 첫 번째 릴리스는 2018 년 12 월 18 일이었습니다. 이 값은 '20181218'로 표시됩니다. 지정된 릴리스에 사용된 날짜는 이 문서의 시작 부분에 있는 개정 내역 표의 '날짜' 열에 있는 해당 항목에서 가져와야 합니다. *appId* 는 Launch Broadcaster Application API 를 호출한 Broadcaster Application 의 *HELD@appId*(A/331 [3] 참조)이어야 합니다(section 9.6.6 참조). 이것은 한 가지 수준의 수익만 제공한다는 점에 유의하십시오.

다음은 이러한 쿼리 문자열을 사용하여 Broadcaster Application 을 시작하는 방법의 예를 보여줍니다. 이 예에서 항목 페이지 URL 이 다음과 같은 경우:

http://localhost/xbc.org/x.y.z/home.html,

WebSocket API 는 2018 년 7 월 20 일에 출시된 표준 개정판을 기반으로 하며 Broadcaster Application 은 *appId="pbs.org/kids/1"*를 사용하여 이전 Broadcaster Application 에 의해 시작되었으며, Broadcaster Application 은 다음과 같이 실행됩니다.

```
http://localhost/xbc.org/x.y.z/home.html?wsURL=wss://localhost2:8000
                                        &rev=20180720
                                        &callerId=pbs.org/kids/1
```

wsURL 및 *rev* 쿼리 매개 변수는 broadcast-delivered 애플리케이션의 Entry Page URL 을 로드하기 위해 추가됩니다. Broadband 웹 서버는 HTTP 요청의 URL 에 있는 *wsURL* 쿼리 매개 변수가 표시되는 경우 이를 무시할 것으로 예상됩니다. *rev* 쿼리 용어는 방송망 및 broadband 망에서 Broadcaster Application 을 시작하는 데 적용할 수 있습니다.

8.2.1 Interface Binding

모든 **Receivers**는 section 9에 설명된 API의 통신에 사용되는 **WebSocket** 인터페이스에 대한 액세스를 지원해야 합니다. 이진 미디어 데이터(비디오, 오디오 및 캡션)의 푸시 모드 배달을 지원하는 Receivers는 각 미디어 데이터 유형에 대해 하나씩 세 개의 추가 **WebSocket** 인터페이스도 지원합니다. A/338 Companion Device 표준[37]을 지원하는 Receivers는 Receiver 내에서 **CD Manager**와 통신할 수 있는 추가 **WebSocket** 인터페이스도 제공합니다(섹션 6.7 참조). 표 8.2.1-1 은 5개의 인터페이스를 설명합니다. 표에서 "*WSPath*"라는 용어는 위의 절차에서 검색된 *wsURL* 매개 변수의 값을 나타냅니다.

<표 8.2.1-1> WebSocket Server Functions and URLs
[출처: A344]

WebSocket Interface Function	URL	Receiver Support
Command and Control	WSPath/atscCmd	Required
Video	WSPath/atscVid	Optional
Audio	WSPath/atscAud	Optional
Captions	WSPath/atscCap	Optional
Companion Device	WSPath/atscCD	Optional

표 8.2.1-1 에 표시된 선택적 **WebSocket** URL 이 지원되지 않는 경우, **Broadcaster Application** 이 선택적 인터페이스에 연결을 시도할 때 **Receiver**는 HTTP 상태 코드 "*404 Not Found*"으로 응답해야 합니다. 이 상태를 수신하면 특정 **WebSocket** API에 대한 **WebSocket** 연결이 실패합니다.

푸시 모델에서 비디오/오디오/캡션 **WebSocket** 인터페이스를 통해 전달되는 각 MPEG DASH 미디어 세그먼트 파일은 **WebSocket** 프로토콜의 바이너리 프레임으로 전달됩니다. 명령 및 제어 인터페이스는 텍스트 프레임 전달을 사용합니다.

8.2.1.1 Initializing Pushed Media WebSocket Connections

표 8.2.1-1(*atscVid, atscAud, atscCap*)에 나열된 미디어 **WebSocket** 연결이 설정되면 이러한 연결을 통해 **Receiver** 가 전송한 첫 번째 데이터는 페이로드 "*/S*" 다음에 **Initialization Segment**가 있는 텍스트 메시지(IETF RFC 6455 [24]의 section 5.2 에 정의된 opcode 0x1)일 것으로 예상됩니다. **Initialization Segment** 후에 **Receiver 는** 페이로드 "*/S_end*"와 미디어 세그먼트가 포함된 다른 문자 메시지를 보내야 합니다. 미디어 전달 WebSocket 연결이 설정된 후 새 **Initialization Segment**가 수신되면 **Receiver**는 이전 **Initialization Segment**와 연결된 마지막 미디어 세그먼트 바로 뒤에 페이로드 "*/S*"가 있는 동일한 **WebSocket** 연결을 통해 문자 메시지를 보내야 합니다. 그런 다음 **Receiver**는 새 **Initialization Segment**를 보낸 다음 페이로드 "*/S_end*"이 포함된 문자 메시지를 보낸 다음 미디어 세그먼트를 전송해야 합니다.

8.2.1.2 Initializing Pushed Media WebSocket Connections

Broadcaster Application 이 미디어 WebSocket 연결을 요청하면, Receiver 는 연결이 설정된 후 해당 콘텐츠를 전송하기 시작해야 함. Receiver 가 미디어 WebSocket 연결을 통해 전송하는 데이터는 표 8.2.1-1 에 정의된 WebSocket Interface Function 에서 지정한 미디어 유형과 일치해야 함. 따라서 A/331 [3]에서 정의된 형식에 부합하는 비디오는 수신기에 의해 URL *WSPath/atscVid* 로 식별되는 WebSocket 을 통해 전송되어야 함. 마찬가지로, A/331 [3]에서 정의된 형식에 부합하는 오디오는 *WSPath/atscAud*, 자막은 *WSPath/atscCap* WebSocket 연결을 통해 각각 전송되어야 함. 현재 열려 있는 미디어 WebSocket 연결의 경우, 예를 들어 특정 시점에 자막이 존재하지 않는 경우와 같이, 모든 시간에 콘텐츠가 전송되지 않을 수도 있음.

8.3 Data Binding

Receiver WebSocket 명령 및 제어 서버에 대한 연결이 설정되면, 메시지를 송수신할 수 있음. 그러나 WebSocket 인터페이스는 message framing 을 제외하고는 별도의 구조가 없는 단순한 양방향 인터페이스이므로, 메시지 형식을 정의할 필요가 있음.

명령 및 제어용 WebSocket 인터페이스는 부록 B 에 정의된 JSON-RPC 2.0 표준을 따라야 하며, Section B.4(*batch mode operation*)에 기술된 기능은 포함하지 않아도 됨. JSON-RPC 는 JavaScript Object Notation (JSON) 데이터 구조 [22]를 사용하여, 단방향 알림(unidirectional notifications)과 명확하게 정의된 오류 처리(well-defined error handling)를 포함하는 RPC(Remote Procedure Call, 원격 프로시저 호출) 방식의 메시징을 제공함.

이 절에서는 메시지의 기본 형식을 정의하며, 다음 절에서는 지원되는 구체적인 메시지들을 정의함. 본 문서는 메시지의 의미론(semantics)을 설명하고, 별도의 스키마 파일에서 규범적 구문(normative syntax)을 제공함. 별도의 스키마 파일에서 지정된 메시지 구문은 JSON Schema 표준 [19]에 정의된 대로 따라야 함. JSON schema 에 대한 추가적인 참고 정보는 [47]에서 확인할 수 있음.

Receiver WebSocket 명령 및 제어 서버와의 연결이 한 번 열리면, 해당 연결은 Broadcaster Application 의 수명 동안 유지되는 것이 일반적임. 명령 및 제어용 WebSocket 인터페이스를 닫았다가 다시 연결을 재설정(reestablishing)하는 동작은 정의되어 있지 않으며, 상태 손실(state loss)을 초래할 수 있음. 예를 들어, 수신기로부터 키 타임아웃(key timeout)을 수신하지 못하거나, 수신기 자원이 재할당되는 등의 문제가 발생할 수 있음.

이 문서에 제시된 표에서 암시적으로 정의된 JSON 스키마 정의와, 별도의 JSON 스키마 정의 파일에 포함된 정의 간에 불일치가 발생할 경우, JSON 스키마 정의 파일에 포함된 정의가 우선하며, 이를 권위 있는 기준으로 간주함.

“Data Type” 열에 사용된 용어들은 JSON Schema [19]에 정의된 데이터 타입(datatype)을 간략히 표현한 약어이며, 해당 문서에 정의된 대로 적용되어야

함.

향후 스키마 변경에 대한 유연성을 확보하기 위해, 본 문서에서 정의된 JSON 문서의 디코더는 인식할 수 없는 데이터 타입을 오류로 처리하지 말고 무시하도록 해야 함.

JSON 스키마 및 메시지는 UTF-8 로 인코딩된 문자열 형식으로 표현되어야 함. 수신기는 JSON 메시지를 분석하여, 메시지의 메서드를 올바른 핸들러에 전달하고, 이후의 처리 절차를 수행해야 함.

명령 및 제어용 WebSocket 인터페이스에 대해서는 여러 유형의 데이터 메시지가 정의되어 있음.

- 메시지 요청 - 정보를 요청하거나 작업을 시작하는 데 사용됩니다.
- 동기식 응답 - 즉시 제공된 요청에 대한 확실한 답변
- 비동기 응답 - 비동기적으로 제공된 요청에 대한 확실한 답변
- 오류 응답 - 제공된 요청에 대한 최종 오류
- 알림 - 단방향 알림, 동기 또는 비동기 응답이 예상되지 않음

다른 세 개의 **WebSocket** 인터페이스는 **Receive** 에서 **Broadcaster Application** 으로 바이너리 데이터를 전달하는 데 사용됩니다.

이 표준의 예에서 데이터 흐름을 설명하는 데 사용되는 표기법은 다음과 같습니다.

<표 8.3-1> 데이터 흐름 표기법
[출처: A344]

```
--> data sent to Receiver
<-- data sent to Broadcaster Application
```

참고: 인터페이스는 양방향이므로 요청, 응답 및 알림은 수신기 또는 Broadcaster Application 에서 시작할 수 있습니다

<표 8.3-1> Request/response example
[출처: A344]

```
--> {"jsonrpc": "2.0",
       "method": "exampleMethod1",
       "params": 1,
       "id": 1
}
<-- {
       "jsonrpc": "2.0",
       "result": 1,
       "id": 1
}
```

<표 8.3-2> Notification example
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "update",
       "params": [1,2,3,4,5]
}
```

'*id*'속성이 없다는 것은 알림의 경우 응답이 예상되지 않음을 나타냅니다.

<표 8.3-2> Error example

[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "faultyMethod",
        "params": 1,
        "id": 6
}
<-- {
        "jsonrpc": "2.0",
        "error": {"code": -32601, "message": "Method not found"},
        "id": 6
}
```

8.3.1 General JSON Property Considerations

다음 절에서 설명하는 **Cancel Request Command** 및 Section 9 에 정의된 API 들은 요청 및 응답 메서드에서 사용되는 JSON 객체의 허용 가능한 구문(syntax)을 정의하기 위해 JSON schema [19]를 사용함. 각 스키마는 별도의 파일로 관리되며, 스키마 내에 지정된 요소(elements)와 속성(properties)을 설명하는 본문의 내용과 함께 모두 규범적(normative)으로 간주됨. 반면, 제공되는 예제(examples)는 참고용(informative) 으로서 API 의 의미와 사용 방식을 명확히 설명하기 위한 것이며, 예제와 규범적 스키마 간에 차이가 있을 경우, 그러한 차이는 의도된 것이 아님(unintentional)을 명시함.

API 용 스키마는 일부 속성이 필수(required) 임을 명시할 수 있음. 이는 해당 속성이 스키마에 부합하는 JSON 구조 내에 key 로 반드시 존재해야 함을 의미함. 단순 타입(simple types)의 경우, 별도로 명시되지 않는 한, **Receiver** 또는 **Broadcaster Application** 이 해당 키에 대한 충분한 정보를 제공할 수 없을 때는 "*null*" 값을 사용할 수 있음. 의미적으로 "*null*" 값은 구현체가 해당 키가 필수임을 인지하고 있으나, 의미 있는 데이터를 제공할 수 없음을 나타냄. 필수 객체나 배열 구조의 경우, 정보가 없는 경우라도 별도의 명시가 없는 한 빈 구조체를 제공해야 함. 단, 해당 구조의 스키마에서 내부 속성이 필수로 정의되어 있는 경우에는 예외로 함.

8.3.2 Cancel Request Command

Cancel Request Command 을 사용하여 선택한 수 또는 모든 미해결 **JSON-RPC** 요청을 종료할 수 있습니다. 요청 메시지는 요청 ID 에 해당하는 ID 목록을 제공합니다. 응답 메시지에는 종료된 요청이 나열됩니다. 요청된 ID 와 일치하는 요청을 찾을 수 없는 경우 오류 응답이 제공됩니다. 요청이 성공적으로 취소되면 **Receiver** 는 요청이 취소되었음을 나타내는 오류 코드와 함께 해당 요청에 대한 응답을 발행해야 합니다. 취소 요청은 이전 취소 요청을 취소하는 데 사용되지 않을 것으로 예상됩니다.

Cancel Request Command 의미 체계는 표 8.3.2-1 에 정의된 대로이며 구문은 스키마 파일 *cancel-request.json* 에 정의된 대로입니다. 매개 변수의

추가 의미 체계 정의는 표를 따릅니다.

<표 8.3.2-1> Cancel Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	“cancel”
params	0..1		
requestIDs	1		An array of one or more request IDs to be canceled.
items	1..N	integer	

requestIDs - 선택적 *params* 개체의 이 목록에는 미해결 요청의 하나 이상의 ID 가 포함되어야 합니다. *params* 개체 및 해당 하위 *requestIDs* 목록이 제공되지 않으면 모든 미해결 요청이 취소되어야 합니다.

Cancel Response 의미 체계는 표 8.3.2-1 에 정의되어 있으며 구문은 스키마 파일 *cancel-response.json* 에 정의된 대로입니다. 추가 의미 체계 정의가 있는 경우는 표를 따릅니다.

<표 8.3.2-2> Cancel Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	Matches the request id value
result	oneOf X	string	Returned on successful request otherwise the error structure is returned
cancelList	1		An array of requests and their disposition in response to the cancel request
Items	1..N	integer	
requestID	1		The request ID of a request that was requested to be canceled
disposition	1		One of "CANCELED", "UNKNOWN", or "FAILED"
description	0..1		Information regarding the cancel operation
error	oneOf X		See Section 8.3.3

Receiver 는 단일 응답을 제공할 수 있도록 취소된 요청 ID 를 누적해야 합니다. 각 요청은 요청이 취소되었음을 나타내는 별도의 오류 응답을 받습니다.

cancelList - 이 필수 속성은 취소 요청에 대한 응답으로 개체 목록을 제공해야 합니다. 목록에는 취소 명령의 요청 ID 수 또는 모든 미해결 요청을 취소하도록 요청하는 요청 ID 가 제공되지 않은 경우 미해결 요청 수와 동일한 수의 오브젝트가 있을 것으로 예상됩니다.

requestID - 이 필수 속성에는 취소 명령에 제공된 요청 ID 중 하나가 포함되거나 모든 요청이 취소되는 경우 취소된 요청의 *requestID* 가 포함되어야 합니다.

disposition - 이 필수 속성에는 다음 값 중 하나가 포함되어야 합니다.

CANCELED - 객체의 *requestID* 에 해당하는 요청이 성공적으로 취소되었음을 나타냅니다.

UNKNOWN - 객체의 *requestID* 에 해당하는 요청을 찾을 수 없음을 나타냅니다. 수신자가 지정된 *requestID* 에 해당하는 미해결 요청을 식별할 수 없거나 취소 요청이 처리되기 전에 해당 ID 값을 가진 요청이 완료되었기 때문에 요청이 취소되지 않았습니다.

FAILED - 객체의 *requestID* 에 해당하는 요청을 취소할 수 없음을 나타냅니다. 이 경우 **Receiver** 가 지정된 요청을 찾았지만 요청을 취소할 수 없습니다.

description - 이 선택적 속성에는 취소 작업에 대한 설명이 포함되어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application** 은 ID '12'로 쿼리 요청을 만들고, 쿼리가 필요하지 않은 사용자 작업으로 인해 **Broadcaster Application** 은 다음 명령으로 요청을 취소합니다.

<표 8.3.2-3> Example of Cancel Request 1
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "cancel",
       "params": {
       "requestIDs": [12]
       },
       "id": 913
}
```

Receiver 는 다음과 같이 취소 요청에 응답할 수 있습니다.

<표 8.3.2-4> Example of Cancel Response 1-1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
              "cancelList": [
                     {"requestID": 12,
                     "disposition": "CANCELED",
                     "description": "Request canceled successfully"}
              ]
       },
       "id": 913
}
```

Receiver 는 ID 12 의 쿼리 요청에 대해 다음 응답을 실행할 수도 있습니다.

<표 8.3.2-5> Example of Cancel Response 1-2
[출처: A344]

```
<-- {
  "jsonrpc": "2.0",
  "error": {"code": -20, "message": "Request Canceled" },
  "id": 12
}
```

또 다른 예로, **Broadcaster Application**은 사용자가 동작 모드를 전환했기 때문에 함수의 모든 미해결 요청을 취소하기를 원할 수 있다. 이 경우 **Broadcaster Application**은 다음과 같이 해당 미해결 요청을 취소하려고 시도합니다.

<표 8.3.2-6> Example of Cancel Request 2
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "cancel",
       "params": {
       "requestIDs": [42, 216, 922]
       },
       "id": 226
}
```

Receiver는 다음과 같이 취소 요청에 응답할 수 있습니다.

<표 8.3.2-7> Example of Cancel Response 2
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
              "cancelList": [
                     {"requestID": 42,
                     "disposition": "UNKNOWN",
                     "description": "Request is not currently pending"},
                     {"requestID": 216,
                     "disposition": "CANCELED",
                     "description": "Request canceled successfully"},
                     {"requestID": 922,
                     "disposition": "CANCELED",
                     "description": "Request canceled successfully"}
              ]
       },
       "id": 226
}
```

Receiver는 ID가 **216과 922**인 query request에 대해 error response을 보낸다. 단, *requestID* 42의 상태가 **UNKNOWN**으로 표시되는 것은 오류가 아닐 수도 있다. 이는 취소 명령이 전송되는 시점에 해당 요청이 이미 완료되었을 가능성이 있기 때문이다.

또 다른 예로, **Broadcaster Application**이 모든 동작을 종료하고 현재 처리 중인 모든 요청을 종료하고자 하는 경우가 있다. 이때 방송사 애플리케이션은 다음과

같이 outstanding requests 을 취소하려 시도한다.

<표 8.3.2-8> Example of Cancel Request 3
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "cancel",
        "id": 226
}
```

Receiver 는 다음과 같이 취소 요청에 응답할 수 있습니다.

<표 8.3.2-9> Example of Cancel Response 3
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "cancelList": [
                        { "requestID": 324,
                        "disposition": "CANCELED",
                        "description": "Request canceled successfully" },
                        { "requestID": 167,
                        "disposition": "CANCELED",
                        "description": "Request canceled successfully" }
                ]
        },
        "id": 226
}
```

또한 **Receiver** 는 ID 324 및 167 을 사용하는 쿼리 요청에 대한 error response 을 발행합니다. **Broadcaster Application** 은 모든 outstanding requests 이 취소되었다고 가정할 수 있습니다

8.3.3 Error Handling

JSON-RPC 2.0 은 예약된 오류 코드 집합을 정의합니다. Section B.3.1 의 표를 참조하십시오. 오류 구조체의 의미론(semantics)은 Section B.3.1 에도 정의되어 있으며 구조는 표 8.3.3-2 에 나와 있습니다.

<표 8.3.3-1> Cancel Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치합니다.
result	oneOf X		API 에 종속된 구조로 성공적인 요청 시 반환되며 그렇지 않으면 오류 구조가 반환됩니다.
error	oneOf X		Section B.3.1 참조
code	1	Integer	어떤 문제가 발생했는지를 나타내는 오류 코드입니다.
message	1	String	오류를 설명하는 간결한 메시지

	data	0..1		오류에 대한 추가 정보를 포함하는 선택적 기본 형식 또는 개체

error – 오류가 발생할 경우 "*result*" 구조 대신 제공됩니다. 오류 반환에 대한 규범적 설명은 Section B.3.1 오류 개체를 참조하세요. 일반 JSON RPC 오류 코드는 부록 B 에 정의되어 있습니다. 개별 API 응답에서 제공하는 특정 오류 코드는 각 API 에 대한 응답 정의에 자세히 설명되어 있습니다. 수신기의 ATSC 정의 오류 코드 요약은 표 8.3.3-2 에 나열되어 있습니다. 별표 열은 API 요청에 대한 오류 응답에서 관련 오류 코드가 발생할 수 있음을 나타냅니다.

<표 8.3.3-2> JSON-RPC ATSC Error Codes
[출처: A344]

Code	*	Message	Meaning
-1	*	비인가	도메인 제한으로 인해 요청을 수락할 수 없습니다.
-2	*	자원 부족	요청을 준수하는 데 사용할 수 있는 리소스가 없습니다..
-3	*	시스템 대기	시스템이 대기 상태입니다. 요청을 수락할 수 없습니다.
-4		콘텐츠 없음	요청된 콘텐츠를 찾을 수 없습니다. 예를 들어 잘못된 URL 입니다.
-5		Broadband 망 연결 불가	요청을 준수할 수 있는 broadband 망 연결이 없습니다.
-6		서비스 없음	요청된 서비스를 찾을 수 없습니다.
-7		비인가 서비스	요청된 서비스를 획득했지만 조건부 액세스 제한으로 인해 볼 수 있는 권한이 없습니다.
-8		비디오 크기 조정/위치 실패	비디오 크기 조정 및/또는 위치 지정에 대한 요청이 성공하지 못했습니다.
-9		XLink 를 확인 불가	XLink 확인 요청이 실패했습니다.
-10		트랙을 선택 불가	미디어 트랙 선택 API 에서 식별된 미디어 트랙을 찾거나 선택할 수 없습니다.
-11		표시된 MPD 에 액세스 불가	RMP URL 설정 API 에 대한 응답으로 제공된 URL 에서 참조되는 MPD 에 액세스할 수 없습니다
-12		콘텐츠를 재생불가	RMP URL 설정 API 에 대한 응답으로 요청된 콘텐츠를 재생할 수 없습니다.
-13		요청된 MPD 앵커에 연결 불가	RMP URL 설정 API 에 대한 응답으로 표시된 MPD 앵커에 도달할 수 없습니다(예: 파일 끝 너머).
-14		지원되지 않거나 알 수 없는 콘텐츠 보호 시스템	지정된 콘텐츠 보호 시스템이 수신자에서 지원되지 않거나 알 수 없습니다.
-15		잘못된 URL 형식	요청의 sourceURL 또는 targetURL 에 지정된 URL 형식이 잘못되었습니다.
-16		잘못된 URL 형식	요청된 목록의 하나 이상의 URL 에 지정된 URL 형식은 illegal 입니다.
-17		잘못된 형식의 DASH Period	Period 에 지정된 MPEG DASH 조각의 형식이 잘못되었습니다.
-18		MPD 없음	참조된 MPD 파일을 찾을 수 없습니다.
-19		rmpSyncTime 으로 지정된 동기화를 수행 불가	rmpSyncTime 을 사용하여 RMP URL API 설정에 대한 응답으로 rmpSyncTime 으로 표시된 동기화를 달성할 수 없습니다.
-20	*	요청 취소	브로드캐스터 애플리케이션은 이 요청에 해당하는 ID 를 사용하여 취소 요청 명령(section 8.3.1)을 실행했습니다.

-21		현재 소스에서 RMP 재생을 변경하는 것은 지원 불가	RMP URL 설정 API 에 대한 응답으로 수신기는 현재 소스에서 대체 소스(예: broadband 망 또는 로컬로 캐시된 콘텐츠)로 재생을 변경하는 것을 지원하지 않습니다.
-22		dialogEnhancement 실패	대화 개선 데이터를 설정하거나 수신하기 위한 요청이 실패했습니다.
-23		appId 없음	요청된 Broadcaster Application 의 appId 가 HELD 에 없습니다.
-24		구독하지 않음	Broadcaster Application 이 어떤 요청된 알람도 구독하지 않았습니다.
-25		Referenced BA 사용 불가	appId 가 HELD 에서 발견되었지만 사용할 수 없거나 브로드캐스트 전용이며 아직 획득되지 않았습니다.
-26		referenced broadband BA 엑세스 불가	appId 는 HELD 에서 발견되었으며 broadband 망 전용이지만 네트워크 연결 부족으로 인해 사용할 수 없습니다
-27		수신기 기능 없음	수신기가 필요한 기능을 지원하지 않습니다.
-28		알 수 없는 필터 코드	제공된 필터 코드 중 하나 이상이 현재 정의되지 않았습니다.
-29		더 이상 사용되지 않음(알 수 없는 DRM 시스템)	더 이상 사용되지 않음, 오류 코드 -14 참조
-30		매우 늦은	기본 콘텐츠를 대체하기 위해 요청이 제때 수신되지 않았습니다.
-31		알려지지 않은 XLink	제공된 elementType 및/또는 elementId 를 찾을 수 없습니다.
-32		잘못된 요소	제공된 MPD 요소가 유효하지 않습니다.
-33		Asset 선택 불가	자산을 선택할 수 없습니다. MMT 미디어 자산 선택 API 에서 식별된 미디어 자산을 찾거나 선택할 수 없습니다.
-34		MMT Asset 없음	assetLink, assetType 및/또는 assetId 로 설명된 MMT assets 을 찾을 수 없습니다.
-35		대체 MMT asset 이 유효하지 않음	대체 콘텐츠로 제공된 MMT asset 콘텐츠는 유효하지 않습니다.
-100		EME TypeError	See EME Section 6.5 [31]
-101		EME NotSupportedError	See EME Section 6.5 [31]
-102		EME InvalidStateError	See EME Section 6.5 [31]
-103		EME QuotaExceededError	See EME Section 6.5 [31]
-200..-229		예약	A/338 오류 코드용으로 예약됨 [37]

9 SUPPORTED METHODS

이 장에서는 명령 및 제어 **WebSocket** 인터페이스에서 지원되는 메서드에 대해 설명합니다. 이러한 API 는 섹션 8 에 설명된 대로 **WebSocket** 을 통한 JSON-RPC 2.0 을 기반으로 합니다. 인터페이스 및 데이터 바인딩에 대한 자세한 내용은 위의 장을 참조하십시오. 모든 메서드는 점 "."로 구분된 역방향 도메인 표기법입니다. 인터페이스를 통해 사용할 수 있는 모든 ATSC 메서드에는 "*org.atsc*"라는 접두사가 붙어 향후 새 **Receiver** API 에 대해 다른 메서드를 정의할 수 있는 여지를 남깁니다

이 섹션에 설명된 API 와 section 8.3.2 에서 이전에 설명한 취소 API 에 대한 스키마 및 예제는 다음에서 찾을 수 있습니다.

https://atsc-schemas.org/atsc3.0/a344/20250226/

스키마 저장소에 있는 스키마는 신뢰할 수 있습니다. 이 문서에 제공된 구문 및 예제는 유익한 것으로 간주됩니다.

9.1 Receiver Query APIs

Receiver 소프트웨어 스택은 다음 section에 설명된 대로 사용자 설정 및 정보를 검색하기 위해 **Broadcaster Application**에 **WebSocket API** 세트를 노출합니다.

Receiver에서 이러한 설정을 사용할 수 없는 경우 **Broadcaster Application**은 자체 비즈니스 정책 및 로직에 따라 기본값을 사용할 수 있습니다. **Broadcaster Application**은 자체 설정 사용자 인터페이스를 제공하고 수집된 설정을 **Receiver**에 쿠키로 저장하도록 선택할 수 있습니다.

다음 API는 **Broadcaster Application**이 이러한 설정 및 정보를 검색할 수 있도록 정의됩니다

9.1.1 Query Content Advisory Rating API

Broadcaster Application은 **Receiver**에 의해 현재 렌더링되고 있는 콘텐츠에서 신호된 현재 콘텐츠 권고 등급과 해당 콘텐츠가 차단되고 있는지 여부를 검색하기를 원할 수 있습니다. 콘텐츠가 차단되면 특정 API에 대한 액세스가 제한될 수 있지만 브로드캐스터 애플리케이션은 차단되지 않고 계속 실행된다고 가정합니다.

Query Content Advisory Rating Request의 Semantics는 표 9.1.1-1에 정의되어 있으며 구문은 스키마 파일에 정의된 대로여야 org.atsc.query.ratingLevel-request.json.

<표 9.1.1-1> Query Content Advisory Rating Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.query.ratingLevel

Query Content Advisory Rating Response의 Semantics는 표 9.1.1-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.ratingLevel-response.json에 정의된 대로여야 합니다.

매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.1.1-2> Query Content Advisory Rating Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	요청 ID 값과 일치
result	oneOf X	string	요청이 성공하면 빈 객체(empty object) 가 반환

				요청에 실패하면 오류 구조(error structure) 가 반환
	blocked	1	boolean	"true"인 경우 현재 콘텐츠가 표시되지 않도록 차단됨
	contentRating	1	string	현재 재생 중인 콘텐츠에서 발견된 등급
error		oneOf X		Section 8.3.3 참조

blocked - 이 필수 Boolean 값은 서비스의 콘텐츠 권고 등급이 콘텐츠 권고 등급 기본 설정보다 높기 때문에 **Receiver** 가 현재 콘텐츠를 차단하고 있는지 여부를 나타냅니다.

contentRating - 이 필수 문자열 값은 A/331 [3], section 7.3 에 정의된 대로 문자열 형식으로 **Receiver Media Player** 에서 현재 렌더링되는 콘텐츠의 콘텐츠 권고 등급을 제공해야 합니다. 콘텐츠 등급 문자열에는 해당하는 경우 여러 등급 영역을 포함하여 모든 등급 값이 포함되어야 합니다. 여러 등급 지역에 대한 콘텐츠 권고 정보 데이터를 지정하려면 추가 3 부분 문자열(각 지역당 하나씩)을 연결하여 여러 개의 연결된 3 단계 문자열로 구성된 하나의 문자열을 만들어야 합니다. 이 경우 마지막 부분을 제외한 각 콘텐츠 권고 정보 문자열의 세 번째 부분 뒤에는 쉼표(",")가 옵니다. 따라서 전체 콘텐츠 권고 등급 문자열의 마지막 문자는 오른쪽 중괄호("}")입니다. 콘텐츠 등급이 없는 경우 이 속성은 존재하고 빈 문자열(예: "*contentRating*":"")으로 설정되어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 콘텐츠 권고 등급 설정이 미국 등급 지역 1 에 대해 TV-PG-D-L 이고 현재 콘텐츠 권고 등급이 TV-G 인 경우를 고려합니다. **Broadcaster Application** 은 요청을 할 수 있습니다.

<표 9.1.1-3> Example of Query Content Advisory Rating Request [출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.ratingLevel",
      "id": 37
}
```

Receiver 는 다음과 같이 응답할 것입니다.

<표 9.1.1-4> Example of Query Content Advisory Rating Response [출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {
            "blocked": false,
            "contentRating": "1,'TV-G', {0 'TV-G'}"
      },
      "id": 37
}
```

9.1.2 Query Closed Captions Enabled/Disabled API

Broadcaster Application 은 사용자가 자막을 켠 것인지 여부를 알고 싶어할 수 있습니다. **Broadcaster Application** 은 **Receiver WebSocket Server** 인터페이스를 통해 **Receiver** 에 자막 설정을 요청합니다.

Query Closed Captions Enabled/Disabled Request 의 Semantics 는 표 9.1.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.cc-request.json 에 정의된 대로여야 합니다.

<표 9.1.2-1> Query Closed Captions Enabled/Disabled Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.query.cc"

Query Closed Captions Enabled/Disabled Response 의 Semantics 는 표 9.1.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.cc-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.2-2> Query Closed Captions Enabled/Disabled Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	요청 ID 값과 일치
result	oneOf X	string	요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
ccEnabled	1	boolean	자막 사용 여부
error	oneOf X		Section 8.3.3 참조

ccEnabled - 이 필수 부울은 자막이 현재 표시되고 있는 경우 true 를 나타내고 그렇지 않으면 false 를 나타냅니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 자막이 현재 활성화된 경우:

<표 9.1.2-3> Example of Query Closed Captions Enabled/Disabled Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.cc",
       "id": 49
}
```

Receiver 는 다음과 같이 응답할 것입니다.

<표 9.1.2-4> Example of Query Closed Captions Enabled/Disabled Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {"ccEnabled": true},
       "id": 49
}
```

9.1.3 Query Service ID API

동일한 애플리케이션이 동일한 브로드캐스트 패밀리 내의 여러 서비스에 사용될 수 있기 때문에, **Broadcaster Application** 은 현재 선택된 서비스를 알기를 원할 수 있다. 이를 통해 **Broadcaster Application** 은 사용자 인터페이스를 조정하고 한 서비스와 다른 서비스에서 사용할 수 있는 추가 기능을 제공할 수 있습니다.

Query Service ID Request Semantics 는 표 9.1.3-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.service-request.json 에 정의된 대로여야 합니다.

<표 9.1.3-1> Query Service ID Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.query.service"

Query Service ID Response Semantics 는 표 9.1.3-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.service-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.3-2> Query Service ID Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	요청 ID 값과 일치
result	oneOf X	string	요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
service	1	string(uri)	현재 선택된 서비스의 *globalServiceID* 를 지정
error	oneOf X		Section 8.3.3 참조

service - 이 필수 속성은 현재 선택된 서비스에 연결된 *globalServiceID* 를 나타내야 하며, 이는 SLT 내의 **SLT.Service**@*globalServiceID* 에 정의되어 있다. (A/331 [3] section 6.3 참조) *globalServiceID* 는 사용자가 선택할 수 있는

"일반적인(normal)" 서비스 유형(예: *@serviceCategory* 1 및 2)에 대해 필수이며, 반드시 SLT 에 존재해야 한다. *globalServiceID* 가 필요하지 않은 서비스 유형(예: DRM, ESG, NRT)의 경우, **Receiver** 는 section 8.3.1 에서 설명된 동작에 따라 "null" 값을 반환해야 한다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 현재 선택한 서비스에 대한 *globalServiceID* 가 "https://doi.org/10.5239/8A23-2B0B"이고 **Broadcaster Application** 이 **Receiver** 에 요청을 발행하는 경우:

<표 9.1.3-3> Example of Query Service ID Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.service",
        "id": 55
}
```

Receiver 는 다음과 같이 응답할 것입니다.

<표 9.1.3-4> Example of Query Service ID Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"service": "https://doi.org/10.5239/8A23-2B0B"},
        "id": 55
}
```

9.1.4 Query Language Preferences API

Broadcaster Application 은 오디오 출력, 사용자 인터페이스 디스플레이 및 자막/캡션을 위해 선택한 언어를 포함하여 **Receiver** 의 언어 설정을 알고 싶어할 수 있습니다. **Broadcaster Application** 은 **Query Language Preferences API** 를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Language Preferences Request 의 Semantics 는 표 9.1.4-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.languages-request.json 에 정의된 대로여야 합니다.

<표 9.1.4-1> Query Language Preferences Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.query.languages"

Query Language Preferences Response 의 Semantics 는 표 9.1.4-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.languages-

response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.1.4-2> Query Language Preferences Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
preferredUILang	1	string	수신기 사용자 인터페이스의 기본 언어를 제공함
preferredAudioLang	0..1	string	오디오 출력에 기본 언어를 제공함
preferredCaption-SubtitleLang	0..1	string	자막 또는 자막의 기본 언어를 제공함
error	oneOf X		Section 8.3.3 참조

preferredUILang, *preferredAudioLang*, *preferredCaptionSubtitleLang* - 이러한 각 문자열은 BCP 47[21]에 따라 코딩된 각 항목의 현재 설정된 언어 기본 설정을 나타냅니다. 최소한 **Receivers** 는 현재 UI 언어를 *preferredUILang* 으로 제공해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.1.4-3> Example of Query Language Preferences Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.languages",
        "id": 95
}
```

또한 사용자가 미국에 거주하지만 오디오 트랙 및 캡션/자막에 대한 언어 기본 설정을 스페인어로 설정한 경우 **Receiver** 는 다음과 같이 응답할 수 있습니다.

<표 9.1.4-4> Example of Query Language Preferences Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "preferredAudioLang": "es",
                "preferredUILang": "en",
                "preferredCaptionSubtitleLang": "es"
        },
        "id": 95
}
```

9.1.5 Query Caption Display Preferences API

Broadcaster Application 은 글꼴 선택, 색상, 불투명도 및 크기, 배경색 및 불투명도, 및 기타 특성을 포함하여 자막 디스플레이에 대한 사용자의 선호도를 알고 싶어할 수 있습니다. Broadcaster Application 은 Query Caption Display Preferences API 를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Caption Display Preferences Request Semantics 는 표 9.1.5-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.captionDisplay-request.json 에 정의된 대로입니다.

<표 9.1.5-1> Query Caption Display Preferences Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.query.captionDisplay"

Query Caption Display Preferences Response 의미 체계는 표 9.1.5-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.captionDisplay-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.1.5-2> Query Caption Display Preferences Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
cta708	0..1	object	section 9.1.5.1 의 semantics 정의를 참조
imsc1	0..1	object	section 9.1.5.2 의 semantics 정의를 참조
error	oneOf X		Section 8.3.3 참조

자막 표시 기본 설정을 정의하는 하나 이상의 캡션 체계 객체가 알림 메시지에 포함될 수 있습니다. 이러한 개체는 다음 하위 섹션에 정의되어 있습니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

9.1.5.1 CTA 708 Semantics

"cta708" 객체가 있는 경우, 표 9.1.5.1-1 에 설명된 대로 하나 이상의 자막 표시 기본 설정을 제공할 것으로 예상됩니다.

참고: "cta708"을 사용하는 것은 이전 구문 호환성을 위한 것입니다. CTA 708 구문과 어느 정도 일치하지만 구문이 반드시 CTA 708 을 준수하는 것은 아닙니다.

<표 9.1.5.1-1> Caption Display Preferences CTA 708 Object Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
cta708	0..1	object	캡션 표시 기본 설정 속성이 포함된 개체
characterColor	0..1	string	문자의 색상을 나타내는 문자열
characterOpacity	0..1	number	문자의 불투명도를 나타내는 0~1 범위의 정수 또는 고정 소수점 숫자
characterSize	0..1	integer	기본 글꼴 크기의 백분율 승수
fontStyle	0..1	examples	선호하는 글꼴 스타일을 나타내는 문자열 값
backgroundColor	0..1	string	캐릭터 배경의 색상을 정의하는 문자열
backgroundOpacity	0..1	number	문자 배경의 불투명도를 나타내는 0~1 범위의 정수 또는 고정 소수점 숫자
characterEdge	0..1	examples	선호하는 문자 가장자리를 나타내는 문자열 값
characterEdgeColor	0..1	string	해당되는 경우 문자 가장자리의 색상을 지정하는 문자열
windowColor	0..1	string	캡션 창 배경의 색상을 나타내는 문자열
windowOpacity	0..1	number	캡션 창의 불투명도를 나타내는 0~1 범위의 정수 또는 고정 소수점 숫자

characterColor - 이 매개 변수는 CEB35 [10], section 7.3 "펜 스타일, 문자 전경색"에 해당하며 문자의 색을 나타내는 문자열이어야 합니다. 색 값은 "#" 24 비트 sRGB 표기법을 사용하여 CSS3 색[9]에 대한 W3C 권장 사항에 지정된 색 인코딩을 준수해야 합니다. 예를 들어 빨간색은 "#FF0000"로 표시됩니다.

characterOpacity - 이 매개변수는 CEB35 [10], section 7.4 "배경색 및 불투명도, 문자 전경 불투명도"에 해당하며 문자의 불투명도를 나타내는 0 에서 1 사이의 정수 또는 고정 소수점 숫자여야 합니다. 예를 들어 값이 .33 이면 33%가 불투명하고 값이 0 이면 완전히 투명하다는 의미입니다.

characterSize - 이 매개변수는 CEB35 [10], section 7.1 "펜 크기"에 해당하며 100 은 변경되지 않고 50 은 1/2 크기, 200 은 크기의 두 배인 기본 글꼴 크기의 백분율 승수여야 합니다. 구문 및 의미는 A/343[7]에 정의된 IMSC1 과 일치해야 합니다.

fontStyle - 이 문자열은 CEB35 [10], section 7.2 "글꼴 스타일"에 해당하며 기본 캡션 글꼴의 스타일을 나타냅니다. 8 가지 가능한 선택 사항은 섹션 7.2 의 CEB35 [10] 번호가 매겨진 글꼴 스타일에 해당해야 합니다.

"*Default* "(정의되지 않음)

"*MonospacedSerifs*" - 세리프가 있는 고정폭(Courier 와 유사)

"*ProportionalSerifs*" - 세리프와 비례적으로 간격(Times New Roman 과 유사)

"*MonospacedNoSerifs*" - 세리프 없이 고정폭(Helvetica Monospaced 와 유사)

"*ProportionalNoSerifs*" - 세리프 없이 비례적으로 간격(Arial 및 Swiss 와 유사)

"*Casual*" - 캐주얼 글꼴 유형(Dom 및 Impress 와 유사)

" *Cursive* " - 필기체 글꼴 유형(Coronet 및 Marigold 와 유사)

"*SmallCaps*" - 작은 대문자(Engravers Gothic 과 유사)

backgroundColor - 이 매개변수는 CEB35 [10], section 7.4 "배경색 및 불투명도, 배경색"에 해당하며 *characterColor* 와 동일한 CSS 호환 형식으로 제공되는 문자 배경의 색상을 나타냅니다.

backgroundOpacity - 이 매개변수는 CEB35 [10], section 7.4 "배경색 및 불투명도, 전경 불투명도"에 해당하며 캐릭터 배경의 불투명도를 나타내는 0 에서 1 사이의 정수 또는 고정 소수점 숫자여야 합니다. 값 1 은 100% 불투명함을 의미합니다. 값 0 은 완전히 투명함을 의미합니다.

characterEdge - 이 매개변수는 CEB35 [10], section 7.5 "문자 가장자리, 유형 속성"에 해당하며 문자 가장자리에 대해 선호하는 표시 형식을 나타냅니다. 문자의 가장자리(또는 윤곽선)의 기본 색상은 *characterEdgeColor* 에 지정된 대로여야 합니다. 가장자리 불투명도는 캐릭터 전경 불투명도와 동일한 속성을 가져야 합니다. 선택 사항은 CEB35[10], section 7.5: "None", " Raised", "Depressed", “Uniform", "LeftDropShadow" 및 "RightDropShadow"에 명시되어 있습니다.

characterEdgeColor - 이 매개변수는 CEB35 [10], section 7.5 "Character Edges, edge color 에 해당하며 해당되는 경우 *characterColor* 와 동일한 형식으로 제공된 문자 가장자리의 색상을 나타냅니다.

windowColor - 이 매개변수는 CEB35 [10], section 8.1 "Window, color"에 해당하며 *characterColor* 와 동일한 형식으로 제공되는 캡션 창 배경의 색상을 나타냅니다.

windowOpacity - 이 매개변수는 CEB35 [10], section 8.1 "Window, opacity"에 해당하며 캡션 창의 불투명도를 나타내는 0 에서 1 사이의 정수 또는 고정 소수점 숫자여야 합니다. 값 1 은 100% 불투명함을 의미합니다. 값이 0 이면 완전히 투명함을 의미합니다.

9.1.5.2 IMSC1 Extensions Semantics

이 section 에 정의된 키/값 쌍은 "imsc1" 개체 내에서 IMSC1(A/343 [7]에 정의된 대로) 속성 기본 설정을 나타내는 수단을 제공합니다(있는 경우).

IMSC1 기본 설정의 키/값 쌍은 다음과 같은 형식을 취해야 합니다.

```
"<imsc1_key>": "<imcs1_value>"
```

<imsc1_key>의 구문은 RFC 5234 [12]에 정의된 ABNF(Augmented Backus-Naur Form) 문법을 사용하여 아래에 지정되어야 합니다.

```
<imsc1_key> = ("region_" / "content_") + imsc1_attribute
```

imsc1_attribute - 이 부분은 IMSC1 정의 속성의 이름이어야 합니다.

<imsc1_key> - 값의 값 데이터 유형 범위 및 해당 인코딩은 *imsc1_attribute*에 명명된 속성에 대해 IMSC1에서 지원하는 값이어야 합니다.

*<imsc1_key>*의 두 번째 부분은 *<imsc1_value>*이 지역에 적용되는 경우 "*region_*", 콘텐츠(텍스트)에 적용되는 경우 "content_"를 표시해야 합니다.

*<imsc1_key>*의 유효한 예로는 각각 영역 또는 콘텐츠(텍스트)의 배경색을 나타내는 "*region_backgroundColor*" 및 "*content_backgroundColor*"이 있습니다.

9.1.5.3 Caption Display Preferences Query Example

Broadcaster Application은 캡션 표시 기본 설정을 얻기 위해 다음 쿼리를 수행할 수 있습니다.

<표 9.1.5-3-1> Example of Caption Display Preferences Query
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.captionDisplay",
      "id": 932
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.1.5-3-2> Example of Caption Display Preferences Query Response
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {
            "cta708": {
                  "characterColor": "#F00000",
                  "characterOpacity": 0.5,
                  "characterSize": 80,
                  "fontStyle": "MonospacedNoSerifs",
                  "backgroundColor": "#808080",
                  "backgroundOpacity": 0,
                  "characterEdge": "None",
                  "characterEdgeColor": "#000000",
                  "windowColor": "#000000",
                  "windowOpacity": 0
            },
      "imsc1": {
            "region_textAlign": "center",
            "content_fontWeight": "bold"
```

```
        }
        },
        "id": 932
}
```

9.1.6 Query Audio Accessibility Preferences API

Broadcaster Application은 비디오 설명 서비스, 긴급 정보의 오디오/청각 표현 및 해당 언어 기본 설정의 자동 렌더링이 활성화되어 있는지 여부를 포함하여 **Receiver**의 오디오 접근성 설정을 알고 싶어할 수 있습니다. **Broadcaster Application**은 **Query Audio Accessibility Preferences API**를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Audio Accessibility Preferences Request의 Semantics는 표 9.1.6-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.audioAccessibilityPref-request.json에 정의된 대로여야 합니다.

<표 9.1.6-1> Query Audio Accessibility Preferences Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.query.audioAccessibilityPref”

Query Audio Accessibility Preferences Response의 Semantics는 표 9.1.6-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.audioAccessibilityPref-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.6-2> Query Audio Accessibility Preferences Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	요청 ID 값과 일치
result	oneOf X	string	요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
videoDescriptionService	0..1		
enabled	0..1	boolean	비디오 설명 서비스가 활성화되었는지 여부
language	0..1	string	동영상 설명 서비스의 기본 언어
audioElService	0..1		
enabled	0..1	boolean	긴급 정보 오디오가

				활성화되었는지 여부
		language	0..1	string	긴급 정보 오디오의 기본 언어
error			oneOf X		Section 8.3.3 참조

result - 비디오 설명 서비스나 오디오 긴급 정보 렌더링이 모두 활성화되지 않은 경우 결과 구조에는 요소가 포함되지 않아야 합니다. JSON 에서는 "result": {}로 표시됩니다.

videoDescriptionService.enabled, audioEIService.enabled - 이러한 각 Boolean 값은 각각 VDS(비디오 설명 서비스)의 자동 렌더링 기본 설정, 긴급 정보의 오디오/청각 표현의 현재 상태를 나타냅니다.

videoDescriptionService.language - BCP 47[21]에 따라 코딩된 VDS 렌더링의 기본 언어를 나타내는 문자열입니다.

audioEIService.language - BCP 47[21]에 따라 코딩된 긴급 정보 렌더링의 오디오/청각 표현의 기본 언어를 나타내는 문자열입니다.

수신기에 *videoDescriptionService.enabled, videoDescriptionService.language, audioEIService.enabled, audioEIService.language* 에 대한 설정이 없는 경우 응답에 해당 속성이 포함되지 않을 것으로 예상됩니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.1.6-3> Example of Query Audio Accessibility Preferences Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.audioAccessibilityPref",
        "id": 90
}
```

또한 사용자가 비디오 설명 서비스의 자동 렌더링 기본 설정을 ON 으로 설정하고 수신기에 나머지 설정이 없는 경우 **Receiver** 는 다음과 같이 응답할 수 있습니다.

<표 9.1.6-4> Example of Query Audio Accessibility Preferences Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "videoDescriptionService": {
        "enabled": true
        }
        },
        "id": 90
}
```

9.1.7 Query Receiver Web Server URI API

Broadcaster Application 은 **Receiver** 가 제공하는 **Application Context Cache** 의 위치에 액세스하기를 원할 수 있습니다. 이 개념적 캐시는 현재 로드된 **Broadcaster Application** 에 대해 정의된 **Application Context Identifier** 의 후원 하에 전달되는 리소스에 대한 액세스를 제공합니다. 이러한 리소스는 기본 URI 를 사용하여 **Receiver Web Server** 를 통해 사용할 수 있습니다(section 5.3 참조). 이 API 는 해당 URI 에 대한 액세스를 제공합니다.

이 API 는 **Broadcaster Application** 이 broadband 서버에서 시작된 경우에 유용합니다. 이 경우 **Application Context Cache** URI 를 인식하지 못합니다. **Application Context Cache** 에서 실행되는 **Broadcaster Application** 은 표준 W3C DOM 매개변수를 통해 서버 위치를 확인할 수 있습니다.

Query Receiver Web Server URI Request Semantics 는 표 9.1.7-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.baseURI-request.json 에 정의된 대로여야 합니다.

<표 9.1.7-1> Query Receiver Web Server URI Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.query. baseURI"

Query Receiver Web Server URI Response Semantics 는 표 9.1.7-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.baseURI-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.7-2> Query Receiver Web Server URI Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	요청 ID 값과 일치
result	oneOf X	string	요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
baseURI	1		활성 **Application Context Cache** 에 액세스할 수 있는 URI 를 제공
error	oneOf X		Section 8.3.3 참조

baseURI – 이 반환 매개 변수에는 **Application Context Identifier** 와 연결된 리소스에 액세스할 수 있는 URI 가 포함되어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None – 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.1.7-3> Example Query Receiver Web Server URI Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.baseURI",
        "id": 90
}
```

Receiver 는 현재 **Application Context Identifier** 에 대해 정의된 **Application Context Cache** 에 대한 **Receiver Web Server** 의 URI 로 응답합니다.

<표 9.1.7-4> Example Query Receiver Web Server URI Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "baseURI": "http://localhost:8080/contextA"
        },
        "id": 90
}
```

결과 URI 는 **Receiver** 에서 해당 리소스에 액세스하기 위해 리소스에 대한 상대 참조 앞에 추가할 수 있습니다.

9.1.8 Query Alerting Signaling API

Broadcaster Application 은 현재 브로드캐스트에서 신호를 받은 다양한 경고 메타데이터 구조에 액세스하기를 원할 수 있습니다. **Query Alerting Signaling API** 는 **Broadcaster Application** 이 요청한 특정 경고 메타데이터 목록을 반환합니다.

Query Alerting Signaling Request 의 Semantics 는 표 9.1.8-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.alerting-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.1.8-1> Query Alerting Signaling Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.alerting”
alertingTypes	1	enum	요청된 경고 유형 목록. 빈 목록은 모든 경고 유형

alertingTypes - 다음과 같이 경고 유형 중 하나 또는 둘 다의 배열입니다.
AEAT - 가장 최근의 AEAT XML 조각(있는 경우)을 요청합니다.

OSN - 가장 최근의 OSN XML 조각(있는 경우)을 요청합니다.

빈 목록은 모든 값을 제공하는 것과 같습니다.

Query Alerting Signaling Response 의 Semantics 는 표 9.1.8-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.alerting-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.8-2> Query Signaling Data Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
alertList	1	array	요청을 기반으로 하는 경고 조각 목록. 요청된 유형과 일치하는 경고 신호가 활성화되지 않은 경우 목록이 비어 있을 수 있음.
items	0..N		
alertingType	1	enum	"AEAT" or "OSN"
alertingFragment	1	string (xml)	연결된 경고 유형의 XML 조각
receiveTime	0..1	string (date-time)	alertingType = "OSN"인 경우 프래그먼트가 수신된 날짜 및 시간
filteredEventList	0..1	array	수신기에 의해 필터링된 AEA ID 배열을 제공
items	1..N	string	
error	oneOf X		Section 8.3.3 참조

alertList - 요청에 지정된 경고 신호 조각의 배열입니다. 요청된 경고 신호가 활성 상태가 아닌 경우 배열이 비어 있을 수 있습니다.

alertingType - 이 필수 매개 변수에는 경고 유형 "*AEAT*" 또는 "*OSN*" 중 하나가 포함되어야 합니다. 해당 *alertingFragment* 에는 표시된 경고 메타데이터 조각 유형에 해당하는 데이터가 포함되어야 합니다.

alertingFragment - 이 필수 문자열에는 연결된 *alertingType* 에 대한 경고 XML 조각이 포함되어야 합니다. AEAT XML 및 OSN XML 단편은 A/331 [3]에 설명된 각 LLS 테이블에서 추출됩니다.

receiveTime - 경고 조각이 수신된 날짜 및 시간입니다. 이 값은 개체가 "OSN"일 때 제공되어야 합니다. (참고: OnscreenMessageNotification 요소에는 OSN 을 수신한 시간부터 시작하는 KeepScreenClear 메시지의 지속 기간인 KeepScreenClear@notificationDuration 속성이 포함되어 있습니다. 따라서 OSN 이 수신된 시간은 브로드캐스터 애플리케이션이 OSN 정보를 최대한 활용하기 위해 필요합니다.) 날짜-시간 JSON 데이터 유형은 JSON 스키마 표준[19]에 정의된 대로 형식이 지정되어야 합니다.

filteredEventList – **Receiver**에 의해 필터링된 AEA 이벤트의 목록을 제공합니다. 수신기는 사용자 선호도, 위치 또는 기타 기준에 따라 다양한 이유로 특정 이벤트를 필터링할 수 있습니다. AEA 이벤트가 필터링된 경우, 해당 *AEAT.AEA@aeald* 값이 *filteredEventList* 속성에 표시되어야 합니다. AEA 이벤트가 필터링되지 않은 경우, 해당 *AEAT.AEA@aeald* 값은 목록에 포함되지 않습니다. *filteredEventList* 가 비어 있거나 존재하지 않는 경우, 이는 수신기에서 필터링된 이벤트가 없음을 의미합니다. 이 속성은 ***alertingType***이 "AEAT"인 경우에만 적용됩니다. "필터링된" AEA 이벤트는 수신기에 의해 이미 처리되었거나 처리 중인 이벤트로, **Broadcaster Application**에서 별도로 처리할 필요가 없는 이벤트를 의미합니다.

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None – 이 API와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application**은 다음과 같은 쿼리를 만듭니다.

<표 9.1.8-3> Example of Query Alerting Signaling Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.alerting",
       "params": {
              "alertingTypes": ["AEAT", "OSN"]
       },
       "id": 913
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.1.8-4> Example of Query Alerting Signaling Request
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
       "alertList": [
              { "alertingType": "AEAT",
                "alertingFragment": "<AEAT>…</AEAT>" },
              { "alertingType": "OSN",
               "alertingFragment": "<OSN>…</OSN>",
                "receiveTime": "2017-01-01T23:54:59.590Z" }
              ]
       },
       "id": 913
}
```

9.1.9 Query Service Guide URLs API

Broadcaster Application은 현재 브로드캐스트에서 제공되는 다양한 서비스 가이드 데이터 구조에 액세스하기를 원할 수 있다. **Query Service Guide URLs API**는 **Broadcaster Application**이 브로드캐스트에 제공된 특정 서비스 가이드

데이터 구조를 검색하는 데 사용할 수 있는 URL 목록을 반환합니다(예: XHR로).

Query Service Guide URLs Request의 Semantics는 표 9.1.9-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.serviceGuideUrls-request.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.9-1> Query Service Guide URLs Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
method	1	string	"org.atsc.query.serviceGuideUrls"
service	0..1	String(uri)	특정 서비스에 해당하는 서비스 안내 정보를 요청

service – Section 9.1.3의 **Query Service ID API**에 정의된 선택적 서비스 필드입니다. 생략하면 모든 서비스에 대해 모든 서비스 가이드 조각이 반환됩니다. 있는 경우 제공된 서비스와 관련된 조각만 반환됩니다.

Query Service Guide URLs Response의 Semantics는 표 9.1.9-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.serviceGuideUrls-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.9-2> Query Service Guide URLs Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
urlList	1	array	지정된 경우 요청된 서비스를 기반으로 서비스 가이드 URL 집합을 나열하고, 그렇지 않은 경우 모든 서비스 가이드 URL을 나열
items	0..N		
sgType	1	enum	"Service", "Schedule" or "Content"
sgUrl	1	string (url)	관련 서비스 가이드 유형의 XML 조각 URL
service	1	string (uri)	서비스 가이드 유형과 관련된 서비스의 URI
content	0..1	string (uri)	sgType = "Content"인 경우 이 매개변수는 가능한 경우 콘텐츠의 고유 ID를 제공
error	oneOf X		Section 8.3.3 참조

urlList – **Service Guide URLs request**에 대한 응답으로 발견된 서비스

가이드 URL의 배열(비어 있을 수 있음)을 제공합니다.

sgType - 서비스 가이드 XML 조각 유형 중 하나입니다. 해당 *sgUrl*을 사용하여 표시된 서비스 가이드 조각 유형에 해당하는 XML 조각에 액세스할 수 있습니다. 동일한 *sgType* 목록 내에 여러 조각이 있을 수 있습니다. *sgType*을 사용하여 관심 있는 조각에 빠르게 액세스할 수 있습니다.

sgUrl - **Broadcaster Application** (예: XHR 요청)에서 연결된 *sgType*에 대한 현재 브로드캐스트 서비스 가이드 XML 조각을 검색하는 데 사용할 수 있는 정규화된 URL 입니다. 서비스 가이드는 구문이 A/332[4]에 정의된 대로 XML 조각으로 제공됩니다. 이렇게 하면 정확히 하나의 서비스 가이드 조각이 반환됩니다.

service - Section 9.1.3의 **Query Service ID API**에 정의된 필수 서비스 필드입니다. 참고: 이는 *globalServiceID* 필드로 더 일반적으로 알려져 있습니다. 적절한 작동을 위해서는 SLT에 *globalServiceID*가 있어야 합니다. A/331 [3] 섹션 6.3 및 A/351 [38] 섹션 5 참조. 이렇게 하면 정확히 하나의 서비스 가이드 조각이 반환됩니다.

content - *sgType="Content"*일 때 필요한 정규화된 URI 는 **Broadcaster Application**에서 서비스의 특정 콘텐츠 항목(많을 수 있음)을 고유하게 식별하는 데 사용할 수 있는 콘텐츠 조각에 있는 *globalContentID*를 제공해야 합니다. 서비스 가이드는 A/332[4]에 구문이 정의된 XML 조각으로 제공됩니다. 이렇게 하면 정확히 하나의 서비스 가이드 조각이 반환됩니다. *globalContentID*가 서비스 안내서 컨텐츠 조각에 없는 경우 이 필드는 비어 있을 수 있습니다.

섹션 B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- -4 - ESG 서비스가 없어 반환할 프래그먼트가 없는 경우 오류는 -4 입니다. "콘텐츠를 찾을 수 없습니다".
- -6 - 서비스를 찾을 수 없습니다. 요청된 서비스를 찾을 수 없습니다.

예를 들어 **Broadcaster Application**은 다음과 같은 쿼리를 만듭니다.

<표 9.1.9-3> Example of Query Service Guide URLs Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.serviceGuideUrls",
       "id": 913
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.1.9-4> Example of Query Service Guide URLs Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
              "urlList": [
```

```
                        { "sgType": "Service",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Service.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B" },
                        { "sgType": "Schedule",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Schedule.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B" },
                        { "sgType": "Content",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Content.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B",
                        "content": "urn:eidr:10.5240:7791-8534-2C23-9030-8610-5" }
                ]
        },
        "id": 913
}
```


제공된 URL은 예시일 뿐입니다. 파일 이름을 포함하여 사용되는 실제 URL은 **Receiver** 구현 및 HTTP 서버를 통해 ESG 파일을 사용할 수 있도록 선택하는 방법에 따라 전적으로 달라집니다. **Broadcaster Application**은 URL 경로에 대해 가정하지 않아야 하며 이를 사용하여 프래그먼트 데이터에 직접 액세스해야 합니다.

참조된 서비스 가이드 파일(이 예에서 Service.xml, Schedule.xml 및 Content.xml)은 각각 A/332[4]에 설명된 대로 서비스, 일정 및 콘텐츠 XML 단편을 포함해야 합니다. **Receiver**는 **Broadcaster Application**에서 사용할 수 있도록 하기 전에 바이너리 SGDU 구조에서 각 XML 조각을 추출해야 합니다.

ESG 파일을 **Broadcaster Application**과 연결하려면 ESG 서비스 ROUTE 세션의 LCT 채널에서 ESG 파일을 보낼 때 정의된 EFDT(Extended FDT) 요소인 FDT-Instance@appContextIdList 에 해당 **Application Context Identifier**가 제공되어야 합니다. FDT 연장 및 ESG 서비스에 대한 설명은 A/331[3]에서 확인할 수 있습니다. **Broadcaster Application** 에 ESG 데이터가 필요하지 않은 경우 **Application Context Identifier**를 EFDT 에 포함할 필요가 없습니다.

9.1.10 Query Signaling Data API

Broadcaster Application은 현재 브로드캐스트에서 다양한 신호 메타데이터 구조에 액세스하기를 원할 수 있습니다. 재배포(방송 시그널링 메타데이터를 사용할 수 없는 경우)의 경우, **Broadcaster Application**은 A/336[5], Table 5.30 에 정의된 Recovery Data Table (RDT)을 포함하고 A/331[3]에 열거된 다른 메타데이터를 포함할 수 있는 콘텐츠 복구를 통해 수신기에 의해 획득된 시그널링 메타데이터 구조에 액세스하기를 원할 수 있습니다. 아래 표 9.1.10-2 를 참조하십시오. **Query Signaling Data API**는 **Broadcaster Application**이 주 및 부 채널 번호와 같이 다른 방법으로는 사용할 수 없는 세부 정보를 추출하는 데 사용할 수 있는 신호 테이블 목록을 반환합니다.

Query Signaling Data Request Semantics는 표 9.1.10-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.signaling-request.json 에 정의된

대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.10-1> Signaling Data Change Notification Semantic
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.signaling"
group	1	enum	반환된 신호와 관련된 그룹
nameList	1	array	요청된 신호 객체 목록
items	0..N	string or integer	아래 *names* 정의를 참조

group - 이 선택적 매개변수는 요청된 메타데이터 객체의 신호 그룹([3] section 5.5 참조)을 지정합니다. 요청된 메타데이터 개체는 지정된 신호 그룹의 일부인 경우에만 반환됩니다. 신호 그룹이 지정되지 않은 경우, 수신기는 검색된 모든 메타데이터 객체를 보내거나 요청을 하는 **Broadcaster Application** 을 시작한 HELD 와 동일한 그룹의 메타데이터 객체만 전송하도록 선택할 수 있습니다.

nameList - 아래 *names* 에 설명된 신호 객체 이름의 배열입니다. 비어 있으면 메타데이터 개체가 반환되지 않습니다.

names - 이 필드는 표 9.1.10-2 의 "names" 열에 지정된 값 목록으로 설정되어야 합니다. names 필드가 비어 있으면 이 요청은 메타데이터 개체를 반환하지 않습니다. 일부 메타데이터 개체는 전송에 종속되며(ROUTE 대 MMT) 지정된 수신기에서 사용하지 못할 수 있습니다. LLS 테이블은 SignedMultiTable 을 통해 전달될 수 있습니다. 재배포의 경우 RDT 와 함께 다운로드한 RDT 및 메타데이터 개체만 반환할 수 있습니다.

<other> 이름은 문자열 또는 숫자의 자리 표시자입니다. **Broadcaster Application** 이 **Receiver** 에게 알 수 없는 문자열이나 숫자를 제공하는 경우 **Receiver** 는 요청을 무시해야 합니다. 문자열 또는 숫자가 **Receiver** 에게 알려져 있고 메타데이터 개체가 있는 경우 **Receiver** 는 메타데이터 개체를 반환해야 합니다.

<표 9.1.10-2> Signaling Metadata Object Name Definitions
[출처: A344]

Names	Description	Reference
ROUTE / DASH Signaling		
USBD	User Service Bundle Description	[3]Section7.1.3
STSID	Service-based Transport Session Instance Description	[3]Section7.1.4
MPD	DASH Media Presentation Description	[3]Section7.1.5
APD	Associated Procedure Description	[3]Section7.1.7
MMT Signaling		
USD	User Service Description for MMTP	[3]Section7.2.1
PAT	MMT Package Access Table	[3]Section7.2.3
MPT	MMT Package Table	[3]Section7.2.3
MPIT	MMT Media Presentation Information	[3]Section7.2.3

	Table	
CRIT	MMT Clock Relation Information Table	[3]Section7.2.3
DCIT	MMT Device Capabilities Information Table	[3]Section7.2.3
MMT Message Signaling		
AEI	MMT Application Event Information	[6]Section4.1.2
VSPD	Video Stream Properties Descriptor	[3]Section7.2.3.2
ASD	ATSC Staggercast Descriptor	[3]Section7.2.3.3
IED	Inband Event Descriptor	[6]Section4.1.2
CAD	Caption Asset Descriptor	[3]Section7.2.3.5
ASPD	Audio Stream Properties Descriptor	[3]Section7.2.3.4
SPD	Security Properties Descriptor	[3]Section7.2.4.2
Event Signaling		
EMSG	ROUTE/DASH Application Dynamic Event	[6]Section4.2
EVTI	MMT Application Dynamic Event	[6]Section4.1.2
Other Signaling		
HELD	HTML Entry pages Location Description	[3]Section7.1.8
DWD	Distribution Window Description	[3]Section7.1.9
RSAT	Regional Service Availability Table	[3]Section7.1.10
RDT	Recovery Data Table	[5]Section5.4.1
Low-Level Signaling (LLS)		
SLT or 1	Service List Table, LLS_table_id=1	[3]Section6.3
RRT or 2	Region Rating Table, LLS_table_id=2	[3]Annex F
STT or 3	SystemTime Table, LLS_table_id=3	[3]Section6.2
AEAT or 4	Advance Emergency Information Table, LLS_table_id=4	[3]Section6.5
OSN or 5	Onscreen Message Notifications, LLS_table_id=5	[3]Section6.6
SMT or 254	Signed Multitable, LLS_table_id=0xFE (254)	[3]Section6.7
CDT or 6	CertificateData Table, LLS_table_id=6	[8]Section5.2.2.2
<other>	“names" 열에 명시적으로 명명되지 않은 메타데이터 개체와 연결된 문자열 또는 숫자	

Query Signaling Data Response Semantics는 표 9.1.10-3에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.signaling-response.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.10-3> Query Signaling Data Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
objectList	1	array	요청에 지정된 경우 요청된 이름을 기반으로 신호 테이블을 나열하고 그렇지 않은 경우 사용 가능한 모든 신호 테이블을 나열
items	0..N		
name	1	string or integer	위의 names 정의를 참조
version	1	integer	시그널링 항목의 version
group	0..1	integer	LLS 테이블에 필요. LLS 그룹 ID 제공

			table	1	string (XML or JSON or Base64)	시그널링 table 데이터
			encoding	0..1	string	UTF-8 이 아닌 경우 콘텐츠 인코딩
error				oneOf X		Section 8.3.3 참조

objectList - 신호 데이터 요청에 대한 응답으로 발견된 신호 데이터의 배열(즉, 요청된 테이블이 브로드캐스트에 존재하지 않음)을 제공합니다.

name - 위의 *names* 을 참조하세요.

version - XML 신호 문서의 버전입니다. LLS 의 경우 *LLS_table_version* 값으로 설정해야 합니다. 메타데이터 개체 유형 및 RDT 의 경우 **metadataEnvelope***@version* 로 설정해야 합니다.

group - LLS 테이블의 경우 필수이며 *LLS_group_id* 값이어야 합니다. **Metadata Object Types** 및 RDT 의 경우 생략됩니다.

table - 이 문자열에는 UTF-8 로 인코딩된 *name* 과 일치하는 유형의 단일 객체가 포함되어야 합니다. UTF-8 로 인코딩되지 않은 객체의 경우 먼저 **Receiver** 에 의해 Base64 로 인코딩되어야 합니다. XML 및 JSON 문서는 압축되지 않아야 합니다. LLS 및 SLS 문서 모두에 대해 XML 문서 자체만 반환됩니다. RDT 의 경우 JSON 문서 자체만 반환됩니다. 서명, 바이너리 LLS 필드(예: *group_count_minus1*), MIME 구분 기호 등을 포함한 관련 패키징은 제거되어야 합니다.

encoding - 테이블이 Base64 로 인코딩된 경우 이 선택적 문자열에는 단일 토큰인 "*Base64*"가 포함되어야 합니다. UTF-8 인 개체의 경우 기본값이 UTF-8 이므로 이 필드가 필요하지 않습니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 사용 가능한 테이블이 없어 반환할 테이블이 없는 경우 오류는 -4 "콘텐츠를 찾을 수 없음"입니다.

예를 들어 애플리케이션은 다음과 같이 SLT 및 MPD 에 대한 쿼리를 만듭니다.

<표 9.1.10-4> Example of Signaling Data Change Notification
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.signaling",
      "params": {
            "nameList": [1, "MPD"]
      },
      "id": 913
}
```

Receiver 는 다음과 같이 응답할 수 있습니다.

<표 9.1.10-5> Example of Query Signaling Data Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "objectList": [
                        { "name": 1,
                        "version": 23,
                        "group": 1,
                        "table": "<SLT …… </SLT>" },
                        { "name": "MPD",
                        "version": 65,
                        "table": "<MPD …… </MPD>" }
                ]
        },
        "id": 913
}
```

9.1.11 Query Dialog Enhancement Preferences API

Broadcaster Application 은 Dialog Enhancement 처리의 상태 및 양을 포함하여 Dialog Enhancement 기능의 사용자 기본 설정을 알고 싶어할 수 있습니다. **Broadcaster Application** 은 **Query Dialog Enhancement Preferences API** 를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Dialog Enhancement Preferences Request 의 Semantics 는 표 9.1.11-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.dialogEnhancementPref-request.json 에 정의된 대로여야 합니다.

<표 9.1.11-1> Query Dialog Enhancement Preferences Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.dialogEnhancementPref"

Query Dialog Enhancement Preferences Response 의 Semantics 는 표 9.1.11-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.dialogEnhancementPref-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.1.11-2> Query Dialog Enhancement Preferences Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
dialogEnhancementPref	1	integer	사용자의 대화 강화 선호

				게인 값(dB)
error		oneOf X		Section 8.3.3 참조

dialogEnhancementPref – 오디오 디코더에 적용할 대화 개선 처리에 대한 사용자의 기본 설정 게인 값(dB)입니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None – 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.1.11-3> Example of Query Dialog Enhancement Preferences Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.dialogEnhancementPref",
        "id": 92
}
```

사용자가 경미한 청각 장애가 있고 기본 설정 메뉴에서 대화 개선 사항을 활성화한 경우 **Receiver** 는 다음과 같이 응답할 수 있습니다.

<표 9.1.11-4> Example of Query Dialog Enhancement Preferences Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "dialogEnhancementPref": 6
        },
        "id": 92
}
```

9.1.12 Query Display Components API

Broadcaster Application 은 현재 캡션 표시 영역 및 비디오 창의 위치와 크기, 그리고 **Receiver** 가 자막 또는 비디오 창의 크기 조정을 지원하는지 여부에 대한 **Receiver** 설정을 요청할 수 있습니다. **Broadcaster Application** 은 **Query Display Components API** 를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Display Components Request Semantics 는 표 9.1.12-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.displayComponents-request.json 에 정의된 대로입니다.

<표 9.1.12-1> Query Display Component Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.displayComponents"

Query Display Components Response Semantics는 표 9.1.12-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.displayComponents-response.json에 정의된 대로입니다.

<표 9.1.12-2> Query Display Component Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
videoWindowScalingSupported	1	boolean	"true"인 경우 비디오 창의 크기 조정이 지원됨
captionScalingSupported	1	boolean	"true"인 경우 비디오 창의 크기 조정에 따른 자막 크기 조정이 지원됨
activeCaptionRegions	1	array	렌더링되고 있는 캡션 영역 목록. 자막이 표시되지 않으면 목록이 비어 있을 수 있음
items	0..N		
captionRegionOriginX	1	integer	화면 좌측상단을 기준으로 캡션 표시 영역의 현재 원점 x 축 좌표를 픽셀수로 제공
captionRegionOriginY	1	integer	화면 좌측 상단을 기준으로 캡션 표시 영역의 현재 원점 y 축 좌표를 픽셀 수로 제공
captionRegionExtentX	1	integer	현재 캡션 표시 영역의 너비를 픽셀 수로 제공
captionRegionExtentY	1	integer	현재 캡션 표시 영역의 높이를 픽셀 수로 제공
error	oneOf X		Section 8.3.3 참조

videoWindowScalingSupported - 상기 *videoWindowScalingSupported* 참조

captionScalingSupported - 상기 *captionScalingSupported* 참조

activeCaptionRegions - 상기 *activeCaptionRegions* 참조

captionRegionOriginX - 상기 *captionRegionOriginX* 참조

captionRegionOriginY - 상기 *captionRegionOriginY* 참조

captionRegionExtentX - 상기 *captionRegionExtentX* 참조

captionRegionExtentY - 상기 *captionRegionExtentY* 참조

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None - 이 API와 관련된 오류가 없습니다.

Receiver에서 보고한 *activeCaptionRegions*는 지정된 순간에 캡션의 정확한 영역에 대한 스냅샷을 나타내기 위한 것이 아닙니다. **Receiver**는 현재 글꼴 크기와 위치, 예상되는 최대 문자 및 행 수를 기반으로 가능한 가장 큰 영역을 보고할 수

있습니다. 이렇게 하면 캡션 텍스트가 변경될 때 발생할 수 있는 디스플레이 충돌 상황을 방지하는 데 도움이 됩니다. 캡션을 사용하지 않도록 설정하면 *activeCaptionRegions* 의 수는 0 이어야 합니다.

9.1.13 Query Announcement Time Limit

Broadcaster Application 은 시청자에게 "call-to-action"을 제시하여 자신의 존재를 알리고 시청자에게 참여할 기회를 제공할 수 있는 시간 제한에 대해 수신기 설정을 알도록 요청할 수 있다. **Broadcaster Application** 은 **Query Announcement Time Limit API** 를 사용하여 이러한 설정을 결정할 수 있습니다.

Query Announcement Time Limit Request 의 의미는 표 9.1.13-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.announcementTimeLimit-request.json 에 정의된 대로여야 합니다.

<표 9.1.13-1> Query Time Limit Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.announcementTimeLimit"

Query Announcement Time Limit Response 의 의미는 표 9.1.13-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.announcementTimeLimit-response.json 에 정의된 대로여야 합니다.

<표 9.1.13-2> Query Time Limit Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
announcementTimeLimit	1	integer	Broadcaster Application 이 자체적으로 알릴 수 있도록 클릭 유도 문구가 시청자에게 화면에 표시될 수 있는 시간을 제한하는 시간 값(초)에 대한 현재 설정을 제공합니다.
error	oneOf X		Section 8.3.3 참조

announcmentTimeLimit – 이 필수 정수는 시청자가 **Broadcaster Application** 그래픽을 표시할 수 있도록 선택하기 전에 **Broadcaster**

Application 의 그래픽이 화면에 표시될 수 있는 시간을 제한하는 시간(초)을 나타냅니다. 이 알림 시간 제한은 **Receiver** 가 기본값으로 설정하거나 사용자가 선택한 뷰어에 의해 설정되었을 수 있습니다. *announcementTimeLimit* 를 0 으로 설정하면 **Broadcaster Application** 과 통신할 수 있는 공지 시간 제한이 없음을 나타냅니다. 0 이 아닌 *announcementTimeLimit* 값은 3 보다 크거나 같아야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

9.2 Asynchronous Notifications of Changes

수신자가 이 section 에 정의된 API 를 통해 **Broadcaster Application** 에 제공할 것으로 예상되는 알림 유형은 표 9.2-1 에 명시되어 있습니다. 모두 *org.atsc.notify* 메소드를 사용하고 알림 유형을 나타내는 "*msgType*"이라는 매개변수를 포함합니다.

<표 9.2-1> Asynchronous Notifications
[출처: A344]

msgType	Event Description	Reference
ratingBlock	Content Advisory Rating Block Change -디코딩 중인 현재 콘텐츠가 차단에서 차단 해제로 또는 차단 해제에서 차단으로 변경되도록 사용자가 수신기에서 콘텐츠 자문 등급 설정을 변경할 때마다 제공되는 알림	Section 9.2.2
serviceChange	Service Change - 사용자 작업으로 인해 다른 서비스를 획득한 경우 제공되는 알림이며, 새로운 서비스는 동일한 애플리케이션의 URL 을 신호로 알림	Section 9.2.3
captionState	Caption State - 사용자가 자막 표시 상태를 변경할 때마다(끄기에서 켜기 또는 켜기에서 끄기) 제공되는 알림	Section 9.2.4
langPref	Language Preference -사용자가 선호하는 언어를 변경할 때마다 제공되는 알림	Section 9.2.5
captionDisplayPrefs	자막 표시 속성 기본 설정.	Section 9.2.6
audioAccessibilityPref	오디오 접근성 환경설정.	Section 9.2.7
alertingChange	Alerting Change - SEAT 또는 ON 메시지의 새 버전이 수신되었거나 경고 필터링 기본 설정이 변경되어 이벤트가 필터링되지 않은 경우 알림.	Section 9.2.8
contentChange	Content Change - 새 콘텐츠가 애플리케이션 컨텍스트 캐시에 배치되었으며 브로드캐스터 애플리케이션에서 액세스할 수 있다는 알림	Section 9.2.9
serviceGuideChange	Service Guide Change - 새로운 ESG 조각이 수신되면 제공되는 알림	Section 9.2.10
signalingData	Signaling Data Change - 새로운 신호 데이터가 수신되었다는 알림	Section 9.2.11

dialogEnhancementPrefChange	Dialog Enhancement Preference Change – 사용자가 사용자 기본 설정에서 대화 강화 처리 상태나 양을 변경할 때마다 제공되는 알림	Section 9.2.12
dialogEnhancementLimitChange	Dialog Enhancement Limit Change – 수신 오디오 스트림이 대화 강화 처리에 대한 변경된 제한을 알릴 때마다 제공되는 알림	Section 9.2.13
rfSignalChange	RF Signal Change – 수신 RF 신호의 일부 측면이 변경될 때마다 제공되는 알림	Section 9.2.14
contentRecoveryStateChange	Content Recovery State Change – 콘텐츠 복구를 위해 오디오 워터마크, 비디오 워터마크, 오디오 지문 및/또는 비디오 지문을 사용할 때마다 제공되는 알림	Section 9.8.4
displayOverrideChange	Display Override Change – 표시 재정의 상태 또는 특정 리소스에 대한 애플리케이션 액세스가 차단된 상태가 변경되는 경우 제공되는 알림	Section 9.8.5
recoveredComponentInfoChange	Recovered Component Info Change – 수신기가 수신하는 서비스의 구성 요소가 업스트림에서 변경되는 경우 제공되는 알림	Section 9.8.6
rmpMediaTimeChange	RMP Media Time Change – 재생 중에 주기적으로 제공되는 알림	Section 9.12.5
rmpPlaybackStateChange	RMP Playback State Change – 재생 상태가 변경되면 제공되는 알림	Section 9.12.3
rmpPlaybackRateChange	RMP Playback Rate Change – 재생 속도가 변경되면 제공되는 알림	Section 9.12.7
DRM	DRM Notification – 콘텐츠 보호 시스템에서 Broadcaster Application 으로 메시지를 제공하는 알림	Section 9.13.1
xlinkResolution	XLink Resolution – RMP 가 XLink 속성이 있는 마침표를 발견할 때 제공되는 알림	Section 9.14.1
assetLinkResolution	Asset Link Resolution – RMP 가 교체 가능한 대상 자산을 발견할 때 제공되는 알림	Section 9.16.1

9.2.1 Integrated Subscribe / Unsubscribe API for Notifications

Receiver 가 제공할 수 있는 다양한 유형의 알림 중, section 9.5 의 **Event Stream Notification** 을 제외한 나머지 알림 유형은 section 9.1.5 의 subsection 에서 정의되어 있다.

Broadcaster Application 은 특정 알림을 수신하고자 할 수 있다. **Broadcaster Application** 이 시작될 때는 기본적으로 어떠한 알림에도 구독 되어 있지 않으며, **Receiver** 가 알림을 전송할 것으로 기대되지 않는다. 방송사 애플리케이션은 구독 API 를 사용하여 원하는 알림 수신을 시작할 수 있다. **Receive** 는 **Broadcaster Application** 이 **Integrated Unsubscribe API** 를 요청할 때까지 구독 된 알림을 계속 전송해야 한다. 또한 **Broadcaster Application** 이 종료되면, **Receiver** 는 애플리케이션이 요청했던 모든 구독 알림을 자동으로 해제해야 한다.

이러한 기능을 지원하기 위해 두 가지 API가 필요하다:

- Integrated Subscribe API
- Integrated Unsubscribe API

표 9.2.1-1은 각 알림에 대해 구독할 수 있는 msgType 목록을 설명한다. 표의 msgType 열에 나열된 값은 subscribe 및 unsubscribe API의 열거형(enum) 매개변수와 동일하다.

참고로, Event Stream Notification과 그에 관련된 구독 메서드는 section 9.5에 별도로 정의되어 있다. 이 별도의 인터페이스는 다양한 이벤트 스트림을 필터링하기 위한 매개변수를 지정할 수 있도록 한다.

<표 9.2.1-1> Subscription Parameter List
[출처: A344]

Notification APIs	Reference	msgType
All Notification APIs	-	All
Rating Block Change Notification API	9.2.2	ratingBlock
Service Change Notification API	9.2.3	serviceChange
Caption State Change Notification API	9.2.4	captionState
Language Preference Change Notification API	9.2.5	languagePref
Caption Display Preferences Change Notification API	9.2.6	captionDisplayPrefs
Audio Accessibility Preference Change Notification API	9.2.7	audioAccessibilityPref
Alerting Change Notification API	9.2.8	alertingChange
Content Change Notification API	9.2.9	contentChange
Service Guide Change Notification API	9.2.10	serviceGuideChange
Signaling Data Change Notification API	9.2.11	signalingData
Dialog Enhancement Preference Change Notification API	9.2.12	dialogEnhancementPrefChange
Dialog Enhancement Limit Change Notification API	9.2.13	dialogEnhancementLimitChange
RF Signal Change Notification API	9.2.14	rfSignalChange
Content Recovery State Change Notification API	9.8.4	contentRecoveryStateChange
Display Override Change Notification API	9.8.5	displayOverrideChange
Recovered Component Info Change Notification API	9.8.6	recoveredComponentInfoChange
RMP Media Time Change Notification API	9.12.5	rmpMediaTimeChange
RMP Playback State Change Notification API	9.12.6	rmpPlaybackStateChange
RMP Playback Rate Change Notification API	9.12.7	rmpPlaybackRateChange
RMP Media Asset Change Notification API	9.12.8	rmpMediaAssetChange
DRM Notification API	9.13.1	DRM
XLink Resolution Notification API	9.14.1	xlinkResolution
AssetLink Resolution Notification API	9.16.1	assetLinkResolution

† Section 9.2.5에 정의된 언어 기본 설정 변경 알림 API "msgType"의 이름은 "langPref"입니다.

9.2.1.1 Integrated Subscribe API

Subscribe Request Semantics 는 표 9.2.1.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.subscribe-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.1.1-1> Subscribe Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
method	1	string	"org.atsc.subscribe"
msgType	1	array	**Broadcaster Application** 이 구독을 요청하는 알림 목록
items	0..N	examples	표 9.2-1 "msgType" 열의 msgTypes 중 하나
signalingDataList	0..1	array	"signalingData" msgType 이 요청된 경우 요청된 신호 개체 목록
items	0..N	string or integer	Section 9.1.4 의 *names* 정의를 참조

msgType – **Broadcaster Application** 이 구독을 요청하는 표 9.2-1 의 "msgType" 열에 있는 알림 msgTypes 배열입니다. 비어 있으면 이 요청은 작업을 수행하지 않습니다. "All" 열거형 값을 사용하여 모든 알림을 구독합니다.

signalingDataList – Section 9.1.10 의 "names description"에서 설명된 시그널링 객체 이름들의 배열이다. 이 필드는 "signalingData" msgType 이 msgType 목록에 포함되어 있을 때만 적용된다. 만약 이 필드가 비어 있으면, 메타데이터 객체는 반환되지 않는다.

구독 응답 의미 체계는 표 9.2.1.1-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.subscribe-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.2.1.1-2> Subscribe Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
msgType	1	array	**Broadcaster Application** 이 구독을 요청하는 알림 목록
items	0..N	examples	표 9.2-1 "msgType" 열의 msgTypes 중 하나
error	oneOf X		Section 8.3.3 참조

msgType – **Broadcaster Application** 이 구독된 표 9.2-1 의 "msgType" 열에 있는 알림 msgTypes 배열입니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다. 중복 구독 요청에 대한 오류는 없습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application** 은 "*alertingChange*", "*ratingBlock*" 및 "*contentChange*" 알림을 구독하려고 합니다.

<표 9.2.1.1-3> Example of Subscribe Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.subscribe",
        "params": {
                "msgType":      ["alertingChange",      "ratingBlock",
        "contentChange"]
        },
        "id": 51
}
```

성공하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.2.1.1-4> Example of Subscribe Response 1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "msgType":      ["alertingChange",      "ratingBlock",
        "contentChange"]
        },
        "id": 51
}
```

Broadcaster Application 이 모든 알림을 구독하려고 시도할 수 있는 경우:

<표 9.2.1.1-5> Example of Subscribe Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.subscribe",
        "params": {
                "msgType": ["All"]
        },
        "id": 51
}
```

Receiver 가 콘텐츠 복구를 제외한 모든 기능을 지원하는 경우 다음과 같이 응답할 수 있습니다.

<표 9.2.1.1-6> Example of Subscribe Response 2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
```

```
        "result": {
              "msgType":        ["ratingBlock",        "serviceChange",
        "captionState",       "languagePref",       "captionDisplayPrefs",
        "audioAccessibilityPref",    "alertingChange",    "contentChange",
        "rmpMediaTimeChange",                  "rmpPlaybackStateChange",
        "rmpPlaybackRateChange", "DRM", "xlinkResolution"]
        },
        "id": 51
}
```

9.2.1.2 Integrated Unsubscribe API

Unsubscribe Request Semantics 는 표 9.2.1.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.unsubscribe-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.1.2-1> Unsubscribe Request Semantic
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
method	1	string	"org.atsc.unsubscribe"
msgType	1	array	**Broadcaster Application** 이 구독 취소를 요청하는 알림 목록
items	0..N	examples	표 9.2-1 "msgType" 열의 msgTypes 중 하나

msgType – **Broadcaster Application** 이 구독 취소를 요청하는 표 9.2-1 의 "msgType" 열에 있는 알림 msgTypes 배열입니다. 비어 있으면 이 요청은 작업을 수행하지 않습니다. "All" 열거형 값을 사용하여 모든 알림을 구독 취소합니다.

Unsubscribe Response Semantics 는 표 9.2.1.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.unsubscribe-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.1.2-2> Unsubscribe Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
msgType	1	array	**Broadcaster Application** 이 구독을 요청하는 알림 목록
items	0..N	examples	표 9.2-1 "msgType" 열의 msgTypes 중 하나
error	oneOf X		Section 8.3.3 참조

msgType – **Broadcaster Application** 이 구독 취소된 표 9.2-1 의 "msgType" 열에 있는 알림 msgTypes 배열입니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -24: **Broadcaster Application** 이 요청된 알림을 구독하지 않았습니다.

"모두" 알림을 구독 취소하면 표 9.2-1 에 나열된 모든 알림의 구독이 취소됩니다. 오류는 현재 알림에 대한 미해결 구독이 없는 경우에만 발생할 것으로 예상됩니다.

예를 들어, **Broadcaster Application** 은 모든 알림을 구독 취소하려고 합니다.

<표 9.2.1.2-3> Example of Unsubscribe Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.unsubscribe",
       "params": {
             "msgType": ["All"]
       },
       "id": 52
}
```

성공하면 수신기는 **Broadcaster Application** 이 성공적으로 구독을 취소한 msgTypes 목록으로 응답합니다.

<표 9.2.1.2-4> Example of Unsubscribe Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
       "msgType":          ["alertingChange",          "contentChange",
"contentRecoveryStateChange",                "displayOverrideChange",
"recoveredComponentInfoChange",               "rmpMediaTimeChange",
"rmpPlaybackStateChange", "rmpPlaybackRateChange"]
       },
       "id": 52
}
```

9.2.2 Content Advisory Rating Block Change Notification API

Content Advisory Rating Block Change Notification 은 현재 표시된 서비스의 콘텐츠 권고 등급 차단이 차단 해제에서 차단으로 또는 그 반대로 변경되는 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다.

서비스가 차단되면 **Broadcaster Application** 은 디스플레이와 같은 API 에 대한 액세스를 제한할 수 있습니다. 서비스 차단이 해제되면 **Broadcaster Application** 은 정상 작업을 재개할 것으로 예상됩니다.

차단된 상태 외에도, 수신자는 **Broadcaster Application** 이 콘텐츠를 차단한 이유를 결정하고 사용자에게 알릴 수 있도록 현재 표시된 서비스의 콘텐츠 권고 등급을 제공할 수도 있습니다. 콘텐츠 권고 등급이 변경될 수 있는 경우에도 차단 상태에 변화가 없는 경우 이 알림이 발행되지 않을 것으로 예상됩니다.

콘텐츠 권고 등급, 다운로드 가능한 등급 지역 표 및 자녀 보호 기능은 법률 또는 규정에 의해 다룰 수 있습니다.

Content Advisory Rating Block Change Notification 의 Semantics 는 표 9.2.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-ratingBlock.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.2-1> Content Advisory Rating Block Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	array	"ratingBlock"
blocked	1	boolean	콘텐츠가 차단되었는지 여부
contentRating	0..1	string	현재 서비스 시그널링에서 제공하는 컨텐츠 자문 등급의 새로운 값

blocked – 필수 Boolean 값은 상태가 변경된 후 차단 상태를 나타냅니다. 이는 사용자 작업(예: 자녀 보호 설정 변경) 또는 현재 콘텐츠의 등급 변경의 결과일 수 있습니다.

contentRating – 서비스 신호에 제공된 현재 표시된 서비스의 콘텐츠 권고 등급을 포함하는 선택적 문자열입니다. *contentRating* 문자열은 A/331 [3], section 7.3 에 지정된 인코딩을 준수해야 합니다. 콘텐츠 등급 문자열에는 해당하는 경우 여러 등급 지역을 포함하여 모든 등급 값이 포함되어야 합니다. 여러 등급 지역에 대한 콘텐츠 권고 정보 데이터를 지정하려면 추가 3 부분 문자열(각 지역당 하나씩)을 연결하여 여러 개의 연결된 3 단계 문자열로 구성된 하나의 문자열을 만들어야 합니다. 이 경우 마지막 부분을 제외한 각 콘텐츠 권고 정보 문자열의 세 번째 부분 뒤에는 쉼표(",")가 옵니다. 따라서 전체 콘텐츠 권고 등급 문자열의 마지막 문자는 오른쪽 중괄호("}")입니다. A/331 에서 참조된 인코딩은 ROUTE DASH 에서 사용하는 인코딩입니다. MMT 는 A/332[4]에 따라 등급을 인코딩합니다.

콘텐츠 차단 상태가 차단 해제에서 차단으로 변경된 예:

<표 9.2.2-2> Example of Content Advisory Rating Block Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "ratingBlock",
                "blocked": true,
                "contentRating": "1,'TV-PG-L', {0 'TV-PG'}{1 'L'}"
        }
}
```

9.2.3 Service Change Notification API

Service Change Notification 는 **Application Lifecycle**, section 6.3 에 설명된 조건에 따라 수신자가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다.

Service Change Notification Semantics 는 ine2e4w 표 9.2.3-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-serviceChange.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.3-1> Service Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
method	1	string	"org.atsc.notify"
msgType	1	enum	"serviceChange"
service	1	string(uri)	새로 획득한 서비스의 *globalServiceID* 를 제공
requested	0..1	boolean	사용자 또는 **Broadcaster Application** 이 새로운 서비스를 요청하고 있음을 나타냄

service – 이 필수 속성은 새로 획득한 서비스에 연관된 *globalServiceID* 를 제공해야 하며, 이는 **Query Service ID response API** (section 9.1.3 참조)에 정의되어 있다. *globalServiceID* 가 정의될 필요가 없는 서비스 유형(예: DRM, ESG, NRT 서비스)에 대해서는, **Receiver** 가 **Broadcaster Application** 에 알림(notification)을 제공할 필요는 없다.

requested – 선택적 *requested* 속성을 "true"로 설정하면 서비스가 요청 중이며 아직 획득되지 않았음을 나타냅니다. *requested* 가 "false"로 설정되거나 없는 경우 새 서비스가 획득되었습니다 (**Broadcaster Application** 에 새 서비스에 대한 공통 *@appId* 및 *@appContextId* 이 있음을 암시적으로 나타냅니다).

다음 예제에서 사용자는 globalServiceID 가 https://doi.org/10.5239/8A23-2B0B 를 사용하여 서비스를 변경했습니다.

<표 9.2.3-2> Example of Service Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "serviceChange",
        "service": "https://doi.org/10.5239/8A23-2B0B"
        }
}
```

9.2.4 Caption State Change Notification API

Caption State Change Notification 은 캡션 디스플레이의 상태가 변경되면

Receiver 에 의해 현재 실행 중인 Broadcaster Application 에 발행될 것으로 예상됩니다.

Caption State Change Notification 의 Semantics 는 표 9.2.4-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-captionState.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.4-1> Caption State Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"captionState"
captionDisplay	1	boolean	자막이 표시되는지 여부

captionDisplay – 자막 표시의 새 상태를 나타내는 필수 Boolean 값입니다. "true" 값은 캡션이 표시되고 있음을 나타내고 "false" 값은 표시되지 않음을 나타냅니다.

예를 들어, Receiver 는 캡션 표시가 켜져 있음을 Broadcaster Application 에 알립니다.

<표 9.2.4-2> Example of Caption State Change Notification
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "method": "org.atsc.notify",
      "params": {
            "msgType": "captionState",
            "captionDisplay": true
      }
}
```

9.2.5 Language Preference Change Notification API

Language Preference Change Notification 은 사용자가 오디오, 사용자 인터페이스, 자막/캡션 또는 전체에 적용할 수 있는 기본 언어를 변경하는 경우 Receiver 가 현재 실행 중인 Broadcaster Application 에 발행할 것으로 예상됩니다. 변경된 기본 설정만 이 알림에 제공되어야 합니다.

Language Preference Change Notification 의 Semantics 는 표 9.2.5-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-langPref.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.2.5-1> Language Preference Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"captionState"

captionDisplay	1	boolean	자막이 표시되는지 여부

preferredUILang, *preferredAudioLang*, *preferredCaptionSubtitleLang* – 이러한 각 선택적 문자열은 쿼리 언어 기본 설정 응답 API 에 설명된 의미 체계를 준수해야 합니다(section 9.1.4 참조).

예를 들어 사용자가 캡션의 기본 언어를 캐나다에서 사용되는 프랑스어로 변경한 경우:

<표 9.2.5-2> Example of Language Preference Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "langPref",
                "preferredCaptionSubtitleLang": "fr-CA"}
        }
}
```

9.2.6 Caption Display Preferences Change Notification API

Caption Display Preferences Change Notification 이 있는 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 캡션 표시 기본 설정 변경 알림을 발행할 것으로 예상됩니다.

Caption Display Preferences Change Notification Semantics는 표 9.2.6-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-captionDisplayPrefs.json 에 정의된 대로여야 합니다. 자막 표시 환경설정인 "cta708"의 Semantics는 section 9.1.5.1 에 규정되어 있으며, IMSC1 [48] 속성 "imsc1"(A/343 [7]에 정의됨)은 section 9.1.2.2 에 명시되어 있다. IMSC1 속성들은 section 9.1.5.2 의 본문에서 정의되지만, 이 절의 JSON 스키마의 통합적인 구성 요소로 간주된다.

<표 9.2.6-1> Caption Display Preferences Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"captionDisplayPrefs"
cta708	0..1	object	Section 9.1.5.1 의 semantics 정의를 참조
imsc1	0..1	object	Section 9.1.5.2 의 semantics 정의를 참조

예를 들어, **Receiver** 는 사용자가 캡션 표시 기본 설정을 회색 배경에 빨간색 텍스트로 변경했음을 **Broadcaster Application** 에 알립니다. 두 개의 IMSC1 매개변수와 함께 사용 가능한 모든 708 매개변수가 이 예에 포함되어 있습니다.

<표 9.2.6-2> Example of Caption Display Preferences Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "captionDisplayPrefs",
                "cta708": {
                        "characterColor": "#FF0000",
                        "characterOpacity": 0.5,
                        "characterSize": 100,
                        "fontStyle": "MonospacedSerifs",
                        "backgroundColor": "#808080",
                        "backgroundOpacity": 0.25,
                        "characterEdge": "Raised",
                        "characterEdgeColor": "#000000",
                        "windowColor": "#000000",
                        "windowOpacity": 0
                },
                "imsc1": {
                        "region_textAlign": "center",
                        "content_fontWeight": "bold"
                }
        }
}
```

9.2.7 Audio Accessibility Preference Change Notification API

Audio Accessibility Preference Change Notification 은 사용자가 비디오 설명 서비스 및/또는 긴급 정보(EI)의 오디오/청각 표현에 대한 접근성 설정을 변경하는 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다.

Audio Accessibility Preference Change Notification 의 Semantics 는 표 9.2.7-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-audioAccessibilityPref.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.7-1> Audio Accessibility Preference Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify”
msgType	1	enum	"audioAccessibilityPref"
videoDescriptionService	0..1		
enabled	0..1	boolean	영상 설명 서비스 활성화 여부
language	0..1	string	동영상 설명 서비스의 기본 언어
audioEIService	0..1		
enabled	0..1	boolean	긴급 정보 오디오 활성화 여부
language	0..1	string	비상 정보 오디오의 기본 언어

videoDescriptionService.enabled - 비디오 설명 서비스(VDS) 렌더링의 새 상태를 나타내는 Boolean 값입니다.

videoDescriptionService.language - BCP 47[21]에 따라 코딩된 VDS 렌더링의 기본 언어를 나타내는 문자열입니다. 이 속성은 *videoDescriptionService.enabled* 가 true 와 같고 **Receiver** 에서 VDS 렌더링의 기본 언어를 사용할 수 있는 경우 알림에 있어야 합니다. *videoDescriptionService.enabled* 가 true 와 같고 VDS 렌더링의 기본 언어를 사용할 수 없는 경우 **Receiver** 에는 *videoDescriptionService.language* 속성이 포함되지 않습니다.

audioEIService.enabled - 긴급 정보 렌더링의 오디오/청각 표현의 새로운 상태를 나타내는 Boolean 값입니다.

audioEIService.language - BCP 47[21]에 따라 코딩된 긴급 정보 렌더링의 오디오/청각 표현의 기본 언어를 나타내는 문자열입니다. 이 속성은 *audioEIService.enabled* 가 true 와 같고 긴급 정보 렌더링의 오디오/청각 표현의 기본 언어를 **Receiver** 에서 사용할 수 있는 경우 알림에 있어야 합니다. *audioEIService.enabled* 가 true 와 같고 기본 언어를 사용할 수 없는 경우 **Receiver** 에는 *audioEIService.language* 속성이 포함되지 않습니다.

예를 들어, 사용자가 비디오 설명 서비스의 접근성 기본 설정을 ON 으로 변경한 경우 **Receiver** 는 아래와 같이 비디오 설명 서비스의 현재 상태와 VDS 언어 기본 설정(있는 경우)을 **Broadcaster Application** 에 알립니다.

<표 9.2.7-2> Example of Audio Accessibility Preference Change Notification
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "org.atsc.notify",
       "params": {
              "msgType": "audioAccessibilityPref",
              "videoDescriptionService": {
                     "enabled": true,
                     "language": "en"
              }
       }
}
```

9.2.8 Alerting Change Notification API

AEAT 또는 OSN 경보 데이터 구조의 버전이 변경되고 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 이러한 알림을 수신하도록 가입한 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 **Alerting Change Notification** 을 발행할 것으로 예상됩니다. 이전 수신 없이 새 경고 개체를 수신하는 것은 버전 변경으로 간주됩니다. 경고 이벤트 필터링이 변경되어

필터링된 이벤트 목록이 변경된 경우에도 **Alerting Change Notification** 이 발행될 수 있습니다.

알림 메시지에는 새 경고 조각 또는 업데이트된 경고 조각 목록이 포함되어 있습니다.

Alerting Change Notification Semantics 는 표 9.2.8-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-alertingChange.json 에 정의된 대로입니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.2.8-1> Alerting Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify”
msgType	1	enum	“alertingChange"
alertList	1	array	새로운 알림 조각이나 업데이트된 알림 조각이 포함된 목록
items	0..N		
alertingType	1	enum	"AEAT" or "OSN"
alertingFragment	1	string (XML)	연관된 경고 유형의 XML 조각
receiveTime	0..1	string (date-time)	alertingType = "OSN"인 경우 조각이 수신된 날짜 및 시간
filteredEventList	0..1	array	**Receiver** 에 의해 필터링된 AEA ID 배열을 제공합니다.
items	1..N	string	

alertList – 새 경고 조각 또는 업데이트된 경고 조각 목록을 포함하는 필수 배열입니다. 이 알림은 배열이 비어 있는 경우, 즉 변경사항이 없는 경우에는 발생하지 않을 것으로 예상됩니다.

alertingType – "*AEAT*" 또는 "*OSN*" 중 하나를 포함하는 필수 매개 변수입니다. 해당 *alertingFragment* 에는 *alertingType* 에 해당하는 XML 조각이 포함되어야 합니다.

alertingFragment – 필수 문자열에는 연결된 *alertingType* 에 대한 경고 XML 조각이 포함되어야 합니다. AEAT XML 및 OSN XML 단편은 A/331 [1]에 설명된 각 LLS 테이블에서 추출됩니다.

receiveTime – OSN 프래그먼트가 수신된 날짜 및 시간입니다. 이 값은 *alertingType* 이 "OSN"일 때 제공되어야 하며 그렇지 않으면 선택 사항입니다. (참고: OSN 테이블에는 OSN 이 수신된 시간부터 시작되는 *KeepScreenClear* 메시지의 지속 기간인 알림 기간 필드가 포함되어 있습니다. 따라서 OSN 이 수신된 시간은 브로드캐스터 애플리케이션이 OSN 정보를 최대한 활용하기 위해 필요합니다.) 날짜-시간 JSON 데이터 유형은 JSON 스키마 사양[19]에 정의된 대로 형식이 지정되어야 합니다.

filteredEventList – **Receiver** 에 의해 필터링된 AEA 이벤트 목록을 제공합니다. **Receiver** 는 사용자 선호도, 위치 또는 기타 기준에 따라 다양한 이유로 이벤트를 필터링할 수 있습니다. AEA 이벤트가 필터링된

경우, 해당 이벤트의 **AEAT.AEA***@aeald* 가 *filteredEventList* 속성에 표시됩니다. 반대로, AEA 이벤트가 필터링되지 않은 경우에는 해당 **AEAT.AEA***@aeald* 가 이 목록에 포함되지 않습니다. 비어 있거나 존재하지 않는 *filteredEventList* 는 **Receiver** 에서 필터링된 이벤트가 없음을 의미합니다. 이 속성은 *alertingType* 이 "AEAT"인 경우에만 적용됩니다. "filtered out" AEA 이벤트란 **Receiver** 에 의해 이미 처리되었거나 처리 중이며, **Broadcaster Application** 에서 별도로 처리할 필요가 없는 이벤트를 의미합니다.

예를 들어 **Receiver** 가 이동하거나 사용자가 기본 설정을 변경하여 필터링 기준이 변경되는 경우 이전에 필터링된 이벤트가 필터링되지 않을 수 있으며 이전에 필터링되지 않은 이벤트가 이제 필터링될 수 있습니다. 이 경우 경고 변경 알림은 수신자가 새 *filteredEventList* 와 함께 발행해야 합니다. **Broadcaster Application** 은 새 이벤트 목록에 따라 대체 조치를 취해야 할 수 있습니다.

예를 들어, **Receiver** 는 다음 JSON-RPC 명령을 실행하여 새 AEAT 가 수신되었음을 나타낼 수 있습니다.

<표 9.2.8-2> Example of Alerting Change Notification 1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "org.atsc.notify",
       "params": {
              "msgType": "alertingChange",
              "alertList": [
                     { "alertingType": "AEAT",
                     "alertingFragment": "<AEAT>…</AEAT>" }
              ]
       }
}
```

추가 예로, **Receiver** 는 이 JSON-RPC 명령을 실행하여 새 AEAT 및 OSN 이 수신되었음을 나타낼 수 있습니다:

<표 9.2.8-3> Example of Alerting Change Notification 2
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "org.atsc.notify",
       "params": {
              "msgType": "alertingChange",
              "alertList": [
                     { "alertingType": "AEAT",
                     "alertingFragment": "<AEAT>…</AEAT>" },
                     { "alertingType": "OSN",
                     "alertingFragment": "<OSN>…</OSN>",
                     "receiveTime": "2017-01-01T23:54:59.590Z" }
              ]
       }
```

```
}
```

9.2.9 Content Change Notification API

Content Change Notification는 새 서명된 패키지 또는 서명된 패키지의 새 버전이 수신되고 **Broadcaster Application**이 section 9.2.1에 지정된 API를 통해 그러한 알림을 수신하도록 가입한 경우 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 발행할 것으로 예상됩니다. 포함된 파일은 **Receiver Web Server**를 통해 사용할 수 있습니다.

알림 메시지에는 수신된 패키지를 참조하는 URI 목록이 포함되어 있습니다.

Content Change Notification의 Semantics는 표 9.2.9-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-contentChange.json에 정의된 대로입니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.2.9-1> Content Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
method	1	string	"org.atsc.notify"
msgType	1	enum	"contentChange"
packageList	1	array	새로 수신된 패키지 URI가 포함된 목록
items	0..N	string(uri)	이제 Application Context Cache에서 파일을 사용할 수 있는 ROUTE를 통해 전달된 특정 패키지의 패키지 URI

packageList – 콘텐츠가 수신되고 서명을 확인했으며 이제 **Broadcaster Application**에 액세스할 수 있는 패키지 배열입니다. ROUTE를 통해 전달되는 특정 패키지의 패키지 URI는 EFDT에 신호된 값에서 가져와야 합니다. A/331에 따른 FDT-Instance.File@Content-Location 속성[3]. **Broadcaster Application**은 이러한 패키지 URI를 사용하여 수신 또는 업데이트된 파일 컬렉션을 확인할 수 있습니다.

예를 들어, 특정 패키지에서 다양한 콘텐츠 파일의 새 버전이 수신되었고, 콘텐츠 서명이 확인되었으며, 이제 **Receiver Web Server**를 통해 파일을 사용할 수 있음을 **Broadcaster Application**에 알리기 위해 **Receiver**는 다음 JSON-RPC 명령을 실행할 수 있습니다.

<표 9.2.9-2> Example of Content Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "contentChange",
        "packageList": ["http://192.168.1.10/items/zz/content"]
        }
}
```

Broadcaster Application은 이러한 알림이 수신되면 자체 또는 자체 일부를

다시 로드하도록 선택할 수 있습니다

9.2.10 Service Guide Change Notification API

Service Guide Change Notification 는 서비스 가이드 데이터 구조의 일부에 변경이 있고 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 그러한 알림을 수신하도록 가입한 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다. 이전 영수증 없이 새 서비스 가이드 조각을 수신하는 것은 버전 변경으로 간주됩니다.

알림 메시지에는 새 서비스 안내서 XML 단편 또는 업데이트된 서비스 안내서 XML 단편을 참조하는 URL 목록이 포함되어 있습니다.

Service Guide Change Notification 의 Semantics 는 표 9.2.10-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-serviceGuideChange.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.10-1> Service Guide Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"serviceGuideChange
urlList	1	array	새로 변경된 서비스 가이드 조각을 가리키는 서비스 가이드 URL 집합을 나열
items	0..N		
sgType	1	enum	"Service", "Schedule" or "Content"
sgUrl	1	string(uri)	관련 서비스 가이드 유형의 XML 조각
Service	1	string(uri)	서비스 가이드 유형과 관련된 서비스의 URI
ontent	0..1	string(uri)	sgType = "Content"인 경우 이 매개변수는 가능한 경우 콘텐츠의 고유 ID 를 제공하는 데 사용

urlList - 새 서비스 가이드 조각 또는 업데이트된 서비스 가이드 조각에 대한 URL 목록을 포함하는 필수 배열입니다.

이 알림은 배열이 비어 있는 경우, 즉 변경사항이 없는 경우에는 발생하지 않을 것으로 예상됩니다. *sgType*, *sgUrl*, *service* 및 *content* 속성의 의미 체계는 section 9.1.9 쿼리 서비스 가이드 URL API 응답에 지정된 대로입니다.

예를 들어, **Receiver** 는 다음 JSON-RPC 명령을 실행하여 새 일정이 수신되었음을 나타낼 수 있습니다.

<표 9.2.10-2> Example of Service Guide Change Notification 1
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "method": "org.atsc.notify",
      "params": {
```

```
                "msgType": "serviceGuideChange",
                "urlList": [
                        { "sgType": "Schedule",
                        "sgUrl":
                        "http://127.0.0.1:8080/wmbc.appctx/Schedule.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B" }
                ]
        }
}
```

제공된 URL은 예시일 뿐입니다. 파일 이름을 포함하여 사용되는 실제 URL은 수신기 구현 및 HTTP 서버를 통해 ESG 파일을 사용할 수 있도록 선택하는 방법에 따라 전적으로 달라집니다.

추가 예로, **Receiver**는 다음 JSON RPC 명령을 실행하여 새 서비스 정보와 관련 일정 및 콘텐츠가 수신되었음을 나타낼 수 있습니다.

<표 9.2.10-3> Example of Service Guide Change Notification 2 [출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "serviceGuideChange",
                "urlList": [
                        { "sgType": "Service",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Service.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B" },
                        { "sgType": "Schedule",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Schedule.xml",
                        "service": "https://doi.org/10.5239/8A23-2B0B" },
                        { "sgType": "Content",
                        "sgUrl": "http://127.0.0.1:8080/wmbc.appctx/Content.xml",
                        "servict": "https://doi.org/10.5239/8A23-2B0B",
                        "content":    "urn:eidr:10.5240:7791-8534-2C23-9030-8610-
                5" }
                ]
        }
}
```

이 예제에 표시된 접두사는 정보 제공용일 뿐입니다. **Broadcaster Application**은 경로에 대해 가정하지 않아야 하며 단순히 프래그먼트 데이터에 직접 액세스하는 데 사용해야 합니다.

참조된 서비스 가이드 파일(이 예에서 Service.xml, Schedule.xml 및 Content.xml)은 각각 A/332[4]에 설명된 서비스, 일정 및 콘텐츠 XML 단편을 포함해야 합니다. **Receiver**는 **Broadcaster Application**에서 사용할 수 있도록 하기 전에 바이너리 SGDU 구조에서 각 XML 조각을 추출해야 합니다.

ESG 파일을 **Broadcaster Application**과 연결하려면 ESG 서비스 ROUTE 세션의 LCT 채널에서 ESG 파일을 보낼 때 정의된 EFDT(Extended FDT) 요소인 FDT-Instance@appContextIdList 에 해당 **Application Context Identifier**가 제공되어야 합니다. FDT 연장에 대한 설명은 A/331[3]에서, ESG 서비스는 A/332[4]에서 확인할 수 있습니다. **Broadcaster Application** 에 ESG 데이터가 필요하지 않은 경우 **Application Context Identifier**를 EFDT에

포함할 필요가 없습니다.

9.2.11 Signaling Data Change Notification API

Signaling Data Change Notification 는 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 나열된 테이블에 대한 업데이트 알림을 수신하도록 가입했거나 버전이 이전에 통지된 이후 LLS 테이블 또는 SLS 단편의 새 버전이 수신되는 경우 **Receiver** 에 의해 현재 실행 중인 **Broadcaster Application** 에 발행될 것으로 예상됩니다. **Broadcaster Application** 은 section 9.1.10 에 지정된 **Query Signaling Data API** 를 사용하여 새 복사본을 가져와 신호 데이터 변경 알림에 응답할 수 있습니다.

이 알림은 AEAT 및 OSN 신호를 포함하여 LLS 변경이 감지될 때마다 발행됩니다. 이 신호는 **Alerting Change Notification API** (section 9.2.8)와 독립적이며, 즉, **Broadcaster Application** 이 **Signaling Data Change** 및 **Alerting Change Notifications** 을 모두 구독하는 경우 새 AEAT 또는 OSN 조각이 감지될 때 두 알림이 모두 발행될 것으로 예상됩니다.

Signaling Data Change Notification Semantics 는 표 9.2.11-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-signalingData.json 에 정의된 대로여야 합니다.

<표 9.2.11-1> Signaling Data Change Notification Semantic
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"signalingData"
objectLlst	0..1	array	이 알림의 결과로 변경된 신호 테이블을 나열
items	0..N		
name	1	string or integer	section 9.1.10 의 *names* 정의를 참조
version	1	integer	신호 요소의 버전
group	0..1	integer	LLS 테이블에 필요. LLS 그룹 ID 제공
table	1	string (XML or JSON or Base64)	시그널링 table 데이터
encoding	0..1	string	UTF-8 이 아닌 경우 콘텐츠 인코딩

objectList – 이 선택적 속성의 의미는 Section 9.1.10 의 **Query Signaling Data response API** 에 정의된 동일한 이름의 속성과 동일합니다. **Receiver** 는 **Broadcaster Application** 이 알림을 수신한 후 **Query Signaling Data API** 를 호출할 때 발생할 수 있는 타이밍 문제를 방지하기 위해, 알림의 objectList 에 변경된 테이블들을 포함하는 것이 권장됩니다.

다음은 알림(notification)의 예시입니다:

<표 9.2.11-2> Example of Signaling Data Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
                "params": {
                        "msgType": "signalingData",
                        "objectList": [
                                { "name": 1,
                                "version": 23,
                                "group": 1,
                                "table": "<SLT …… </SLT>" },
                                { "name": "MPD",
                                "version": 65,
                                "table": "<MPD …… </MPD>" }
                ]
        }
}
```

9.2.12 Dialog Enhancement Preference Change Notification API

Dialog Enhancement Preference Change Notification 은 사용자가 Dialog Enhancement 처리에 대한 기본 설정을 변경하고 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 그러한 알림을 수신하도록 가입한 경우 **Receiver** 에 의해 현재 실행 중인 **Broadcaster Application** 발행될 것으로 예상됩니다.

Dialog Enhancement Preference Change Notification 의 Semantics 는 표 9.2.12-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-dialogEnhancementPrefChange.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.12-1> Dialog Enhancement Preference Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"dialogEnhancementPrefChange"
dialogEnhancementPref	1	integer	사용자의 새로운 대화 강화 선호 이득 값(dB)

dialogEnhancementPref – 이 필수 속성은 **Query Dialog Enhancement Preference response API** 에서 동일한 이름의 속성에 대해 설명된 의미 체계를 준수해야 합니다(section 9.1.11 참조).

예를 들어, 사용자가 기본 설정에서 Dialog Enhancement 처리를 9dB 의 게인 값으로 변경하면 **Receiver** 는 다음 알림을 보냅니다.

<표 9.2.12-2> Example of Dialog Enhancement Preference Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "dialogEnhancementPrefChange",
                "dialogEnhancementPref": 9
        }
}
```

9.2.13 Dialog Enhancement Limit Change Notification API

Dialog Enhancement Limit Change Notification 은 현재 디코딩된 오디오 스트림에서 Dialog Enhancement 처리에 대한 제한이 변경되고 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 이러한 알림을 수신하도록 가입한 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다.

Dialog Enhancement Limit Change Notification 의 Semantics 는 표 9.2.13-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-dialogEnhancementLimitChange.json 에 정의된 대로여야 합니다. 매개변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.2.13-1> Dialog Enhancement Limit Change Notification Semantics
[출처: A344]

Property Name		Use	Data Type	Short Description
Jsonrpc		1	string	“2.0”
method		1	string	"org.atsc.notify"
msgType		1	enum	"dialogEnhancementLimitChange”
dialogEnhancementLimit		1		
	max	1	integer	허용되는 최대 대화 향상 게인 값(dB)
	min	1	integer	허용되는 최소 대화 향상 게인 값(dB)

dialogEnhancementLimit – 오디오 디코더에 적용할 Dialog Enhancement 처리에 허용되는 게인 값의 범위(dB)입니다.

max – 오디오 디코더에 적용될 Dialog Enhancement 처리에 허용되는 최대 게인 값(dB)입니다.

min – 오디오 디코더에 적용할 Dialog Enhancement 처리에 허용되는 최소 게인 값(dB)입니다.

예를 들어 디코딩된 오디오 스트림이 오디오 스트림 메타데이터에서 Dialog Enhancement 처리를 0dB 에서 6dB 범위로 제한하는 경우 **Receiver** 는 다음 알림을 제공합니다.

<표 9.2.13-2> Example of Dialog Enhancement Limit Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "dialogEnhancementLimitChange",
                "dialogEnhancementLimit": {
                        "max": 6,
                        "min": 0
                }
        }
}
```

9.2.14 RF Signal Change Notification API

RF Signal Change Notification 은 표 9.2.14-1 에 나열된 속성 중 하나라도 현재 조정된 RF 채널에서 변경되고 **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 이러한 알림을 수신하도록 가입한 경우 수신기가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다

RF Signal Change Notification 의 Semantics 는 표 9.2.14-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-rfSignalChange.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

구독 시 **Receiver** 는 즉시 초기 알림을 발행해야 합니다. 알림은 초당 한 번 이상 없어야 합니다. 필드 값의 계산은 **Receiver** 에 따라 다릅니다

<표 9.2.14-1> RF Signal Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"rfSignalChange"
rfChannel	1	integer	RF 채널 번호
frequency	1	integer	주파수(Hertz)
signalQuality	1	enum	"lost", "weak" or "strong"
signalStrength	0..1	integer (0..100)	신호 강도는 0~100 의 백분율로 표시됨
gainLevel	0..1	integer (0..100)	게인 수준은 0~100 의 백분율로 표시됨
bootstrapLock	0..1	boolean	성공적인 부트스트랩 기호 획득
alpLock	0..1	boolean	성공적인 ALP 데이터 수집

rfChannel - 이 필수 속성에는 현재 조정된 RF 채널의 RF 채널 번호가 포함되어야 합니다.

frequency - 이 필수 속성에는 현재 조정된 RF 채널의 중심 주파수(Hz)가 포함되어야 합니다.

signalQuality - 이 필수 속성은 Android Media TV *onSignalStrengthUpdated* API[52]에 맞춰 단순화된 전체 RF 품질을 제공해야 합니다. 사용 가능한 값은 "lost", "weak" 또는 "strong"입니다.

signalStrength - 이 선택적 속성은 백분율로 보고된 수신된 신호 강도를 제공해야 합니다. 사용 가능한 값은 0 에서 100 까지의 범위입니다.

gainLevel - 이 선택적 속성은 백분율로 보고된 튜너의 평균 신호 게인 레벨을 제공해야 합니다. 사용 가능한 값은 0 에서 100 까지의 범위입니다.

bootstrapLock - 이 선택적 Boolean 속성은 "true"인 경우 A/321 에서 정의한 기호 획득[1]을 확인해야 합니다. "false" 값은 부트스트랩 기호 획득을 확인할 수 없음을 나타냅니다.

alpLock - 이 선택적 Boolean 속성은 "true"인 경우 A/330 ATSC ALP(Link-Layer Protocol) 데이터 수집을 확인합니다[2]. "false" 값은 ALP 데이터 수집을 확인할 수 없음을 나타냅니다.

예를 들어, **Receiver**는 다음 알림을 제공할 수 있습니다.

<표 9.2.14-2> Example of RF Signal Change Notification
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "org.atsc.notify",
       "params": {
              "msgType": "RFSignalChange",
              "rfChannel": 15,
              "frequency": 479000000,
              "signalQuality": "strong",
              "signalStrength": 75,
              "gainLevel": 50,
              "bootstrapLock": true,
              "alpLock": true
       }
}
```

9.3 Cache Request APIs

Cache Request APIs는 현재 실행 중인 **Broadcaster Application**에서 **Receiver** 가 broadband 서버에서 하나 이상의 오브젝트를 다운로드하여 **Application Context Cache**의 지정된 위치에 배치하도록 요청하는 데 사용할 수 있습니다. 파일은 URL로 개별적으로 식별되거나 DASH MPD 또는 Period를 지정할 수 있으며, 이 경우 MPD 또는 Period에서 참조하는 모든 미디어 파일이 요청됩니다.

9.3.1 Cache Request API

Broadcaster Application은 **Cache Request API**를 사용하여 수신자에게 하나 이상의 표시된 파일을 다운로드하도록 요청할 수 있습니다. **Broadcaster Application** 은 광고가 실시간으로 스트리밍되는 경우 네트워크 정체로 인해 발생할 수 있는 재생 문제를 방지하기 위해 광고 교체 시간 전에 broadband를 통해 광고 콘텐츠를 다운로드하도록 요청할 수 있습니다.

Cache Request API에 대한 **Receiver**의 응답은 표시된 파일이 **Application Context Cache**에 이미 있는지 여부를 나타냅니다. 따라서 API를 사용하여 하나 이상의 표시된 파일이 **Application Context Cache**에 있는지 여부를 확인할 수도 있습니다. 상태 확인 기능은 방송망 또는 broadband망 전송 경로에 의해

도착했을 수 있는 파일에 대해 작동합니다.

Section 6.2 에 설명된 것처럼 **Application Context Cache** 의 스토리지 기능 및 관리는 수신자에 따라 다르므로 **Application Context Cache** 의 상태에 따라 이 API 를 통해 요청된 파일이 저장되거나 저장되지 않을 수 있습니다. 그러나 **Broadcaster Application** 은 section 9.4 에 정의된 **Query Cache Usage API** 를 사용하여 **Application Context ID** 에 할당된 **Application Context Cache** 의 스토리지 할당량을 확인할 수 있습니다. Section 9.7 에 정의된 **Mark Unused API** 를 사용하여 캐시된 파일이 사용되지 않음을 **Application Context Cache** 시스템에 나타낼 수 있습니다. 현재 실행 중인 **Broadcaster Application** 이 종료되면 **Receiver** 는 진행 중인 모든 파일 검색 프로세스를 취소하고 이 API 에서 요청한 모든 캐시된 파일을 해제할 수 있습니다.

메서드 이름 "*org.atsc.CacheRequest*"는 이 표준의 다른 부분의 메서드 이름과 일치하지 않는 대문자 "C"로 시작합니다. 독자는 문제를 피하기 위해 메서드 이름을 그대로 사용해야 합니다.

Cache Request Request 의 Semantics 는 표 9.3.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.CacheRequest-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.3.1-1> Cache Request Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.CacheRequest"
sourceUrl	0..1	string (uri)	파일을 검색할 기본 URL
targetUrl	0..1	string(uri-reference)	**Application Context Cache** 에 파일이 배치될 대상 URL
URLs	1		
items	0..N	string(uri-reference)	검색하거나 상태를 제공할 파일의 상대 URL(설명 참조)

sourceURL - *sourceURL* 이 있는 경우 이 API 는 **Receiver** 에게 제공된 URL 배열에서 참조되는 파일을 검색하도록 요청합니다. 여기서 *sourceURL* 은 각 파일의 기본 URL 입니다. *sourceURL* 이 있는 경우 https 프로토콜 식별자를 포함해야 합니다. *sourceURL* 이 없는 경우 API 는 **Receiver** 가 **Application Context Cache** 내에서 식별된 파일의 존재 여부에 대한 정보를 반환하도록 요청해야 합니다.

targetURL - 이 상대 URL 은 파일이 배치될 베이스를 기준으로 **Application Context Cache** 내의 위치를 나타냅니다. *sourceURL* 이 없는 경우 *targetURL* 은 수신자가 URL 배열에 지정된 파일을 찾고 모든 파일이 있는지 여부를 표시하여 응답해야 하는 기본 기준을 기준으로 **Application Context Cache** 내의 위치를 나타내야 합니다. *sourceURL* 이 있고 *targetURL* 이 없는 경우 파일은 **Application Context Cache** 의 루트를 기준으로 각 URL 아래에 저장되어야 합니다.

URL - *sourceURL* 이 있는 경우 URL 배열은 **Application Context**

Cache 에 검색하고 저장할 파일의 상대 URL 을 포함하는 하나 이상의 문자열 배열을 나타내야 합니다. URL 의 각 URL 문자열은 상대 URL 이어야 합니다. Broadband 서버에서 파일을 검색하기 위한 유효 URL 은 *sourceURL* 과 파일의 URL 문자열을 연결한 것입니다. *sourceURL* 이 없는 경우 각 URL 문자열은 *targetURL* 과 파일의 URL 을 연결하여 응용 프로그램 컨텍스트 캐시에 있을 수 있는 파일을 참조해야 하며, API 에 대한 응답은 참조된 모든 파일이 캐시에 있는지 여부를 나타냅니다.

Cache Request Response 의 Semantics 는 표 9.3.1-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.CacheRequest-response.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.3.1-2> Cache Request Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		구독이 성공하면 빈 객체(empty object) 가 반환 구독에 실패하면 오류 구조(error structure) 가 반환
cached	1	boolean	요청한 파일이 캐시되었는지 여부
error	oneOf X		Section 8.3.3 참조

cached - 이 필수 boolean 결과는 URL 에서 참조되는 모든 파일이 **Application Context Cache** 에 있고 만료된 파일이 없음을 "true"인 경우 나타냅니다. "false"인 경우 cached 는 하나 이상의 파일이 없거나 만료되었음을 나타냅니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 요청된 콘텐츠를 찾을 수 없습니다.
- -5: 요청을 준수하기 위해 광대역 연결을 사용할 수 없습니다.
- -15: 요청의 *sourceURL* 또는 *targetURL* 에 지정된 URL 형식이 잘못되었습니다.
- -16: 요청의 하나 이상의 URL 에 지정된 URL 형식이 잘못되었습니다.

예를 들어, **Broadcaster Application** 은 https://foo.com/service1 의 광대역 서버로부터 3 개의 PNG 파일의 다운로드를 요청하고 이러한 파일을 images/하위 디렉토리 또는 그 아래에 지정된 하위 디렉토리의 **Application Context Cache** 에 저장하기를 원할 수 있습니다. 이 세 가지 파일 각각의 원본 및 대상은 다음과 같습니다.

1. 파일 1:
   - 출처: https://foo.com/service1/A/big-image1.png
   - **App Context Cache** 의 대상 위치: 이미지/A/big-image1.png
2. 파일 2:
   - 출처: https://foo.com/service1/B/big-image2.png
   - **App Context Cache** 의 대상 위치: images/B/big-image2.png

3. 파일 3:

- 출처: https://foo.com/service1/C/big-image3.png
- App Context Cache 의 대상 위치: images/C/big-image3.png

다운로드를 수행하기 위해 Broadcaster Application 은 다음 API 를 실행할 수 있습니다.

<표 9.3.1-3> Example of Cache Request Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.CacheRequest",
       "params": {
              "sourceURL": "https://foo.com/service1/",
              "targetURL": "images/",
              "URLs":["A/big-image1.png","B/big-image2.png","C/big-
       image3.png"]
       },
       "id": 37
}
```

이 예에서는 URL https://foo.com/service1/A/big-image1.png 를 사용하여 첫 번째 PNG 파일을 가져와 하위 디렉토리 images/A/big-image1.png 의 Application Context Cache 에 배치합니다. 검색 프로세스를 성공적으로 시작하면 이전에 파일을 검색하지 않은 경우 Receiver 는 다음과 같이 응답합니다.

```
<-- {
       "jsonrpc": "2.0",
       "result": {"cached": false},
       "id": 37
}
```

cached 는 "false"로 이러한 파일이 아직 없음을 나타냅니다. 모든 파일이 이미 Application Context Cache 에 있고 만료된 파일이 없는 경우 cached 는 "true"를 반환했을 것입니다. 그렇지 않으면 Receiver 가 파일을 다시 다운로드하기 시작합니다.

Broadcaster Application 이 나중에 이러한 파일 중 처음 두 파일이 성공적으로 다운로드되었는지 확인하려는 경우 Receiver 에 다음 API 를 실행할 수 있습니다.

<표 9.3.1-4> Example of Cache Request Resonse
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.CacheRequest",
       "params" {
              "targetURL": "images/",
              "URLs":["A/big-image1.png", "B/big-image2.png"]
       },
       "id": 38
```

```
}
```

9.3.2 Cache Request DASH API

Cache Request DASH API 는 현재 실행 중인 **Broadcaster Application** 에서 특정 파일을 broadband 망을 통해 검색하고 **Application Context Cache** 에 저장해야 함을 **Receiver** 에게 표시하는 데 사용할 수 있습니다. 이 API 를 사용하여 각 URL 을 개별적으로 나열하는 대신 파일이 MPEG DASH 기간 XML 조각 또는 전체 DASH MPD 에 지정됩니다. 전체 DASH MPD 가 지정된 경우 MPD 파일 및 MPD 파일에 지정된 MPEG DASH 세그먼트는 broadband 을 통해 검색되어 저장되어야 합니다. 각 MPEG DASH 세그먼트 파일의 URL 은 MPEG DASH 사양[29]에 따라 생성되어야 합니다. XLink 해결 알림 API(섹션 9.13.1)에 대한 응답으로 **Broadcaster Application** 은 동일한 DASH 기간 XML 조각을 제공할 수 있습니다.

Cache Request DASH API 를 사용하여 DASH Period 또는 MPD 에 표시된 파일이 **Application Context Cache** 에 있고 만료되지 않았는지 여부를 확인할 수도 있습니다. 상태 확인 기능은 방송망 또는 broadband 망 전송 경로에 의해 도착했을 수 있는 파일에 대해 작동합니다.

이 API 는 리소스가 캐시될 때까지 기다리지 않습니다. **Broadcaster Application** 은 cached = "true"가 될 때까지 이 API 를 반복적으로 호출해야 합니다. **Receiver** 가 하나 이상의 리소스를 캐시할 수 없다고 판단하면(예: 인터넷 손실, HTTPS 가 404 반환 등) 오류 -4 를 반환합니다.

메서드 이름 "org.atsc.CacheRequestDASH"는 이 표준의 다른 부분의 메서드 이름과 일치하지 않는 대문자 "C"로 시작합니다. 독자는 문제를 피하기 위해 메서드 이름을 그대로 사용해야 합니다.

캐시 요청 DASH 요청 Semantics 는 표 9.3.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.CacheRequestDASH-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.3.2-1> Cache Request DASH Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.CacheRequestDASH"
object A	oneOf X		기간 요청을 지정
sourceUrl	0..1	string (uri)	기간 참조 세그먼트 파일을 검색할 기본 URL
targetUrl	1	string(uri-reference)	**Application Context Cache** 에 파일이 배치될 대상 URL
Period	1	string (XML)	참조된 세그먼트가 캐시되도록 요청된 DASH 기간 XML 조각
object B	oneOf X		MPD 요청을 지정
sourceUrl	0..1	string	MPD 참조 세그먼트 파일을 검색할

			(uri)	기본 URL
	targetUrl	0..1	string(uri-reference)	Application Context Cache 에 파일이 배치될 대상 URL
	mpdFileName	1	string	참조된 세그먼트가 캐시되도록 요청된 DASH MPD 파일

API 가 DASH Period 과 함께 사용되는 경우:

sourceURL - *sourceURL* 이 있는 경우 이 API 는 **Receiver** 에게 제공된 DASH 기간 XML 조각에서 참조되는 미디어 파일을 검색하도록 요청합니다. 여기서 *sourceURL* 은 기간의 URL 에 지정된 파일의 기본 URL 입니다. *sourceURL* 이 있는 경우 https 프로토콜 식별자를 포함해야 합니다. *sourceURL* 이 없는 경우 API 는 **Receiver** 가 **Application Context Cache** 내에서 식별된 파일의 존재 여부에 대한 정보를 반환하도록 요청해야 합니다.

targetURL - 이 상대 URL 은 파일이 배치될 베이스를 기준으로 **Application Context Cache** 내의 위치를 나타냅니다. *sourceURL* 이 없는 경우 *targetURL* 은 수신자가 Period 에서 참조하는 파일을 찾고 모든 파일이 있고 만료되지 않았는지 여부를 표시하여 회신해야 하는 기본 기준을 기준으로 **Application Context Cache** 내의 위치를 나타내야 합니다. *sourceURL* 이 있고 *targetURL* 이 없는 경우 파일은 **Application Context Cache** 의 루트를 기준으로 각 URL 아래에 저장되어야 합니다.

Period - 기간은 A/331[3]을 준수하는 MPEG DASH 의 기간으로 정의된 XML fragment 를 나타냅니다. 각 미디어 세그먼트 및 초기화 세그먼트 URL 은 MPEG DASH [29] subclause 5.6 의 처리 규칙을 사용하여 구성됩니다. 기간은 상대 URL 참조만 사용합니다. **Period***@duration* 속성이 있어야 합니다. *sourceURL* 이 포함된 경우 기간의 URL 은 참조된 광대역 서버에 있는 미디어 파일로 확인됩니다

API 를 DASH MPD 와 함께 사용하는 경우:

sourceURL - 이 API 는 **Receiver** 에게 *mpdFileName* 으로 식별되는 DASH MPD 에서 참조되는 미디어 파일을 검색하도록 요청하며, 여기서 *sourceURL* 은 MPD 를 검색할 수 있는 광대역 서버의 URL 입니다. *sourceURL* 이 있는 경우 https 프로토콜 식별자를 포함해야 합니다. *sourceURL* 이 없는 경우 API 는 **Receiver** 가 **Application Context Cache** 내에서 참조된 MPD 로 식별된 파일의 가용성에 대한 정보를 반환하도록 요청해야 합니다. sourceURL 이 없는 경우 MPD 자체가 **Application Context Cache** 에 없거나 참조하는 파일이 없거나 만료된 경우 요청에 대한 응답에 "cached":false 가 표시됩니다.

targetURL - *sourceURL* 이 있는 경우 API 는 **Receiver** 에게 표시된 MPD 및 MPD 자체와 연결된 파일을 검색하여 **Application Context Cache** 에 배치하도록 요청합니다. 이 경우 *targetURL* 은 파일이 배치될 베이스를 기준으로 **Application Context Cache** 내의 위치를 나타내야 합니다. **Receiver** 는 MPD 를 검색하여 targetURL 에서 제공하는

Application Context Cache 의 위치에 배치해야 합니다. *sourceURL* 이 없는 경우 *targetURL* 은 Receiver 가 MPD 및 MPD 에서 참조하는 파일을 찾고 MPD 및 MPD 가 참조하는 모든 파일이 존재하고 만료되지 않았는지 여부를 표시해야 하는 기본 위치를 기준으로 Application Context Cache 내의 위치를 나타내야 합니다. *sourceURL* 이 있고 *targetURL* 이 없는 경우 파일은 Application Context Cache 의 루트를 기준으로 각 URL 아래에 저장되어야 합니다.

mpdFileName - 필수 *mpdFileName* 은 A/331[3]을 준수하는 MPEG DASH MPD 의 파일 이름을 나타냅니다. 표시된 MPD 에서 참조되는 각 미디어 세그먼트 및 초기화 세그먼트의 URL 은 MPEG DASH [29] subclause 5.6 의 처리 규칙을 사용하여 구성됩니다. 참조된 MPD 에는 상대 URL 만 포함되어야 합니다. *sourceURL* 이 포함된 경우 MPD 의 URL 은 참조된 broadband 서버에 있는 미디어 파일로 확인되어야 하며, MPD 자체는 *mpdFileName* 에 지정된 파일 이름으로 *sourceURL* 에 표시된 서버 위치에 있어야 합니다.

MPEG DASH [29] subclause 5.6.4 에 따르면, "MPD 의 각 레벨에 있는 URL 은 문서의 해당 레벨 또는 기본 URL 자체를 확인하는 경우 위의 레벨에 지정된 *BaseURL* 요소와 관련하여 RFC 3986 에 따라 확인됩니다(RFC 3986 [25] section 5.1 에 정의된 문서 '기본 URI'는 MPD 레벨 위의 레벨로 간주됨)." 이 API 의 경우 *sourceURL* 은 광대역 서버의 문서 "기본 URI"이고 *targetURL* 은 Application Context Cache 의 문서 "기본 URI"입니다.

Cache Request DASH Response 의 Semantics 는 표 9.3.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.CacheRequestDASH-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.3.2-2> Cache Request Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		구독이 성공하면 빈 객체(empty object) 가 반환 구독에 실패하면 오류 구조(error structure) 가 반환
cached	1	boolean	요청한 파일이 캐시되었는지 여부
error	oneOf X		Section 8.3.3 참조

cached - 이 boolean 결과는 "true"인 경우 기간 또는 MPD 에서 참조되는 모든 파일이 표시된 위치의 Application Context Cache 에 있고 만료된 파일이 없음을 나타냅니다. "false"인 경우 cached 는 하나 이상의 파일이 만료되었거나 존재하지 않음을 나타냅니다. 요청에 sourceURL 이 있는 경우 표시된 파일이 Application Context Cache 에 이미 있고 만료된 파일이 없는 경우 "true" 결과가 반환됩니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 콘텐츠를 찾을 수 없습니다.
- -5: 광대역 연결을 사용할 수 없습니다.
- -11: 표시된 MPD 에 액세스할 수 없습니다.
- -15: 요청의 *sourceURL* 또는 *targetURL* 에 지정된 URL 형식이 잘못되었습니다.
- -17: Period 에 지정된 MPEG DASH 조각의 형식이 잘못되었습니다.
- -18: 참조된 MPD 세그먼트 파일을 찾을 수 없습니다.

예를 들어, **Broadcaster Application** 이 broadband 서버에서 https://wxyz.com/svc4.4/content/ 의 broadband 서버에서 하나의 기간에 해당하는 MPEG DASH 미디어 세그먼트 파일을 가져오도록 요청하고 advertising1/의 **Application Context Cache** 에 배치하려는 경우 다음 API 를 실행할 수 있습니다.

<표 9.3.2-3> Example of Cache Request DASH Request 1
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.CacheRequestDASH",
      "params": {
            "sourceURL": "https://wxyz.com/svc4.4/content/",
            "targetURL": "advertising1/",
            "Period":   "<Period   start='PT9H'   duration='PT30S'>
      <AdaptationSet   mimeType='video/mp4'/>   <SegmentTemplate
      timescale='9000'              media='video/xbc$Number$.mp4v'
      duration='90000' startNumber='32401' /><Representation id='v2'
      width='1920' height='1080'/></Period>"
      },
      "id": 38
}
```

결과 비디오 미디어 세그먼트 파일은 advertising1/video/하위 디렉터리의 **Application Context Cache** 에 검색되어 저장됩니다. 검색 프로세스를 성공적으로 시작하면 이전에 파일을 검색하지 않은 경우 **Receiver** 는 다음과 같이 응답합니다.

<표 9.3.2-4> Example of Cache Request DASH Response 1
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {"cached": false},
      "id": 38
}
```

응답에서 캐시된 값 "false"는 요청된 파일이 모두 캐시에 이미 존재하지 않음을 나타냅니다. 모든 파일이 **Application Context Cache** 에 이미 있고 만료된 파일이 없는 경우 캐시된 파일은 "true"를 반환하고 그렇지 않으면 수신자가 파일을 다시 다운로드하기 시작합니다.

Broadcaster Application 이 나중에 표시된 DASH 기간과 관련된 파일이 성공적으로 다운로드되었는지 확인하려는 경우 Receiver 에 다음 API 를 실행할 수 있습니다.

<표 9.3.2-5> Example of Cache Request DASH Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.CacheRequestDASH",
        "params": {
                "targetURL": "advertising1/",
                "Period":    "<Period    start='PT9H'    duration='PT30S'>
        <AdaptationSet    mimeType='video/mp4'/>    <SegmentTemplate
        timescale='9000'                media='video/xbc$Number$.mp4v'
        duration='90000' startNumber='32401' /> <Representation id='v2'
        width='1920' height='1080'/></Period>"
        },
        "id": 37
}
```

초기화 세그먼트를 포함하여 표시된 모든 미디어 세그먼트 파일이 있는 경우 Receiver 는 다음과 같이 응답할 수 있습니다.

<표 9.3.2-6> Example of Cache Request DASH Response 2-1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"cached": true},
        "id": 37
}
```

표시된 미디어 세그먼트 파일 또는 초기화 세그먼트가 누락된 경우 Receiver 는 다음과 같이 응답할 수 있습니다.

<표 9.3.2-7> Example of Cache Request DASH Response 2-2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"cached": false},
        "id": 37
}
```

9.4 Query Cache Usage API

Broadcaster Application 이 연결된 Application Context ID 에 할당된 캐시의 총 할당량 크기와 Application Context ID 계층 구조에서 캐시의 총 현재 사용량을 알고 싶다면 Query Cache Usage API 를 사용할 수 있습니다.

Query Cache Usage Request 의 Semantics 는 표 9.4-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.cacheUsage-request.json 에 정의된 대로여야 합니다.

<표 9.4-1> Query Cache Usage Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.query.cacheUsage”

Query Cache Usage Response의 Semantics는 표 9.4-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.cacheUsage-response.json에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.4-2> Query Cache Usage Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
usageSize	1	Integer	현재 Application Context Cache에 사용된 총 바이트 수
quotaSize	1	Integer	현재 Application Context Cache에 할당된 총 바이트 수
error	oneOf X		Section 8.3.3 참조

usageSize – **Broadcaster Application**의 **Application Context ID**와 연결된 캐시의 총 사용량 바이트 크기입니다.

quotaSize – **Broadcaster Application**의 **Application Context ID**에 할당된 할당량의 총 크기(바이트)입니다

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None – 이 API와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application**션이 캐시 사용 상태를 쿼리하려는 경우 다음과 같이 할 수 있습니다.

<표 9.4-3> Example of Query Cache Usage Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.cacheUsage",
       "id": 39
}
```

사용 크기가 8,475,337 바이트이고 **Application Context ID**에 사용할 수 있는 할당량 크기가 209,715,200 바이트인 경우 **Receiver**는 다음과 같이 응답할 수 있습니다.

<표 9.4-4> Example of Query Cache Usage Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
              "usageSize": 8475337,
```

```
            "quotaSize": 209715200
        },
        "id": 39
}
```

9.5 Event Stream APIs

Broadcast Applications 을 위한 **Event** 는 브로드캐스트 미디어에서 미디어와 함께 대역 내 이벤트 메시지('emsg' 또는 'evti') 상자 또는 DASH MPD 의 기간 수준에서 정적 *EventStream* 요소로 발생하거나, 신호 서버 또는 콘텐츠 복구 서버에서 broadband 망을 통해 얻거나, 비디오 워터마크에서 디코딩된 비디오 콘텐츠에서 감지될 수 있습니다. 이러한 **Event** 는 **Broadcast Applications** 측에서 대화형 작업을 시작하거나 새 버전의 파일이 전달되고 있음을 나타낼 수 있습니다. ATSC 3.0 애플리케이션에 대한 이벤트 전달 및 이러한 애플리케이션 이벤트와 기본 콘텐츠의 동기화에 대한 사양은 A/337 표준[6]에서 찾을 수 있습니다.

AMP 미디어 재생의 경우 **Events** 의 구문 분석 및 처리는 **Broadcast Applications** 에서 수행될 것으로 예상됩니다. RMP 미디어 재생의 경우 이 기능을 지원하려면 세 가지 API 가 필요합니다.

- **Event Stream** 구독
- **Event Stream** 구독 취소
- 구독한 **Event Stream** 에서 **Event** 수신

9.5.1 Event Stream Subscribe API

현재 **Event Stream** 알림을 구독하는 **Broadcast Applications** 은 MPD 또는 미디어 세그먼트에서 RMP 재생 중에 특정 **Event Stream** 이벤트가 발생할 때 알림을 받을 것으로 예상됩니다. MPEG DASH 의 경우 이벤트 메시지 상자('emsg') 상자에는 대역 내 이벤트가 포함되며 MPD 는 기간 수준의 *EventStream* 요소에 정적 이벤트를 포함할 수 있습니다. MMT 기반 서비스의 이벤트는 MPU 의 'evti' 상자로 전달될 수 있습니다[6]. 특정 유형의 이벤트가 발생할 때 알림을 받으려는 **Broadcast Applications** 은 *schemeIdUri* 및 필요에 따라 함께 제공되는 값 매개 변수를 사용하여 해당 유형의 이벤트에 등록할 수 있습니다.

Event Stream Subscribe Request 의 Semantics 는 표 9.5.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.eventStream.subscribe-request.json 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.5.1-1> Event Stream Subscribe Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.eventStream.subscribe

schemeIdUri	1	string	Broadcaster Application 으로 전송하도록 요청된 Event Stream 스키마 ID
value	0..1	string	탐지할 특정 이벤트
dispatchMode	0..1	string	이벤트 알림의 상대적 타이밍을 설정

schemeIdUri – Broadcaster Application 에 관심 있는 Event Stream 이벤트와 연결된 *schemeIdUri* URI 문자열입니다. *schemeIdUri* 의 구문은 [6]에 정의된 AEI.EventStream*@schemeIdUri* 의 구문을 준수해야 합니다.

value – 특정 Event Stream 이벤트를 식별하는 데 사용되는 선택적 문자열입니다.

dispatchMode – 이벤트가 설정되는 시기를 지정하는 선택적 문자열입니다. 값은 다음과 같습니다.

onReceive – (기본값). 이 값으로 설정되거나 *dispatchMode* 속성이 없는 경우 이벤트는 수신된 후 가능한 한 빨리 디스패치되어야 합니다(시작될 때가 아님).

onStart – 이 값으로 설정하면 이벤트가 시작될 때 이벤트가 전달됩니다.

이 작업에 대한 자세한 내용은 [42]의 DASH-IF 이벤트 참조 모델을 참조하세요.

Event Stream Subscribe Response 의 Semantics 는 표 9.5.1-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.eventStream.subscribe-response.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.5.1-2> Event Stream Subscribe Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		구독이 성공하면 빈 객체(empty object) 가 반환 구독에 실패하면 오류 구조(error structure) 가 반환
error	oneOf X		Section 8.3.3 참조

result – 구독에 성공하면 결과 구조에 요소가 포함되지 않아야 합니다. JSON 에서는 "result": {}로 표시됩니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -16: *schemeIdUri* 속성에 잘못된 형식의 URI 가 포함되어 있음을 나타냅니다.

예를 들어 Broadcaster Application 이 *schemeIdUri* *"urn:uuid:1AD2F3EF-87C8-46B4-BD1D-94C174C278EE"*와 연결된 Event Stream 이벤트를 등록하려는 경우 다음과 같이 구독할 수 있습니다.

<표 9.5.1-3> Example of Event Stream Subscribe Request 1
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
```

```
      "method": "org.atsc.eventStream.subscribe",
      "params":    {"schemeIdUri":    "urn:uuid:1AD2F3EF-87C8-46B4-
BD1D- 94C174C278EE"},
      "id": 22
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.5.1-4> Example of Event Stream Subscribe Response 1
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {},
      "id": 22
}
```

그런 다음 **Receiver**는 아래 section 9.5.3에 정의된 **Event Stream Event API**를 사용하여 *schemeIdUri "urn:uuid:1AD2F3EF-87C8-46B4-BD1D-94C174C278EE"*로 태그가 지정된 **Event Stream** 이벤트를 **Broadcaster Application**에 전달하도록 설정됩니다.

Broadcaster Application이 수반되는 값 = "17"일 때 이 *schemeIdUri*와 연결된 **Event Stream** 이벤트에만 관심이 있는 경우 value 매개 변수를 포함하는 동안 구독할 수 있습니다.

<표 9.5.1-5> Example of Event Stream Subscribe Request 2
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.eventStream.subscribe",
      "params": {
            "schemeIdUri":          "urn:uuid:1AD2F3EF-87C8-46B4-
BD1D94C174C278EE",
            "value": "17"
      },
      "id": 23
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.5.1-6> Example of Event Stream Subscribe Response 2
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {},
      "id": 23
}
```

그런 다음 **Receiver**는 *schemeIdUri "urn:uuid:1AD2F3EF-87C8-46B4-BD1D-94C174C278EE"* 및 값 = "17"으로 태그가 지정된 **Event Stream** 이벤트를 아래 section 9.5.3에 정의된 알림 API를 사용하여 **Broadcaster**

Application 에 전달하도록 설정됩니다. Broadcaster Application 은 구독 취소된 *schemeIdUri* 값으로 태그가 지정된 Event Stream 이벤트 또는 구독된 *schemeIdUri*가 있지만 지정된 값과 일치하지 않는 이벤트에 대해 알림을 받지 않습니다.

Broadcaster Application 은 여러 Event Stream 이벤트(서로 다른 *schemeIdUri* 값 또는 서로 다른 *schemeIdUri*/값 조합 사용)를 구독할 수 있습니다.

일단 구독되면 Broadcaster Application 은 section 9.5.2 에 설명된 API 를 사용하여 구독을 취소할 수 있습니다.

9.5.2 Event Stream Unsubscribe API

Broadcaster Application 이 section 9.5.1 에 정의된 Event Stream Subscribe API 를 사용하여 Event Stream 을 구독한 경우 여기에 정의된 Event Stream Unsubscribe API 를 사용하여 Receiver 에게 식별된 이벤트와 관련된 알림을 중단하도록 요청할 수 있습니다.

Event Stream Unsubscribe Request 의 Semantics 는 표 9.5.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.eventStream.unsubscribe-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.5.2-1> Event Stream Unsubscribe Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.eventStream.unsubscribe"
schemeIdUri	1	string	Broadcaster Application 으로 전송하도록 요청된 Event Stream 스키마 ID
value	0..1	string	탐지할 특정 이벤트

schemeIdUri – Broadcaster Application 이 구독을 제거하려는 이벤트 스트림 이벤트와 연결된 *schemeIdUri* URI 문자열입니다. *schemeIdUri* 의 구문은 [6]에 정의된 AEI.EventStream@*schemeIdUri* 의 구문을 준수해야 합니다.

value – 구독을 제거할 특정 Event Stream 이벤트를 식별하는 데 사용되는 선택적 문자열입니다.

Event Stream Unsubscribe Response 의 Semantics 는 표 9.5.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.eventStream.unsubscribe-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.5.2-2> Event Stream Unsubscribe Response Semantics

[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치합니다.
result	oneOf X		구독이 성공하면 빈 객체(empty object) 가 반환된다. 구독에 실패하면 오류 구조(error structure) 가 반환된다.
error	oneOf X		Section 8.3.3 참조

result - 구독 취소 요청에 성공하면 결과 구조에 요소가 포함되지 않아야 합니다. JSON 에서는 "result": {}로 표시됩니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -16: *schemeldUri* 속성에 잘못된 형식의 URI 가 포함되어 있음을 나타냅니다.
- -24: **Broadcaster Application** 이 요청된 알림을 구독하지 않았습니다.

예를 들어 **Broadcaster Application** 이 *schemeldUri "urn:uuid:1AD2F3EF-87C8-46B4-BD1D-94C174C278EE"*와 연결된 모든 **Event Stream** 이벤트의 구독을 취소하려는 경우 값 매개 변수의 값에 관계없이 다음 API 를 사용할 수 있습니다.

<표 9.5.2-3> Example of Event Stream Unsubscribe Request 1

[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.eventStream.unsubscribe",
      "params":   {"schemeldUri":   "urn:uuid:1AD2F3EF-87C8-46B4-
BD1D-94C174C278EE"},
      "id": 26
}
```

작업이 성공하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.5.2-4> Example of Event Stream Unsubscribe Response 1

[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {},
      "id": 26
}
```

Broadcaster Application 이 value="47" 및 value="48"을 사용하여 동일한 *schemeldUri*를 구독하고 이제 후자의 구독을 취소하려는 경우 다음 API 를 사용할 수 있습니다.

<표 9.5.2-5> Example of Event Stream Unsubscribe Request 2

[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.eventStream.unsubscribe",
      "params": {
```

```
            "schemeIdUri":      "urn:uuid:1AD2F3EF-87C8-46B4-BD1D-
      94C174C278EE",
            "value": "48"
      },
      "id": 29
}
```

작업이 성공하면 **Receiver**는 다음과 같이 응답합니다.

<표 9.5.2-6> Example of Event Stream Unsubscribe Response 2
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {},
      "id": 29
}
```

9.5.3 Event Stream Event API

Event Stream Event는 현재 선택된 서비스의 콘텐츠 또는 현재 재생 중인 콘텐츠에서 이전 이벤트 스트림 구독에서 제공된 schemeIdUri 값(및 구독에 제공된 경우 수반되는 값)과 일치하는 이벤트가 발생하는 경우 RMP 재생 중에 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 발행할 것으로 예상됩니다.

Event Stream Event의 Semantics는 표 9.5.3-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.eventStream.event-notification.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.5.3-1> Event Stream Event Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.eventStream.event"
schemeIdUri	1	string(uri)	이 이벤트가 발생한 소스 이벤트 스트림을 식별
value	0..1	string	스키마로 정의
currentTime	0..1	number (>=0)	부동 소수점 초 단위의 미디어 타임라인의 현재 오프셋
eventTime	1	number (>=0)	이 이벤트가 발생한 미디어 프레젠테이션 시간(부동 소수점 초)
duration	0..1	number (>=0)	부동 소수점 초 단위의 이벤트 기간
id	0..1	integer (0 … 4294967295)	이 이벤트의 상대 ID
data	0..1	string	선택적으로 *contentEncoding* 메소드로 인코딩된 이벤트의 데이터 세트
contentEncoding	0..1	string	데이터 속성에 적용된 인코딩을 식별

schemeIdUri - 이벤트가 연결된 **Event Stream**을 식별하는 필수

문자열입니다. *schemeIdUri* 의 구문은 [6]에 정의된 **AEI.EventStream***@schemeIdUri* 의 구문을 준수해야 합니다.

value – *schemeIdUri* 로 식별되는 **Event Stream** 스키마의 소유자가 정의한 의미 체계가 있는 선택적 문자열입니다.

currentTime – RMP 미디어 프레젠테이션 타임라인의 현재 시간을 나타내는 선택적 부동 소수점 숫자로, *startDate* 에서 오프셋(초)으로 표현됩니다(**Query RMP Media Time API**, section 9.12.1 에 지정됨).

eventTime – RMP 미디어 프레젠테이션 타임라인에서 이벤트가 시작되는 프레젠테이션 시간을 나타내는 필수 부동 소수점 숫자로, *startDate* 에서 오프셋(초)으로 표시됩니다(**Query RMP Media Time API**, section 9.12.1 에 지정됨).

duration – 이벤트 기간(초)을 나타내는 선택적 부동 소수점 숫자입니다.

id – 이 이벤트의 상대 ID 를 나타내는 선택적 숫자입니다.

data – 이 이벤트와 관련된 추가 데이터를 나타내는 선택적 문자열 또는 객체로, *schemeIdUri* 로 식별되는 **Event Stream** 스키마의 소유자가 정의한 의미 체계가 있습니다. *contentEncoding* 속성은 데이터에 적용된 인코딩을 나타냅니다. 콘텐츠 인코딩이 표시되는 경우 데이터를 디코딩하는 것은 **Broadcaster Application** 의 책임입니다.

contentEncoding – **Receiver** 가 데이터에 적용한 콘텐츠 인코딩을 지정하는 선택적 문자열입니다. *contentEncoding* 속성에 대해 지원되는 유일한 값은 "base64"이며, 이는 데이터 매개 변수 값이 base64 로 인코딩되었음을 나타냅니다[27]. 지정된 인코딩이 없으면 응답에 *contentEncoding* 속성이 없음으로 표시됩니다.

RMP 프레젠테이션 타임라인에서 이벤트의 시작 시간은 *currentTime*(RMP **Query RMP Media Time API**, section 9.12.1 에 제공된 대로)이 **Event** 의 *eventTime* 과 같을 때입니다.

MPD Event[41]의 경우, **Receiver** 는 **Event Stream Event** 응답의 데이터 속성을 MPD 이벤트의 **Event***@messageData* 매개 변수의 디코딩되지 않은 값(즉, **Event***@contentEncoding* 로 식별된 디코딩을 적용하지 않고) 또는 MPD 이벤트에 **Event***@messageData* 속성이 없는 경우 MPD 이벤트의 이벤트 요소의 디코딩되지 않은 값으로 채워야 합니다. **Receiver** 는 응답의 *contentEncoding* 속성을 MPD 이벤트의 **Event***@contentEncoding* 매개 변수(있는 경우)로 채워야 합니다.

AEI 이벤트[6]의 경우 수신자는 **Event Stream Event** 응답의 데이터 속성을 AEI 이벤트의 Event 매개 변수로 채워야 하며 응답의 *contentEncoding* 속성은 없어야 합니다.

'emsg' 상자 [6], 'evti' 상자 [6] 또는 **Dynamic Event Message** [5]를 통해 전달되고 수신된 이벤트 데이터가 UTF-8 을 준수하지 않는 경우 수신자는 수신된 이벤트 데이터에 base64 인코딩[27]을 적용하고, **Event Stream Event** 응답의 data 속성을 인코딩된 이벤트 데이터로 채우고, 응답의 *contentEncoding*

속성을 값 "base64"로 채워야 합니다. 수신자는 UTF-8을 준수하는 수신된 이벤트 데이터에 인코딩을 적용해서는 안 됩니다. (스트림 이벤트 데이터의 UTF-8 준수를 설정하는 효율적인 방법은 수신기 제조업체에서 널리 사용할 수 있습니다[51].)

Receiver가 *contentEncoding* 속성에 "base64"를 표시하는 경우 예상되는 이진 또는 UTF-8 페이로드 가정에 관계없이 이벤트 스트림 이벤트 응답의 데이터 속성을 base64로 디코딩하는 것은 **Broadcaster Application**의 책임입니다.

Broadcaster Application 이 tag:xyz.org:evt:xyz.aaa.9 의 *schemeIdUri*를 사용하여 **Event Stream Event**에 등록한 경우 발생할 수 있는 이벤트 스트림 알림 메시지의 예:

<표 9.5.3-2> Example of Event Stream Event
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.eventStream.event",
        "params": {
                "schemeIdUri": "tag:xyz.org:evt:xyz.aaa.9",
                "value": "ev47",
                "currentTime": 1460.2,
                "eventTime": 1450.6,
                "id": 60,
                "data": "d8a0c98fs08-d9df0809s"
        }
}
```

이 예에서 **Broadcaster Application**이 구독에 value 매개변수를 포함하고 해당 매개변수가 "ev47"이 아닌 경우 이 특정 이벤트는 **Broadcaster Application**으로 전달되지 않습니다.

9.6 Request Receiver Actions

9.6.1 Acquire Service API

현재 서비스는 **Receiver**에 대한 요청을 통해 **Broadcaster Application**에 의해 변경되거나 Receiver를 통해 직접 사용자가 변경될 수 있습니다. 이는 동일하거나 다른 RF 채널 내에 있을 수 있습니다. **Broadcaster Application** 시그널링에서 전송된 정보에 따라 **Receiver**는 section 6.3, **Broadcaster Application** 수명 주기에 설명된 작업을 수행합니다.

Broadcaster Application 이 **Receiver** 에게 서비스 선택을 변경하도록 요청할 수 있는 이유는 사용자가 관심을 가질 수 있는 콘텐츠에 대해 동일한 브로드캐스터의 다른 서비스로 이동하기 위해서일 수 있습니다. **Receiver**는 요청을 처리하고 가능한 경우 서비스 선택을 변경합니다.

Acquire Service Request 의미 체계는 표 9.6.1-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.acquire.service-request.json 에 정의된 대로입니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.6.1-1> Acquire Service Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.acquire.service"
svcToAcquire	1	string(uri)	획득할 서비스의 *globalServiceID*

svcToAcquire - 이 필수 문자열은 획득하려는 서비스의 *globalServiceID* (A/331 [3] section 6.3 의 **SLT.Service**@*globalServiceID* 에 정의됨)와 일치해야 한다.

Acquire Service Response 의 의미론적 정의(semantics)는 표 9.6.1-2 에 제시되어 있으며, 구문(syntax)은 org.atsc.acquire.service-response.json 스키마 파일에 정의된 대로 따라야 한다. 매개변수에 대한 추가적인 의미론적 정의(semantics)는 표 이후에 제시된다.

<표 9.6.1-2> Acquire Service Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		구독이 성공하면 빈 객체(empty object) 가 반환 구독에 실패하면 오류 구조(error structure) 가 반환
error	oneOf X		Section 8.3.3 참조

result - 서비스 획득에 성공하면 수신자는 비어 있는 "{}" 결과 객체가 있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -6: 서비스를 찾을 수 없음
- -7: 서비스가 승인되지 않음

예를 들어 **Broadcaster Application** 이 *globalServiceID* "https://doi.org/10.5239/8A23-2B0B"로 표시되는 서비스에 대한 액세스를 요청하는 경우 **Receiver** 에게 이 요청을 실행할 수 있습니다:

<표 9.6.1-3> Example of Acquire Service Request 1
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.acquire.service",
      "params":{"svcToAcquire":"https://doi.org/10.5239/8A23-
2B0B"},
      "id": 59
}
```

The Receiver would respond, if acquisition were successful with:

<표 9.6.1-4> Example of Acquire Service Response 1-1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 59
}
```

If globalServiceID "https://doi.org/10.5239/8A23-2B0B" is unknown to the Receiver, the response would be:

<표 9.6.1-5> Example of Acquire Service Response 1-2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "error": {"code": -6, "message": "Service not found"},
        "id": 59
}
```

9.6.2 Video Scaling and Positioning API

애플리케이션-enhanced 서비스의 **Broadcaster Application** (예: **Receiver Media Player**에서 생성된 비디오 위에 배치된 비디오 평면 내에서 재생)은 비디오 크기 조정 및 위치 지정 JSON-RPC 메서드를 사용하여 RMP가 전체 크기(전체 화면) 미만으로 비디오를 렌더링하도록 요청하고 디스플레이 창 내의 지정된 위치에 배치하도록 요청할 수 있습니다.

Video Scaling and Positioning Request Semantics는 표 9.6.2-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.scale-position-request.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.2-1> Video Scaling and Positioning Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.scale-position
scaleFactor	1	number (10.0 ... 100.0)	비디오 크기를 10.0%에서 100.0%까지 비율로 조정
xPos	1	number (0.0 …100.0-scaleFactor)	전체 화면 너비의 백분율로 비디오를 배치하는 X 축 위치
yPos	1	number (0.0 …100.0-scaleFactor)	전체 화면 높이의 백분율로 비디오를 배치하는 Y 축 위치

scaleFactor - 10.0에서 100.0 사이의 이 필수 숫자 값은 비디오 크기 조정 매개 변수를 나타내며, 여기서 100.0은 전체 화면(크기 조정 없음)을 나타냅니다.

xPos - 0.0에서 100.0 사이의 이 필수 숫자 값-scaleFactor는 RMP

비디오 창 왼쪽의 X 축 위치를 나타내며, 화면의 전체 너비에 대한 백분율로 표시됩니다. 값 0.0 은 비디오 창의 왼쪽이 디스플레이 창의 왼쪽과 정렬되어 있음을 나타냅니다. 값 50.0 은 비디오 창의 왼쪽이 디스플레이 창의 수직 중심선에 정렬되어 있음을 나타냅니다.

yPos - 0.0 에서 100.0 사이의 이 필수 숫자 값-scaleFactor 는 화면 전체 높이의 백분율로 표시되는 RMP 비디오 창 위쪽의 Y 축 위치를 나타냅니다. 값 0.0 은 비디오 창의 위쪽이 디스플레이 창의 위쪽에 정렬되어 있음을 나타냅니다. 값 50.0 은 비디오 창의 상단이 디스플레이 창의 수평 중심선에 정렬되어 있음을 나타냅니다.

좌표계의 0 축은 CSS 와 마찬가지로 왼쪽 상단 모서리여야 합니다.

매개 변수 값은 비디오 창의 어떤 부분도 디스플레이 창 외부에 렌더링되지 않도록 설정되어야 합니다.

Video Scaling and Positioning Request 이 성공하면 *xPos* 및 *yPos* 값을 사용하여 비디오 창의 변환을 수행하기 전에 *scaleFactor* 값을 사용한 크기 조정이 적용되어야 합니다. **Video Scaling and Positioning Request** 이 추가로 성공하면 변환이 이전에 성공한 **Video Scaling and Positioning Request** 에서 이미 변환된 비디오 창이 아니라 재설정된 전체 화면 비디오 창에 적용되어야 합니다.

Video Scaling and Positioning Response Semantics 는 표 9.6.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.scale-position-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.6.2-2> Video Scaling and Positioning Response Semantics [출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	요청 ID 값과 일치
result	oneOf X		크기 조정 및 위치 지정이 성공하면 빈 객체(empty object) 가 반환 크기 조정 및 위치 지정에 실패하면 오류 구조(error structure) 가 반환
error	oneOf X		Section 8.3.3 에서 아래의 내용이 확장
code	1	integer	어떤 문제가 발생했는지 나타내는 오류 코드
message	1	string	오류를 설명하는 간결한 메시지
data	0..1	object	
minimumScaleFactor	0..1	number (10.0 ...100.0)	**Receiver** 가 지원하는 최소 scaleFactor

result - **Video Scaling and Positioning Request** 이 성공하면 수신자는 비어 있는 "{}" 결과 개체가 있는 JSON-RPC 응답 개체로

응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -8: 비디오 크기를 조정/배치하지 못했습니다.

이 경우 수신기는 필요에 따라 오류 데이터 개체의 속성으로 *minimumScaleFactor* 를 포함할 수 있습니다.

minimumScaleFactor – **Receiver** 에서 지원하는 최소 *scaleFactor* 를 제공합니다. 위에 정의된 *scaleFactor* 의미 체계를 참조하세요.

예를 들어 **Broadcaster Application** 이 표시된 비디오를 전체 화면의 25%로 조정하고 디스플레이의 왼쪽 가장자리를 화면 너비의 10%에서 가로로, 디스플레이의 상단 가장자리를 화면 높이의 15%에서 수직으로 배치하려는 경우 **Receiver** 에 이 JSON-RPC API 를 발급합니다.

<표 9.6.2-3> Example of Video Scaling and Positioning Request 1
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.scale-position",
       "params": {
              "scaleFactor": 25.0,
              "xPos": 10.0,
              "yPos": 15.0
       },
       "id": 589
}
```

스케일링/포지셔닝이 성공하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.2-4> Example of Video Scaling and Positioning Request 1-1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 589
}
```

크기 조정/포지셔닝이 성공하지 못하면 **Receiver** 는 "오류" 개체를 포함하는 JSON 개체로 응답합니다.

<표 9.6.2-5> Example of Video Scaling and Positioning Request 1-2
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "error": {
       "code": -8,
       "message": "Video scaling/position failed",
       "data":{
              "mimimumScaleFactor": 30.0
              }
       },
       "id": 589
}
```

9.6.3 Set RMP URL API

Broadcaster Application 은 브로드캐스트로 전달되는 콘텐츠 대신 대체 소스(예를 들어, broadband 또는 로컬로 캐시된 콘텐츠)에서 유래된 비디오 콘텐츠를 재생하기 위해 **Receiver Media Player (RMP)**를 사용하도록 선택할 수 있습니다. 이러한 방식으로 **Broadcaster Application** 은 **Receiver** 에서 제공하는 최적화된 미디어 플레이어를 활용할 수 있습니다. **Broadcaster Application**은 RMP URL 설정 API를 사용하여 **Receiver**에게 RMP를 사용하여 **Broadcaster Application**에서 제공한 URL에서 시작된 콘텐츠를 재생하도록 요청할 수 있습니다. **Broadcaster Application** 제공 URL의 콘텐츠를 재생하라는 알림을 받으면 RMP는 브로드캐스트 콘텐츠(또는 요청 시 렌더링 중인 콘텐츠) 렌더링을 중지하고 새 URL에서 참조하는 콘텐츠 렌더링을 시작합니다.

지정된 MPD를 통해 제공되는 콘텐츠는 현재 선택된 서비스의 일부로 간주됩니다. 이 URL을 변경하면 일시적인 효과가 발생하며, 서비스를 다시 선택하는 경우(예: **Acquire Service API**를 통한 **Broadcaster Application**에 의해) RMP는 *endOperation* 함수를 처리해야 합니다. 브로드캐스터 애플리케이션은 *stopRMP* 작업을 사용하여 현재 RMP 재생을 중지할 수 있지만 서비스를 선택하면 이 작업이 재정의됩니다.

Broadcaster Application 은 RMP가 요청된 작업을 현재 프레젠테이션 타임라인의 미래 시간으로 동기화하도록 요청할 수 있습니다. 이 명시적 동기화는 현재 재생 중인 콘텐츠가 재배포에서 찾을 수 있는 것과 같은 대체 전송에 의해 배달되는 시나리오에서 필요합니다. RMP는 한 번에 두 개 이상의 보류 중인 요청을 큐에 대기할 수 없을 것으로 예상됩니다. 동기화는 다음 단락에 지정된 대로 rmpSyncTime 매개 변수를 사용하여 표시됩니다.

Broadcaster Application에서 RMP URL API 설정을 호출하고 *startRMP* 조작이 지정된 경우:

- 현재 요청에 *rmpSyncTime* 값이 지정되지 않은 경우 **Receiver**는 보류 중인 RMP URL 설정 요청을 취소하고 현재 요청에 제공된 MPD의 재생을 즉시 시작해야 합니다.
- RMP가 현재 서비스 수준 신호에 지정된 콘텐츠를 재생 중이고 현재 요청에 *rmpSyncTime* 값이 지정된 경우 **Receiver**는 보류 중인 RMP URL 설정 요청을 취소하고 *rmpSyncTime*으로 지정된 프레젠테이션 시간에 도달하면 현재 요청에 지정된 MPD의 재생을 시작해야 합니다.
- RMP가 현재 이전 RMP URL 설정 요청에 지정된 MPD를 재생 중이고 *rmpSyncTime*에 현재 요청에 지정된 값이 -1.0인 경우 수신자는 보류 중인 RMP URL 설정 요청을 취소하고 RMP에서 현재 재생 중인 프레젠테이션의 끝에 도달하면 현재 요청에 제공된 MPD의 재생을 시작해야 합니다.
- 그렇지 않으면 수신자는 현재 요청을 무시하고 현재 콘텐츠를 계속 재생해야 합니다.

Broadcaster Application에서 RMP URL API 설정을 호출하고 *stopRMP*

조작이 지정되면 다음을 수행합니다.

- RMP가 현재 service-level 시그널링에 지정된 콘텐츠 또는 이전 RMP URL 설정 요청에 지정된 MPD를 재생하고 있고 현재 요청에 *rmpSyncTime* 값이 지정되지 않은 경우 **Receiver**는 보류 중인 RMP URL 설정 요청을 취소하고 RMP의 표시를 즉시 중지해야 합니다.
- RMP가 현재 service-level 시그널링에 지정된 콘텐츠 또는 이전 RMP URL 설정 요청에 지정된 MPD를 재생하고 있고 현재 요청에 *rmpSyncTime* 값이 지정된 경우 **Receiver**는 보류 중인 RMP URL 설정 요청을 취소하고 *rmpSyncTime*으로 표시된 프레젠테이션 시간에 도달할 때까지 현재 콘텐츠의 재생을 계속해야 합니다. 이때 RMP 발표를 중단할 것으로 예상됩니다. 그리고
- RMP 재생이 현재 중지된 경우 **Receiver**는 현재 요청을 무시해야 합니다.

Broadcaster Application에서 RMP URL API 설정을 호출하고 *resumeService* 조작이 지정되면 다음을 수행합니다.

- RMP가 현재 이전 RMP URL 설정 요청에 지정된 MPD를 재생 중이거나 이전 RMP URL 설정 요청의 *stopRmp* 작업에 의해 중지되고 현재 요청에 *rmpSyncTime*이 지정되지 않은 경우 수신자는 보류 중인 RMP URL 설정 요청을 취소하고 *endOperation* 함수를 처리해야 합니다.
- RMP가 현재 이전 RMP URL 설정 요청에 지정된 MPD를 재생 중이고 현재 요청에 *rmpSyncTime*이 지정된 경우 RMP는 보류 중인 RMP URL 설정 요청을 즉시 취소하고 *rmpSyncTime*으로 표시된 프레젠테이션 시간에 도달할 때까지 현재 재생 중인 MPD의 재생을 계속해야 합니다. 이때 *endOperation* 함수를 처리할 것으로 예상됩니다.
- 그렇지 않으면 수신자는 현재 요청을 무시하고 현재 콘텐츠를 계속 재생해야 합니다.

Receiver가 요청된 대로 작업을 수행할 수 없다고 판단하는 경우 오류 코드를 반환해야 하며 요청된 작업을 수행할 것으로 예상되지 않습니다.

RMP가 RMP URL 설정 요청에 제공된 MPD의 재생을 시작할 때 **Broadcaster Application**은 **Signaling Data Change Notification API**(section 9.2.11)를 통해 알림을 받을 수 있습니다. 어떤 경우든 MPD 변경 또는 업데이트로 인해 프레젠테이션 타임라인이 불연속성이 발생할 때마다 RMP는 보류 중인 RMP URL 설정 요청을 취소해야 합니다.

Broadcaster Application은 MPD의 URL을 제공하여 RMP에서 재생할 콘텐츠를 지정합니다. MPD는 A/331[3]에 따라 구성되어야 합니다.

URL은 RMP가 재생을 시작해야 하는 미디어 프리젠테이션 타임라인의 진입점(예: MPD의 시작으로부터의 오프셋, 명명된 기간의 시작으로부터의 오프셋, UTC 시간 또는 "라이브 에지")을 식별하는 MPD 앵커를 포함할 수 있다. MPD 앵커는 MPEG DASH[29]에 정의된 대로여야 합니다. 이를 통해 북마크를 포함한 많은 사용 사례에 유연성을 제공합니다. 지정된 MPD 앵커에 의해 표시된 재생 위치를 RMP에서 사용할 수 없는 경우 RMP는 지정된 URL에서 MPD를 재생하지 않을

것으로 예상되며 오류 코드가 반환될 것으로 예상됩니다.

Set RMP URL Request의 Semantics는 표 9.6.3-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.setRMPURL-request.json에 정의된 대로입니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.6.3-1> Set RMP URL Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.setRMPURL”
operation	0..1	enum	"startRmp", "stopRmp", "pauseRmp", "resumeRmp", "fastForwardRmp", "rewindRmp", "skipForwardRmp", "skipBackwardsRmp", "resumeService"
rmpurl	0..1	string(uri)	"operation" = "startRmp"인 경우 RMP에서 재생할 MPD의 URI를 제공
rmpSyncTime	0..1	number	지정된 operation이 발생해야 하는 시간 오프셋
endOperation	0..1	enum	"stopRmp", "pauseRmp", "resumeService”

operation – 이 선택적 문자열은 RMP에서 수행할 작업을 정의해야 합니다. 작업이 없는 경우 수신자는 보류 중인 RMP URL API 설정 요청을 취소해야 합니다. 열거된 값의 의미는 다음과 같이 정의되어야 합니다.

"*startRmp*"는 아래 속성 설명에 설명된 대로 RMP가 *rmpurl* 속성에서 제공하는 URI 재생을 시작해야 함을 나타냅니다.

"*stopRmp*"는 RMP가 재생을 중지함을 나타냅니다. 이 작업의 경우 *rmpurl* 속성이 필요하지 않으며 있는 경우 무시해야 합니다.

"*pauseRmp*"는 RMP가 재생을 일시 중단하고, 프레임을 고정하고, 콘텐츠의 현재 위치를 표시해야 함을 나타냅니다.

"*resumeRmp*"는 RMP가 이전 *pauseRmp* 작업에서 계속 재생되어야 함을 나타냅니다.

"*fastForwardRmp*"는 RMP가 재생 속도를 높여야 함을 나타냅니다. 이 작업의 초기 속도 및 순차 호출은 **Receiver**에 따라 다릅니다.

"*rewindRmp*"는 RMP가 역방향으로 재생되어야 함을 나타냅니다. 이 작업의 초기 속도 및 순차 호출은 **Receiver**에 따라 다릅니다.

"*skipForwardRmp*"는 RMP가 앞으로 건너뛰고 재생을 다시 시작해야 함을 나타냅니다. 건너뛴 시간은 **Receiver**에 따라 다릅니다.

"*skipBackwardsRmp*"는 RMP가 뒤로 건너뛰고 재생을 다시 시작해야 함을 나타냅니다. 건너뛴 시간은 **Receiver**에 따라 다릅니다.

"*resumeService*"는 RMP가 현재 서비스의 정상적인 재생을 재개해야 함을 나타냅니다. 이 작업의 경우 "*rmpurl*" 속성이 필요하지 않으며

있는 경우 무시됩니다.

rmpurl - 작업 값이 *startRmp* 로 설정된 경우 이 문자열을 지정하고 broadband 망을 통해 MPD 를 참조하는지 또는 **Application Context Cache** 에서 MPD 를 참조하는지 여부에 관계없이 RMP 에서 재생할 MPD 를 참조하는 정규화된 URI 를 제공해야 합니다. URI 는 **Receiver** 가 액세스할 수 있어야 합니다. URI 에 액세스할 수 없는 경우 적절한 오류 코드(아래 참조)가 반환됩니다. 응용 프로그램 컨텍스트 캐시의 MPD 의 경우 섹션 9.1.3 에 설명된 대로 **Query Receiver Web Server URI API** 를 사용하여 제공된 기본 URI 를 사용하여 전체 URI 를 생성할 수 있습니다

rmpSyncTime - 이 선택적 부동 소수점 숫자는 RMP 에서 현재 재생 중인 프레젠테이션의 미디어 프레젠테이션 타임라인의 미래 시간(즉, *currentTime* 보다 이후)을 나타냅니다(Query RMP Media Time API, section 9.11.1 에 지정된 대로 *startDate* 에 지정된 미디어 프레젠테이션 시간을 기준으로 초)작업으로 지정된 작업을 수행해야 합니다. *rmpSyncTime* 을 지정하지 않으면 작업으로 표시된 작업이 즉시 재생을 시작해야 합니다. *rmpSyncTime* 의 값이 -1.0 인 경우 RMP 에서 현재 재생 중인 프레젠테이션의 끝에 도달하면 작업에 의해 표시된 작업을 수행해야 합니다. (프레젠테이션 종료는 추가 프레젠테이션 설명이 표시되지 않을 때 발생하는 것으로 간주됩니다.)

endOperation - 이 선택적 문자열은 MPD 프레젠테이션의 끝에 도달할 때 RMP 가 수행해야 하는 작업을 나타냅니다. 값은 매개변수 연산에 대해 정의된 대로여야 합니다. 이 매개 변수가 없는 경우 기본값은 *resumeService* 입니다.

Set RMP URL Response Semantics 는 표 9.6.3-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.setRMPURL-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.3-2> Set RMP URL Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 빈 객체(empty object) 가 반환 요청에 실패하면 오류 구조(error structure) 가 반환
error	oneOf X		Section 8.3.3 참조

result - RMP URL 이 성공하면 수신자는 비어 있는 "{}" 결과 객체가 있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -5: Broadband 연결을 사용할 수 없습니다.
- -11: 표시된 MPD 에 액세스할 수 없습니다.

- -12: 콘텐츠를 재생할 수 없습니다.
- -13: 요청된 MPD 앵커에 연결할 수 없습니다.
- -15: 제공된 *rmpurl* 속성 값이 잘못된 URL 임을 나타냅니다.
- -19: *rmpSyncTime* 에 지정된 동기화를 수행할 수 없습니다.
- -21: 현재 소스에서 RMP 재생 변경은 지원되지 않습니다.

예를 들어, **Broadcaster Application** 이 RMP 에 https://stream.wxyz.com/33/program.mpd 에 위치한 DASH 서버의 broadband 소스에서 콘텐츠를 재생하도록 요청하는 경우 다음과 같이 **Receiver** 에 명령을 실행할 수 있습니다.

<표 9.6.3-3> Example of Set RMP URL Request 1
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.setRMPURL",
       "params": {"operation": "startRmp",
              "rmpurl": "https://stream.wxyz.com/33/program.mpd"},
       "id": 104
}
```

성공하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.3-4> Example of Set RMP URL Response 1-1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 104
}
```

Receiver 의 RMP 가 콘텐츠를 재생할 수 없는 경우 수신자는 다음과 같이 응답할 수 있습니다.

<표 9.6.3-5> Example of Set RMP URL Response 1-2
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "error": { "code": -12, "message": "The content cannot be
played"},
       "id": 104
}
```

예를 더 나아가면, 사용자는 이제 비디오가 없는 전체 화면 주문형 비디오 선택 화면을 표시하고자 하는 **Broadcaster Application** 과 상호 작용합니다. **Broadcaster Application** 은 수신기에 다음과 같은 요청을 합니다.

<표 9.6.3-6> Example of Set RMP URL Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.setRMPURL",
        "params": {"operation": "stopRmp"},
        "id": 113
}
```

성공하면 RMP 는 비디오 표시를 중지하고 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.3-7> Example of Set RMP URL Response 2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 113
}
```

마지막으로 사용자는 주문형 비디오 시나리오를 종료하고 이제 방송 서비스 시청으로 돌아가려고 합니다. **Broadcaster Application** 은 다음과 같은 요청을 합니다.

<표 9.6.3-8> Example of Set RMP URL Request 3
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.setRMPURL",
        "params": {"operation": "resumeService"},
        "id": 106
}
```

성공하면 **Receiver** 는 정상적인 재생 작업을 재개하고 다음과 같이 응답합니다.

<표 9.6.3-9> Example of Set RMP URL Response 3
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 106
}
```

네 번째 예로, **Query RMP Media Time API** 에서 제공하는 *currentTime* 이 *startDate* 보다 1740 초 지난 경우 이전 RMP URL 설정 요청에 제공된 MPD 가 재생되지 않고 **Set RMP URL API** 은 RMP 가 현재 재생 중인 프레젠테이션의 *currentTime* 이 지정된 MPD 에 대한 진입점이 있는 1800 초에 도달할 때 RMP 가 https://stream.wxyz.com/33/program.mpd 에 있는 MPD 의 재생을 시작하기를 원합니다 즉, 시작부터 5 분이 지나면 애플리케이션은 다음과 같이 수신기에 명령을 내릴 수 있습니다.

<표 9.6.3-10> Example of Set RMP URL Request 4
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.setRMPURL",
       "params": {"operation": "startRmp",
              "rmpurl": "https://stream.wxyz.com/33/program.mpd#t=5:00",
              "rmpSyncTime": 1800.00},
       "id": 107
}
```

보류 중인 MPD 재생을 성공적으로 예약하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.3-11> Example of Set RMP URL Response 4-1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 107
}
```

Receiver 의 RMP 가 콘텐츠의 예약된 재생 요청을 수락할 수 없는 경우(예: 지정된 동기화 시간이 지났거나 RMP 가 준비하기에는 너무 빠르기 때문에) 수신자는 다음과 같이 응답할 수 있습니다.

<표 9.6.3-12> Example of Set RMP URL Response 4-2
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "error": { "code": -19,
       "message": "The synchronization specified by rmpSyncTime
cannot be achieved"},
       "id": 107
}
```

다섯 번째 예로, RMP 가 이전 **Set RMP URL API** 요청에 따라 MPD 를 재생하고 있고 *currentTime* 이 **Query RMP Media Time API** 에서 제공하는 *startDate* 보다 10 초 지난 경우, **Broadcaster Application** 이 현재 재생 중인 프레젠테이션의 *currentTime* 이 60 초에 도달할 때 RMP 가 서비스 수준 신호에 지정된 콘텐츠의 재생을 재개하기를 원하는 경우, 다음과 같이 **Receiver** 에게 요청을 발행할 수 있습니다.

<표 9.6.3-13> Example of Set RMP URL Request 5
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.setRMPURL",
       "params": {"operation": "resumeService",
                     "rmpSyncTime": 60.00},
       "id": 108
```

```
}
```

Service-level signaling 에 지정된 콘텐츠의 요청된 재생을 성공적으로 예약하면 수신기는 다음과 같이 응답합니다.

<표 9.6.3-14> Example of Set RMP URL Response 5
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 108
}
```

여섯 번째 예로, RMP 가 service-level signaling (또는 이전 **Set RMP URL API** 에 지정된 MPD)에 지정된 콘텐츠를 재생하는 경우 *currentTime* 이 **Query RMP Media Time API** 에서 제공하는 startDate 보다 10 초 지난 경우 브로드캐스터 애플리케이션은 다음 요청을 사용하여 currentTime 이 30 초에 도달하면 수신자에게 현재 재생을 중지하도록 요청할 수 있습니다.

<표 9.6.3-15> Example of Set RMP URL Request 6
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.setRMPURL",
       "params": {"operation": "stopRmp",
                  "rmpSyncTime": 30.00},
       "id": 109
}
```

요청된 현재 재생 중지를 성공적으로 예약하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.3-16> Example of Set RMP URL Response 6
[출처: A344]

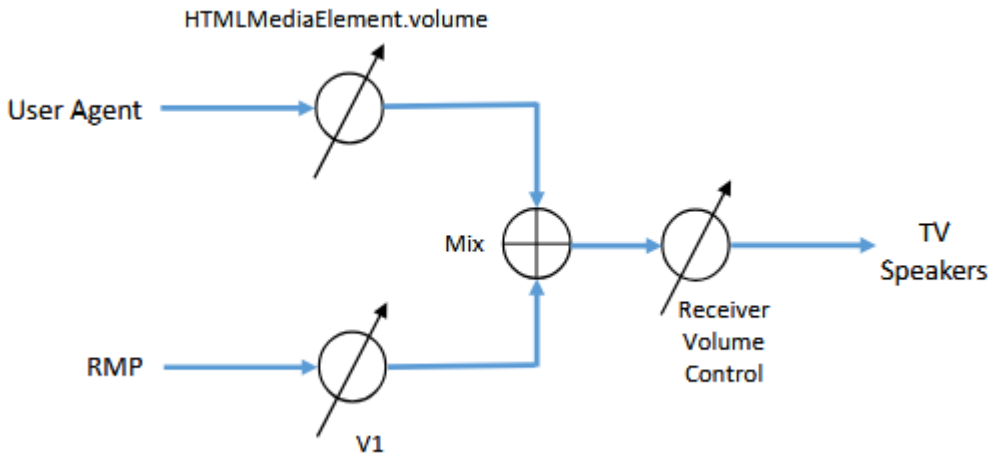
```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 109
}
```

9.6.4 Audio Volume API

기본적으로 **Receiver Media Player** 의 오디오 출력과 **User Agent** 의 오디오 출력이 혼합됩니다. **Broadcaster Application** 은 *.volume* 속성을 사용하여 HTML5 미디어 요소의 볼륨을 설정하고 가져올 수 있습니다. **Receiver Media Player** 의 오디오 볼륨을 설정하고 가져올 수 있습니다. 예를 들어 **Broadcaster Application** 은 사용자가 HTML5 미디어 요소로 렌더링된 광대역 콘텐츠를 시청하도록 선택할 때 브로드캐스트 서비스의 오디오 출력을 음소거할 수 있습니다.

이러한 경우에는 **Audio Volume API**를 사용할 수 있습니다.

그림 9.6.4-1은 사용자에게 프레젠테이션하기 위해 **User Agent**의 오디오 출력이 **Receiver Media Player**의 오디오 출력과 혼합되는 예제 **Receiver**에서 오디오 처리를 보여줍니다. **Broadcaster Application**은 HTMLMediaElement의 *.volume* 속성을 사용하여 출력 볼륨을 제어합니다. 유사하게, 여기에 정의된 **Audio Volume API**는 그림에서 "V1"로 표시된 수신기 미디어 플레이어의 볼륨을 설정하는 데 사용할 수 있습니다. API는 RMP 볼륨("V1")만 변경합니다. 전체 **Receiver Volume Control**은 **Broadcaster Application**에서 관리할 수 없으며 이 오디오 볼륨의 제어는 본 문서의 범위에 속하지 않습니다



(그림 9.6.4-1) RMP audio volume.
[출처: A344]

요청에 볼륨 요소가 제공되는 경우 **Receiver**는 RMP 볼륨을 설정하기 위한 요청을 처리합니다. **Receiver**의 응답은 두 경우 모두 현재 볼륨을 제공합니다.

Audio Volume Request의 Semantics는 표 9.6.4-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.audioVolume-request.json에 정의된 대로여야 합니다.

<표 9.6.4-1> Audio Volume Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.audioVolume"
audioVolume	0..1	number(0.0…1.0)	존재하는 경우 요청된 오디오 볼륨은 0.0(음소거) ~ 1.0(최대 볼륨) 범위

audioVolume – 0에서 1 사이의 이 선택적 부동 소수점 숫자(있는 경우)는 **Receiver Media Player**에서 설정할 오디오 볼륨 값에 해당해야 합니다. number의 값은 0.0(최소 또는 음소거)에서 1.0(전체 볼륨) 사이여야

합니다. 인코딩은 HTML5 미디어 요소의 *.volume* 속성과 동일합니다. 요청에 볼륨이 지정되지 않은 경우 볼륨은 이 요청에 의해 변경되지 않습니다. 이는 현재 볼륨 설정을 결정하는 데 사용할 수 있습니다.

Audio Volume Response의 Semantics는 표 9.6.4-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.audioVolume-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.4-2> Audio Volume Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
audioVolume	1	number(0.0…1.0)	0.0(음소거)~1.0(최대 볼륨) 범위의 현재 오디오 볼륨
error	oneOf X		Section 8.3.3 참조

audioVolume – 0 에서 1 사이의 이 부동 소수점 숫자는 **Receiver Media Player**의 현재 오디오 볼륨을 나타내며, 여기서 0 은 최소 볼륨 또는 음소거를 나타내고 1.0 은 전체 볼륨을 나타냅니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None – 이 API 와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application**이 **Receiver Media Player**를 원하는 경우 오디오 볼륨을 절반 볼륨(50%)으로 설정합니다.

<표 9.6.4-3> Example of Audio Volume Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.audioVolume",
       "params": {"audioVolume": 0.5},
       "id": 239
}
```

요청이 성공적으로 처리되면 **Receiver**는 다음과 같이 응답할 수 있습니다.

<표 9.6.4-4> Example of Audio Volume Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {"audioVolume": 0.5},
       "id": 239
}
```

9.6.5 Dialog Enhancement API

기본적으로 **Receiver**의 **Receiver Media Player**의 오디오 디코더는 사용자가

기본 설정에서 구성한 대로 Dialog Enhancement 처리를 적용합니다. Broadcaster Application은 처리량을 가져오거나 설정하거나 Broadcaster Application 에 의해 이전에 수행된 설정을 해제하기 위한 인터페이스를 제공하고자 할 수 있습니다. 이 경우 Dialog Enhancement API를 사용할 수 있습니다.

사용자가 Receiver 환경 설정에서 원하는 Dialog Enhancement 처리 수준을 변경하면 이러한 변경 사항이 즉시 적용되어야 합니다. 따라서 Broadcaster Application 시작 기본 설정 설정이 대체될 것으로 예상되며 Receiver는 기본 설정 설정의 게인 값을 다시 사용해야 합니다. 동시에 Broadcaster Application 은 Dialog Enhancement Preference Change Notification API를 통해 이 변경 사항에 대한 정보를 받을 수 있으며 결과적으로 이 이벤트에 따라 조치를 취할 수 있습니다.

Receiver는 요청을 처리하고 가능한 경우 처리량을 변경합니다. 사용자 기본 설정의 설정은 변경되지 않을 것으로 예상됩니다.

Dialog Enhancement Request의 Semantics는 표 9.6.5-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.dialogEnhancement-request.json 에 정의된 대로여야 합니다.

<표 9.6.5-1> Dialog Enhancement Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.dialogEnhancement”
dialogEnhancementGain	0 or oneOfX	integer	존재하는 경우 요청된 Dialog Enhancement 게인(dB)
dialogEnhancementReset	0 or oneOfX	boolean	존재하고 "true"인 경우 방송 애플리케이션에서 설정한 대화 강화 값을 재설정

dialogEnhancementGain - 이 선택적 정수는 오디오 디코더에 적용할 Dialog Enhancement 처리의 게인 값(dB)을 나타내는 경우입니다. 값이 0 이면 Dialog Enhancement 처리가 비활성화됩니다. 현재 디코딩된 오디오 스트림의 메타데이터로 표시된 대로 Dialog Enhancement 처리의 허용 값 범위를 벗어난 *dialogEnhancementGain* 값은 Receiver에 의해 제한되어야 합니다.

이 값이나 *dialogEnhancementReset*이 요청에 지정되지 않은 경우 이 요청에 의해 처리 양이 변경되지 않습니다. 이는 오디오 디코더가 적용하는 현재 처리 양과 제한을 결정하는 데 사용할 수 있으며, 이러한 값은 응답에 반환되기 때문입니다.

dialogEnhancementReset - "*true*"로 설정하면 이 선택적 Boolean 값은 Broadcaster Application 제어 대화 향상 처리를 해제합니다. Receiver는 기본 설정 설정에서 사용자가 구성한 Dialog Enhancement 처리로 되돌아갈 것으로 예상됩니다. 없거나 "false"로 설정된 경우 Dialog

Enhancement 처리의 상태 및 양은 변경되지 않습니다.

이 값이나 *dialogEnhancementGain* 모두 요청에 지정되지 않은 경우 이 요청에 의해 처리 양이 변경되지 않습니다. 이는 오디오 디코더가 적용하는 현재 처리 양과 제한을 결정하는 데 사용할 수 있으며, 이러한 값은 응답에 반환되기 때문입니다.

참고: 사용자의 기본 설정은 section 9.1.11 에 지정된 **Query Dialog Enhancement Preferences API** 를 사용하여 얻을 수 있습니다.

Dialog Enhancement Response 의 Semantics 는 표 9.6.5-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.dialogEnhancement-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.5-2> Dialog Enhancement Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
dialogEnhancementGain	1	integer	현재 대화 향상 게인 값(dB)
dialogEnhancementLimit	1		현재 오디오 스트림 신호 대화 상자 향상 제한을 제공
max	1	integer	허용되는 대화 강화 처리 게인 값의 상한(dB)
min	1	integer	허용되는 대화 강화 처리 게인 값의 하한(dB)
error	oneOf X		Section 8.3.3 참조

dialogEnhancementGain - 이 필수 정수는 오디오 디코더에 적용되도록 구성된 **Dialog Enhancement** 게인 값(dB)을 나타내야 합니다. 원하는 게인 값이 허용된 게인 값의 범위를 벗어나면 현재 적용된 게인 값이 가장 가까운 지정된 한계로 잘립니다.

dialogEnhancementLimit - 이 필수 개체는 현재 디코딩된 오디오 스트림에서 신호를 받은 **Dialog Enhancement** 처리의 신호 제한에 대한 정보를 제공합니다.

max - 이 필수 정수는 허용된 대화 향상 처리 게인 값의 현재 신호된 상한(dB)을 제공해야 합니다.

min - 이 필수 정수는 허용된 대화 향상 처리 게인 값의 현재 신호 하한(dB)을 제공해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -22: 대화 상자 향상 실패

예를 들어 **Broadcaster Application** 이 **Receiver Media Player** 의 오디오 디코더에 있는 **Dialog Enhancer** 가 8dB 의 게인 값에서 처리를 적용하도록 하려면 다음 요청을 제출합니다.

<표 9.6.5-3> Example of Dialog Enhancement Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.dialogEnhancement",
        "params": {"dialogEnhancementGain": 8},
        "id": 192
}
```

요청이 성공적으로 처리되면 **Receiver**는 다음과 같이 응답합니다.

<표 9.6.5-4> Example of Dialog Enhancement Response 1-1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "dialogEnhancementGain": 8,
                "dialogEnhancementLimit": {
                        "max": 12,
                        "min": 0
                }
        },
        "id": 192
}
```

요청이 성공하지 못하면 **Receiver**는 오류 코드 -22로 응답할 수 있습니다.

<표 9.6.5-5> Example of Dialog Enhancement Response 1-2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "error": {"code": -22, "message": "Dialog Enhancement failed"},
        "id": 192
}
```

9.6.6 Launch Broadcaster Application API

이 API를 사용하면 현재 실행 중인 **Broadcaster Application**이 HELD에서 새 **Broadcaster Application**을 시작할 수 있습니다.

호출 **Broadcaster Application**의 *@appId* 문자열은 진입점 호출의 매개변수로 포함됩니다. section 8.2를 참조하십시오.

Launch Broadcaster Application Request Semantics는 표 9.6.6-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.launchApp-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.6-1> Launch Broadcaster Application Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.launchApp"
appId	1	string(uri)	실행할 Broadcaster

			Application 의 appId
parameters	0..1	string	Broadcaster Application 호출에서 Broadcaster Application 실행까지의 불투명 텍스트 문자열

appId - 이 필수 문자열은 A/331 [3] 섹션 7.1.8 에 정의된 *appId* 입니다.

parameters - 호출 **Broadcaster Application** 에서 시작된 **Broadcaster Application** 으로 전달되는 선택적 텍스트 문자열입니다. 구문과 의미 체계는 두 **Broadcaster Application** 간에 비공개입니다.

appId 및 매개 변수 문자열은 section 8.2 에 정의된 쿼리 문자열을 통해 전달됩니다.

Launch Broadcaster Application Response Semantics 는 표 9.6.6-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.launchApp-response.json 에 정의된 대로여야 합니다. 성공하면 이 API 에서 반환되지 않습니다. 요청이 실패하면 아래에 정의된 오류 응답이 반환됩니다.

<표 9.6.6-2> Launch Broadcaster Application Response Semantics [출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
error	oneOf X		Section 8.3.3 참조

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -23: HELD 에서 *appId* 를 찾을 수 없습니다.
- -25: *appId* 가 HELD 에서 발견되었지만 사용할 수 없거나 브로드캐스트 전용이며 아직 획득되지 않았습니다.
- -26: HELD 에서 발견된 *appId*, 광대역 전용, 네트워크 연결 없음
- -27: **Receiver** 가 필요한 기능을 지원하지 않습니다.

다음 예에서 **Broadcaster Application** 은 다른 **Broadcaster Application** 을 시작합니다.

<표 9.6.6-3> Example of Launch Broadcaster Application Request [출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.launchApp",
       "params": {
              "appId": "pbs.org/kids/1"
       },
       "id": 42
}
```

성공하면 수신기는 새 **Broadcaster Application** 이 시작되었으므로 응답하지 않을 것으로 예상됩니다. 실패 시 수신기는 가능한 경우 오류로 응답해야 합니다.

9.6.7 Media Track Selection API for DASH

Broadcaster Application 은 **Receiver** 의 **Receiver Media Player** 에게

서비스에서 사용할 수 있는 특정 비디오 스트림(예: 대체 카메라 각도)을 선택하도록 요청할 수 있습니다. 또는 **Receiver Media Player**에 사용자의 기본 설정에 따라 선택한 오디오 프레젠테이션이 아닌 다른 오디오 프레젠테이션을 선택하도록 요청할 수 있습니다. 이러한 경우에는 **DASH Media Track Selection API**를 사용할 수 있습니다.

Receiver는 요청을 처리하고 가능한 경우 선택 항목을 변경합니다.

DASH Media Track Selection Request의 Semantics는 표 9.6.7-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.track.selection-request.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.7-1> DASH Media Track Selection Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.track.selection"
selectionId	1	string(uri)	선택할 트랙 ID

selectionId - 이 필수 정수는 현재 **Period**의 *AdaptationSet* 또는 사전 선택과 관련된 복잡한 오디오 프레젠테이션의 경우 현재 **Period**의 DASH *Period.Preselection@id* 값에 있는 *@id* 특성 값에 해당해야 합니다. 하나의 트랙 또는 오디오 프레젠테이션을 명확하게 선택하려면 Period 내의 모든 id 값이 고유해야 합니다.

DASH Media Track Selection Response의 Semantics는 표 9.73에 정의되어 있으며 구문은 스키마 파일 org.atsc.track.selection-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.7-2> DASH Media Track Selection Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result - 미디어 트랙 선택 요청이 성공하면 수신자는 빈 *"{}" result* 객체가 있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- -10: 지정된 트랙을 선택할 수 없습니다.

예를 들어 **Broadcaster Application**이 **Receiver Media Player**가 id 값이 5506인 비디오 AdaptationSet을 찾아 선택하기를 원하는 경우 다음

WebSocket 메시지를 보낼 수 있습니다.

<표 9.6.7-3> Example of DASH Media Track Selection Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.track.selection",
        "params": {"selectionId": 5506},
        "id": 329
}
```

요청된 AdaptationSet 이 성공적으로 선택되면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.6.7-4> Example of DASH Media Track Selection Response 1-1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 329
}
```

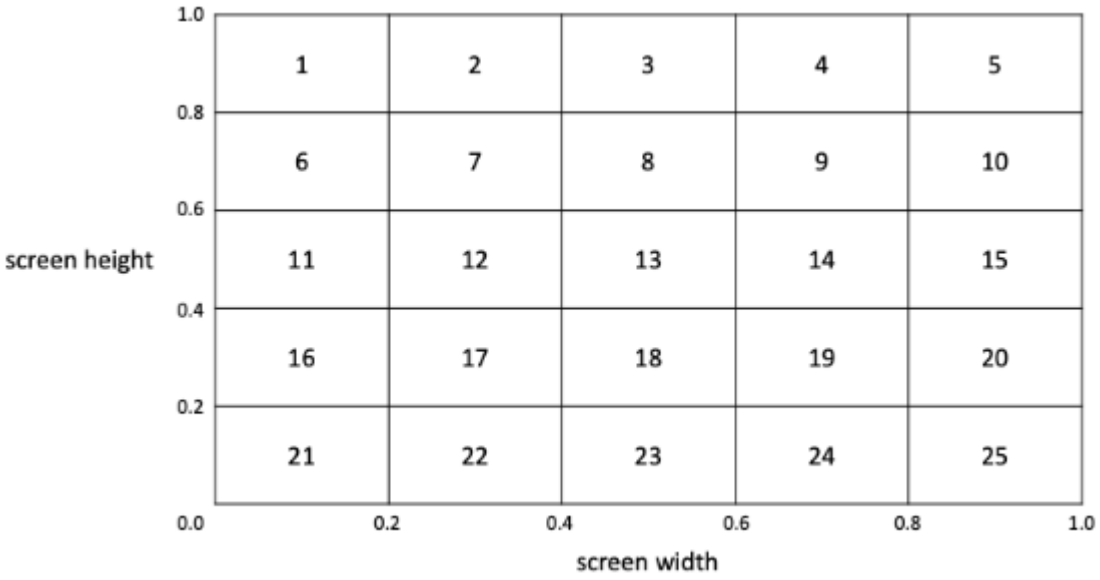
요청된 트랙을 선택할 수 없는 경우 **Receiver**는 오류 코드 10 으로 응답해야 합니다.

<표 9.6.7-5> Example of DASH Media Track Selection Response 1-2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "error": {"code": -10, "message": "Track cannot be selected"},
        "id": 329
}
```

9.6.8 Graphics Display Regions API

Broadcaster Application 은 표시하려는 그래픽 레이아웃이 차지할 화면 영역을 제공해야 할 수 있습니다. 수신자는 이러한 정보를 사용하고 잠재적인 디스플레이 충돌을 완화하는 등의 목적으로 렌더링되는 다른 디스플레이 구성 요소를 조정하도록 요청할 수 있습니다. **Graphics Display Regions API** 와 함께 사용하기 위한 그래픽 표시 영역 레이아웃 및 번호 매기기는 그림 9.6.8-1 에 나와 있습니다



(그림 9.6.8-1) Graphics Display Regions Layout and Numbers.
[출처: A344]

Graphics Display Regions Request 의 Semantics 는 표 9.6.8-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.graphicsDisplayRegions-request.json 에 정의된 대로여야 합니다.

<표 9.6.8-1> Graphics Display Regions Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.graphicsDisplayRegions"
occupiedRegions	1	integer (0 … 33554431)	정수로 설정된 적절한 비트로 표시된 대로 **Broadcaster Application** 그래픽 레이아웃이 차지할 각 그래픽 표시 영역 번호를 제공합니다. (참고: 33,554,431 = 2^25 - 1)

occupiedRegions - 이 필수 정수에서 '1'로 설정된 비트는 위의 그림 9-1 에 표시된 5x5 디스플레이 그리드의 디스플레이 영역이 **Broadcaster Application** 이 표시를 위해 준비한 그래픽 레이아웃에 의해 부분적으로도 점유됨을 나타냅니다. '1'로 설정된 정수의 최하위 비트는 **Broadcaster Application** 이 그림 9-1 에 표시된 디스플레이 영역 번호 1 을 차지하는 그래픽 레이아웃을 제공할 계획임을 나타내고, '1'로 설정된 다음 유효 비트는 디스플레이 영역 번호 2 를 나타내는 식입니다. 5x5 디스플레이 그리드에는 25 개의 디스플레이 영역이 있으므로 정수의 처음 25 비트만 '1'로 설정될 수 있습니다. 따라서 최상위 7 비트는 '0'으로 설정되어야 합니다.

Graphics Display Regions Response 의 의미 체계 (semantics)는 표 9.6.8-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.graphicsDisplayRegions-response.json 에 정의된 대로여야

합니다.

<표 9.6.8-2> Graphics Display Regions Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
error	oneOf X		

result – 그래픽 표시 영역 요청이 성공하면 **Receiver**는 빈 "{}" 결과 개체가 있는 JSON-RPC 응답 개체로 응답해야 합니다.

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None – 이 API와 관련된 오류가 없습니다.

9.6.9 Media Asset Selection API for MMT

Broadcaster Application은 **Receiver**의 **Receiver Media Player**에게 서비스에서 사용할 수 있는 특정 비디오 asset, 예를 들어 대체 카메라 각도를 선택하거나, MMT 스트림에서 사용자의 선호도에 따라 선택했을 오디오 asset이 아닌 다른 오디오 asset을 선택하도록 요청할 수 있다. 이러한 경우에는 **MMT Media Asset Selection API**를 사용할 수 있습니다.

MMT Media Asset Selection Request의 이미 체계(semantics)은 표 9.6.9-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.asset.selection-request.json에 정의되어 있습니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.9-1> MMT Media Asset Selection Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.asset.selection"
assetId	1	string	선택할 asset ID

assetId – 이 필수 문자열은 MP 테이블[30]의 asset ID 값에 해당해야 합니다. Asset ID에는 UUID(Universally Unique Identifier) 또는 URI(Uniform Resource Identifier) 체계가 있을 수 있으며 형식은 길이가 32비트인 바이트 배열입니다.

MMT Media Asset Selection Response의 Semantics는 표 9.6.9-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.asset.selection-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.6.9-2> MMT Media Asset Selection Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description

Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청이 성공하면 반환되고, 그렇지 않으면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result – **MMT Media Asset Selection Request** 가 성공하면 **Receiver** 는 비어 있는 "{}" 결과 객체가 있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -33: 지정된 자산을 선택할 수 없습니다.

9.7 Mark Unused API

현재 실행 중인 **Broadcaster Application** 에서 **Mark Unused API** 를 사용하여 캐시 내의 요소가 사용되지 않음을 **Application Context Cache** 에 나타낼 수 있습니다. 그런 다음 **Receiver** 는 사용되지 않는 요소에서 사용하는 리소스를 회수하기 위해 적절한 작업을 수행할 수 있습니다.

Mark Unused Request 의 Semantics 는 표 9.7-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.cache.markUnused-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.7-1> Mark Unused Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.cache.markUnused”
elementUri	1	string (uri-reference)	사용되지 않은 것으로 표시할 요소의 **Application Context Cache** 내 상대 경로

elementUri – 이 필수 URI 는 사용되지 않는 것으로 표시될 **Broadcaster Application** 의 **Application Context Cache** 내의 요소 경로여야 합니다.

Mark Unused Response 의 Semantics 는 표 9.7-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.markUnused-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.7-2> Mark Unused Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result – 사용하지 않은 것으로 표시하는 요청이 성공하면 **Receiver** 는 비어 있는 "{}" 결과 개체가 있는 JSON-RPC 응답 개체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 콘텐츠를 찾을 수 없음

예를 들어, **Broadcaster Application**은 특정 대체 광고가 사용된 후 더 이상 필요하지 않음을 표시하기를 원할 수 있습니다. 대체 광고를 구성하는 파일, 즉 기간 XML 조각과 연결된 오디오 및 비디오 세그먼트는 이 예에서 'ads/16'이라는 레이블이 지정된 특정 디렉터리 계층 구조로 전송됩니다. **Broadcaster Application**은 기간 XML 조각 파일을 사용되지 않는 것으로 표시하고 기간 XML 조각 파일에서 참조하는 모든 세그먼트도 사용되지 않은 것으로 표시합니다. **Broadcaster Application**은 광고의 모든 리소스를 사용하지 않는 것으로 표시할 책임이 있으며, **Receiver**는 참조된 리소스를 검색하기 위해 기간 XML 조각 파일을 처리할 책임이 없습니다. 또는 **Broadcaster Application**은 디렉터리에 대체 광고 기간 XML 단편 파일 및 관련 세그먼트만 포함된 경우 전체 디렉터리 "ads/16"을 사용되지 않는 것으로 표시할 수 있습니다.

"ads/16" 디렉터리를 사용하지 않는 것으로 표시하기 위한 RPC 요청은 다음과 같이 형식이 지정됩니다.

<표 9.7-3> Example of Mark Unused Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.cache.markUnused",
        "params": {
                "elementUri": "ads/16"
        },
        "id": 42
}
```

Receiver는 성공 시 다음과 같이 응답할 수 있습니다.

<표 9.7-4> Example of Mark Unused Response 1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 42
}
```

마찬가지로 단일 파일을 사용하지 않는 것으로 표시하기 위해 **Broadcaster Application**은 다음 요청을 할 수 있습니다.

<표 9.7-5> Example of Mark Unused Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.cache.markUnused",
        "params": {
                "elementUri": "news/storyImages/photo12.png"
        },
        "id": 42
}
```

Receiver는 성공 시 다음과 같이 응답할 수 있습니다.

<표 9.7-6> Example of Mark Unused Response 2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 42
}
```

표준 HTTP 실패 코드는 URI 형성에 문제가 있고 참조된 파일 또는 디렉터리를 사용되지 않은 것으로 표시할 수 없음을 나타내는 데 사용되어야 합니다. 요소가 사용하지 않는 것으로 성공적으로 표시되면 나중에 해당 요소에 액세스하려는 시도는 일부 수신자가 요소를 사용할 수 없게 만들지 않아 요청에 긍정적으로 응답하는 반면 다른 **Receiver**는 즉시 오류 상태로 응답할 수 있습니다.

9.8 Content Recovery APIs

9.8.1 Query Content Recovery State API

Broadcaster Application은 A/336[5]에 명시된 대로 워터마킹 및/또는 지문을 통한 콘텐츠 복구를 사용하여 관리되고 있는지 여부를 알고 싶어할 수 있습니다. 이를 통해 **Broadcaster Application**은 브로드캐스트 시그널링이 존재할 때 제공될 수 있는 것과 다른 기능을 콘텐츠 복구 시나리오에서 제공할 수 있으며, 콘텐츠 복구 시나리오에서 애플리케이션은 A/336[5]의 Annex A에서 논의된 바와 같이 업스트림 장치(예: Set-Top Box(STB))에 의해 도입될 수 있는 수정의 존재를 실행 중에 지속적으로 식별하고 그에 따라 동작을 변경할 수 있습니다.

Query Content Recovery State Request의 Semantics는 표 9.8.1-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.contentRecoveryState-request.json에 정의된 대로여야 합니다.

<표 9.8.1-1> Query Content Recovery State Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.query.contentRecoveryState"

Query Content Recovery State Response의 Semantics는 표 9.8.1-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.contentRecoveryState-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.8.1-2> Query Content Recovery State Response Semantics

[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
audioWatermark	0..1	integer (0..2)	오디오 워터마크 감지 상태
videoWatermark	0..1	integer (0..2)	비디오 워터마크 감지 상태
audioFingerprint	0..1	integer (0..2)	오디오 핑거프린트 감지 상태
videoFingerprint	0..1	integer (0..2)	비디로 핑거프린트 감지 상태
error	oneOf X		Section 8.3.3 참조

audioWatermark - 이 정수 값은 오디오 워터마크 검색의 다음 상태 중 하나를 나타냅니다.

0: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 지정된 대로 복구된 오디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하지 않는 경우;

1: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 지정된 대로 복구된 오디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하고 수신기가 현재 A/336 에 정의된 대로 VP1 오디오 워터마크 세그먼트를 감지하지 않는 경우;

2: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 지정된 대로 복구된 오디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하고 수신기가 현재 A/1 에 정의된 대로 VP336 오디오 워터마크 세그먼트를 감지하고 있는 경우.

videoWatermark - 이 정수 값은 비디오 워터마크 감지의 다음 상태 중 하나를 나타냅니다.

0: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 지정된 대로 복구된 비디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하지 않는 경우;

1: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 명시된 대로 복구된 비디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 VP1 비디오 워터마크 세그먼트 또는 A/336 에 정의된 다른 비디오 워터마크 메시지를 감지하지 않는 경우;

2: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 명시된 대로 복구된 비디오 워터마크 감지 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 VP1 비디오 워터마크 세그먼트 또는 A/336 에 정의된 기타 비디오 워터마크 메시지를 감지하고 있는 경우.

audioFingerprint - 이 정수 값은 오디오 지문 인식의 다음 상태 중 하나를 나타냅니다.

0: **Receiver** 가 애플리케이션 관리를 위해 A/336 에 지정된 대로

복구된 오디오 지문 인식 및 애플리케이션 신호를 모두 사용하지 않는 경우;

1: **Receiver**가 A/336에 명시된 대로 복구된 오디오 지문 인식 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 오디오 지문을 인식하지 못하는 경우;

2: **Receiver**가 A/336에 지정된 대로 복구된 오디오 지문 인식 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 오디오 지문을 인식하고 있는 경우.

videoFingerprint - 이 정수 값은 비디오 지문 인식의 다음 상태 중 하나를 나타냅니다.

0: **Receiver**가 A/336에 명시된 대로 복구된 비디오 지문 인식 및 애플리케이션 신호를 모두 사용하지 않는 경우;

1: **Receiver**가 A/336에 명시된 대로 복구된 비디오 지문 인식 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 비디오 지문을 인식하지 못하는 경우;

2: **Receiver**가 A/336에 명시된 대로 복구된 비디오 지문 인식 및 애플리케이션 신호를 모두 사용하고 있고 수신기가 현재 비디오 지문을 인식하고 있는 경우.

결과에 키/값 쌍이 없으면 키/값 쌍의 값이 0임을 나타냅니다(즉, 연결된 기능은 수신자에서 지원되지 않음).

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None - 이 API와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application**은 다음과 같은 쿼리를 만듭니다.

<표 9.8.1-3> Example of Query Content Recovery State Request
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.contentRecoveryState",
      "id": 122
}
```

Receiver가 A/336에 지정된 대로 오디오 및 비디오 워터마크 모두에서 복구된 애플리케이션 신호를 사용하여 애플리케이션 관리를 지원하고 현재 둘 다 감지되는 경우 **Receiver**는 다음과 같이 응답해야 합니다.

<표 9.8.1-4> Example of Query Content Recovery State Response
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {
            "audioWatermark": 2,
            "videoWatermark": 2
      },
      "id": 122
}
```

9.8.2 Query Display Override API

Broadcaster Application 은 Receiver 가 워터마킹(A/336 [5]에 정의됨)을 통해 얻은 비디오 및 오디오 프레젠테이션의 수정을 수행해서는 안 된다는 것을 나타내는 "display override" 신호를 수신하고 있는지, Receiver 가 Broadcaster Application 의 프레젠테이션 리소스에 대한 액세스를 억제하여 해당 신호를 적극적으로 시행하고 있는지 여부를 알 수 있습니다("리소스 차단").

이 정보는 예를 들어 Broadcaster Application 에 의해 사용될 수 있습니다.

- Receiver 및 네트워크 리소스의 효율적인 활용을 보장합니다(예: 해당 리소스를 사용자에게 제공할 수 없는 경우 광대역 서버에서 리소스를 요청하지 않도록 선택할 수 있음).
- 사용자 경험의 정확한 표현을 유지합니다(예: 광고 조회가능성 표준에서 요구할 수 있는 동적으로 삽입된 광고의 조회 가능성을 정확하게 보고하기 위해) 또는
- 수신기가 리소스 차단을 수행하지 않는 경우, 오디오나 비디오 수정 작업을 중단하는 등의 display override state 의 요구사항을 준수해야 합니다.

Query Display Override Response 의 Semantics 는 표 9.8.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.displayOverride-request.json 에 정의된 대로여야 합니다.

<표 9.8.2-1> Query Display Override Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.query.displayOverride”

Query Display Override Response 의 Semantics 는 표 9.8.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.displayOverride-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.8.2-2> Query Display Override Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
resourceBlocking	0..1	boolean	"true"는 수신기가 브로드캐스터 애플리케이션의 출력을 차단하고 있음
displayOverride	0..1	boolean	"true"는 표시 재정의 조건이 적용됨
error	oneOf X		Section 8.3.3 참조

resourceBlocking – 이 선택적 Boolean 값은 Receiver 가 A/336[5]에

정의된 활성 디스플레이 재정의 상태에 따라 **Broadcast Application** 이 비디오 및 오디오를 표시하지 못하도록 차단하는지 여부를 나타냅니다.

displayOverride – A/336[5]의 section 5.1.9 에 지정된 비디오 워터마크 **Display Override Message** 또는 A/336[5]의 section 5.2.4 및 5.4.2 에 지정된 오디오 워터마크 디스플레이 재정의 표시에 따라 디스플레이 재정의 조건이 현재 적용되는 경우 이 선택적 Boolean 값은 true 입니다. 그렇지 않으면 값은 false 가 됩니다.

결과에 키/값 쌍 중 하나 또는 둘 다 없는 경우 부재한 키/값 쌍의 값이 false 임을 나타냅니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None – 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcast Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.8.2-3> Example of Query Display Override Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.displayOverride",
       "id": 62
}
```

디스플레이 재정의 조건이 현재 A/336[5]의 section 5.2.4 에 지정된 대로 오디오 워터마크를 통해 표시되고 **Receiver** 가 **Broadcast Application** 이 비디오 및 오디오를 표시하지 못하도록 차단하는 경우 **Receiver** 는 다음과 같이 응답합니다.

<표 9.8.2-4> Example of Query Display Override Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {
              "resourceBlocking": true,
              "displayOverride": true
       },
       "id": 62
}
```

9.8.3 Query Recovered Component Info API

워터마킹 또는 지문 인식을 통한 콘텐츠 복구가 사용되는 경우, **Broadcaster Application** 이 **Receiver** 에 의해 수신되고 있는 서비스의 비디오 또는 오디오 구성 요소를 결정할 수 있는 것이 유용하다(예를 들어, 업스트림 장치에서 사용자가 선택한 결과). 이를 통해 **Broadcaster Application** 은 수신된 구성 요소의 특성에 맞게 화면 배치 또는 오버레이된 그래픽 또는 오디오의 언어를 수정할 수 있습니다.

워터마킹 또는 지문을 통한 콘텐츠 복구 중에 **Receiver** 는 A/336[5]의 섹션 5.4.2 에 지정된 복구 파일에서 구성 요소 설명자를 수신합니다. 이 API 는

Broadcaster Application 이 워터마크 또는 지문을 사용하여 식별된 구성 요소에 대해 복구된 설명자에 액세스할 수 있는 수단을 제공합니다.

Query Recovered Component Info Request 의 Semantics 는 표 9.8.3-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.recoveredComponentInfo-request.json 에 정의된 대로여야 합니다.

<표 9.8.3-1> Query Recovered Component Info Request Semantics [출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.query.recoveredComponentInfo"

Query Recovered Component Info Response 의 Semantics 는 표 9.8.3-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.recoveredComponentInfo-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.8.3-2> Query Recovered Component Info Response Semantics [출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
component	1	array	복구된 미디어 구성요소 배열
items	1..N		
mediaType	1	examples	"audio", "video", "both"
componentID	0..1	string	
descriptor	0..1	string	
error	oneOf X		Section 8.3.3 참조

mediaType, *componentID* 및 *descriptor* 는 A/336[5]의 표 5.29 에 지정된 복구 파일의 componentDescription 요소에 있는 동일한 이름의 필드에 제공된 대로 **Receiver** 가 수신한 미디어 구성 요소와 관련된 데이터 값입니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 은 다음과 같은 쿼리를 만듭니다.

<표 9.8.3-3> Example of Query Recovered Component Info Request [출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.recoveredComponentInfo",
      "id": 39
}
```

복구 파일의 *componentDescription* 요소에 *componentID* 값이 "1"이고 설명자 값이 "구성 요소 설명자 문자열 1"인 오디오 구성 요소와 *componentID* 값이 "2"이고 설명자 값이 "구성 요소 설명자 문자열 2"인 비디오 구성 요소가 **Receiver**가 수신한 구성 요소와 연결된 경우 **Receiver**는 다음과 같이 응답할 수 있습니다.

<표 9.8.3-4> Example of Query Recovered Component Info Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"component": [
                {
                        "mediaType": "audio",
                        "componentID": "1",
                        "descriptor": "component descriptor string 1"
                },
                {
                        "mediaType": "video",
                        "componentID": "2",
                        "descriptor": "component descriptor string 2"
                }
        ]},
        "id": 39
}
```

9.8.4 Content Recovery State Change Notification API

Section 9.8.1의 **Query Content Recovery State API**에 정의된 콘텐츠 복구 상태가 적어도 하나의 속성 값이 변경되는 다른 상태로 변경되고 **Broadcaster Application**이 section 9.2.1에 지정된 API를 통해 이러한 알림을 수신하도록 구독한 경우 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 **Content Recovery State Change Notification**를 발행할 것으로 예상됩니다

Content Recovery State Change Notification의 Semantics은 표 9.8.4-1에 정의되어 있으며, 구문은 스키마 파일 org.atsc.notify-contentRecoveryStateChange.json 에 정의된 대로여야 한다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.8.4-1> Query Content Recovery State Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"contentRecoveryStateChange”
audioWatermark	0..1	integer (0..2)	오디오 워터마크 감지 상태
videoWatermark	0..1	integer	비디오 워터마크 감지 상태

		(0..2)	
audioFingerprint	0..1	integer (0..2)	오디오 핑거프린트 감지 상태
videoFingerprint	0..1	integer (0..2)	비디로 핑거프린트 감지 상태
error	oneOf X		Section 8.3.3 참조

audioWatermark, videoWatermark, audioFingerprint 및 *videoFingerprint* 는 section 9.8.1 의 **Query Content Recovery State API** 에 정의되어 있습니다.

매개변수에 키/값 쌍이 없으면 키/값 쌍의 값이 0 임을 나타냅니다.

예를 들어, 사용자가 워터마크가 없는 서비스에서 오디오 및 비디오 워터마크가 모두 표시된 새 서비스로 변경하고 오디오 및 비디오 워터마크가 모두 감지되어 **Receiver** 에서 콘텐츠 복구에 사용되는 경우, **Receiver** 는 아래와 같이 **Broadcaster Application** 에 콘텐츠 복구 상태 변경을 알립니다.

<표 9.8.4-2> Example of Query Content Recovery State Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "contentRecoveryStateChange",
                "audioWatermark": 2,
                "videoWatermark": 2
        }
}
```

9.8.5 Display Override Change Notification API

Display Override Change Notification API 는, **Receiver** 의 디스플레이 재정의 상태 또는 리소스 차단 상태가 Section 9.9.2 의 **Query Display Override API** 에서 정의된 상태 중 하나에서 다른 상태로 변경될 때, 그리고 **Broadcaster Application** 이 Section 9.3.1 에 명시된 API 를 통해 해당 알림을 수신하도록 구독한 경우, 현재 실행 중인 **Broadcaster Application** 에 의해 발행되도록 예상됩니다.

<표 9.8.5-1> Display Override Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"displayOverrideChange”
resourceBlocking	0..1	boolean	"true"는 Receiver 가 Broadcaster Application 의 출력을 차단하고 있음
displayOverride	0..1	boolean	"true"는 표시 재정의 조건이 적용됨

resourceBlocking 및 *displayOverride* 는 section 9.8.2 의 **Query Display Override API** 에 정의되어 있습니다.

params 에 키/값 쌍이 없으면 키/값 쌍의 값이 false 임을 나타냅니다.

예를 들어, 디스플레이 재정의 상태가 비활성에서 활성으로 변경되고 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 이 비디오 및 오디오를 표시하지 못하도록 차단하는 경우 **Receiver** 는 아래와 같이 **Broadcaster Application** 에 디스플레이 재정의 변경 사항을 알립니다.

<표 9.8.5-2> Example of Display Override Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "displayOverrideChange",
                "resourceBlocking": true,
                "displayOverride": true
        }
}
```

9.8.6 Recovered Component Info Change Notification API

Recovered Component Info Change Notification API 는 **Receiver** 가 수신하는 서비스의 비디오 또는 오디오 컴포넌트가 변경되고(예: 업스트림 디바이스에서 사용자가 선택한 결과로) **Broadcaster Application** 이 section 9.2.1 에 지정된 API 를 통해 그러한 알림을 수신하도록 가입한 경우 수신자에 의해 현재 실행 중인 **Broadcaster Application** 에 발행될 것으로 예상됩니다

Recovered Component Info Change Notification 의 Semantics 는 표 9.8.6-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-recoveredComponentInfoChange.json 에 정의된 대로여야 합니다. 매개변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.8.6-1> Recovered Component Info Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"displayOverrideChange”
resourceBlocking	0..1	boolean	"true"는 Receiver 가 Broadcaster Application 의 출력을 차단하고 있음
displayOverride	0..1	boolean	"true"는 표시 재정의 조건이 적용됨

mediaType, *componentID* 및 *descriptor* 는 section 9.2.3 의 **Query Recovered Component Info API** 에 정의되어 있습니다.

예를 들어, **Receiver** 의 업스트림 장치에 있는 사용자가 *componentID* 값 "1" 및 *descriptor* 값 "구성 요소 설명 문자열 3"으로 설명되는 스페인어에서 영어 오디오 트랙으로 변경된 경우, **Receiver** 는 아래와 같이 변경된 복구된 구성 요소를 브로드캐스터 애플리케이션에 알립니다.

<표 9.8.6-2> Example of Recovered Component Info Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "recoveredComponentInfoChange",
                "mediaType": "audio",
                "componentID": "1",
                "descriptor": "component description string 3"
        }
}
```

9.9 Filter Codes APIs

Receiver 는 **Filter Codes** 를 사용하여 저장된 **Filter Codes** 를 EFDT 의 NRT 데이터 파일과 연결된 **Filter Codes** 와 비교하여 NRT 데이터 파일을 선택적으로 다운로드할 수 있습니다. **Filter Code** 처리에 대한 전체 설명은 섹션 6.5.3 을 참조하십시오.

9.9.1 Set Filter Code Instances API

Broadcaster Application 에서 **Set Filter Code Instances API** 를 발급하여 **Receiver** 에게 지정된 **Filter Code Instances** 를 저장하도록 알릴 수 있습니다.

Set Filter Code Instances Request Semantics 는 표 9.9.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.setFilterCodes-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.9.1-1> Set Filter Code Instances Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	
method	1	string	"org.atsc.setFilterCodes”
filters	1	array	Filter Code Instance 정의 목록
items	1..N		각 항목 개체는 Filter Code Instance 를 설명
filterCode	1	integer	Filter Code Instance 의 Filter Code 값
expires	0..1	string (date-time)	Filter Code Instance 가 만료되는 날짜와 시간

filters - **Filter Code Instance** 정의의 필수 배열입니다.

filterCode - 브로드캐스터가 결정한 개인화 범주와 연결된 부호 없는 정수입니다. 이 특성은 **Filter Code Instance** 의 값 부분을 설정합니다. *AppContextID* 내의 **Filter Code Instance** 간에 값의 고유성 범위를 유지하는 것은 브로드캐스터의 책임입니다.

expires - 이 문자열은 **Filter Code Instance** 의 만료를 나타내기 위해 JSON 스키마 사양[19]에 정의된 날짜-시간 JSON 데이터 유형으로 표시됩니다. **Filter Code Instance** 는 만료 후 사용할 수 없습니다.

expires 값을 생략하면 만료가 표시되지 않으며 **Broadcaster Application** 이 종료될 때 **Filter Code Instance** 가 만료됩니다. "xs:dateTime"에는 JSON "날짜-시간" 형식을 준수하지 않는 법적 인스턴스가 있으므로 "xs:dateTime" 형식의 XML 문자열을 사용하여 이 필드를 채우는 것은 피해야 합니다.

Set Filter Code Instances Response Semantics 는 표 9.9.1-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.setFilterCodes-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.9.1-2> Set Filter Code Instances Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result - **Set Filter Code Instances Request** 가 성공하면 **Receiver** 는 비어 있는 "{}" 결과 객체가 있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

다음 예제에서 **Broadcaster Application** 은 **Receiver** 가 사용할 두 개의 **Filter Code Instances** 를 설정합니다.

<표 9.9.1-3> Example of Set Filter Code Instances Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.setFilterCodes",
       "params": {
              "filters": [
              {"filterCode": 101, "expires": "2016-07-17T09:30:47Z"},
              {"filterCode": 102}]
       },
       "id": 57
}
```

성공하면 **Receiver** 는 다음과 같이 응답합니다.

<표 9.9.1-4> Example of Set Filter Code Instances Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 57
}
```

9.9.2 Clear Filter Code Instances API

Broadcaster Application 에서 **Clear Filter Code Instances API** 를 발행하여 정의된 모든 **Filter Code Instance** 또는 지정된 **Filter Code Instance** 집합을 지우도록 **Receiver** 에게 알릴 수 있습니다.

Clear Filter Code Instances Request Semantics 는 표 9.9.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.clearFilterCodes-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.9.2-1> Clear Filter Code Instances Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	
method	1	string	"org.atsc.clearFilterCodes"
filters	1	array	삭제될 Filter Code Instance 목록
items	1..N	integer	삭제될 Filter Code Instance 를 식별하는 Filter Code

filters - **Receiver** 가 처리에서 일치하는 **Filter Code Instance** 를 제거하는 데 사용해야 하는 부호 없는 정수 **Filter Code** 의 선택적 목록입니다. *filters* 속성이 없거나 비어 있는 경우 모든 활성 **Filter Code Instance** 가 지워집니다. 이러한 방식으로 지워진 **Filter Code Instance** 는 정의된 만료 시간에 관계없이 더 이상 NRT 파일 처리에 포함되지 않습니다(section 6.5.3 참조).

Clear Filter Code Instances Response Semantics 는 표 9.9.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.clearFilterCodes-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.9.2-2> Clear Filter Code Instances Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result - **Clear Filter Code Instances request** 이 성공하면 **Receiver** 는 빈 "{}" 결과 개체가 있는 JSON-RPC 응답 개체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -28: 제공된 **Filter Code** 중 하나 이상을 알 수 없습니다. 요청된 다른 모든 **Filter Code** 는 지워질 것으로 예상됩니다.

다음 예에서 **Broadcaster Application** 은 **Receiver** 가 지울 수 있는 두 개의

Filter Code 를 제공합니다.

<표 9.9.2-3> Example of Clear Filter Code Instances Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.clearFilterCodes",
        "params": {
                "filters": [101, 102]
        },
        "id": 62
}
```

성공하면 Receiver 는 다음과 같이 응답합니다.

<표 9.9.2-4> Example of Clear Filter Code Instances Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 62
}
```

9.10 Keys APIs

이 section 의 API 를 사용하면 Broadcaster Application 이 Receiver 의 권한을 받아 지정된 특정 원격 제어 키에 액세스할 수 있습니다. Query Device Info API(section 9.11)은 Broadcaster Application 이 요청할 수 있는 탐색 키, 선택 키 및 뒤로 키를 포함하여 입력 키 세트를 정의해야 합니다. *BAAppear* 키로 지정된 키는 Receiver 가 사용자가 상호 작용하기를 원하는 Broadcaster Application 에 알리는 방법을 제공하는 Query Device Info API 에 의해 정의됩니다. *BAAppear* 키의 예상 용도는 아래에 설명되어 있습니다. Query Device Info API 는 Broadcaster Application 이 요청할 수 있는 추가 키를 제공할 수도 있습니다. Broadcaster Application 사용 가능한 입력 키를 관리할 수 있도록 하는 API 는 이 section 에 정의되어 있습니다.

Broadcaster Application 은 사용자의 입력으로 처리할 것으로 예상되는 모든 입력 키를 수신기에 요청해야 합니다. Broadcaster Application 은 현재 사용자 상호 작용에 적용할 수 있는 입력 키만 요청해야 하며 현재 대화형 작업에 필요하지 않은 입력 키를 요청하지 않아야 합니다. 이를 통해 Receiver 가 키의 용도를 변경할 수 있습니다. *BAAppear* 키를 초과하는 키가 요청되지 않은 경우 Query Device Info API 응답 *deviceInput* 개체(section 9.11 참조)에 의해 나열되고 Receiver 가 사용하지 않는 모든 키로 인해 *BAAppear* 키가 Broadcaster Application 에 발급될 수 있습니다.

예를 들어 시작 시 Broadcaster Application 은 Query Device Info API 를 사용하여 Receiver 에서 사용할 수 있는 키를 확인하고 의도한 사용자 환경을 제공하는 데 필요한 키에 매핑해야 합니다. 이 예에서, Broadcaster

Application 은 날씨, 뉴스 또는 광대역 기반 VOD 콘텐츠와 같이 사용자가 관심을 가질 수 있는 다양한 기타 기능을 제공한다. Broadcaster Application 이 존재하고 사용 가능함을 사용자에게 알리려면 Broadcaster Application 이 Query Device Info API Response 에서 제공하는 *BAAppear* 텍스트 및 이미지를 표시해야 합니다.

사용자가 이미 Broadcaster Application 과 상호 작용하지 않는 한 Broadcaster Application 은 *BAAppear* 입력 키를 요청합니다. 이를 시작 화면이라고 합니다. 이 이후에는 Broadcaster Application 디자인에 따라 실행 화면이 보이지 않을 수 있습니다. 그러나 보이지 않는 경우 상호 작용을 사용할 수 있는 경우 Broadcaster Application 에 전달된 모든 키가 가시성을 다시 가져와야 합니다. Broadcaster Application 디자인은 *BAAppear* 입력 키를 수신할 때 대화형 기능을 직접 실행하도록 선택할 수 있습니다. Broadcaster Application 은 사용자에게 상호 작용이 가능하다는 것을 알리기 위해 언제든지 시작 페이지를 표시하도록 선택할 수 있습니다.

BAAppear 입력 키를 수신하면 Broadcaster Application 은 요청 키 API 를 통해 수신기에 적절한 입력 키를 요청하고 사용자가 Broadcaster Application 과 상호 작용할 수 있는 대화 상자를 표시하기 시작합니다. 사용자가 리모컨의 키를 클릭하면 Broadcaster Application 이 적절하게 응답하여 새 페이지와 대화 상자를 표시해야 합니다. Broadcaster Application 은 Broadcaster Application 의 Query Device Info API 호출에 대한 응답으로 Receiver 가 열거한 대로 키를 요청하거나 포기하도록 선택할 수 있습니다. Receiver 는 성공적인 요청 키 API 호출을 통해 Broadcaster Application 에 제공된 키의 용도를 변경하지 않아야 합니다. 시작 화면과 유사하게, 이 인터랙티브 컨텍스트의 Broadcaster Application 은 Receiver 가 키를 제공할 때 볼 수 있을 것으로 예상됩니다. 이를 통해 사용자는 시작 페이지로 이동할 수 있으며 Receiver 는 필요에 따라 포기된 키의 용도를 변경할 수 있습니다.

일부 상황에서 수신기는 BA 에서 초점을 빼앗는 Broadcaster Application 의 "앞" 또는 "위"에 대한 사용자 입력에 대한 대화 상자 또는 기타 프롬프트를 표시할 수 있습니다. 예를 들어 사용자가 Receiver 설정을 변경하기 위해 대화 상자를 요청한 경우 발생합니다. 그러면 Broadcaster Application 이 동일한 입력 키를 요청했더라도 사용자의 입력은 Receiver 대화 상자에 의해 처리됩니다. Broadcaster Application 은 W3C "onblur" 이벤트를 사용하여 현재 초점이 맞춰져 있는지 여부를 감지할 수 있습니다. CTA-5000-G [9]의 [UI 이벤트]를 참조하십시오.

사용자가 Broadcaster Application 과의 상호 작용을 완료하면 Broadcaster Application 은 모든 대화 상자와 표시되는 페이지를 닫고 요청된 모든 키를 포기해야 합니다. 해당 *BAAppear* 키를 상기시키기 위해 Broadcaster Application 은 *BAAppear* 텍스트 및 이미지를 표시할 수 있습니다.

Broadcaster Application 은 뒤로 키를 사용하거나 그렇지 않은 방식으로 시작 페이지에 대한 경로를 제공할 것으로 예상됩니다. 시작 페이지에서 뒤로 키를 더

누르면 **Broadcaster Application**이 보이지 않게 될 것으로 예상됩니다. 시작 페이지에서 **Receiver**는 필요한 경우 *BAAppear* 키를 제외한 키의 용도를 변경할 수 있습니다.

9.10.1 Keycode Consistency

이 section 및 section 9.11의 API에서 제공하는 키 코드는 사용자 입력을 활성화하기 위한 물리적, 가상 및 대체 방법에서 일관성이 있을 것으로 예상됩니다. 예를 들어 실제 키보드의 키와 디스플레이의 가상 키보드는 동일한 키에 대해 동일한 코드를 반환해야 합니다.

9.10.2 Keycode Consistency

Broadcaster Application은 일반적으로 **Receiver**에서 사용하고 처리하는 선택적 키 누름 수신을 요청할 수 있습니다. 이 컨텍스트에서 "*key press*"는 키패드 또는 리모컨에서 키를 누르는 사용자 작업을 나타내며 W3C 이벤트 메커니즘과 혼동해서는 안 됩니다. 예를 들어, 리모컨의 숫자 키 누름은 일반적으로 기본 **Receiver**에서 특정 채널에 직접 튜닝하는 데 사용됩니다. 그러나, **Broadcaster Application**은 특정 동작을 수행하거나 시청자로부터 입력을 요청하기 위해 사용자로부터 숫자 데이터를 수락하기 위해 데이터 입력 UI를 제시하기를 원할 수 있다. 이 경우 **Broadcaster Application**은 **Receiver**에 숫자 키 누름을 일시적으로 자신에게 다시 라우팅하도록 요청할 수 있습니다. **Receiver** 제조업체에 따라 수신기는 이 요청을 거부할 수 있으며, 이 경우 **Broadcaster Application**은 TV 화면에 소프트 키보드를 표시하거나 다른 유형의 장치 입력을 사용하도록 선택할 수 있습니다.

Request Keys Request Semantics는 표 9.10.2-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.request.keys-request.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.10.2-1> Request Keys Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	
method	1	string	"org.atsc.request.keys"
keys	1	array	Broadcaster Application과 연결되도록 요청된 키 목록
items	1..N	string	요청된 키 이름

keys - 이 필수 매개변수는 문자열 배열이어야 하며, 각 매개변수는 **Broadcaster Application**이 **Receiver**가 전달하기를 원하는 특정 원격 제어 키 또는 키 유형을 나타냅니다. 사용 가능한 키 문자열은 다음과 같이 정의됩니다.

"Numeric" - 숫자 키 0-9를 나타냅니다.

"ArrowUp" - "ArrowUp" 입력 키를 나타냅니다.

"ArrowDown" - "ArrowDown" 입력 키를 나타냅니다.

"ArrowRight" - "ArrowRight" 입력 키를 나타냅니다.

"ArrowLeft" - "ArrowLeft" 입력 키를 나타냅니다.

"뒤로" - "뒤로" 입력 키를 나타냅니다.

"BAAppear" - **Broadcaster Application** 전용 입력 키를 나타냅니다. section 9.11 를 참조하십시오

<other> - W3C "UI 이벤트 KeyboardEvent 키 값", 섹션 3 [32]의 문자열을 나타냅니다.

Receiver 에게 알려지지 않은 요청된 키는 무시될 것으로 예상됩니다.

Request Keys Response Semantics는 표 9.10.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.request.keys-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.10.2-2> Request Keys Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
accepted	1		허용되는 키 목록
items	0..N	string	
error	oneOf X		Section 8.3.3 참조

accepted - **Receiver** 는 "결과" 객체를 포함한 JSON-RPC 응답 객체로 응답해야 합니다. 결과 개체에는 요청이 성공한 키를 나타내는 문자열 배열이 포함됩니다. 제공된 문자열은 위의 요청 작업의 "*keys*" 매개 변수에 허용된 문자열에 해당해야 합니다. "*accepted*" 배열에 포함되지 않은 요청된 키는 수락되지 않았다고 가정할 수 있습니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application** 이 숫자 키와 채널 위쪽 및 채널 아래쪽 화살표의 수신을 요청하는 경우 **Receiver** 에 다음 요청을 발행할 수 있습니다.

<표 9.10.2-3> Example of Request Keys Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.request.keys",
        "params": {"keys": ["Numeric", "ChannelUp", "ChannelDown"]},
        "id": 43
}
```

Receiver 가 숫자 키 누름을 허용하지만 채널 업 및 채널 다운 키 누름은 허용하지 않는 경우 수신기는 다음과 같이 응답할 수 있습니다.

<표 9.10.2-4> Example of Request Keys Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"accepted": ["Numeric"]},
        "id": 43
}
```

9.10.3 Relinquish Keys API

Broadcaster Application은 키 누름에 대한 이전 요청을 포기할 수 있습니다. 이는 **Request Keys API**(section 9.10.1)를 통해 요청한 후 키 누름 처리를 **Receiver**에게 반환하는 데 사용됩니다.

Relinquish Keys Request의 Semantics는 표 9.10.3-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.relinquish.keys-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.10.3-1> Relinquish Keys Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	
method	1	string	"org.atsc.relinquish.keys"
keys	oneOf X	array	Broadcaster Application 에서 포기할 키 목록
items	0..N	string	포기할 키 이름

keys - 이 필수 매개변수는 문자열 배열로, 각 매개변수는 **Broadcaster Application**이 더 이상 처리하지 않고 **Receiver**에게 양도하는 특정 원격 제어 키 또는 키 유형을 나타냅니다. *keys* 매개 변수가 제공되지 않았거나, "*All*"과 같거나, 빈 배열인 경우, 이전에 **Broadcaster Application**에서 전달을 요청한 모든 키는 포기됩니다. 이전에 요청되지 않았거나 **Receiver**에게 알려지지 않은 지정된 키는 무시됩니다. 사용 가능한 키 문자열은 **Request Keys API**의 **keys** 속성에 대한 Semantics 정의가 명시된 Section 9.11.1 에서 정의됩니다.

Relinquish Keys Response의 Semantics는 표 9.10.3-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.relinquish.keys-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.10.3-2> Relinquish Keys Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

result - 키 포기 요청이 성공하면 수신자는 비어 있는 "{}" 결과 객체가

있는 JSON-RPC 응답 객체로 응답해야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어, **Broadcaster Application** 은 **Receiver** 에 다음 요청을 발행할 수 있습니다.

<표 9.10.3-3> Example of Relinquish Keys Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.relinquish.keys",
        "params": {
                "keys": ["ChannelUp", "ChannelDown"]
        },
        "id": 44
}
```

Receiver 는 다음과 같이 응답할 수 있습니다.

<표 9.10.3-4> Example of Relinquish Keys Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {},
        "id": 44
}
```

9.10.4 Request Keys Timeout

응용 프로그램의 오작동을 방지하기 위해 **Receiver** 는 각 **Receiver** 제조업체에서 정의한 특정 시간이 지난 후 **Broadcaster Application** 에 의해 키 요청을 포기하도록 강제할 수 있습니다. 이를 요청 키 제한시간이라고 합니다. 요청 키 시간 초과 전에 **Receiver** 는 **Broadcaster Application** 에 경고 알림을 보내 애플리케이션 시간 초과에 응답할 시간을 제공합니다. 이 시점에서 **Broadcaster Application** 은 **Request Keys API** 호출을 실행하도록 선택하거나 키 요청 시간이 초과되도록 허용할 수 있습니다. 다른 **Request Keys API** 호출이 실행되면 **Receiver** 가 수락하거나 수락하지 않을 수 있습니다.

Request Key Timeout Notification 의 Semantics 는 표 9.10.4-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-requestKeyTimeout.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.10.4-1> Request Key Timeout Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”

method		1	string	"org.atsc.notify"
msgType		1	enum	"requestKeyTimeout"
timeout		1	array	Broadcaster Application 입력에서 제거될 키 목록을 제공
	items	1..N		
		key	1	string	키의 이름
		time	1	integer	Broadcaster Application 에서 키를 더 이상 사용할 수 없는 시간(초)

timeout - 제한 시간이 발생할 때까지의 시간(초)과 함께 제한 시간에 가까운 각 키를 식별하는 필수 배열입니다.

Key - 이 매개변수는 **Receiver** 가 **Broadcaster Application** 에서 되찾고자 하는 특정 원격 제어 키 또는 키 유형을 나타내는 문자열입니다. 사용 가능한 키 문자열은 **Request Keys API** 의 **keys** 속성에 대한 Semantics 정의가 명시된 Section 9.11.1 에서 정의됩니다.

time - 지정된 키 또는 키 유형에 대한 제한 시간을 나타내는 정수(초)입니다. 시간 제한 값은 최소 3 초여야 합니다.

예를 들어, **Receiver** 는 채널 업 및 채널 다운 키 요청이 3 초 안에 시간 초과될 것으로 예상됨을 나타내기 위해 다음 알림을 보낼 수 있습니다.

<표 9.10.4-2> Example of Request Key Timeout Notification
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "method": "org.atsc.notify",
       "params": {
              "msgType": "requestKeyTimeout",
              "timeout": [
                     {"key": "ChannelUp", "time": 3},
                     {"key": "ChannelDown", "time": 3}
              ]
       }
}
```

9.11 Query Device Info API

Query Device Info API 는 **Broadcaster Application** 과 **Receiver** 간의 인터페이스를 제공하여 장치별 정보를 검색합니다. **Receiver** 와 **Broadcaster Application** 사이의 일반 도관으로, 장치에 대한 선택적 추가 키/값 쌍 정보와 함께 장치의 제조업체 및 모델을 포함한 기본 장치 정보를 제공합니다. 선택적 추가 키/값 쌍의 형식과 정의는 제조업체에 따라 다르며 여기에 지정되지 않습니다. 특정 매개변수는 방송사와 장치 제조업체 간의 비즈니스 관계의 일부로 정의될 수 있습니다.

아래 응답에 있는 두 개의 고유 ID 매개 변수(*deviceId*, *advertisingId*)는 모든 서비스 및 **Receiver** 전원 주기에 걸쳐 장기적인 지속성을 제공하기 위해 **Receiver** 가 한 번 초기화하고 사용자가 부여한 권한에 따라 **Receiver** 가 제공할 것으로 예상됩니다. 브로드캐스터, **Broadcaster Application** 당 또는 전체적으로 고유한 *deviceId* 또는 *advertisingId* 매개변수의 공개를 승인하는

기능은 사용자에게 개인 정보 보호를 강화합니다.

Query Device Info API Request *params* 개체는 선택 사항입니다. params 가 생략된 경우(또는 *deviceInfoProperties* 가 생략되거나 빈 배열인 경우) **Receiver** 는 디바이스 제조업체 및 모델 및 *deviceInput* 개체로만 응답해야 합니다. 그런 다음 **Broadcaster Application** 은 디바이스 제조업체 및 모델을 사용하여 쿼리할 추가 속성을 결정할 수 있습니다. *deviceInfoProperties* 는 원하는 속성의 배열이며 **Receiver** 는 응답에서 이러한 속성의 값을 제공합니다.

Query Device Info Request Semantics 는 표 9.11-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.deviceInfo-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.11-1> Query Device Info Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	
method	1	string	"org.atsc.query.deviceInfo"
deviceInfoProperties	1	array	반환할 장치 속성 목록
items	0..N	string	Broadcaster Application 에 관심이 있는 특정 장치 속성의 이름

deviceInfoProperties – 이 매개 변수는 문자열 배열로, 각 문자열은 **Broadcaster Application** 이 관심 있는 디바이스의 특정 측면을 나타냅니다.

Query Device Info Response Semantics 는 표 9.11-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.deviceInfo-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.11-2> Query Device Info Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
deviceMake	1	string	장치 제조사를 포함하는 문자열
deviceModel	1	string	장치 모델을 포함하는 문자열
deviceInput	1	object	키 이름과 값의 매핑
ArrowUp	1	Integer	ArrowUp 키에 대한 키 코드 값 제공
ArrowDown	1	Integer	ArrowDown 키에 대한 키 코드 값 제공
ArrowRight	1	Integer	ArrowRight 키에 대한 키 코드 값 제공
ArrowLeft	1	Integer	ArrowLeft 키에 대한 키 코드 값 제공
Select	1	Integer	Select 키에 대한 키 코드 값 제공
Back	1	Integer	Back 키에 대한 키 코드 값

			제공
BAAppear	1	object	Broadcaster Application "Launch" 키를 정의
label	1	string	키에 대해 인식 가능한 이름을 제공(예: "NextGen App")
keycode	1	Integer	"BAAppear" 키에 대한 키 코드 값을 제공
img	0..1	string	키 또는 라벨의 인라인 인코딩 이미지를 제공
<other>	0..N	integer	<other>는 값이 관련 키 코드를 제공하는 W3C "UI 이벤트 KeyboardEvent 키 값", 섹션 3 [32]의 문자열
deviceInfo	0..1	object	장치의 특정 속성을 정의하는 키/값 쌍의 모음
deviceId	0..1	string	장치와 연결된 영구적이고 전역적으로 고유한 UUID
advertisingId	0..1	string	기기의 광고와 관련된 영구적이고 전역적으로 고유한 UUID
deviceCapabilities	0..1	string	
deviceSupported-WebSocketAPIs	0..1	array	지원되는 A/344 API 메소드 세트
items	0..N		항목은 아래 설명된 대로 문자열이거나 객체일 수 있음
	anyOf	string	A/344 지원되는 메소드, 완전한 메소드 이름(이름만)으로 식별됨
	anyOf	object	객체 기술 방법 및 개정 지원
method	1	string	A/344 지원되는 메소드, 완전한 메소드 이름으로 식별됨
rev	0..1	string	지원되는 방법의 특정 개정 날짜
deviceSupportedDRMs	0..1	array	지원되는 DRM 시스템 ID URN 세트
items	0..N	string	
error	oneOf X		Section 8.3.3 참조

deviceMake - 이 필수 문자열은 **Receiver** 제조업체를 나타냅니다.

deviceModel - 이 필수 문자열은 **Receiver**의 모델을 나타냅니다.

deviceInput - 이 필수 객체는 **Receiver** 사용자 인터페이스의 사용 가능한 사용자 입력 키 이름 및 코드와 **Broadcaster Application**에 필요한 "launch key"를 정의합니다. *deviceInput* 개체에는 키가 사용자 입력 이름이고 값이 연결된 정수 코드인 입력 키/값 쌍의 컬렉션이 포함되어 있습니다. **Receiver**는 **Broadcaster Application**이 요청할 수 있는 모든 입력 키와 관련 값을 제공해야 합니다. **Broadcaster Application**은 section 9.10에 정의된 키 API를 사용하여 *deviceInput* 개체에 정의된 키의 일부 또는 전부를 요청할 수 있습니다. 제공된 키 코드는 모든 입력 방법에서 일관성이 있을 것으로 예상됩니다(section 9.10.1 참조). 입력 키 컬렉션 외에도 *deviceInput* 개체에는 아래에 설명된 대로 *BAAppear* 개체가 포함되어야 합니다.

ArrowUp - 이 키 이벤트는 사용자가 리모컨에서 위쪽 화살표 방향 키를 트리거하거나 동등한 작업을 통해 전송됩니다.

ArrowDown – 이 키 이벤트는 사용자가 리모컨에서 아래쪽 화살표 방향 키를 트리거하거나 동등한 작업을 통해 전송됩니다.

ArrowRight – 이 키 이벤트는 사용자가 리모컨에서 오른쪽 화살표 방향 키를 트리거하거나 동등한 작업을 통해 전송됩니다.

ArrowLeft – 이 키 이벤트는 사용자가 리모컨에서 왼쪽 화살표 키를 트리거하거나 동등한 작업을 통해 전송됩니다.

Select – 이 키 이벤트는 사용자가 원격 제어 키 신호 선택을 트리거할 때 전송됩니다. 선택 입력을 제공할 수 있는 일반적인 원격 제어 키의 예로는 'OK', 'Enter', 'Select' 및 'Return' 키가 있습니다. *Select* 이벤트는 사용자가 **Broadcaster Application**이 현재 포커스를 기반으로 작업을 수행하기를 원함을 나타냅니다. 예를 들어, 아이콘이 강조 표시되면 선택 키 코드는 사용자가 관련 기능에 참여하기를 원함을 나타냅니다.

Back – 이 키 이벤트는 사용자가 현재 **Broadcaster Application** 상태에 따라 한 단계 뒤로 이동하려는 의사를 알리는 원격 제어 키를 트리거할 때 전송됩니다. 예를 들어, *Back* 이벤트는 현재 디스플레이에서의 활동이 완료되었고 사용자가 이전 디스플레이로 돌아가기를 원한다는 것을 **Broadcaster Application**에 나타낼 수 있다.

BAAppear – 이 필수 객체는 **Broadcaster Application** "launch key" 레이블, 키 코드 및 선택적 이미지 인코딩을 정의합니다.

label – 이 필수 문자열에는 원격 제어 장치에서 "launch key"를 정의하는 데 도움이 되는 사용자에게 표시하려는 텍스트가 포함되어 있습니다.

keycode – 이 필수 정수는 "launch key"의 키 코드입니다.

img – 이 선택적 문자열에는 사용자가 가상일 수 있는 "launch key" (즉, 리모컨의 키가 아님)를 찾는 데 도움이 되는 이미지 인코딩이 포함되어 있습니다. 문자열은 IETF RFC 2397[28]에 정의된 데이터 URL 체계와 RFC 4648[27]에 정의된 "base64" 인코딩을 준수해야 합니다. 수신기는 CTA-5000-G, 섹션 3.6[9]에 제공된 대로 PNG, JPEG 및 GIF 를 지원해야 합니다.

<other> – 이러한 키는 W3C "UI 이벤트 *KeyboardEvent* 키 값", 섹션 3 [32]에 설명된 대로 해당 키 코드에 대한 다양한 키 문자열의 매핑을 제공합니다. 표 9.11-2 에 명시적으로 지정된 키(예: "*Select*")는 다시 포함되지 않아야 합니다. 이 매핑에 포함된 모든 입력 키는 section 9.10 에 설명된 대로 요청할 수 있습니다.

deviceInfo – 이 선택적 개체에는 키/값 쌍이 포함됩니다. 요청에 **Receiver** 가 제공할 수 있는 정보에 해당하는 하나 이상의 *deviceInfoProperties* 문자열이 포함된 경우 *deviceInfo* 가 응답에 포함됩니다.

deviceId – 이 선택적 문자열은 특정 **Broadcaster Application**에 대해 권한이 부여된 경우 urn:uuid 구문을 사용하여 RFC 4122 [26]에 정의된 대로 전역적으로 고유한 UUID 를 반환합니다. 없는 경우 **Receiver**에서

지원하지 않습니다. 존재하고 모두 0 인 경우("urn:uuid:00000000-0000-0000-0000-000000000000"), 이 식별자를 제공하는 사용자 설정의 값은 지정된 서비스에 대해 비활성화되어 사용자가 값의 제공을 명시적으로 승인하지 않았음을 나타냅니다.

advertisingId - 이 선택적 문자열은 특정 **Broadcaster Application** 에 대해 승인된 경우 urn:uuid 구문을 사용하여 RFC 4122[26]에 정의된 대로 전역적으로 고유한 UUID 를 반환합니다.

없는 경우 **Receiver** 에서 지원하지 않습니다. 존재하고 모두 0 인 경우("urn:uuid:00000000-0000-0000-0000-0000-0000000000000"), 이 식별자를 제공하는 사용자 설정의 값은 지정된 서비스에 대해 비활성화되어 사용자가 값의 제공을 명시적으로 승인하지 않았음을 나타냅니다.

deviceCapabilities - 이 선택적 문자열은 **Receiver** 의 기능을 설명하며 A/332, section 5.2.2.3.3.2 "장치 기능 구문 및 의미 체계"[4]를 준수해야 합니다. 이것은 본질적으로 *deviceInfo* 문자열의 표준화된 버전입니다.

deviceSupportedWebSocketAPI - 이 선택적 객체 배열은 정규화된 메서드 이름(예: "org.atsc.query.deviceInfo")을 사용하여 이 표준에 설명된 대로 수신기가 지원하는 A/344 **WebSocket API** 메서드를 식별합니다. 배열의 객체는 문자열, "*method*"/"*rev*" 객체 또는 문자열과 "*method*"/"*rev*" 객체의 조합이어야 합니다. 문자열인 경우 각각은 정규화된 메서드 이름이며, 이 경우 지원되는 메서드는 section 8.2 에 설명된 대로 이 표준의 개정판에 설명된 대로입니다. 배열의 "*method*"/"*rev*" 객체는 아래에 설명된 "*method*"와 선택적으로 "*rev*"로 구성됩니다.

method - 정규화된 메서드 이름을 사용하여 이 표준(또는 다른 표준, 아래 "*rev*" 참조)에 설명된 대로 수신기가 지원하는 메서드입니다.

rev - 이 선택적 필드는 section 8.2 인터페이스 바인딩, "*rev*=" 구조에 설명된 대로 형식이 지정된 A/344 의 특정 개정판에 따라 메서드가 지원됨을 식별합니다. 이 필드가 없는 경우 식별된 버전은 section 8.2 의 "*rev*=" 매개변수에 설명된 대로 있습니다.

deviceSupportedDRMs - 이 선택적 문자열 배열은 수신자가 지원하는 DRM 시스템을 식별합니다. 각 문자열은 [43]의 "보호 시스템별 식별자"에 설명된 것과 같이 "urn:uuid:<uuid>" 형식의 DRM 시스템 ID URN 이어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 의 쿼리는 다음과 같을 수 있습니다.

<표 9.11-3> Example of Query Device Info Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.deviceInfo",
        "id": 92
}
```

모델 번호가 "A300"인 "Acme" 회사에서 제조한 **Receiver**는 다음과 같이 응답할 수 있습니다.

<표 9.11-4> Example of Query Device Info Response 1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "deviceMake": "Acme",
                "deviceModel": "A300",
                "deviceInput": {
                        "ArrowUp": 38,
                        "ArrowDown": 40,
                        "ArrowRight": 39,
                        "ArrowLeft": 37,
                        "Select": 13,
                        "Back": 461,
                        "BAAppear": {
                                "label": "NextGen App",
                                "keycode": 500,
                                "img":
                                "
                                BAAD=="
                        }
                },
                "deviceSupportedWebSocketAPIs": [
                        "org.atsc.query.ratingLevel",
                        {"method": "org.atsc.query.service", "rev": "20190502"},
                        {"method": "org.atsc.scale-position"}
                ],
                "deviceSupportedDRMs": [
                        "urn:uuid:edef8ba9-79d6-4ace-a3c8-27dcd51d21ed",
                        "urn:uuid:e2719d58-a985-b3c9-781a-b030af78d30e"
                ]
        },
        "id": 92
}
```

Broadcaster Application이 이 제조사 및 모델을 인식하는 경우 요청에서 특정 장치 속성을 표시하여 이 **Receiver**에 대한 추가 정보를 검색할 수 있음을 알 수 있습니다. 모든 **Receiver**가 이 예제에 제공된 *deviceInfoProperties* 문자열을 인식할 것으로 예상되는 것은 아닙니다:

<표 9.11-5> Example of Query Device Info Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.deviceInfo",
```

```
        "params":       {"deviceInfoProperties":        ["numberOfTuners",
        "yearOfMfr"]},
        "id": 93
}
```

이 **Receiver**는 다음과 같이 응답할 수 있습니다.

<표 9.11-6> Example of Query Device Info Response 2
[출처: A344]

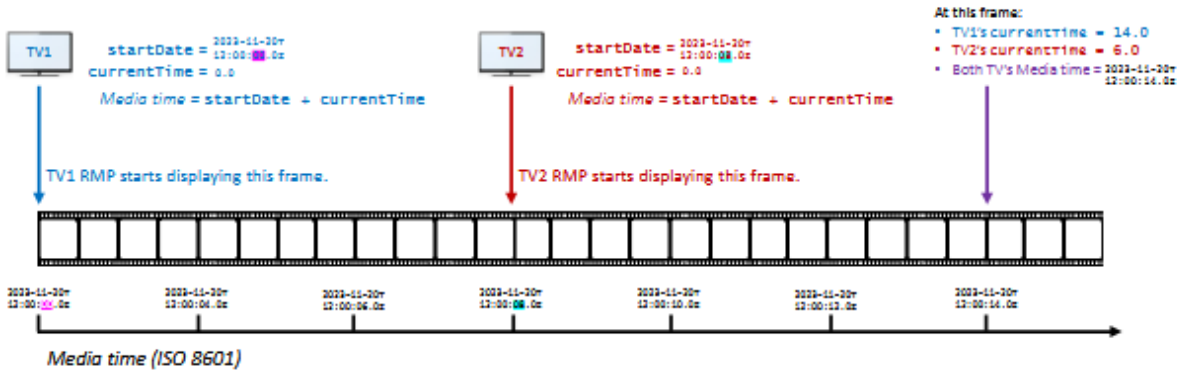
```
<-- {
        "jsonrpc": "2.0",
        "result": {
                "deviceMake": "Acme",
                "deviceModel": "A300",
                "deviceInput": {
                        "ArrowUp": 38,
                        "ArrowDown": 40,
                        "ArrowRight": 39,
                        "ArrowLeft": 37,
                        "Select": 13,
                        "Back": 461,
                        "BAAppear": {
                                "label": "NextGen App",
                                "keycode": 500,
                                "img":
                        "
                        ="
                        }
                },
                "deviceInfo": {
                        "numberOfTuners": 1,
                        "yearOfMfr": 2017
                },
                "deviceSupportedWebSocketAPIs": [
                        "org.atsc.query.ratingLevel",
                        {"method": "org.atsc.query.service", "rev": "20190502"},
                        {"method": "org.atsc.scale-position"}
                ],
                "deviceSupportedDRMs": [
                        "urn:uuid:edef8ba9-79d6-4ace-a3c8-27dcd51d21ed",
                        "urn:uuid:e2719d58-a985-b3c9-781a-b030af78d30e"
                ]
        },
        "id": 93
}
```

9.12 RMP Content Synchronization APIs

이 section에 정의된 **RMP Content Synchronization APIs**는 [9]에 제공된 대로 W3C *HTMLMediaElement*에 사용할 수 있는 API와 유사한 기능을 제공합니다. 독자는 *HTMLMediaElement* API를 검토하여 여기에 설명된 RMP API의 의미 체계에 대한 배경 정보를 얻는 것이 좋습니다.

이 section의 API에 사용되는 시간은 RMP 제어의 적절한 작업에 필수적입니다. API에 사용되는 *startDate* 및 *currentTime* 매개변수와 미디어 시간 간의 관계는 그림 9.12-1에 나와 있습니다. Section 9.12.1의 **Query RMP Media Time**

API 에서 *startDate* 및 *currentTime* 매개 변수에 대한 자세한 정의를 참조하십시오.



(그림 9.12-1) RMP Media Time Representation
[출처: A344]

이 표현에서 wall-clock 시간은 미디어 시간에 브로드캐스트 체인 지연을 더한 값에 배포 지연을 더한 값과 수신기 내부 지연을 더한 값입니다. 각 TV 는 서로 다른 wall-clock 시간에 주어진 미디어 시간을 표시할 수 있습니다.

9.12.1 Query RMP Media Time API

Broadcaster Application 은 수신기에 의해 제시되는 브로드캐스트 콘텐츠의 현재 프리젠테이션 시간을 알기를 원할 수 있다. 이를 통해 **Broadcaster Application** 은 브로드캐스트 콘텐츠와 동기화된 추가 콘텐츠를 제공할 수 있습니다. 예를 들어, **Broadcaster Application** 은 프레젠테이션 중 특정 순간에 그래픽 오버레이를 표시할 수 있습니다.

Query RMP Media Time RequestSemantics 는 표 9.12.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpMediaTime-request.json 에 정의된 대로여야 합니다.

<표 9.12.1-1> Query RMP Media Time Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	integer	
method	1	string	"org.atsc.query.rmpMediaTime"

Query RMP Media Time Response Semantics 는 표 9.12.1-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpMediaTime-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.1-2> Query RMP Media Time Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
currentTime	1	number (>=0.0)	RMP 의 현재 발표 시간
startDate	0..1	string (date-time)	미디어 타임라인의 시작 시간
error	oneOf X		Section 8.3.3 참조

currentTime - 이 필수 부동 소수점 숫자 값은 RMP 에서 제공하는 콘텐츠의 현재 프레젠테이션 시간을 나타내며, *startDate* 에 대한 오프셋(초)으로 표시됩니다. 콘텐츠의 공식 재생 위치를 나타내는 [9]에 주어진 *HTMLMediaElement* 의 *@currentTime* 속성과 동일한 의미를 갖습니다.

startDate - 이 선택적 날짜-시간 값은 RMP 에서 제공하는 콘텐츠의 미디어 타임라인의 시작(즉, 0 시간)에 대한 절대 시간 참조를 나타냅니다. [9]의 "타임라인 오프셋"과 같은 의미를 갖습니다(즉, *HTMLMediaElement* 의 *getStartDate()* 메서드에서 제공되는 값). 날짜-시간 JSON 데이터 유형은 JSON 스키마 사양[19]에 정의된 대로 형식이 지정되어야 합니다.

RMP 가 [41]을 준수하는 콘텐츠를 제공하는 경우 *startDate* 및 *currentTime* 의 보고된 값에 다음 요구 사항이 적용됩니다.

- *startDate* 값은 RMP 가 프레젠테이션 재생 또는 녹화를 시작할 때 RMP 에서 사용 중이었던 MPD 의 **MPD***@availablilityStartTime* 합계와 RMP 가 프레젠테이션 재생 또는 녹화를 시작한 DASH 미디어 프레젠테이션 타임라인의 시간 오프셋을 나타냅니다. 광대역을 통해 전달된 콘텐츠가 RMP 가 콘텐츠 재생 또는 녹화를 시작한 시간(예: 라이브 타임 시프트)보다 먼저 프레젠테이션의 위치를 탐색할 수 있는 경우, DASH 미디어 프레젠테이션 타임라인의 시간 오프셋은 콘텐츠에서 가장 빠른 검색 가능한 시간 오프셋이어야 합니다. 녹화된 콘텐츠의 미디어 형식은 수신기에 따라 다릅니다.
- 녹화된 콘텐츠가 표시될 때 *startDate* 및 *currentTime* 값은 모두 녹화된 콘텐츠의 라이브 버전을 표시하는 동안 적용된 *startDate* 및 *currentTime* 값과 동일한 값을 가져야 합니다.

존재하는 콘텐츠(예: 다운로드한 콘텐츠)에 대해 명시적인 날짜와 시간을 사용할 수 없는 경우 응답에 *startDate* 가 없습니다. 그렇지 않으면 *startDate* 가 응답에 있어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 이 다음을 쿼리하는 경우

<표 9.12.1-3> Example of Query RMP Media Time Request
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.query.rmpMediaTime",
       "id": 61
}
```

Receiver는 다음과 같이 응답할 수 있습니다.

<표 9.12.1-4> Example of Query RMP Media Time Response
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {"currentTime": 3600.033,
              "startDate": "2019-01-01T23:59:59.590Z"
       },
       "id": 61
}
```

9.12.2 Query RMP UTC Time API DEPRECATED

이 API는 더 이상 사용되지 않습니다. 현재 시간이 필요한 **Broadcaster Applications**은 **User Agent**에서 사용할 수 있는 W3C Date.now() 메서드를 사용해야 합니다.

Broadcaster Application은 RMP에서 사용 중인 UTC 시간을 알고 싶어할 수 있습니다. 이 시간은 JavaScript API 대신 UTC 프레젠테이션 타임라인(예: 사용자 에이전트의 Date.now())으로 작업할 때 사용해야 합니다.

Query RMP UTC Time Request의 Semantics는 표 9.12.2-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpUTCTime-response.json에 정의된 대로여야 합니다.

<표 9.12.2-1> Query RMP UTC Time Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	
method	1	string	"org.atsc.query.rmpUTCTime”

Query RMP UTC Time Response의 Semantics는 표 9.12.2-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpUTCTime-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.2-2> Query RMP UTC Time Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치

result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
utcTime	1	string (date-time)	현재 UTC 시간
error	oneOf X		Section 8.3.3 참조

utcTime - 이 필수 date-time 값은 현재 UTC 시간을 나타내며, 방송 PLP 의 L1D_time 필드에 신호화된 PTP 시간(Precision Time Protocol time)을 LLS SystemTime 필드를 이용해 UTC 로 변환(조정)한 값이어야 합니다. 이 date-time JSON 데이터 타입은 JSON Schema Specification [19]에 정의된 형식에 따라 포맷되어야 합니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None - 이 API 와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application** 이 다음을 쿼리하는 경우

<표 9.12.2-3> Example of Query RMP UTC Time Request
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.rmpUTCTime",
      "id": 62
}
```

Receiver 는 다음과 같이 응답할 수 있습니다.

<표 9.12.2-4> Example of Query RMP UTC Time Response
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "result": {"utcTime": "2019-01-01T23:59:56.320Z"},
      "id": 62
}
```

9.12.3 Query RMP Playback State API

Broadcaster Application 은 RMP 에 의해 발표되거나 발표를 위해 준비되는 콘텐츠의 재생 상태를 알고 싶어할 수 있습니다. 이를 통해 애플리케이션은 콘텐츠의 재생 상태에 따라 추가 콘텐츠를 표시할 때 조정할 수 있습니다. 예를 들어, 콘텐츠의 재생이 콘텐츠 버퍼 언더플로("버퍼링") 또는 사용자 입력으로 인해 일시 중지되거나 VOD 콘텐츠 스트림의 끝에 도달하여 중지된 경우 애플리케이션은 보충 콘텐츠의 프레젠테이션을 일시 중단할 수 있습니다.

Query RMP Playback State Request 의 Semantics 는 표 9.12.3-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpPlaybackState-request.json 에 정의된 대로여야 합니다.

<표 9.12.3-1> Query RMP Playback State Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description

Jsonrpc	1	string	“2.0”
Id	1	integer	
method	1	string	"org.atsc.query.rmpPlaybackState"

Query RMP Playback State Response의 Semantics은 표 9.12.3-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpPlaybackState-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.3-2> Query RMP Playback State Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
playbackState	1	integer (-1 … 3)	현재 RMP 재생 상태
error	oneOf X		Section 8.3.3 참조

playbackState - 이 정수 값은 RMP의 다음 재생 상태 중 하나를 나타냅니다.

-1: 콘텐츠가 초기화, 연결 중이고 상태를 확인할 수 없는 경우(예: 채널 변경, SLT, MPD 액세스 및 상태가 불분명한 채널 초기화 사이에 시간 창이 있는 경우);

0: 콘텐츠가 활발하게 재생되고 암호화된 경우 유효한 DRM 라이선스도 있어야 하며 CDM이 콘텐츠를 해독하는 경우.

1: 어떤 이유로든 재생이 일시 중지되고 종료되지 않은 경우(예: 검색 또는 중단, 사용자 상호 작용을 위해 일시 중지, 사용자 입력 대기, 오류로 인해 중지됨)

2: 재생이 종료된 경우(예: 콘텐츠의 끝에 도달한 경우).

3: 콘텐츠가 암호화되어 볼 수 없는 경우(즉, 유효한 키가 없고 CDM이 콘텐츠를 해독하지 않는 경우).

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None - 이 API와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application**은 다음과 같은 쿼리를 만듭니다.

<표 9.12.3-3> Query RMP Playback State Request
[출처: A344]

```
--> {
      "jsonrpc": "2.0",
      "method": "org.atsc.query.rmpPlaybackState",
      "id": 63
}
```

RMP가 콘텐츠를 재생하는 경우 **Receiver**는 다음과 같이 응답합니다.

<표 9.12.3-4> Query RMP Playback State Response

[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"playbackState": 0},
        "id": 63
}
```

9.12.4 Query RMP Playback Rate API

Broadcaster Application은 RMP에서 콘텐츠를 제공하는 속도를 알고 싶어할 수 있습니다. 이를 통해 애플리케이션은 콘텐츠의 재생 속도에 따라 보충 콘텐츠를 표시할 때 조정할 수 있습니다. 예를 들어, 애플리케이션은 트릭 플레이 모드 동안 보충 콘텐츠의 프레젠테이션을 일시 중단하도록 선택할 수 있습니다.

Query RMP Playback Rate Request의 Semantics는 표 9.12.4-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpPlaybackRate-request.json에 정의된 대로여야 합니다.

<표 9.12.4-1> Query RMP Playback Rate Request Semantics

[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	integer	
method	1	string	"org.atsc.query.rmpPlaybackState"

Query RMP Playback Rate Response의 Semantics는 표 9.12.4-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.query.rmpPlaybackRate-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.4-2> Query RMP Playback Rate Response Semantics

[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
playbackRate	1	number	현재 RMP 재생 비율
error	oneOf X		Section 8.3.3 참조

playbackRate - 이 필수 부동 소수점 값은 [9]에 제공된 *HTMLMediaElement*의 *playbackRate* 특성과 동일한 의미로 RMP에서 현재 제공하는 콘텐츠의 재생 속도를 나타냅니다.

Section B.3.1의 오류 외에도 표 8.3.3-2의 다음 오류가 반환될 수 있습니다.

- None - 이 API와 관련된 오류가 없습니다.

예를 들어 **Broadcaster Application**은 다음과 같은 쿼리를 만듭니다.

<표 9.12.4-3> Example of Query RMP Playback Rate Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.query.rmpPlaybackRate",
        "id": 65
}
```

그리고 RMP가 정상 속도로 콘텐츠를 재생하는 경우 **Receiver**는 다음과 같이 응답합니다.

<표 9.12.4-4> Example of Query RMP Playback Rate Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"playbackRate": 1.0},
        "id": 65
}
```

9.12.5 RMP Media Time Change Notification API

RMP에서 제공하는 콘텐츠의 현재 재생 위치가 변경된 경우 **RMP Media Time Change Notification API**는 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 발행할 것으로 예상됩니다. 이 API는 [9]에 주어진 *HTMLMediaElement*의 *timeupdate* 이벤트와 기능적으로 동일한 의미를 갖습니다. **Receiver**는 RMP의 공식 재생 위치가 일반 또는 트릭 재생의 일부로 변경될 때 이 API를 사용하여 **Broadcaster Application**에 알립니다.

RMP Media Time Change Notification의 Semantics는 표 9.12.5-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-rmpMediaTimeChange.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

Period를 수정하거나 바꾸는 데 필요한 단일 알림에 데이터가 포함되어 있지만 이러한 필드 또는 관련 필드의 상태를 사용할 때 여전히 경합 상태가 발생할 위험이 있습니다

<표 9.12.5-1> RMP Media Time Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"rmpMediaTimeChange”
currentTime	1	number (>=0.0)	RMP의 현재 재생 시간
startDate	0..1	string (xs:timeDate)	미디어 타임라인의 시작 시간
playbackState	0..1	number	Section 9.12.3에 정의됨
refClock	0..1	string (xs:timeDate)	L1D 프리앰블에서 파생되고 LLS SystemTime 테이블에 의해 수정된 현재 참조 시간
adaptationId_video	0..1	string	MPD.Period.AdaptationSet.

			AdaptationSet@id 의 공백으로 구분된 목록
adaptationId_audio	0..1	string	MPD.Period.AdaptationSet. AdaptationSet@id 의 공백으로 구분된 목록
adaptationId_text	0..1	string	MPD.Period.AdaptationSet. AdaptationSet@id
periodId	0..1	string	MPD.Period@id
periodStart	0..1	string (xs:timeDate)	현재 Period 의 시작 시간
duration	0..1	number (>0)	현재 Period 의 길이입니다.
source	0..1	array	각 적응 ID 에 대한 미디어 재생의 id, uriType 및 sourceType
items	0..N		
id	1	string	요소의 AdaptationSet ID
uriType	1	enum	"GSID", "XLinkURN" or "RMPURL"
sourceType	1	enum	표 9.12.5-2 에 정의됨

currentTime 및 *startDate* 는 section 9.12.1 의 **Query RMP Media Time API** 에 정의되어 있습니다. *params* 에 *startDate* 키/값이 없으면 키/값 쌍의 값이 변경되지 않음을 나타냅니다.

playbackState - Section 9.12.3 에 정의된 대로입니다.

refClock - L1D 프리앰블에서 파생되고 LLS SystemTime 테이블[3]에 의해 수정된 현재 참조 시간이어야 합니다. *playbackState* 가 있는 경우 이 매개 변수가 있어야 합니다.

adaptationId_video - 이 속성은 현재 렌더링 중인 비디오 트랙들의 **MPD.Period.AdaptationSet***@id* 값을 공백으로 구분된 문자열 형태로 나타내야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시 포함되어야 합니다.

adaptationId_audio - 이 속성은 현재 렌더링 중인 오디오 트랙들의 **MPD.Period.AdaptationSet***@id* 값을 공백으로 구분된 문자열 형태로 나타내야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시 포함되어야 합니다.

adaptationId_text - 이 속성은 현재 렌더링 중인 텍스트 트랙(subtitle/caption)의 **MPD.Period.AdaptationSet***@id* 값을 나타내야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시 포함되어야 합니다.

periodId - 이 속성은 현재 재생 중인 Period 의 **MPD.Period***@id* 의 값을 지정해야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시 포함되어야 합니다.

periodStart - 이 속성은 현재 재생 중인 Period 의 **MPD.Period***@start* 의 값을 지정해야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시 포함되어야 합니다.

duration - 이 속성은 현재 재생 중인 Period 의 *duration* 값을 지정해야 합니다. *playbackState* 속성이 존재하는 경우, 해당 매개변수가 반드시

포함되어야 합니다.

source – 현재 재생 중인 미디어를 정의하는 *id*, *uriType* 및 *sourceType* 매개 변수로 구성된 개체의 배열이어야 합니다. *playbackState*가 있는 경우 이 배열이 있어야 하며 현재 재생 중인 미디어에 있는 각 적응 집합을 설명하는 요소를 포함해야 합니다.

id – Adaptation set ID를 포함해야 합니다.

uriType – Section 9.6.3의 *setRMPUrl*에 따라 "*GSID*", "*XLinkURN*" 또는 "*RMPURL*" 중 하나여야 합니다. "*GSID*"는 SLT 내 SLT.Service*@globalServiceID* 에 명시된 서비스에 연관된 *globalServiceID*여야 합니다. A/331 [3] Section 6.3을 참조하십시오.

sourceType – 표 9.12.5-2에 따라 상단에서 하단 순으로 적용 가능한 첫 번째 *mediaType*을 사용하여 *sourceType*을 제공해야 합니다. *playbackState*가 존재하는 경우, 이 속성도 반드시 포함되어야 합니다.

<표 9.12.5-2> *sourceType* Definition
[출처: A344]

sourceType	Type of RMP Source
"broadcast"	RMP가 브로드캐스트 스트림을 재생 중이고 현재 기간이 XLink 해상도로 대체되지 않은 경우
"broadband"	RMP가 broadband 스트림을 재생 중이고 현재 기간이 XLink 해상도로 대체되지 않은 경우.
"setrmpurl"	RMP가 *setRMPURL*을 사용하여 설정된 스트림을 재생하는 경우.
"xlink"	RMP가 스트림 유형을 재생 중이고 현재 기간이 XLink 해상도를 통해 대체된 경우
"other"	RMP가 미리 알려지지 않은 소스에서 재생 중이거나 스트림을 재생하도록 구성되지 않은 경우.

기본적인 예를 들어, Receiver는 현재 서비스의 정상 재생 중에 250 내지 500msec 마다 미디어 시간 변경을 Broadcaster Application에 알릴 수 있습니다. currentTime 값 3600.033을 포함하는 이전 알림 직후에 제공되는 알림은 다음과 같을 수 있습니다.

<표 9.12.5-3> Example of RMP Media Time Change Notification 1
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "method": "org.atsc.notify",
      "params":{
            "msgType": "rmpMediaTimeChange",
            "currentTime": 3600.283
      }
}
```

*playbackState*가 있고 서비스에 두 개의 오디오 트랙(그 중 하나는 광대역을 통해 전달됨)이 포함된 경우 더 복잡한 예는 다음과 같습니다.

<표 9.12.5-4> Example of RMP Media Time Change Notification 2 [출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgtype": "rmpMediaTimeChange",
                "currentTime": 350.424,
                "startDate": "2019-01-01T23:59:59.590Z",
                "playbackState": 0,
                "refClock": "2019-01-01T23:59:53.590Z",
                "adaptationId_video": "2",
                "adaptationId_audio": "1 4",
                "adaptationId_text": "3",
                "periodId": "P1",
                "periodStart": "2019-01-01T23:50:30.000Z",
                "duration": 600,
                "source": [
                        {
                                "id": "2",
                                "uriType": "GSID",
                                "sourceType": "broadcast"
                        },
                        {
                                "id": "1",
                                "uriType": "GSID",
                                "sourceType": "broadcast"
                        },
                        {
                                "id": "4",
                                "uriType": "GSID",
                                "sourceType": "broadband"
                        },
                        {
                                "id": "3",
                                "uriType": "GSID",
                                "sourceType": "broadcast"
                        }
                ]
        },
        "id": 913
}
```

9.12.6 RMP Playback State Change Notification API

RMP Playback State Change Notification API는 section 9.12.3의 **Query RMP Playback State API**에 정의된 RMP의 재생 상태가 한 값에서 다른 값으로 변경되는 경우 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 발행할 것으로 예상됩니다.

RMP Playback State Change Notification의 Semantics는 표 9.12.6-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-rmpPlaybackStateChange.json에 정의된 대로여야 합니다. 매개 변수의 추가 의미 체계 정의는 표를 따릅니다.

<표 9.12.6-1> RMP Playback State Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"rmpPlaybackStateChange
playbackState	1	number (-1…3)	새로운 RMP 재생 상태

playbackState - 이 정수 값은 section 9.12.3 의 **Query Playback State API** 에 정의된 재생 상태 중 하나인 새 재생 상태를 나타냅니다.

예를 들어, **Receiver** 의 사용자가 타임 시프트 브로드캐스트 콘텐츠의 재생을 일시 중지하는 경우, **Receiver** 는 아래와 같이 **Broadcaster Application** 에 재생 상태를 알립니다.

<표 9.12.6-2> Example of RMP Playback State Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "rmpPlaybackStateChange",
                "playbackState": 1
        }
}
```

9.12.7 RMP Playback Rate Change Notification API

RMP Playback Rate Change Notification API 는 section 9.12.4 의 **Query RMP Playback Rate API** 에 정의된 재생 속도가 한 값에서 다른 값으로 변경되는 경우 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상됩니다. 이 API 는 [9]에 주어진 대로 *HTMLMediaElement* 의 *ratechange* 이벤트와 기능적으로 동일한 의미를 갖습니다.

RMP Playback Rate Change Notification 의 Semantics 는 표 9.12.7-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-rmpPlaybackRateChange.json 에 정의된 대로여야 합니다. 매개변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.7-1> RMP Playback Rate Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"rmpPlaybackStateChange”
playbackRate	1	number	새로운 RMP 재생 속도

playbackRate 는 section 9.12.4 의 **Query RMP Playback Rate API** 에 정의된 대로입니다.

예를 들어, **Receiver**의 사용자가 정상 재생 속도의 2배로 타임 시프트 콘텐츠의 빨리 감기 재생을 수행하는 경우, 수신기는 아래와 같이 재생 속도 변경을 **Broadcaster Application**에 알립니다.

<표 9.12.7-2> Example of RMP Playback Rate Change Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "rmpPlaybackRateChange",
                "playbackRate": 2
        }
}
```

9.12.8 RMP Media Asset Change Notification API

RMP Media Asset Change Notification API는 RMP에서 제공하는 asset의 현재 재생 위치가 변경된 경우 **Receiver**가 현재 실행 중인 **Broadcaster Application**에 발행할 것으로 예상됩니다.

RMP Media Asset Change Notification의 Semantics는 표 9.12.8-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-rmpMediaAssetChange.json에 정의된 대로여야 합니다. 매개변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.12.8-1> RMP Media Asset Change Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"rmpMediaAssetChange”
currentTime	1	number (>=0.0)	RMP의 현재 재생 시간
startDate	0..1	string (xs:timeDate)	미디어 타임라인의 시작 시간
assets	1..N	array	자산 목록
items			
assetId	1	string	MP 테이블 - Identifier_mapping() [30], 10.6.3 식별자 매핑
assetType	0..1	string	Asset의 유형
presentationTime	1	string (date-time)	MPU의 첫 번째 MFU 프리젠테이션 시간
duration	1	number (>0)	현재 Asset의 지속 시간(초)
sourceType	0..1	string	broadcast 혹은 broadband
packageId	1	integer	Package의 고유 식별자 [30], 6.2 Package

currentTime 및 *startDate*는 section 9.12.1의 **Query RMP Media Time API**에 정의되어 있습니다. 매개 변수에 *startDate* 키/값이 없으면 키/값 쌍의 값이 변경되지 않음을 나타냅니다.

assetId - 이 필수 정수는 MP 테이블의 asset ID 값에 해당해야 합니다.

assetId 에는 UUID(Universally Unique Identifier) 또는 URI(Uniform Resource Identifier) 체계가 있을 수 있으며 형식은 길이가 32 비트인 바이트 배열입니다.

assetType – 이 값이 제공되는 경우, 이 값은 MP4REG (http://www.mp4ra.org) [30]의 10.3.9 MP 표에 등록된 네 문자 코드("4CC") 유형이어야 합니다.

presentationTime – 지정된 MPU [30]의 10.5.2 MPU 타임스탬프 설명자에 따른 첫 번째 AU 의 표시 시간이어야 하며, JSON Schema 사양 [19]에 정의된 *date-time* JSON 데이터 유형으로 표현되어야 합니다.

duration – Asset duration 입니다.

sourceType – 표 9.12.5-2 에 따라 *sourceType* 을 제공해야 합니다.

packageId – Asset 을 포함한 패키지의 패키지 ID 여야 합니다.

9.13 DRM APIs

이 section 의 API 는 **Broadcaster Application** 에서 암호화된 콘텐츠의 RMP 또는 AMP 처리를 지원하는 데 사용할 수 있습니다. 두 개의 일반 API 가 정의됩니다. "notification" API 는 **Receiver** 가 식별된 DRM 시스템과 관련된 메시지를 **Broadcaster Application** 에 전달하는 데 사용됩니다. "operation" API 는 **Broadcaster Application** 에서 식별된 DRM 시스템과 관련된 메시지를 **Receiver** 에게 전달하는 데 사용됩니다. 이러한 API 는 AMP 와 RMP 의 요구 사항을 모두 지원합니다.

ATSC 3.0 수신기 환경 내의 DRM 작업에 대한 자세한 내용은 ATSC A/362 DRM(Digital Rights Management)에 대한 권장 사례[39]를 참조하십시오.

9.13.1 DRM Notification API

DRM 관련 알림을 전달하기 위해 **Receiver** 가 **Broadcaster Application** 에 **DRM Notification API** 를 사용할 수 있습니다. 이 알림을 수신하는 **Broadcaster Application** 은 section 9.13.2 에 정의된 **DRM Operation API** 를 사용하여 **Receiver** 의 기본 콘텐츠 보호 시스템과 메시지를 교환할 수 있으며, 궁극적으로 보호된 콘텐츠의 암호 해독을 위해 RMP 또는 AMP 에 필요한 라이선스/키를 전달할 수 있습니다.

DRM Notification 의 Semantics 는 표 9.13.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-DRM.json 에 정의된 대로여야 합니다. 매개변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.13.1-1> DRM Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
method	1	string	"org.atsc.notify"

msgType	1	enum	"DRM"
systemId	1	string	알림을 보내는 DRM 시스템의 식별자
message	0..1	array	systemId 에 종속된 DRM 시스템 독점 JSON 개체의 배열
items			

msgType - 이 알림 API 를 식별하기 위해 이 필수 문자열을 "DRM"으로 설정해야 합니다.

systemId - 이 문자열은 DASH-IF IOP, section 7.6 [41]에 정의된 대로 DRM 시스템 식별자 @schemeIdUri 로 설정되어야 합니다. 예를 들어 UUID 문자열 값 "1077efec-c0b2-4d02-ace3-3c1e52e2fb4b"는 W3C EME 의 공통 시스템 ID 에 해당합니다. DASH 의 경우 값은 MPD 의 ContentProtection decriptor 의 @schemeIdUri 값에 해당합니다. MMT 의 경우 값은 A/331 섹션 7.2.4 [3]에 지정된 MPU security_property_descriptor 의 system_UUID 값에 해당합니다.

message - JSON 개체의 배열로 형식이 지정된 콘텐츠 보호 시스템에서 전달된 메시지입니다.

DRM Notification API 는 **DRM Operation API** (section 9.13.2)를 통해 요청된 특정 DRM 라이선스 개체가 브로드캐스트에서 검색되었음을 **Broadcaster Application** 에 알리기 위해 **Receiver** 가 사용할 수 있습니다.

Broadcaster Application 이 라이선스 서버와 상호 작용하는 방식과 이 **DRM Notification API** 를 사용하여 콘텐츠 보호 시스템에서 **Broadcaster Application** 으로 전달되는 메시지의 정확한 형식은 본 사양의 범위를 벗어납니다. 이 API 에 정의된 인터페이스를 통해 전달되는 메시지의 형식은 브로드캐스터에서 사용하는 콘텐츠 보호 시스템에 따라 다릅니다.

9.13.2 DRM Operation API

DRM Operation API 는 **Broadcaster Application** 에서 발급하여 [8]의 section 5.7 에 정의된 대로 보호된 콘텐츠를 재생하기 위해 **Receiver** 에 메시지를 전달할 수 있습니다. 이 API 는 콘텐츠 보호에 대한 정보를 알리기 위해 **Broadcaster Application** 에 메시지를 알리기 위해 **Receiver** 가 발행하는 section 9.13.1 에 정의된 **DRM Notification API** 와 함께 사용할 수 있습니다.

HTML5 미디어 요소의 확장인 W3C EME[31]와 유사하게 **Broadcaster Application** 은 라이선스 서버와 통신하고 이 API 를 통해 라이선스/키 교환을 위한 메시지를 기본 콘텐츠 보호 시스템에 전달할 수 있습니다. 이러한 API 는 메시지 교환 메커니즘만 제공하고 라이선스 설치/업데이트/제거와 같은 제어를 포함하여 API 에 전달되는 메시지의 콘텐츠는 기본 콘텐츠 보호 시스템에만 해당되므로 여기에 지정되지 않으므로 W3C EME 보다 간단합니다. **Broadcaster Application** 은 브로드캐스터의 웹 서버 및 라이선스 서버와의 상호 작용 순서와 **Receiver** 의 기본 콘텐츠 보호 시스템과 메시지를 교환하는 절차를 알아야 합니다.

DRM Operation Request 의 Semantics 는 표 9.13.2-1 에 정의되어 있으며

구문은 스키마 파일 org.atsc.drmOperation-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.13.2-1> DRM Operation Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"DRM”
systemId	1	string	메시지가 의도된 DRM 시스템의 식별자
service	1	string(uri)	현재 선택된 서비스의 globalServiceID 를 지정
message	0..1	array	systemId 에 종속된 DRM 시스템 독점 JSON 개체의 배열
items	0..N	object	

systemId - 이 문자열은 DASH-IF, section 7.6 [39]에 정의된 대로 DRM 시스템 식별자 @schemeIdUri 로 설정되어야 합니다.

service - 이 필수 문자열은 SLT 의 SLT 에 제공된 대로 현재 선택한 서비스와 연결된 globalServiceID 를 나타내야 합니다. 이 DRM 메시지가 연결된 Service@globalServiceID 입니다.

message - JSON 개체의 배열로 형식이 지정된 사용 중인 콘텐츠 보호 시스템과 관련된 메시지입니다.

DRM Operation Response 의 Semantics 는 표 9.13.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.drmOperation-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.13.2-2> DRM Operation Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
elementType	1	string	(MPD) 요소의 이름(예: "Period")
elementId	1	string	elementType 의 @Id
error	oneOf X		Section 8.3.3 참조

message - 사용 중인 콘텐츠 보호 시스템과 관련된 JSON 개체의 배열로 형식이 지정된 명령에 대한 응답입니다.

Seciton B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -6: 서비스 속성에서 제공하는 globalServiceID 를 찾을 수 없습니다.
- -14: 제공된 DRM 시스템 식별자가 지원하지 않거나 **Receiver** 에게 알려지지 않습니다.

AMP 의 경우, **Broadcaster Application** 에서 **DRM Operation API** 를 사용하여 수신자에게 브로드캐스트 **Entitlement Management** 서비스에서 특정 DRM 라이센스 파일을 가져오도록 요청하는 경우, 메시지 객체에는 파일의 Content-Location 을 지정하는 키/값 쌍이 포함될 수 있습니다.

9.14 XLink APIs

이 section의 API는 **Broadcaster Application**이 XLink로 표시된 DASH 요소를 바꿀 수 있는 메커니즘을 제공합니다. 둘 이상의 요소를 반환할 수 있습니다(예: Period를 둘 이상의 Period로 바꾸기). 두 가지 API가 정의되어 있습니다.

- **Receiver**가 RMP가 xlink:href 속성이 있는 MPD 요소를 감지했음을 **Broadcaster Application**에 알릴 수 있는 XLink 확인 알림 API. 알림은 **Broadcaster Application**이 알림을 구독한 경우에만 발생합니다(section 9.2.1.1 참조). 그런 다음 **Broadcaster Application**은 **XLink Resolved API**를 사용하여 대체 요소를 제공할 수 있습니다.
- **XLink Resolved API**는 **Broadcaster Application**에서 xlink:href가 감지된 요소의 대안으로 RMP에서 사용할 대체 MPD URL 또는 인라인 텍스트를 제공하는 데 사용됩니다. 리소스를 찾을 수 없거나 대체하기에는 너무 늦은 경우 수신자가 **XLink Resolved Request**을 무시할 수 있습니다.

9.14.1 XLink Resolution Notification API

XLink 확인 알림은 **Receiver Media Player**(RMP)가 MPD의 요소에서 @xlink:href 속성을 발견할 때 수신기에 의해 현재 실행 중인 **Broadcaster Application**에 발행될 것으로 예상됩니다. **Receiver**는 **Broadcaster Application**에 알리지 않고 URL 체계 "https:"를 사용하여 xlink:href를 확인할 수 있습니다. 다른 모든 URI의 경우, **Broadcaster Application**이 XLink 해결 알림을 구독하고 있는 경우 **Receiver**는 **Broadcaster Application**에 xlink:href를 해결하도록 알려야 합니다(section 9.2.1.1 참조). **Broadcaster Application**은 xlink:href가 나타난 요소를 대체하기 위해 **XLink Resolved API**(section 9.14.2 참조)를 사용하여 응답해야 합니다. MPD에는 둘 이상의 xlink:href가 있을 수 있으며, 이 경우 수신기는 XLink 확인 알림을 여러 번 호출해야 합니다.

Receiver에 의한 xlink:href 확인 타이밍은 xlink:actuate에 의해 제어되며, **Receiver**가 MPD를 수신할 때 XLink를 확인하도록 "onLoad"로 설정해야 합니다. 이는 기본값이 아닙니다. 따라서 xlink:actuate="onLoad"는 일반적으로 **Broadcaster Application**이 링크를 해결하는 데 최대 시간을 제공하기 위해 존재해야 합니다.

XLink Resolution Notification Semantics는 표 9.14.1-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-xlinkResolution.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.14.1-1> XLink Resolution Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"xlinkResolution"
xlink	1	string(uri)	DASH 요소에서 xlink:href URI 가 감지
elementType	0..1	string	"MPD", "Period" 또는 xlink 가 허용되는 기타 MPD 요소 이름 중 하나
elementId	1	string	@id 요소의 값

xlink - DASH 요소에서 감지된 xlink:href 문자열입니다.

elementType - MPD 의 요소 유형 이름(예: "MPD", "Period" 등) 이 속성을 생략하면 기본값은 "기간"입니다.

elementId - 이 XLink 와 연결된 요소의 @id 값입니다.

XLink 해결 알림이 전송되면 **Broadcaster Application** 은 자체 기준을 사용하여 대체 콘텐츠를 제공하기로 결정하거나 제공하지 않기로 선택할 수 있습니다.

예를 들어, 브로드캐스트 스트림에는 다양한 기준에 따라 다양한 시청자를 대상으로 할 수 있는 광고가 포함될 수 있습니다. 기본 보급 알림 및 관련 콘텐츠의 시간 범위는 xlink:href 특성을 포함하는 특정 DASH 요소 구조에 설명됩니다. 광고 교체를 수용하기 위해 **Broadcaster Application** 은 XLink 해상도 알림 API 를 구독합니다.

xlink:href 를 포함하는 요소를 발견하면 **Receiver** 는 다음 메시지를 발행해야 합니다.

<표 9.14.1-2> Example of XLink Resolution Notification
[출처: A344]

```
<-- {
      "jsonrpc": "2.0",
      "method": "org.atsc.notify",
      "params": {
            "msgType": "xlinkResolution",
            "xlink": "urn:rid:xbc4399FB77-3939EA47",
            "elementType": "Period",
            "elementId": "2"
      }
}
```

9.14.2 XLink Resolved API

XLink Resolution Notification (section 9.14.1)이 수신된 후, **Broadcaster Application** 은 기본 콘텐츠를 대체할 대체 콘텐츠를 결정하고 **XLink Resolved API** 를 사용하여 **Receiver** 에게 요청합니다. **Broadcaster Application** 은 대체 콘텐츠의 XML 조각 또는 요청의 실제 요소 텍스트의 URI 를 제공합니다. **Receiver** 는 요소를 하나 이상의 대체 요소로 대체하기 위해 대체 XML 조각 URI 또는 요소 텍스트를 적절하게 처리해야 합니다. **Receiver** 가 기본 콘텐츠를 바꿀 수 없는 경우 반환된 처리 개체에는 실패 이유를 나타내는 적절한 코드 및 설명이 포함되어야 합니다.

XLink Resolved Request Semantics는 표 9.14.2-1에 정의되어 있으며 구문은 스키마 파일 org.atsc.xlinkResolution-request.json 에 정의된 대로입니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.14.2-1> XLink Resolved Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
id	1	integer	
method	1	string	"org.atsc.xlinkResolution"
xlink	1	string(uri)	MPD 요소의 xlink:href 속성에 있는 XLink 값
mpdURL	one of X	string(uri)	xlink:href 속성을 포함하는 기간 요소를 대체할 XML 요소를 포함하는 대체 MPD 조각의 URI
element	one of X	string	xlink:href 속성을 포함하는 요소를 대체할 인라인 텍스트
status	0..1	integer	Broadcaster Application 의 작업 상태
elementType	0..1	string	xlink:href를 포함하는 XML 요소의 유형
elementId	0..1	string	대체되는 요소의 @id

xlink - 이 필수 문자열은 일부 요소의 xlink:href 속성 값이어야 합니다. 이 문자열은 **XLink Resolution Notification** 매개변수에서 수신된 값에 해당합니다(section 9.12.1 참조).

mpdURL - 요소의 대안으로 제공되는 경우 이 문자열은 xlink:href 특성을 포함하는 요소를 대체하기 위해 XML 조각의 URI를 제공해야 합니다. *mpdURL* 또는 요소 속성 중 하나가 제공되어야 합니다. 이 속성이 제공되는 경우 광대역을 통해 XML 요소를 참조하든 응용 프로그램 컨텍스트 캐시에서 참조하든 정규화된 URI여야 합니다. **Application Context Cache**의 XML 요소의 경우 section 9.1.7에 설명된 대로 **Query Receiver Web Server URI API**를 사용하여 제공된 기본 URI를 사용하여 전체 URI를 생성할 수 있습니다.

element - *mpdURL*의 대안으로 제공되는 경우 이 문자열에는 xlink:href 특성이 포함된 요소를 대체하는 실제 요소 텍스트가 포함되어야 합니다. *mpdURL* 또는 요소 속성 중 하나가 제공되어야 합니다.

status - 다음과 같은 **Broadcaster Application** 반응의 상태입니다.

0: 요소가 교체되지 않았으며 **Receiver**가 자유롭게 작업할 수 있습니다.

1: 요소가 교체되었습니다. **Receiver**는 요소를 교체하려고 시도합니다.

-1: 요소가 교체되지 않았습니다. **Receiver**는 이에 대해 행동하는 것이 금지되어 있습니다.

elementType - 대체되는 요소의 이름(예: "MPD", "Period" 등) 기본값은 "Period"입니다.

elementId - 대체되는 요소의 @id입니다.

XLink Resolved Request을 받은 후 **Receiver**는 원래 기본 요소가 요청된 XML 요소 URI 또는 제공된 요소로 성공적으로 대체되었음을 나타내는 응답할 수

있습니다. **Receiver** 가 요청을 성공적으로 완료할 수 없는 경우 오류 코드가 반환될 것으로 예상됩니다. **Receiver** 는 요소가 성공적으로 교체될 때까지 응답을 지연시키거나 오류 조건으로 즉시 응답하도록 선택할 수 있습니다. **Broadcaster Application** 은 **XLink Resolved Response** 를 처리할 때 타이밍의 차이를 고려해야 합니다. **XLink Resolved Response** Semantics 는 표 9.14.2-2 에 정의되어 있으며 구문은 스키마 파일 org.atsc.xlinkResolution-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.14.2-2> XLink Resolved Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
elementType	1	string	(MPD) 요소의 이름(예: "Period")
elementId	1	string	elementType 의 @Id
error	oneOf X		Section 8.3.3 참조

elementType - 대체되는 요소의 이름(예: "MPD", "Period" 등)

elementId - 대체되는 요소의 *@id* 입니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 요청된 콘텐츠를 찾을 수 없습니다. 예를 들어 잘못된 URL 입니다.
- -30: 기본 컨텐츠를 교체하기 위해 요청이 제때 수신되지 않았습니다.
- -31: 제공된 *elementType* 및/또는 *elementId* 를 찾을 수 없습니다.
- -32: 제공된(MPD) 요소 콘텐츠가 유효하지 않습니다.

예를 들어, **Broadcaster Application** 은 기본 URI 가 "http://192.168.32.117:8182/s02gPkwZx14iO"이라고 가정하여 다음 요청을 제출합니다.

<표 9.14.2-3> Example of XLink Resolved Request 1
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.xlinkResolution",
        "params": {
                "xlink": "urn:rid:xbc4399FB77-3939EA47",
                "mpdURL": "http://192.168.32.117:8182/s02gPkwZx14iO",
                "status": 1
        },
        "id": 104
}
```

Receiver 가 기본 콘텐츠를 대체 XML 요소 URI 로 바꿀 수 있는 경우 아래와 같이 사용된 대체 요소로 성공적으로 대체되었음을 나타내는 성향이 응답합니다.

<표 9.14.2-4> Example of XLink Resolved Response 1-1
[출처: A344]

```
<-- {
        "jsonrpc": "2.0"
        "result": {
                "elementType": "Period",
                "elementId": "3"
        },
        "id": 104
}
```

또는 **Receiver**가 요청된 대로 대체 XML 요소 URI를 사용할 수 없는 경우 브로드캐스터 애플리케이션은 실패한 요청의 이유를 나타내는 성향을 반환합니다.

<표 9.14.2-5> Example of XLink Resolved Response 1-2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "error": {
                "code": -30,
                "message": "Too late"
        },
        "id": 104
}
```

추가 예로, **Broadcaster Application** 은 *mpdURL* 대신 실제 대체 마침표 텍스트가 포함된 다음 요청을 제출합니다.

<표 9.14.2-6> Example of XLink Resolved Request 2
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.xlinkResolution",
        "params": {
                "xlink": "urn:xbc4399FB77-3939EA47",
                "element":    "<Period    start='PT0S'>    AdaptationSet
                mimeType='video/mp4'     …      >     SegmentTemplate
                timescale='90000'          …media='alt-$Number$.mp4v'
                duration='90000'startNumber='32401'  />  Representation
                id='v2'       width='1920'       height='1080'       …
                /></AdaptationSet></Period>",
                "status": 1
        },
        "id": 105
}
```

Receiver가 기본 콘텐츠를 대체 기간으로 대체할 수 있는 경우 아래와 같이 사용된 대체 기간으로 성공적으로 교체되었음을 나타내는 성향으로 응답합니다.

<표 9.14.2-7> Example of XLink Resolved Response 2
[출처: A344]

```
<-- {
        "jsonrpc": "2.0"
        "result": {
                "elementType": "Period",
                "elementId": "3"
```

```
        },
        "id": 105
}
```

9.15 Prepare for Service Change API

이 section 의 API 는 **Receiver** 가 서비스 변경 중에 **Broadcaster Application** 이 해체될 수도 있고 그렇지 않을 수도 있는 상황에 대비하여 리소스를 해제하고 기타 정리를 수행하여 **Broadcaster Application** 이 서비스 변경을 준비하도록 요청하는 메커니즘을 제공합니다.

Prepare for Service Change API 를 사용하면 **Receiver** 가 다음을 수행할 수 있습니다.

- 서비스 변경이 시작되고 있음을 **Broadcaster Application** 에 알립니다.
- **Broadcaster Application** 에 리소스를 일반적으로 해제하도록 알리고,
- 선택적으로 **Broadcaster Application** 에 특정 리소스를 해제하도록 지시합니다.

Prepare for Service Change Request 이 구현되는 경우 서비스 변경이 진행 중일 때 **Receiver** 가 현재 실행 중인 **Broadcaster Application** 에 발행할 것으로 예상되며, **Receiver** 는 서비스 변경 중에 **Broadcaster Application** 이 종료 및 해체될 수 있는 가능성에 대비하기 위해 **Broadcaster Application** 이 할당된 리소스를 해제하기를 원합니다. 선택적으로 **Receiver** 는 해제할 특정 리소스를 열거할 수 있습니다.

Prepare for Service Change Request 의 Semantics 는 표 9.15.1-1 에 정의되어 있으며, 구문은 스키마 파일 org.atsc.prepSvcChange-request.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.15.1-1> Prepare for Service Change Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.prepSvcChange"
fromService	1	uri	현재 서비스의 globalServiceID 를 지정
toService	0..1	uri	향후 서비스의 globalServiceID 를 지정
resources	0..1	array	해제할 특정 리소스를 설명하는 토큰 배열
items	1..N	string	요청된 리소스 토큰

fromService – 이 필수 URI 는 SLT 의 SLT.Service@globalServiceID 에 지정된 대로, 현재 선택된 서비스와 연관된 globalServiceID 를 나타내야 합니다.

toService – 이 선택적 URI 가 존재하는 경우, SLT 의 SLT.Service@globalServiceID 에 지정된 대로, 획득될 서비스와 연관된

globalServiceID 를 나타내야 합니다.

resources – 이 선택적 배열(있는 경우)에는 **Broadcaster Application** 에서 릴리스할 특정 리소스를 나타내는 토큰이 포함되어야 합니다. 이 필드에 특정 토큰이 있는지 여부에도 불구하고 **Broadcaster Application** 은 가능한 한 많은 리소스를 릴리스할 것으로 예상됩니다. 이 배열의 값은 표 9.15.1-2 에서 가져와야 합니다.

<표 9.15.1-2> Service Change Resource Tokens
[출처: A344]

Token	Description
AMP	모든 애플리케이션 미디어 플레이어 리소스
(other values)	사용자 개인 또는 확장에 허용됨

Prepare for Service Change Response 의 Semantics 은 표 9.15.1-3 에 정의되어 있으며 구문은 스키마 파일 org.atsc.prepSvcChange-response.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.15.1-3> Prepare for Service Change Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	"2.0"
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
resources	0..1	array	출시된 특정 리소스를 설명하는 토큰 배열
items	1..N	string	요청된 리소스 토큰
error	oneOf X		Section 8.3.3 참조

resources – 이 선택적 배열이 존재하는 경우, **Broadcaster Application** 에 의해 해제된 특정 리소스를 나타내는 토큰을 포함해야 합니다. 이 배열의 값은 표 9.15.1-3 에 정의된 값이어야 합니다. **Broadcaster Application** 이 직접 할당하지 않았지만, **Receiver** 가 해제를 요청한 리소스가 있는 경우, **Broadcaster Application** 은 해당 리소스를 이 응답에 포함해야 합니다. **Broadcaster Application** 이 요청에서 지정된 특정 리소스를 해제하지 않거나(또는 해제된 것으로 식별하지 않는 경우), **Receiver** 는 *toService* 와 연관된 다른 **Broadcaster Application** 과 관계없이 해당 **Broadcaster Application** 을 종료할 수 있습니다.

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- None – 이 API 와 관련된 오류가 없습니다.

예를 들어 **Receiver** 가 서비스 변경 중인 경우 다음과 같이 리소스 해제를 요청합니다.

<표 9.15.1-4> Example of Prepare for Service Change Request
[출처: A344]

```
--> {
        "jsonrpc": "2.0",
        "method": "org.atsc.prepSvcChange",
        "params": {
                "fromService": "https://tv.atsc/OldService",
                "toService": "https://tv.atsc/NewService",
                "resources": ["AMP"]
        },
        "id": 22
}
```

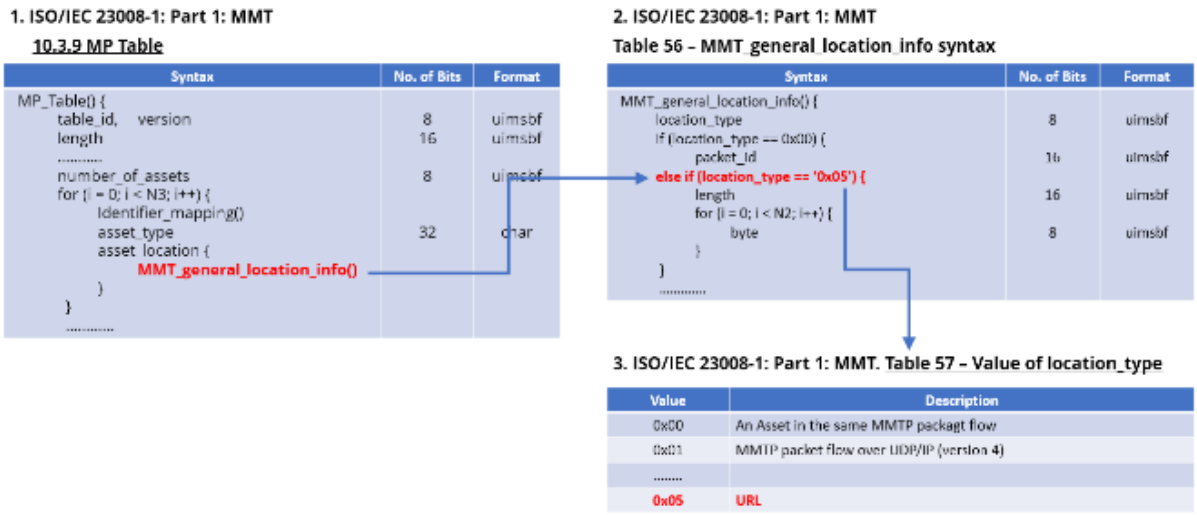
Broadcaster Application은 다음과 같이 응답할 수 있습니다.

<표 9.15.1-5> Example of Prepare for Service Change Response
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "result": {"resources": ["AMP"]},
        "id": 22
}
```

9.16 MMT AssetLink APIs

이 section의 API는 MPT의 *MMT_general_location_info*에 있는 *location_type* 값이 "0x05"로 표시되는 경우와 같이 **Broadcaster Application**이 *asset_location*가 URI인 MMT의 asset을 대체할 수 있는 메커니즘을 제공합니다([30]의 section 7.2.3에 정의). 그림 9.16-1은 MMT 신호 구조 간의 관계에 대한 다이어그램을 제공합니다



(그림 9.16-1) Relationship of MMT signaling tables.
[출처: A344]

두 가지 API가 정의되어 있습니다.

- **AssetLink Resolution Notification API**를 사용하면 **Receiver**가

RMP 가 MMT Asset 에 URI 인 *asset_location* 이 있음을 감지했음을 **Broadcaster Application** 에 알릴 수 있습니다. 이 Asset 은 이 section 에서 "target Asset"이라고 합니다. 알림은 **Broadcaster Application** 알림을 구독한 경우에만 발생합니다(section 9.2.1 참조). 그런 다음 **Broadcaster Application** 은 **AssetLink Resolved API** 를 사용하여 대체 MMT Asset 을 제공할 수 있습니다. MMT Asset 은 동일한 MMTP 무선 패킷 흐름과 MMT 신호 메시지 및 MPU 미디어 데이터가 포함된 단일 파일이 아닌 외부 또는 내부 URL 로 전달됩니다.

- **AssetLink Resolved API** 는 **Broadcaster Application** 에서 RMP 에서 target Asset 의 대안으로 사용할 replacement MMT Asset URL 또는 인라인 텍스트를 제공하는 데 사용됩니다. 리소스를 찾을 수 없거나 대체하기에는 너무 늦은 경우 **Receiver** 가 AssetLink Resolved 요청을 무시할 수 있습니다.

9.16.1 AssetLink Resolution Notification API

AssetLink Resolution Notification 은 RMP 가 URI 로 정의된 *asset_location* 가 있는 MPT 의 Asset 을 발견하면 **Receiver** 에 의해 현재 실행 중인 **Broadcaster Application** 에 발행될 것으로 예상됩니다[30]. **Receiver** 는 **Broadcaster Application** 에 알리지 않고 URL 체계 "https:"를 사용하여 모든 *asset_location* 를 해결할 수 있습니다. 다른 모든 URI 의 경우, **Broadcaster Application** 이 **AssetLink Resolution API** 을 구독하고 있는 경우, 수신기는 **Broadcaster Application** 에 사용 가능한 target Asset 을 알릴 것으로 예상됩니다(section 9.2.1 참조). **Broadcaster Application** 은 **AssetLink Resolved API**(섹션 9.1, 6.2 참조)를 사용하여 수신자에게 대상 자산과 관련하여 수행할 작업을 알려 응답해야 합니다.

AssetLink Resolution Notification 의 Semantics 는 표 9.16.1-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.notify-assetLinkResolution.json 에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.16.1-1> AssetLink Resolution Notification Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
method	1	string	"org.atsc.notify"
msgType	1	enum	"assetLinkResolution"
assetLink	1	string(uri)	대상 Asset 의 *asset_location*
assetType	0..1	string	대상 Asset 의 유형
assetId	1	string	대상 Asset 의 ID 값

assetLink – target Asset 에 제공된 *asset_location* 입니다.

assetType –이 값은 MP4REG (http://www.mp4ra.org) [30]의

10.3.9 MP table 에 등록된 네 문자 코드(“4CC”) 유형이어야 합니다.

assetId - 이 *assetLink* 와 연결된 Asset 의 값입니다.

AssetLink Resolution Notification 이 전송되면 **Broadcaster Application** 은 자체 기준을 사용하여 대체 콘텐츠를 제공하기로 결정하거나 제공하지 않을 수 있습니다. **Broadcaster Application** 은 section 9.16.2 에 설명된 **AssetLink Resolved API** 를 사용하여 **Receiver** 에게 결정을 알려야 합니다.

예를 들어, target Asse 을 감지하면 **Receiver** 는 target Asset 의 asset_location URI 가 "http://192.168.32.117:8182/s02gPkwZx14iO"이라고 가정하여 다음 메시지를 발행할 수 있습니다.

<표 9.16.1-2> Example of AssetLink Resolution Notification
[출처: A344]

```
<-- {
        "jsonrpc": "2.0",
        "method": "org.atsc.notify",
        "params": {
                "msgType": "assetLinkResolution",
                "assetLink": "http://192.168.32.117:8182/s02gPkwZx14iO",
                "assetType": "hevc",
                "assetId": "550e8400-e29b-41d4-a716-446655440000"
        }
}
```

9.16.2 AssetLink Resolved API

AssetLink Resolution Notification (section 9.17.1)이 수신된 후 **Broadcaster Application** 은 target Asset 과 관련하여 수행할 작업과 콘텐츠를 교체할지 아니면 아무 작업도 수행하지 않을지 결정합니다. 콘텐츠를 교체할 수 있다고 판단되면 기본 콘텐츠를 대체할 대체 콘텐츠를 결정하고 **AssetLink Resolved API** 를 사용하여 **Receiver** 에게 요청합니다. **Broadcaster Application** 은 요청에서 대체 콘텐츠의 URI 또는 실제 대체 콘텐츠를 제공합니다. **Receiver** 는 target Asset 을 대체하기 위해 대체 URI 또는 대체 콘텐츠 텍스트를 적절하게 처리해야 합니다. **Receiver** 가 기본 컨텐츠를 바꿀 수 없는 경우 API 응답에는 실패 이유를 나타내는 적절한 코드 및 설명이 포함되어야 합니다.

AssetLink Resolved Request 의 Semantics 는 표 9.16.2-1 에 정의되어 있으며 구문은 스키마 파일 org.atsc.assetLinkResolution-request.json 에 정의되어야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.16.2-1> AssetLink Resolved Request Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
id	1	integer	
method	1	string	"org.atsc.assetLinkResolution"

assetLink	1	string(uri)	대상 Asset의 *asset_location*
assetType	0..1	string	대상 Asset의 유형
assetId	1	string	대상 Asset의 ID 값
status	1	integer	Broadcaster Application의 작업 상태
replacementURL	one of X	string(uri)	대체 MMT 자산 파일의 URI
replacementText	one of X	string	대상 Asset을 대체하기 위해 Base64로 인코딩된 인라인 MMT Asset File

assetLink - target Asset에 제공된 *asset_location*입니다. 이 값은 섹션 9.16.1에 설명된 **AssetLink Resolution Notification**의 *assetLink* 매개 변수에 의해 제공됩니다.

assetType - 이것은 MP4REG(http://www.mp4ra.org)[30]에 등록된 4자 코드("4CC") 유형으로 설명됩니다.

assetId - *assetLink*와 연결된 target Asset의 값입니다.

status - 다음과 같이 알림에 대한 **Broadcaster Application**의 반응 상태입니다.

0: 요소가 교체되지 않았으며 **Receiver**는 자유롭게 조치를 취할 수 있습니다.

1: 요소가 교체되었습니다. **Receiver**는 요소를 교체하려고 시도합니다.

-1: 요소가 교체되지 않았습니다. **Receiver**는 이에 대해 행동하는 것이 금지되어 있습니다.

replacementURL - target Asset의 대안으로 제공되는 경우 이 문자열은 target Asset을 대체할 MMT Asset 파일을 참조하는 URI를 제공해야 합니다. 이 속성이 제공되는 경우 광대역을 통해 또는 **Application Context Cache**에서 대체 MMT Asset 파일을 참조하는지 여부에 관계없이 정규화된 URI여야 합니다. **Application Context Cache**의 경우 seciton 9.1.7에 설명된 대로 **Query Receiver Web Server URI API**를 사용하여 제공된 기본 URI를 사용하여 전체 URI를 구성할 수 있습니다.

replacementText - Base64 문자열로 인코딩된 대체 MMT Asset 파일을 제공합니다.

AssetLink Resolved Request을 받은 후 **Receiver**는 원래 target Asset이 요청된 assetURL URI 또는 제공된 요소로 성공적으로 교체되었음을 나타내는 응답을 할 수 있습니다. **Receiver**가 요청을 성공적으로 완료할 수 없는 경우 오류 코드가 반환될 것으로 예상됩니다. **Receiver**는 요소가 성공적으로 교체될 때까지 응답을 지연시키거나 오류 조건으로 즉시 응답하도록 선택할 수 있습니다. **Broadcaster Application**은 **AssetLink Resolved Response**를 처리할 때 타이밍의 차이를 고려해야 합니다. **AssetLink Resolved Response**의 Semantics는 표 9.16.2-2에 정의되어 있으며 구문은 스키마 파일 org.atsc.assetLinkResolution-response.json에 정의된 대로여야 합니다. 매개 변수의 추가 Semantics 정의는 표를 따릅니다.

<표 9.16.2-2> AssetLink Resolved Response Semantics
[출처: A344]

Property Name	Use	Data Type	Short Description
Jsonrpc	1	string	“2.0”
Id	1	Integer	요청 ID 값과 일치
result	oneOf X		요청 성공 시 개체가 비어 있습니다. 실패하면 오류 구조가 반환
error	oneOf X		Section 8.3.3 참조

Section B.3.1 의 오류 외에도 표 8.3.3-2 의 다음 오류가 반환될 수 있습니다.

- -4: 요청된 콘텐츠를 찾을 수 없습니다. 예를 들어 잘못된 URL 입니다.
- -30: 기본 컨텐츠를 교체하기 위해 요청이 제때 수신되지 않았습니다.
- -34: 제공된 *assetLink*, *assetType* 및/또는 *assetId*를 찾을 수 없습니다.
- -35: 제공된 요소 내용이 유효하지 않습니다.

예를 들어, **Broadcaster Application** 은 target Asset URI 가 "http://192.168.32.117:8182/s02gPkwZx14iO"이라고 가정하여 다음 요청을 제출합니다.

<표 9.16.2-3> Example of AssetLink Resolved Request 1
[출처: A344]

```
--> {
       "jsonrpc": "2.0",
       "method": "org.atsc.assetLinkResolution",
       "params": {
              "assetLink": "http://192.168.32.117:8182/s02gPkwZx14iO",
              "assetType": "hevc",
              "assetId": "550e8400-e29b-41d4-a716-446655440000",
              "status": 1,
              "replacementURL": "/replacement-ad.mpt",
       },
       "id": 104
}
```

Receiver 가 target Asset 을 URI 에서 참조하는 대체 MMT Asset 파일로 대체할 수 있는 경우 아래와 같이 성공적인 결과로 응답합니다.

<표 9.16.2-4> Example of AssetLink Resolved Response 1-1
[출처: A344]

```
<-- {
       "jsonrpc": "2.0",
       "result": {},
       "id": 104
}
```

또는 **Receiver** 가 **Broadcaster Application** 의 요청에 따라 URI 에서 참조하는 대체 MMT Asset 파일을 사용할 수 없는 경우 실패한 요청의 이유를 나타내는 처분(disposition)을 반환합니다.

<표 9.16.2-5> Example of AssetLink Resolved Response 1-2

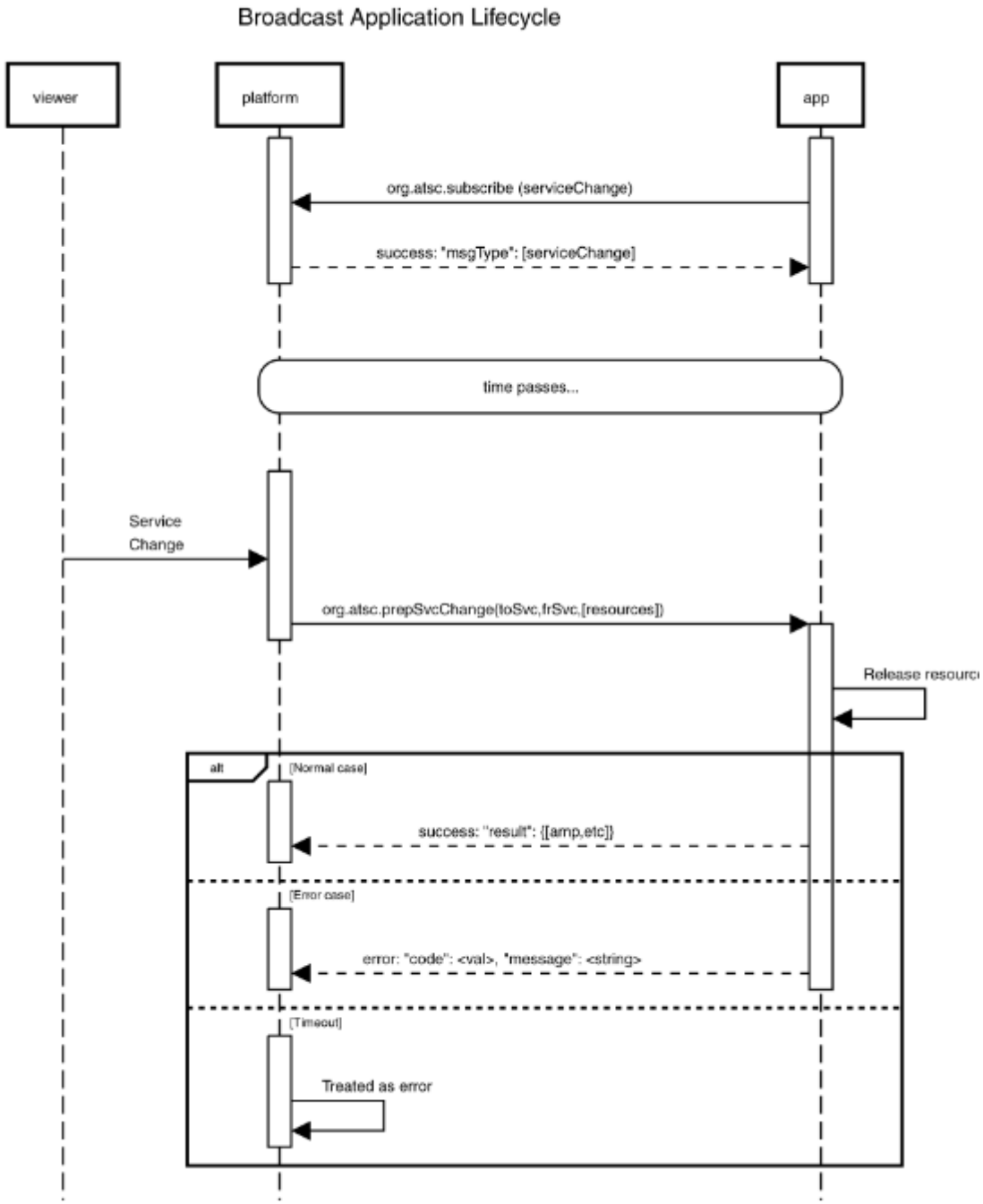
[출처: A344]

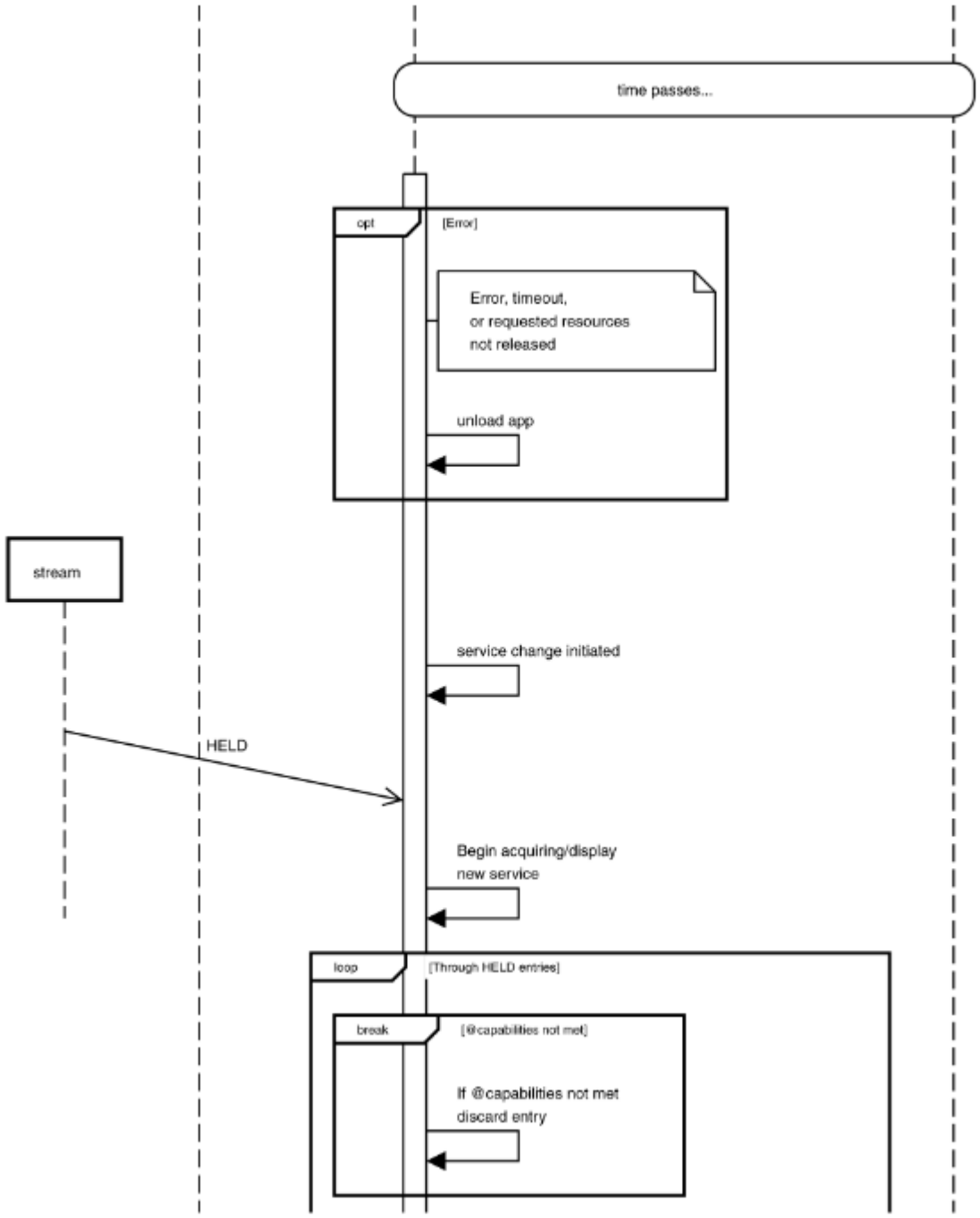
```
<-- {
        "jsonrpc": "2.0",
        "error": {
                "code": -30,
                "message": "Too late"
        },
        "id": 104
}
```

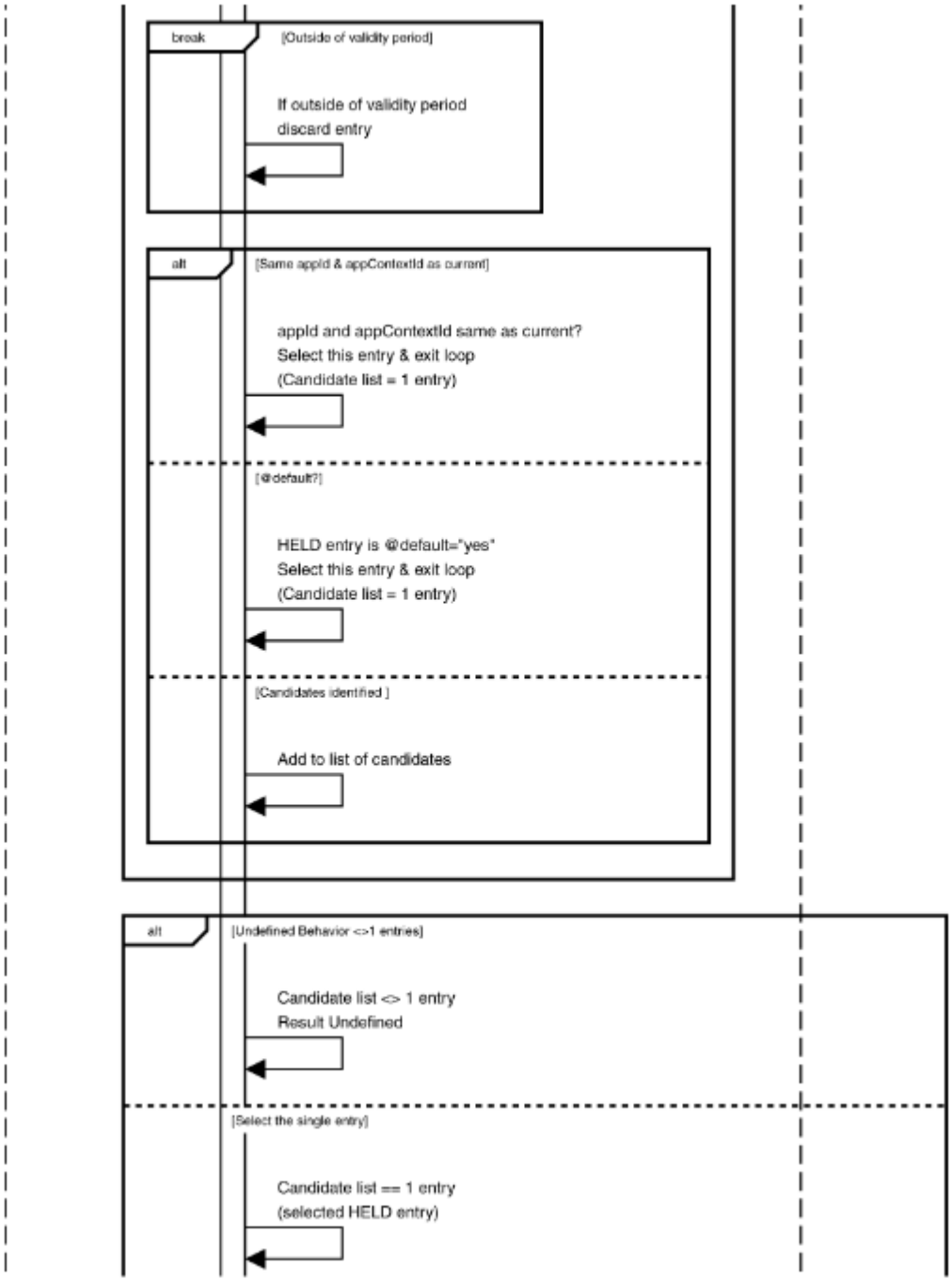
부 속 서 A

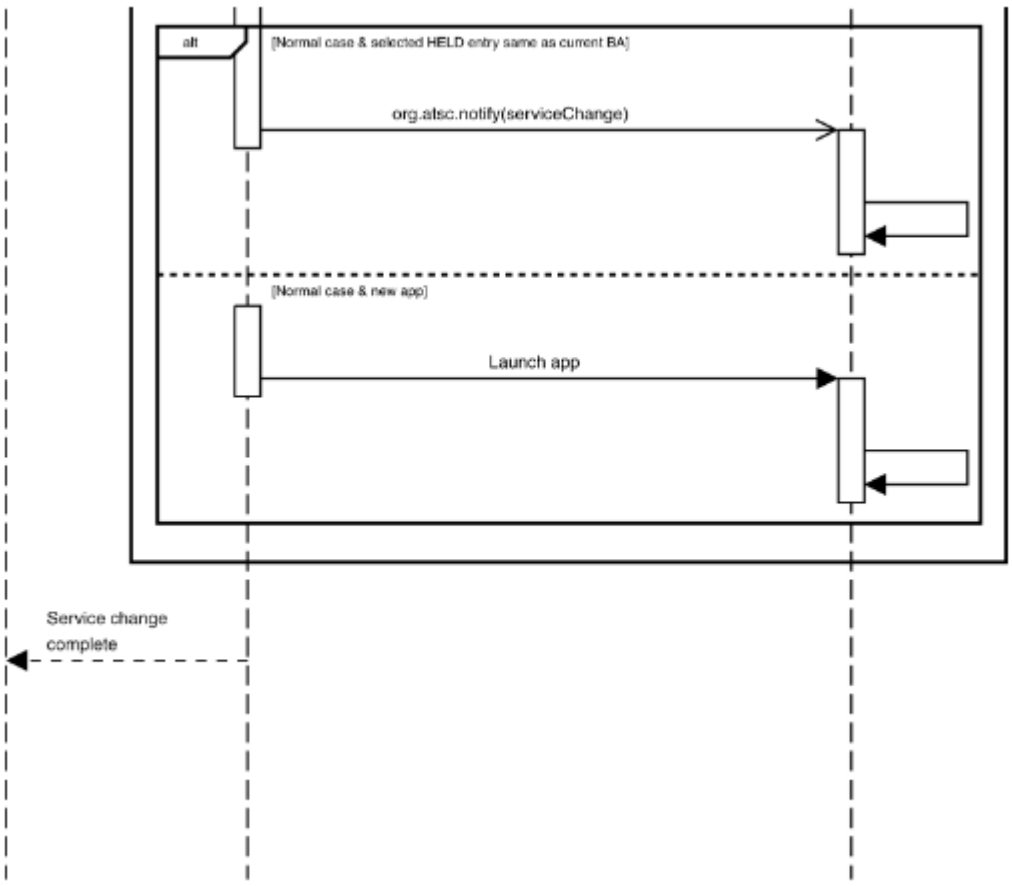
(본 부속서는 표준 내용의 일부임)

Application Lifecycle Sequence Diagram




time passes...
opt
[Error]
Error, timeout,
or requested resources
not released
unload app
stream
service change initiated
HELD
Begin acquiring/display
new service
loop
[Through HELD entries]
break
[@capabilities not met]
If @capabilities not met
discard entry


break
[Outside of validity period]
If outside of validity period
discard entry
alt
[Same appId & appContextId as current]
appId and appContextId same as current?
Select this entry & exit loop
(Candidate list = 1 entry)
[@default?]
HELD entry is @default="yes"
Select this entry & exit loop
(Candidate list = 1 entry)
[Candidates identified]
Add to list of candidates
alt
[Undefined Behavior <>1 entries]
Candidate list <> 1 entry
Result Undefined
[Select the single entry]
Candidate list == 1 entry
(selected HELD entry)


alt
[Normal case & selected HELD entry same as current BA]
org.atsc.notify(serviceChange)
[Normal case & new app]
Launch app
Service change
complete

부 속 서 B

(본 부속서는 표준 내용의 일부임)

JSON-RPC 2.0 Specification

B.1 Overview

JSON-RPC는 상태 비저장, 경량 원격 프로시저 호출(RPC) 프로토콜입니다. 주로 이 사양은 여러 데이터 구조와 해당 처리에 대한 규칙을 정의합니다. 개념이 동일한 프로세스 내에서, 소켓을 통해, http를 통해 또는 다양한 메시지 전달 환경에서 사용될 수 있다는 점에서 전송에 구애받지 않습니다. 데이터 형식으로 JSON(RFC 4627)을 사용합니다.

B.2 Request object

rpc 호출은 Request 개체를 서버에 보내는 것으로 표시됩니다. Request 개체에는 다음과 같은 멤버가 있습니다.

jsonrpc - JSON-RPC 프로토콜의 버전을 지정하는 문자열입니다. 정확히 '2.0'이어야 합니다(MUST).

method - 호출할 메서드의 이름을 포함하는 문자열입니다. 단어 rpc로 시작하고 그 뒤에 마침표 문자(U+002E 또는 ASCII 46)가 오는 메서드 이름은 *rpc-internal* 메서드 및 확장용으로 예약되어 있으며 다른 용도로는 사용해서는 안 됩니다.

params- 메서드 호출 중에 사용할 매개 변수 값을 보유하는 구조화된 값입니다. 이 멤버는 생략될 수 있습니다(MAY).

Id - 클라이언트에서 설정한 식별자로, 포함된 경우 String, Number 또는 NULL 값을 포함해야 합니다. 포함되지 않은 경우 알림으로 간주됩니다. 값은 일반적으로 Null 1이 아니어야 하며 숫자는 소수 부분 2를 포함해서는 안 됩니다(SHOULD NOT).

서버는 포함된 경우 Response 개체에 동일한 값으로 응답해야 합니다. 이 멤버는 두 개체 간의 컨텍스트의 상관 관계를 지정하는 데 사용됩니다.

B.2.1 Notification

알림은 "id" 멤버가 없는 요청 개체입니다. 알림인 Request 개체는 해당 Response 개체에 대한 클라이언트의 관심이 없음을 나타내므로 Response 개체를 클라이언트에 반환할 필요가 없습니다. 서버는 일괄 처리 요청 내에 있는 알림을 포함하여 알림에 회신해서는 안 됩니다.

알림은 반환할 Response 개체가 없으므로 정의에 따라 확인할 수 없습니다. 따라서 클라이언트는 오류(예: "잘못된 매개변수", "내부 오류")를 인식하지 못합니다.

B.2.2 Parameter Structures

있는 경우 rpc 호출에 대한 매개 변수는 구조화된 값으로 제공되어야 합니다. 배열을 통한 위치별 또는 개체를 통한 이름별 지정입니다.

- by-position: params 는 서버 예상 순서의 값을 포함하는 배열이어야 합니다(MUST).
- by-name: params 는 서버 예상 매개 변수 이름과 일치하는 멤버 이름을 가진 개체여야 합니다. 예상 이름이 없으면 오류가 생성될 수 있습니다(MAY). 이름은 대/소문자를 포함하여 메서드의 예상 매개 변수와 정확히 일치해야 합니다(MUST).

B.3 Response object

rpc 호출이 이루어지면 서버는 알림의 경우를 제외하고 응답으로 응답해야 합니다. 응답은 다음 멤버가 있는 단일 JSON 개체로 표현됩니다.

jsonrpc - JSON-RPC 프로토콜의 버전을 지정하는 문자열입니다. 정확히 '2.0'이어야 합니다(MUST).

result - 이 멤버는 성공 시 필수입니다. 메서드를 호출하는 동안 오류가 발생한 경우 이 멤버는 존재하지 않아야 합니다. 이 멤버의 값은 서버에서 호출된 메서드에 의해 결정됩니다.

error - 이 멤버는 오류 시 필수입니다. 호출 중에 트리거된 오류가 없는 경우 이 멤버는 존재하지 않아야 합니다(MUST NOT). 이 멤버의 값은 섹션 B.3.1 에 정의된 Object 여야 합니다(MUST).

id - 이 멤버는 필수입니다. 요청 개체의 id 멤버 값과 동일해야 합니다. Request 객체에서 id 를 감지하는 데 오류가 있는 경우(예: Parse error/Invalid Request) Null 이어야 합니다(MUST). 결과 멤버 또는 오류 멤버는 포함되어야 하지만 두 멤버 모두 포함되어서는 안 됩니다.

B.3.1 Error object

rpc 호출에 오류가 발생하면 Response Object 는 다음 멤버가 있는 Object 값의 error 멤버를 포함해야 합니다(MUST).

code - 발생한 오류 유형을 나타내는 숫자입니다. 이것은 정수여야 합니다.

message - 오류에 대한 간단한 설명을 제공하는 문자열입니다. 메시지는 간결한 한 문장으로 제한되어야 합니다(SHOULD).

data - 오류에 대한 추가 정보를 포함하는 기본 또는 구조화된 값입니다. 생략할 수 있습니다. 이 멤버의 값은 서버에 의해 정의됩니다(예: 자세한 오류 정보, 중첩된 오류 등). -32768 에서 -32000 까지의 오류 코드는 미리 정의된 오류용으로 예약되어 있습니다. 이 범위 내에 있지만 아래에 명시적으로 정의되지 않은 모든 코드는 나중에 사용할 수 있도록 예약되어 있습니다.

오류 코드는 다음 URL 에서 XML-RPC 에 대해 제안된 것과 거의 동일합니다. http://xmlrpc-epi.sourceforge.net/specs/rfc.fault_codes.php

code	message	meaning
-32700	Parse error	서버에서 잘못된 JSON 을 수신했습니다. JSON 텍스트를 구문 분석하는 동안 서버에서 오류가 발생했습니다.
-32600	Invalid Request	전송된 JSON 이 유효한 요청 객체가 아닙니다.
-32601	Method not found	메서드가 존재하지 않거나 사용할 수 없습니다.
-32602	Invalid params	잘못된 메서드 매개 변수
-32603	Internal error	내부 JSON-RPC 오류
-32000 to -32099	Server error	Reserved for implementation-defined server-errors

나머지 공간은 애플리케이션 정의 오류에 사용할 수 있습니다.

B.4 Batch

동시에 여러 Request 개체를 보내기 위해 클라이언트는 Request 개체로 채워진 배열을 보낼 수 있습니다(MAY).

서버는 모든 일괄 처리 요청 개체가 처리된 후 해당 응답 개체가 포함된 배열로 응답해야 합니다. Response 개체는 알림에 대한 Response 개체가 없어야 한다는 점을 제외하고는 각 Request 개체에 대해 존재해야 합니다(SHOULD). 서버는 일괄 rpc 호출을 동시 작업 집합으로 처리하여 임의의 순서와 병렬 처리 폭으로 처리할 수 있습니다(MAY).

일괄 처리 호출에서 반환되는 Response 개체는 배열 내에서 임의의 순서로 반환될 수 있습니다(MAY). 클라이언트는 각 개체 내의 id 멤버를 기반으로 Request 개체 집합과 결과 Response 개체 집합 간의 컨텍스트를 일치시켜야 합니다(SHOULD).

일괄 처리 rpc 호출 자체가 유효한 JSON 또는 하나 이상의 값이 있는 배열로 인식되지 않는 경우 서버의 응답은 단일 응답 개체여야 합니다. 클라이언트로 보낼 Response 배열 내에 포함된 Response 객체가 없는 경우 서버는 빈 배열을 반환해서는 안 되며 all 에서 아무 것도 반환해서는 안 됩니다.