Layered Audio Delivery for Efficient and Resilient Multi-Station Internet Radio

A Case Study of RadioTEDU

Arda Akgül
Broadcast Coordinator of RadioTEDU
arda.akgul@tedu.edu.tr

29 August 2026

Abstract

Internet-radio services must work across many listening conditions while keeping the programme consistent and the service running. Bandwidth, device support, audio quality, loudness, and recovery from failure are closely connected; they cannot be handled as separate encoder settings. This paper presents a design-science case study of RadioTEDU, a multi-channel university radio platform, where "multi-channel" means several editorial station services rather than multichannel audio. The active system has fifteen public outputs from seven editorial stations. It uses a shared signed-16-bit, 48 kHz stereo PCM interface, several codec tiers, common loudness processing, controlled transitions, and separately supervised output workers. Six music stations offer a 192 kbit/s AAC-LC primary stream and a 64 kbit/s HE-AAC v2 efficient stream. Classical and Jazz also offer Ogg-encapsulated FLAC, while Situation Room uses one AAC-LC stream. The study asks how codec tiers can serve different purposes without creating separate schedules, how programme state can remain independent from output-worker failures, and how loudness standards can guide local settings without being confused with them. Public endpoint checks made on 25 August 2026 found all fifteen expected mounts online at the time of the test. Seven sequential 60-second decoded-stream measurements are reported as spot observations, not as programme- or service-loudness compliance tests. Five design lessons are identified: treat codecs as representations of one station; apply shared processing before the outputs split; keep worker recovery separate from editorial state; limit branch state and define recovery clearly; and state the signal boundary for every fidelity, loudness, and peak claim. In this paper, resilience refers to the separation of editorial state from output-worker recovery; no long-term availability rate is claimed.

Keywords: internet radio; AAC-LC; HE-AAC v2; FLAC; broadcast automation; loudness normalization; EBU R 128; ITU-R BS.1770; resilient streaming; design science

1 Introduction

For listeners, internet radio appears simple: they choose a station and expect continuous audio. Behind the player, however, the system must handle mobile and fixed networks, browsers, operating systems, casting devices, vehicles, and hi-fi equipment. It must also manage different source files, programme transitions, loudness changes, encoders, server connections, metadata, storage delays,

and component failures. Improving one part of the system can create problems elsewhere. Higher bitrates use more network capacity, extra codecs require more testing and monitoring, larger buffers add delay, and separate processing paths may drift away from the shared programme.

Research shows that perceived audio quality depends on the codec, bitrate, programme material, and listening situation. HE-AAC adds Spectral Band Replication to an AAC core, while HE-AAC v2 also uses Parametric Stereo to improve efficiency at low bitrates (den Brinker et al., 2009). Broadcast studies show that bitrate and perceived quality involve trade-offs rather than one universal quality threshold (Berg et al., 2013; Gilski & Stefański, 2017). At higher bitrates, controlled tests have found that AAC-LC can come close to a lossless reference under specific conditions (Grivcova et al., 2018). These findings help explain individual formats, but a working radio platform must also decide how several formats can share one schedule and one recovery model.

RadioTEDU addresses this problem with a layered delivery model. Its seven active stations produce fifteen local public outputs. Every station has a 192 kbit/s AAC-LC primary stream. Six music stations also have a 64 kbit/s HE-AAC v2 stream, and Classical and Jazz add lossless Ogg-encapsulated FLAC streams. All representations follow the same programme timeline instead of scheduling content independently. The system completes transitions and applies its shared loudness policy in PCM before creating the codec-specific outputs. Output workers are supervised separately from the schedule, so restarting a worker does not become a new editorial decision.

The platform is studied as a design artifact: knowledge comes from examining how an engineered system is built and used, rather than only from a behavioral experiment (Hevner et al., 2004). Following software case-study guidance, the paper examines a current system in its real operating context and links its claims to a dated case description, public endpoint observations, and standards (Runeson & Höst, 2009). The focus is therefore the active local delivery architecture and its operating policy.

Three research questions guide the analysis:

RQ1: How can a multi-channel internet-radio system structure codec tiers so that network efficiency, primary delivery, and lossless fidelity are treated as distinct roles while preserving one editorial service?

RQ2: How can a shared production domain and independently supervised output branches separate programme continuity from codec, transport, and connection failure domains?

RQ3: How can loudness standards guide the delivery system while keeping measurement rules, operational guidance, and local broadcaster policy separate?

The paper explains how RadioTEDU assigns roles to codec tiers, separates programme state from output-worker failures, connects loudness standards to local policy, and compares the system description with public endpoint observations. It also presents design lessons for other multi-channel internet-radio systems. The claims are architectural and analytical; the paper does not argue that one bitrate always sounds better than another.

2 Related Work

2.1 Perceptual audio coding and broadcast quality

HE-AAC improves coding efficiency by combining a waveform core with parametric tools. Spectral Band Replication rebuilds high-frequency information from compact side data, while HE-AAC v2 adds Parametric Stereo to represent parts of the stereo image more efficiently at low bitrates (den Brinker et al., 2009). This explains why a low-bitrate tier can serve a different purpose from a higher-bitrate AAC-LC primary. The profiles differ not only in bitrate but also in how they use the available data.

Broadcast research also shows that codec choices must be made in context. Berg et al. (2013) studied realistic FM and DAB+ conditions and found that quality depends strongly on programme material and bitrate. Gilski and Stefański (2017) used both listening tests and objective methods to compare DAB+ quality, linking bandwidth limits to listener experience. Objective tools such as ViSQOLAudio can support listening tests for low-bitrate music, although their value still depends on the use case (Hines et al., 2015). At a higher rate, Grivcova et al. (2018) compared 320 kbit/s AAC-LC with an uncompressed reference and found no audible difference under the conditions of that test. Together, these studies suggest that there is no single "best codec" for every situation.

This case study does not repeat those listening experiments. Instead, it uses their main lesson as a design principle: format choice depends on context, and a station can offer several representations without splitting its editorial service.

2.2 Loudness measurement and distribution

ITU-R BS.1770-5 defines methods for measuring programme loudness and true peak (International Telecommunication Union, 2023). EBU R 128 uses these methods with a programme loudness target of -23 LUFS and related measures (European Broadcasting Union, 2023e). EBU Tech 3344 then addresses distribution and playback. It explains that later processing and lossy coding can change peak levels, so enough headroom must be planned (European Broadcasting Union, 2016).

These documents play different roles. A measurement standard explains how to measure a signal, an operational recommendation gives a target or practice, and a broadcaster decides where to apply it. If these roles are mixed together, misleading claims may follow, such as "ITU requires -23 LUFS for internet radio" or "a sample limiter guarantees that decoded audio stays below -1 dBTP." Neither statement correctly describes the standards.

2.3 Compatibility, transport, and codec scope

Codec-family support is not equivalent to endpoint interoperability. Android's platform documentation lists AAC-LC, HE-AAC v1/v2, and ADTS raw AAC among supported audio combinations, while its platform-level FLAC row enumerates native FLAC, MP4, and Matroska containers (Google, 2025). Android Media3's ExoPlayer documentation is more specific for progressive playback and lists both ADTS AAC and Ogg containing FLAC as supported container/sample-format combinations (Google, 2026). Google Cast documents LC-AAC, HE-AAC, and FLAC codec capability and separately defines supported media-type/container combinations (Google, 2024).

These documents show the general range of supported formats, but they do not prove that every client can play RadioTEDU's public Icecast streams. The paper therefore does not claim universal endpoint compatibility. Instead, it states the transport details directly: AAC is sent in ADTS framing over continuous HTTP source streaming, while the lossless stream uses FLAC in an Ogg container. Here, "primary" means the service's default stream, not a stream proven to work on every device.

FLAC is a lossless format: its decoded samples reproduce the PCM information given to the encoder (van Beurden & Weaver, 2024). Opus is another flexible, open codec that supports many bitrates and applications (Valin et al., 2012). RadioTEDU currently uses AAC-LC, HE-AAC v2, and FLAC as an operational choice, not because other codecs are necessarily inferior.

2.4 Design-science and case study approach

Design-science research studies useful artifacts and the knowledge gained by building and using them (Hevner et al., 2004). Software case studies are suitable for examining a current engineering system in its real context, especially when claims can be linked to clear evidence (Runeson & Höst, 2009). This approach fits the present paper because its main contribution is architectural. The question is not whether listeners prefer one codec in a laboratory test, but how several codec roles, shared audio processing, and output supervision can work together in one operating system.

3 Research Design and Scope

3.1 Case definition

The case covers the active local RadioTEDU delivery system and the public endpoint observations recorded on 25 August 2026 in Europe/Istanbul (UTC+03:00). It includes seven editorial stations and fifteen local public audio outputs. Old codec profiles, disabled stations, inactive transport experiments, and external sources are excluded because they are not part of the active local system studied here.

The seven stations are Classical, Lo-Fi, Pop, Jazz, Rock, Energize, and Situation Room. Each one has a single schedule and programme clock. A public output is a technical version of a station, defined by its codec profile, bitrate, framing or container, mount, metadata policy, and worker lifecycle. The difference between a *station* and an *output* is central to the analysis.

3.2 Evidence and analysis

The analysis uses four types of evidence: the active service configuration, the documented system architecture, public endpoint observations, and published standards or research. Case-specific settings are reported as settings of this service, while standards provide technical or normative context. Calculated values come from the equations shown in the paper.

The public verification used HTTP HEAD requests to check whether each expected mount answered and to record the returned media type and advertised `icy-br` value. Seven primary streams were also decoded and measured one after another for 60 s with FFmpeg's `ebur128` filter. These checks support only the claims listed in Table 1; they do not prove long-term availability, universal client support, an exact transport bitrate, or full loudness compliance.

Table 1: Evidence scope used in the case analysis

Claim family	Evidence used	Supported interpretation
Active output matrix	Service configuration and public HEAD observations on 25 August 2026	Fifteen expected mounts answered at the time of the check: seven primary, six efficient, and two lossless outputs.
Codec and transport roles	Configured codec profiles, framing/container definitions, and returned HTTP media types	The paper reports the intended representation roles and observed headers; it does not claim universal endpoint compatibility.
Loudness observations	Seven sequential 60-second decoded-stream windows measured with `ebur128=peak=true`	The values describe the content present in those windows; they are not programme- or service-loudness compliance results.
Failure architecture	Separation of programme state and output-worker lifecycle in the case design	The paper supports a structural recovery claim, not a measured availability percentage or proof of complete branch non-interference.

3.3 Design criteria

The architecture is assessed using five criteria drawn from the practical needs of the service:

1. **One editorial programme:** all outputs of a station keep the same programme identity and order.
2. **Several delivery options:** different bandwidth and fidelity roles are offered without copying the scheduling logic.
3. **Consistent processing:** shared transitions and audio processing take place at one controlled point.
4. **Separate failure boundaries:** output-worker failures remain downstream of the station's programme state.
5. **Clear use of standards:** measurement standards, distribution guidance, and local policy remain separate.

These are properties of the system design and do not depend on a listener-preference experiment.

4 Layered System Architecture

4.1 Station, programme bus, and output

Figure 1 summarizes the system. The schedule and transition logic create one programme. Shared processing then produces a stable PCM interface before each codec is applied. The outputs therefore split only *after* the programme has been fully prepared.

The programme-to-encoder interface is defined at both sides of the boundary. The producer writes raw signed 16-bit little-endian PCM (`s16le`) at 48,000 samples per second with two-channel stereo. An FFmpeg-equivalent producer-side contract is:

```
...,aresample=48000,
aformat=sample_fmts=s16:sample_rates=48000:channel_layouts=stereo
-f s16le -ar 48000 -ac 2 pipe:1
```

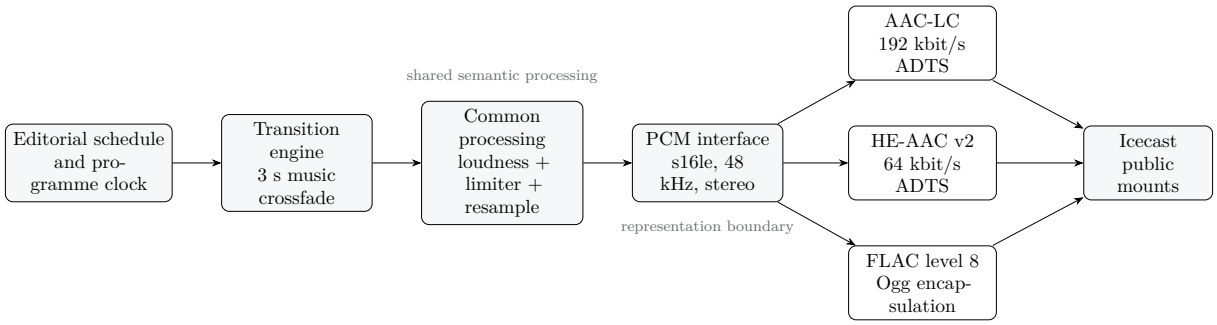

Figure 1: RadioTEDU layered programme path and explicit PCM/representation boundary.

Each encoder worker uses the matching raw-input declaration:

```
-f s16le -ar 48000 -ac 2 -i <PROGRAM_PCM>
```

The first fragment describes the format written by the producer; the second describes how the same bytes are read by a worker. This distinction is important because a consumer-side input declaration alone does not prove the producer format. The command fragments describe the interface contract rather than a complete launch command. The present study does not make a separate claim about the dithering method used during conversion to 16-bit PCM.

4.2 Encoder and transport contract

Table 2 lists the settings used at the encoding boundary. The AAC workers are configured to use `libfdk_aac`. FFmpeg defines `aac_low`, `aac_he`, and `aac_he_v2` as separate profiles of this encoder (FFmpeg Project, n.d.-a). The public checks in this study recorded HTTP headers; they did not independently decode all AAC signalling details. The paper therefore separates the configured profile from what was directly observed at the public endpoint.

Table 2: PCM, encoder, framing, and media-type contract

Representation	**Configured encoder/profile**	**Framing/ container**	**Case contract and public observation**
Primary	`libfdk_aac`, `aac_low`	ADTS	192 kbit/s nominal setting; 48 kHz stereo PCM input; public mounts returned `audio/aac` and `icy-br:192`.
Efficient	`libfdk_aac`, `aac_he_v2`	ADTS	64 kbit/s nominal setting; public mounts returned `audio/aac` and `icy-br:64`.
Lossless	FFmpeg `flac`, compression level 8	Ogg	Variable bitrate; lossless preservation begins at the processed PCM presented to the encoder. The two public mounts returned `application/ogg` and advertised `icy-br:320`.

For this reason, the general label “AAC” is not detailed enough. The encoder backend, configured profile, bitrate, PCM input, and ADTS framing all form part of the output definition. The `icy-br` field is an advertised server value; it is not treated as an independent bitrate measurement. In particular, the value 320 on the FLAC mounts does not make FLAC a fixed-rate 320 kbit/s stream.

RadioTEDU leaves the `libfdk_aac signaling` option at its FFmpeg-defined default (FFmpeg Project, n.d.-a). Because the emitted SBR/PS signalling was not independently inspected,

HE-AAC v2 is reported as the configured encoder profile rather than as a bitstream-verified property.

On 25 August 2026, both audio-only Ogg FLAC mounts returned `application/ogg`. RFC 5334 recommends `audio/ogg` when an Ogg stream mainly contains audio and requires an Ogg Skeleton logical stream when `application/ogg` is used (Goncalves et al., 2008). No Skeleton verification was performed in this study. The observed response therefore could not be confirmed as conforming to RFC 5334. In the absence of verified Ogg Skeleton information, `audio/ogg` is the media type indicated by the RFC for these audio-dominant streams.

4.3 Active output matrix

Table 3 shows the active local distribution matrix.

Table 3: Active local RadioTEDU output matrix

Station	Primary representation	Efficient representation	Lossless representation
Classical	AAC-LC 192 kbit/s, `/classic`	HE-AAC v2 64 kbit/s, `/classic-low`	Ogg-encapsulated FLAC, `/classic-flac`
Lo-Fi	AAC-LC 192 kbit/s, `/lofi`	HE-AAC v2 64 kbit/s, `/lofi-low`	—
Pop	AAC-LC 192 kbit/s, `/radio`	HE-AAC v2 64 kbit/s, `/radio-low`	—
Jazz	AAC-LC 192 kbit/s, `/cazz`	HE-AAC v2 64 kbit/s, `/cazz-low`	Ogg-encapsulated FLAC, `/cazz-flac`
Rock	AAC-LC 192 kbit/s, `/rock`	HE-AAC v2 64 kbit/s, `/rock-low`	—
Energize	AAC-LC 192 kbit/s, `/energize`	HE-AAC v2 64 kbit/s, `/energize-low`	—
Situation Room	AAC-LC 192 kbit/s, `/situation`	—	—
Total	**7**	**6**	**2**

The matrix creates fifteen public outputs without creating fifteen separate schedules. Classical and Jazz each offer three representations, four other music stations offer two, and Situation Room offers one. In every case, the outputs remain versions of one editorial station.

4.4 Public endpoint verification

Table 4 records the successful HEAD observations made on 25 August 2026. The rows list only active public outputs in the case matrix.

Table 4: Public mount observations on 25 August 2026

Mount	Role	Status	Media type	icy-br
`/radio`	AAC-LC primary	OK	`audio/aac`	192
`/radio-low`	HE-AAC v2 efficient	OK	`audio/aac`	64
`/classic`	AAC-LC primary	OK	`audio/aac`	192
`/classic-low`	HE-AAC v2 efficient	OK	`audio/aac`	64
`/classic-flac`	Ogg FLAC lossless	OK	`application/ogg`	320
`/cazz`	AAC-LC primary	OK	`audio/aac`	192
`/cazz-low`	HE-AAC v2 efficient	OK	`audio/aac`	64

Mount	Role	Status	Media type	icy-br
`/cazz-flac`	Ogg FLAC lossless	OK	`application/ogg`	320
`/lofi`	AAC-LC primary	OK	`audio/aac`	192
`/lofi-low`	HE-AAC v2 efficient	OK	`audio/aac`	64
`/energize`	AAC-LC primary	OK	`audio/aac`	192
`/energize-low`	HE-AAC v2 efficient	OK	`audio/aac`	64
`/situation`	AAC-LC primary	OK	`audio/aac`	192
`/rock`	AAC-LC primary	OK	`audio/aac`	192
`/rock-low`	HE-AAC v2 efficient	OK	`audio/aac`	64

A successful HEAD request shows that the mount answered at that time and reports the returned headers. It does not prove continuous availability, actual average bitrate, decoder compatibility, or the exact codec profile. Those questions require longer observation or a stream-level probe.

5 Codec Roles and Capacity Logic

5.1 Different codecs for different roles

Each active codec has a clear role in the system.

Table 5: Codec roles in the active delivery policy

Role	Codec	Nominal rate	Design rationale
Primary access	AAC-LC	192 kbit/s	General-purpose representation emphasizing established decoder support and a comparatively generous perceptual-coding budget.
Efficient access	HE-AAC v2	64 kbit/s	Reduced network payload for constrained links while retaining a stereo music service through SBR/PS-based coding tools.
Lossless fidelity	Ogg-encapsulated FLAC	Variable	Exact preservation of the processed programme PCM for Classical and Jazz, where a lossless representation is an explicit service feature.

AAC-LC at 192 kbit/s is the default public stream. It is not an archival master, and the paper does not claim that it is transparent for every signal, listener, or device. Its role is to provide a general-purpose primary stream. Android and Cast documentation show AAC support in relevant media contexts, but these lists do not prove that every client can play the continuous Icecast stream (Google, 2024, 2025).

HE-AAC v2 at 64 kbit/s is a separate delivery option, not a reduced editorial service. It combines an AAC core with SBR and Parametric Stereo to use limited bitrate more efficiently (den Brinker et al., 2009). The chosen rate is a local policy for efficient access, not a universal best setting.

Ogg-encapsulated FLAC serves a third role. RFC 9639 defines FLAC as lossless because the decoded data reproduce the PCM information given to the encoder (van Beurden & Weaver, 2024). In RadioTEDU, this lossless boundary begins *after* shared programme processing. A FLAC listener receives an exact copy of the processed broadcast PCM, not necessarily an exact copy of

the original source file. Loudness changes, transitions, and resampling may already have changed the source before FLAC encoding.

5.2 Payload arithmetic

For a nominal constant bitrate R in kbit/s and listening duration h in hours, the approximate audio payload is

$$D_{\mathrm{MB}} = \frac{R \times 3600h}{8000}. \tag{1}$$

The calculation uses decimal SI units (1 MB = 10^6 bytes; 1 GB = 10^9 bytes). It does not include HTTP headers, framing, metadata, TCP/IP overhead, retransmissions, or connection setup. Even so, it gives a useful basic comparison between the two lossy tiers.

Table 6: Nominal audio payload for one continuous listener

Representation	**Per hour**	**Per 8 hours**	**Per 30 days**
HE-AAC v2, 64 kbit/s	28.8 MB	230.4 MB	20.736 GB
AAC-LC, 192 kbit/s	86.4 MB	691.2 MB	62.208 GB

At the stated bitrates, the efficient tier uses one third as much audio data as the primary tier. This follows directly from the configured rates and says nothing about perceived quality. FLAC is not included in the fixed-rate comparison because its bitrate changes with the content and the amount of repeated information in the signal.

For concurrent listeners, the audio-payload baseline can be written as

$$B_{\mathrm{total}} = \sum_i N_i R_i, \tag{2}$$

where N_i is the number of simultaneous listeners using representation i. This equation helps with capacity planning because the total now depends on the mix of streams selected by listeners, rather than on one assumed bitrate.

6 Loudness and Signal Control

6.1 Separating standards from local policy

Figure 2 shows how the case separates measurement, guidance, and local settings.

ITU-R BS.1770-5 provides the main measurement method, while EBU Tech 3341 defines EBU-mode meters and their momentary, short-term, and integrated views (European Broadcasting Union, 2023a; International Telecommunication Union, 2023). EBU R 128 recommends programme-loudness normalisation, and its streaming and radio supplements add context for those services (European Broadcasting Union, 2023b, 2023c, 2023d, 2023e). EBU Tech 3344 addresses distribution and codec headroom (European Broadcasting Union, 2016).

In RadioTEDU, −23 LUFS is the *configured integrated-loudness target for the shared processing stage*. It does not mean that every track or every short part of the continuous stream must measure exactly −23 LUFS. Integrated loudness needs a defined start and stop boundary. A

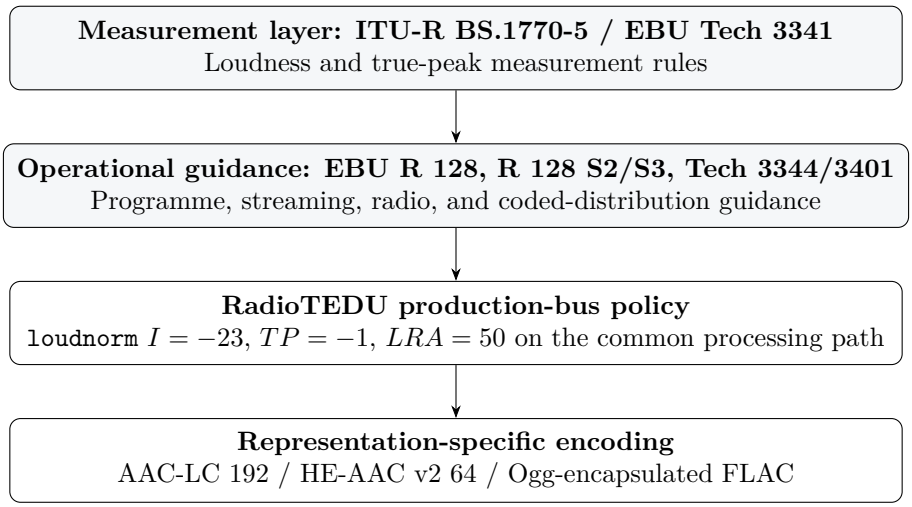

Figure 2: Standards-to-policy mapping. A production-bus setting is not presented as identical to a coded-distribution conformance limit.

continuously running service and a 60-second client observation are not the same measurement object as a complete programme.

The `TP=-1` value and limiter also belong to the production-processing stage; they are not presented as proof of coded-distribution compliance. EBU Tech 3344 separates a production maximum of −1 dBTP from the extra headroom that may be needed before perceptual coding (European Broadcasting Union, 2016). The RadioTEDU value is therefore described as a −1 dBTP *production-bus policy*, not as a guarantee about every decoded AAC output.

6.2 Operational filter chain and signal boundary

The common processing chain uses FFmpeg's `loudnorm`, `alimiter`, and `aresample` filters. FFmpeg documents `loudnorm` as an EBU R 128 normalization filter that supports dynamic and linear modes (FFmpeg Project, n.d.-b). The case configuration is:

```
loudnorm=I=-23.0:TP=-1.0:LRA=50,
alimiter=limit=0.891251:attack=5:release=80:level=false:latency=true,
aresample=48000,
aformat=sample_fmts=s16:sample_rates=48000:channel_layouts=stereo
```

The real-time chain does not provide the measured values needed for two-pass file normalization:

- `measured_I`
- `measured_LRA`
- `measured_TP`
- `measured_thresh`

It is therefore a dynamic streaming policy, not a two-pass file result. The LRA value of 50 LU is a configuration target, not a measured programme range and not proof that the original dynamics remain unchanged.

The limiter threshold is

$$10^{-1/20} \approx 0.891251. \tag{3}$$

This normalized linear amplitude corresponds to a −1 dBFS sample-amplitude threshold; it is not a true-peak measurement. FFmpeg's `alimiter` limits sample amplitude and is treated here as a safety stage rather than proof that true peak remains below the same value after encoding and decoding (FFmpeg Project, n.d.-b).

The loudness state belongs to the running station-processing filtergraph. Recreating that filtergraph starts a new state. The 60-second client measurements below connect to an already running public stream and do not reveal the age of the station-processing filter state. Track changes, station restarts, and host restarts were not instrumented as separate loudness experiments in this study.

6.3 Public decoded-stream spot measurements

On 25 August 2026, the seven primary public mounts were measured one after another for 60 s. FFmpeg decoded each stream and applied `ebur128=peak=true`. The command form was:

```
ffmpeg -hide_banner -nostats -i http://stream.radiotedu.com/radio \
  -t 60 -filter_complex ebur128=peak=true -f null NUL
```

On Linux or macOS, `NUL` is replaced by `/dev/null`. Table 7 reports integrated loudness, LRA, and the larger of the two channel TPK values printed by FFmpeg.

Table 7: Sequential 60-second decoded primary-stream spot measurements

Mount	I (LUFS)	LRA (LU)	Difference from −23 (LU)	TPK (dBFS)
`/radio`	−24.2	4.9	−1.2	−16.0
`/classic`	−27.9	2.0	−4.9	−8.6
`/cazz`	−23.7	6.3	−0.7	−7.5
`/lofi`	−24.1	5.6	−1.1	−10.7
`/energize`	−23.8	4.5	−0.8	−11.6
`/situation`	−21.9	2.9	+1.1	−7.4
`/rock`	−23.5	1.4	−0.5	−16.0

The windows contained whatever material was on air and were not controlled for genre, track position, speech, transitions, or silence. The values therefore describe those seven windows only. They are not acceptance tests against a ±0.5 LU rule, and they do not establish programme- or service-loudness compliance. Within these windows, the largest printed TPK value was −7.4 dBFS, so no near-full-scale peak was observed; this is still not a guarantee for other times or other representations.

6.4 Apply shared processing once

Shared normalization takes place before codec-specific encoding. Transitions and common audio processing are completed at one point, so every public stream starts from the same editorial signal.

This approach also avoids repeated lossy coding. Source material is decoded to PCM, processed once, and then encoded separately for each public output. One AAC stream is never

decoded and used to create another AAC stream. This prevents a second lossy encoder from working on audio that has already lost information.

7 Transitions and Programme Continuity

7.1 Transition policy

Choosing a codec does not make programme transitions smooth. RadioTEDU therefore handles transitions as part of the shared editorial programme, not inside each encoder. Music tracks overlap for three seconds with quarter-sine curves. Music-to-jingle changes are much shorter. Speech, announcements, and operator-controlled items may use a direct cut when an overlap would not suit the content.

For normalized transition time $x \in [0, 1]$, the conceptual quarter-sine pair is

$$g_{\text{out}}(x) = \cos\left(\frac{\pi x}{2}\right), \tag{4}$$

$$g_{\text{in}}(x) = \sin\left(\frac{\pi x}{2}\right). \tag{5}$$

The squared gains sum to one. If the two inputs have zero cross-correlation and equal mean-square power, this gives constant expected electrical power. If their powers differ or they are correlated, the electrical and perceived level can change during the fade. For this study, the key point is that the transition is calculated *once*. Every public output receives the same transition before encoding.

7.2 Keeping the programme clock independent

A resilient radio system must know "what is on air" even when an encoder stops working. RadioTEDU therefore keeps the programme clock separate from the lifecycle of each output. The station owns the schedule position, current item, and transition timing, while each encoder worker owns only its connection and output state.

This leads to a simple recovery rule: when an output worker restarts, it joins the *current* programme instead of playing the interrupted track from the beginning. A technical reconnection therefore does not change the editorial timeline.

8 Failure Boundaries and Operational Resilience

In this paper, resilience means that an output worker does not own the editorial clock and can return to the current programme after a failure. It does not mean that a long-term availability rate has been measured, or that a slow branch has been proved unable to affect every other branch.

8.1 Supervising each output separately

Figure 3 shows the failure model. The station's programme state comes before the output workers. Each branch has limited buffering and its worker is supervised separately. Limiting the buffers

prevents uncontrolled resource use, but it does not prove that every write to every branch is always non-blocking.

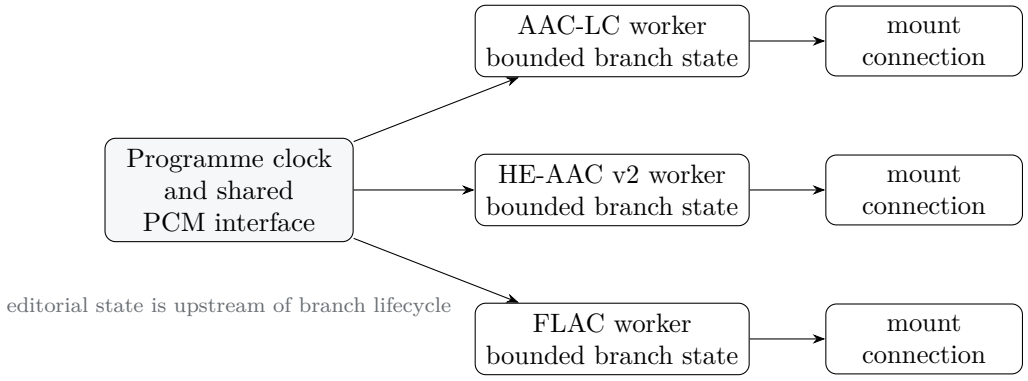

Figure 3: Failure-boundary model. Representation workers are downstream of the station programme state.

The evidence supports a clear structural claim: worker failures are handled separately from the programme clock. A limited queue does not prove that one branch can never affect another. The shared programme path remains a common point of failure.

8.2 Worker lifecycle and staged recovery

Figure 4 shows the recovery process. Retry delays increase through 1, 2, 4, 8, 15, and 30 s. When an output worker is recreated, it connects to the *current* programme PCM instead of restarting the interrupted item. Recovery means returning to the live programme, not replaying older queued audio. This paper reports the configured sequence but does not present measured time-to-audio, jitter, or a complete availability experiment.

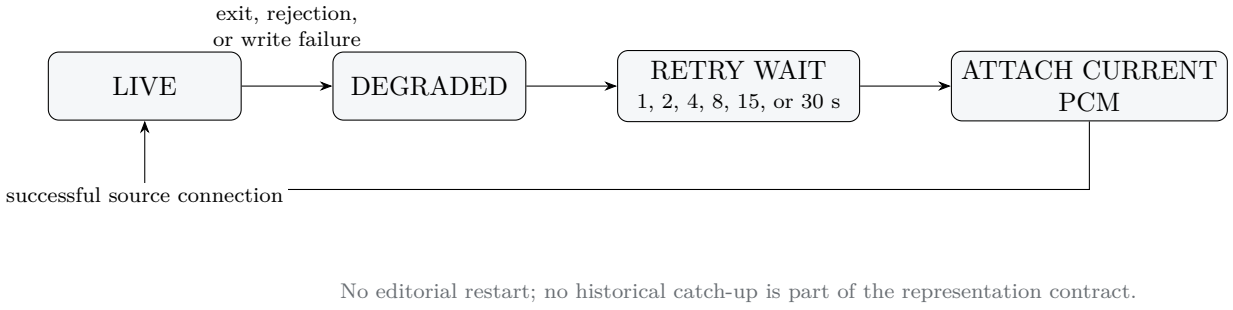

Figure 4: Representation-worker supervision state model used by the case analysis.

8.3 Failure classes

Table 8 separates the system's design rules from stronger guarantees. The response column describes what the system is designed to do; it does not report measured availability.

8.4 Prewarming and cache

Upcoming media are prepared before their scheduled start. A fast-media cache reduces the effect of slow primary storage, but it does not replace the original source. The original file remains

Table 8: Failure-boundary policy

Failure class	State boundary	Case response
Unreadable source	Decode/prewarm path	Handle the item through explicit skip/fallback logic rather than redefining unrelated station state.
Encoder exit	Representation worker	Recreate the representation worker and attach it to current programme PCM.
Mount rejection	Network output	Preserve local programme state and enter staged reconnect.
Network interruption	Transport connection	Keep scheduling state local; re-establish the representation when the path returns.
Slow branch	Branch buffer/write path	Bound branch state and allow supervision to classify persistent non-draining behavior as a representation fault; queue boundedness is not used as a proof of wait-free fan-out.
Storage delay	Media-access layer	Use prewarming and bounded cache to reduce exposure before the scheduled boundary.
Host restart	Host-level service	Restore enabled station workers from persisted configuration.
Metadata error	Metadata path	Keep the audio representation conceptually separate from metadata update state.
Shared PCM failure	Station programme path	Common-mode failure: all representations of the affected station may be lost until the shared path is restored.

authoritative. The cache has a maximum size of 4 GiB.

The cache must have a clear size limit and must identify files correctly. It must also protect files that are currently in use. A fast cache containing the wrong file would be worse than reading the correct file more slowly. Cache design is therefore part of reliable media access, not simply a way to hide output delay.

9 Analytical Findings

9.1 RQ1: Codec tiers as representations, not stations

The output matrix answers RQ1 by showing that stations and outputs are different things. Seven stations can produce fifteen outputs because several outputs may belong to the same station. Classical, for example, offers AAC-LC, HE-AAC v2, and Ogg-encapsulated FLAC while keeping one schedule, one current item, and one transition timeline.

The model gives each codec a role instead of choosing one codec for everything. AAC-LC is the primary stream, HE-AAC v2 is the efficient stream, and Ogg-encapsulated FLAC is the lossless stream. General codec-support documents do not prove that a particular endpoint will work; transport and framing also matter.

9.2 RQ2: Shared programme state with representation-specific lifecycles

The shared PCM interface answers RQ2 by keeping editorial continuity before the output workers. Scheduling, transitions, and shared processing define the station's programme state. Codec and

connection workers then use that state to create the public streams.

This supports a careful claim: an encoder worker does not control the programme clock, and a restarted worker joins the current PCM signal. Limited branch buffers control resource growth, but they do not formally prove that sibling branches can never delay one another. A failure in the shared programme path can still affect every output of the station.

9.3 RQ3: Standards-to-policy traceability

The system answers RQ3 by keeping standards and local choices separate. ITU-R BS.1770-5 explains how to measure loudness and true peak. EBU R 128 recommends a programme-loudness target, and EBU Tech 3344 discusses distribution and headroom. RadioTEDU then applies its own production-bus settings and defines where each output begins.

This difference matters. The -1 dBTP value is reported as a production-bus setting, not as a guarantee for the decoded lossy output. In the same way, `alimiter` is not described as a true-peak meter, and `LRA=50` is not reported as a measured programme range. The 60-second public measurements are kept separate from these configured targets.

9.4 Design lessons from the case

Five design lessons emerge from the case:

P1: Keep outputs below the editorial identity. A codec output is a representation of a station, not a new station.

P2: Complete shared processing before the outputs split. Scheduling, transitions, and common signal settings should be handled at one controlled point when consistency is needed.

P3: Keep output recovery separate from programme state. Restarting an encoder or connection should not reset the editorial clock.

P4: Limit branch state and define recovery clearly. Limited buffering, staged retries, and attachment to current live PCM are clearer than a broad claim of "no back-pressure."

P5: State the boundary of every quality claim. Terms such as "lossless," "loudness target," and "true peak" should say whether they refer to the source, processed PCM, codec input, encoded stream, or decoded output.

These lessons come from one case and are not statistical laws. Their value is that they make the system easier to inspect and prevent its claims from becoming too broad.

10 Discussion

In this paper, multi-channel means that RadioTEDU offers several editorial stations. It does not mean audio with more than two channels. This model is useful because one mixed schedule cannot serve every listener at the same time. A listener may want Classical, Jazz, Rock, Lo-Fi, Pop, Energize, or Situation Room. Separate stations give each type of content a clearer identity. They also allow each station to keep its own schedule while using the same general technical platform.

The multi-channel model becomes more useful when each station can also have more than one technical representation. The editorial programme does not need to be copied for every codec. Instead, one station programme creates one processed PCM signal, and the output workers create the required streams from that signal. In this way, the system has seven editorial stations but fifteen public outputs. The extra outputs increase listener choice without creating fifteen different schedules.

Research does not show that one codec and bitrate is best for every listening condition. Audio quality depends on the content and encoder, while the listener's experience also depends on the network and device (Berg et al., 2013; Gilski & Stefański, 2017; Hines et al., 2015). A listener at home may have a stable connection, while a listener on a mobile network may have less bandwidth or a limited data plan. Offering several codec tiers allows the same station to serve both situations.

The 192 kbit/s AAC-LC stream is the main option for general listening. It gives the station a widely used high-bitrate representation without using the much larger data rate of FLAC. The 64 kbit/s HE-AAC v2 stream has a different role. HE-AAC v2 uses Spectral Band Replication and Parametric Stereo to carry useful audio information at a lower bitrate (den Brinker et al., 2009). It therefore uses less network data and can be a practical option when the connection is limited. The efficient stream is not a smaller editorial service: it carries the same live programme as the primary stream.

FLAC adds a third choice for Classical and Jazz. It preserves the processed PCM given to the encoder without loss (van Beurden & Weaver, 2024). This can be useful for listeners who have a suitable player and a stable connection and who want a lossless representation. However, lossless delivery is not required for every station or every listener. RadioTEDU therefore offers it only where it is part of the station's service policy.

Using AAC-LC, HE-AAC v2, and FLAC together is useful because the formats do not have the same job. AAC-LC is the main general-purpose stream, HE-AAC v2 reduces data use, and FLAC provides lossless delivery. These options support different network conditions and listening goals. They are not competing schedules. A listener who selects another representation still receives the same editorial station and programme timeline.

This model also makes later changes easier. A new codec or bitrate can be added as another output branch without creating a new station schedule. In the same way, an output can be removed without changing the programme clock. This separation helps the platform grow while keeping the meaning of each station clear. It also avoids using one encoded stream as the source for another encoded stream, because every representation starts from the shared PCM signal.

Shared programme processing keeps the representations consistent. Transitions, loudness processing, and format conversion happen before the outputs split. As a result, a codec choice does not create a different edit of the programme. This is important in a multi-channel service because listeners should understand a station as one editorial product, even when it is available in several formats.

Separate output workers also improve recovery. If one encoder or mount connection fails, its worker can restart and return to the current programme. The schedule does not restart with it. The other representations can remain available because they have their own workers. However, the shared PCM path is still a common dependency. A failure in that path can affect every

representation of the station.

This structure also gives clearer monitoring information. The platform can show station health and output health as different conditions. Station health describes the schedule, programme clock, and shared PCM path. Output health describes one codec worker and its mount connection. This difference helps operators see whether a problem affects the whole station or only one listening option.

HE-AAC v2 is also used in DAB+, but the complete systems are different. DAB+ places the codec inside a standard broadcast transport with its own framing and error protection (European Telecommunications Standards Institute, 2017). RadioTEDU sends AAC through an internet streaming system. Therefore, the shared codec family does not make the RadioTEDU stream a DAB+ signal. This example shows why a codec name alone is not enough to describe a delivery service.

11 Conclusion

This paper examined RadioTEDU as a design-science case of layered, multi-channel internet radio. In this use, multi-channel means several station services, not audio with more than two channels. The active local system has seven editorial stations and fifteen public outputs: seven 192 kbit/s AAC-LC primary streams, six 64 kbit/s HE-AAC v2 efficient streams, and two Ogg-encapsulated FLAC streams for Classical and Jazz. The main contribution is not only this codec matrix, but the separation of the editorial programme, processed PCM, output encoding, and worker recovery.

Three conclusions follow. First, codec tiers are easier to manage when they are treated as versions of one station rather than separately scheduled stations. Second, a documented PCM interface and separate worker lifecycles allow programme state and output recovery to be discussed independently. Third, loudness policy is clearest when ITU measurement methods, EBU guidance, local processing targets, and observed public-stream measurements are kept separate.

The public HEAD checks found all fifteen expected mounts online at the time of the test. Seven primary streams were also measured in sequential 60-second windows. These results add a dated runtime observation to the architecture description, but they are not long-term availability or service-loudness tests. The observed `application/ogg` response is not the RFC 5334-recommended media type for these audio-dominant streams; additionally, the RFC's required Ogg Skeleton information for `application/ogg` was not verified.

Together, the findings describe internet radio as a layered system. The schedule creates one programme, shared processing creates a controlled PCM signal, output workers encode that signal for listeners, and supervision returns failed workers to the current programme without giving a codec or connection control of the editorial clock. The term resilience in the title refers to this recovery-oriented separation of responsibilities, not to a measured availability guarantee.

Table 9: RadioTEDU case configuration and observations used in the analysis

Parameter	Case value
Observation date	25 August 2026, Europe/Istanbul (UTC+03:00)
Editorial stations	7 (Classical, Lo-Fi, Pop, Jazz, Rock, Energize, Situation Room)
Local public outputs	15 (7 primary + 6 efficient + 2 lossless)
Programme interface	signed 16-bit little-endian PCM (`s16le`), 48 kHz, stereo
Primary representation	configured AAC-LC (`aac_low`), 192 kbit/s, ADTS; public media type `audio/aac`
Efficient representation	configured HE-AAC v2 (`aac_he_v2`), 64 kbit/s, ADTS; public media type `audio/aac`
Lossless representation	Ogg-encapsulated FLAC, compression level 8; Classical and Jazz only; observed public media type `application/ogg`
Loudness processing set point	`loudnorm I=-23.0`; configured programme-processing target, not a per-track or 60-second result
Production-bus peak setting	`TP=-1.0` and amplitude limiter 0.891251; the limiter threshold is −1 dBFS sample amplitude, not an independent true-peak measurement
FFmpeg LRA parameter	50 LU; configuration parameter, not a measured programme LRA
Music-to-music transition	3.0 s, quarter-sine/cosine gain curves
Representation recovery	staged reconnect delays of 1/2/4/8/15/30 s; recreated worker attaches to current programme PCM
Public endpoint observation	all 15 expected mounts answered the HEAD check at the test time
Primary-stream spot range	integrated loudness −27.9 to −21.9 LUFS; printed TPK −16.0 to −7.4 dBFS across seven sequential 60-second windows

A Case Configuration and Verification Summary

B Public Replication Commands

The following commands use only public stream addresses. They are examples for repeating the endpoint, codec, and 60-second decoded-stream checks. Header results can change over time.

B.1 Endpoint headers

```
for s in radio classic cazz lofi energize situation rock; do
  curl -I "http://stream.radiotedu.com/$s"
done

for s in radio-low classic-low cazz-low lofi-low energize-low rock-low; do
  curl -I "http://stream.radiotedu.com/$s"
done

for s in classic-flac cazz-flac; do
  curl -I "http://stream.radiotedu.com/$s"
done
```

B.2 Codec and stream fields

```
for s in radio classic cazz lofi energize situation rock; do
  echo "=== /$s ==="
  ffprobe -v error -i "http://stream.radiotedu.com/$s" \
    -show_entries stream=codec_name,profile,sample_rate,channels,channel_layout,bit_rate \
```

```
    -of default=noprint_wrappers=1
done

for s in radio-low classic-low cazz-low lofi-low energize-low rock-low; do
  echo "=== /$s ==="
  ffprobe -v error -i "http://stream.radiotedu.com/$s" \
    -show_entries stream=codec_name,profile,sample_rate,channels,channel_layout,bit_rate \
    -of default=noprint_wrappers=1
done

for s in classic-flac cazz-flac; do
  echo "=== /$s ==="
  ffprobe -v error -i "http://stream.radiotedu.com/$s" \
    -show_entries stream=codec_name,sample_rate,channels,channel_layout,bits_per_raw_sample,
    bit_rate \
    -of default=noprint_wrappers=1
done
```

B.3 Sequential 60-second loudness observations

```
for s in radio classic cazz lofi energize situation rock; do
  echo "=== /$s 60s ==="
  ffmpeg -hide_banner -nostats \
    -i "http://stream.radiotedu.com/$s" \
    -t 60 -filter_complex ebur128=peak=true \
    -f null /dev/null 2>&1 | grep -E "Integrated|I:|LRA|TPK"
done
```

On Windows, use `NUL` instead of `/dev/null`. A repeated 60-second observation will normally contain different programme material and should not be treated as a fixed expected value.

References

Berg, J., Bustad, C., Jonsson, L., Mossberg, L., & Nyberg, D. (2013). Perceived audio quality of realistic FM and DAB+ radio broadcasting systems. *Journal of the Audio Engineering Society*, *61*(10), 755–777.

den Brinker, A. C., Breebaart, J., Ekstrand, P., Engdegård, J., Henn, F., Kjörling, K., Oomen, W., & Purnhagen, H. (2009). An overview of the coding standard MPEG-4 audio amendments 1 and 2: HE-AAC, SSC, and HE-AAC v2. *EURASIP Journal on Audio, Speech, and Music Processing*, *2009*, Article 468971. https://doi.org/10.1155/2009/468971

European Broadcasting Union. (2016, July 25). *Guidelines for distribution and reproduction in accordance with EBU R 128* (EBU Tech 3344, Version 2.1). https://tech.ebu.ch/publications/tech3344

European Broadcasting Union. (2023a, November 21). *'EBU Mode' metering to supplement EBU R 128 loudness normalisation* (EBU Tech 3341, Version 4.0). https://tech.ebu.ch/publications/tech3341

European Broadcasting Union. (2023b, November 21). *Guidelines for radio production and distribution in accordance with EBU R 128* (EBU Tech 3401, Version 2.0). https://tech.ebu.ch/publications/tech3401

European Broadcasting Union. (2023c, November 21). *Loudness in radio* (EBU R 128 Supplement 3, Version 2.0). https://tech.ebu.ch/publications/r128s3

European Broadcasting Union. (2023d, November 21). *Loudness in streaming* (EBU R 128 Supplement 2, Version 3.0). https://tech.ebu.ch/publications/r128s2

European Broadcasting Union. (2023e, November 21). *Loudness normalisation and permitted maximum level of audio signals* (EBU R 128, Version 5.0). https://tech.ebu.ch/publications/r128

European Telecommunications Standards Institute. (2017). *Digital audio broadcasting (DAB); DAB+ audio coding (MPEG HE-AACv2)* (ETSI TS 102 563 V2.1.1). https://www.etsi.org/deliver/etsi_ts/102500_102599/102563/02.01.01_60/ts_102563v020101p.pdf

FFmpeg Project. (n.d.-a). *FFmpeg codecs documentation*. Retrieved August 29, 2026, from https://ffmpeg.org/ffmpeg-codecs.html

FFmpeg Project. (n.d.-b). *FFmpeg filters documentation*. Retrieved August 29, 2026, from https://ffmpeg.org/ffmpeg-filters.html

Gilski, P., & Stefański, J. (2017). Subjective and objective comparative study of DAB+ broadcast system. *Archives of Acoustics*, *42*(1), 3–11. https://doi.org/10.1515/aoa-2017-0001

Goncalves, I., Pfeiffer, S., & Montgomery, C. (2008). *Ogg media types* (RFC 5334). RFC Editor. https://doi.org/10.17487/RFC5334

Google. (2024, September 18). *Supported media for Google Cast*. Retrieved August 29, 2026, from https://developers.google.com/cast/docs/media

Google. (2025, May 22). *Supported media formats*. Retrieved August 29, 2026, from https://developer.android.com/media/platform/supported-formats

Google. (2026, March 9). *Progressive media sources*. Retrieved August 29, 2026, from https://developer.android.com/media/media3/exoplayer/progressive

Grivcova, K., Pike, C., & Nixon, T. (2018). A subjective evaluation of high bitrate coding of music [Paper 10001]. *AES Convention 144*. https://aes.org/publications/elibrary-page/?id=19397

Hevner, A. R., March, S. T., Park, J., & Ram, S. (2004). Design science in information systems research. *MIS Quarterly*, *28*(1), 75–105. https://doi.org/10.2307/25148625

Hines, A., Gillen, E., Kelly, D., Skoglund, J., Kokaram, A., & Harte, N. (2015). ViSQOLAudio: An objective audio quality metric for low bitrate codecs. *The Journal of the Acoustical Society of America*, *137*(6), EL449–EL455. https://doi.org/10.1121/1.4921674

International Telecommunication Union. (2023). *Algorithms to measure audio programme loudness and true-peak audio level* (Recommendation ITU-R BS.1770-5). https://www.itu.int/rec/R-REC-BS.1770-5-202311-I

Runeson, P., & Höst, M. (2009). Guidelines for conducting and reporting case study research in software engineering. *Empirical Software Engineering*, *14*(2), 131–164. https://doi.org/10.1007/s10664-008-9102-8

Valin, J.-M., Vos, K., & Terriberry, T. B. (2012). *Definition of the Opus audio codec* (RFC 6716). RFC Editor. https://doi.org/10.17487/RFC6716

van Beurden, M. Q. C., & Weaver, A. (2024). *Free lossless audio codec (FLAC)* (RFC 9639). RFC Editor. https://doi.org/10.17487/RFC9639