

Living Master: A Versioned and Continually Upgradable Music Asset Architecture

Xuanzhen Zhu

Moodify Sound Laboratory, Rongjing Wenchuan (Shenzhen) Technology Co., Ltd., Shenzhen, China

Manuscript type: Engineering Report draft for internal review and reference implementation

Abstract—Digital music services normally treat a released master as a fixed file. This design preserves delivery simplicity, but it also binds the long-term technical state of a work to the tools, budget, and production conditions available at release time. This paper presents Living Master, a versioned music-asset architecture in which an immutable original master, creative-intent metadata, rendering context, engine releases, candidate renders, evaluation evidence, and publication decisions are represented as a single traceable work object. Every candidate is rendered directly from the immutable origin rather than from a previously processed version, preventing deterministic inheritance of cumulative processing errors. New engine releases may generate catalog-wide candidates, but cannot replace a published version without passing quality, risk, and intent-fidelity gates. The architecture separates preservation, transformation, evaluation, publication, and rollback; uses provenance and fixity records to make every audible change explainable; and defines a catalog inheritance mechanism by which historical works may receive future processing improvements. Living Master is compatible with preservation and multimedia metadata standards but adds a longitudinal audio-production layer that those standards do not prescribe. The contribution is an engineering model, lifecycle protocol, metadata schema, and validation plan. A production-scale catalog implementation and longitudinal listening study remain future work.

Keywords—digital music assets; audio provenance; version control; music mastering; digital preservation

1 Introduction

A digital music release is usually distributed as one or more fixed encodes derived from a final master. The content-delivery system may evolve, but the sound of the work remains linked to the production decisions made at one point in time. A platform can improve recommendation, transport, or user interface without improving the acoustic representation of previously released recordings.

The internal Jielu Living Master Theory proposes a different temporal model: a musical work is not identical to one file, but is a persistent asset containing an immutable sound origin, creative intent, context, and a history of validated renderings [1]. This paper translates that theory into an implementable engineering architecture. The term *Living Master* does not mean that a platform continuously modifies a file in place. It means that the platform preserves a stable origin and can generate, evaluate, publish, and roll back distinct versions as technology changes.

The architecture addresses seven requirements:

1. preserve the original master and its fixity evidence;
2. prevent cumulative reprocessing across generations;
3. separate candidate generation from publication authority;
4. preserve creative intent as an explicit constraint;
5. record complete engine, parameter, metric, and decision provenance;
6. support catalog-scale re-rendering after engine upgrades; and
7. allow deterministic rollback to any previously accepted version.

The contribution is architectural rather than perceptual superiority. A Living Master system does not guarantee that every new engine improves every work. It guarantees that every eligible work can be reconsidered, that only validated candidates are promoted, and that the origin and history remain recoverable.

2 Related Standards and Systems

2.1 Preservation and Provenance

The OAIS reference model defines a durable archival framework based on information packages, preservation planning, data management, and access [2]. PREMIS specifies preservation metadata around intellectual entities, objects, events, rights, and agents, including fixity and significant properties [3]. W3C PROV-O provides a general vocabulary for entities, activities, agents, derivation, and generation relationships [4]. These standards are directly relevant to authenticity and provenance, but they do not specify how a released musical work should be repeatedly re-rendered, perceptually evaluated, and promoted as a new default playback version.

2.2 Multimedia Item and Audio Models

MPEG-21 Digital Item Declaration models structured multimedia items consisting of resources, metadata, and relationships [5]. Digital Item Adaptation describes transformations motivated by terminal, network, storage, and quality-of-service constraints [6]. The ITU-R Audio Definition Model (ADM) provides metadata for channel-, object-, and scene-based audio production and interchange [7]. Living Master can use these standards for representation and delivery, but its novel focus is longitudinal production lineage: immutable origin, engine-version rendering, artistic-intent constraints, quality gates, publication aliases, and catalog inheritance.

2.3 Version Control versus Versioned Media

Software version control records source changes and permits reproducible builds. A music asset differs in two ways. First, the acceptance criterion is partly perceptual and artistic rather than only functional. Second, the primary source must remain unchanged while potentially many derived representations coexist. Living Master therefore borrows the ideas of immutable source, explicit build environment, provenance, and rollback, but treats each audible output as an evaluated media representation rather than a source-code commit.

3 Problem Formulation

Let musical work i be represented as

$$W_i = \{O_i, I_i, C_i, V_{i,1}, \dots, V_{i,n}\}, \quad (1)$$

where O_i is the immutable original master, I_i is a structured creative-intent profile, C_i is descriptive and production context, and $V_{i,k}$ are generated versions.

A version at engine release n is rendered by

$$V_{i,n} = \mathcal{R}(O_i, E_n, I_i, C_i, P_{i,n}, T_n), \quad (2)$$

where $P_{i,n}$ is the work-specific parameter set and T_n is the delivery and technical policy profile. Equation (2) separates the work from any single file and separates the engine from the source asset.

3.1 Origin Immutability Invariant

For all system times t ,

$$O_i^{(t)} = O_i^{(0)}, \quad (3)$$

verified by cryptographic digest and preservation metadata. Any operation that mutates the origin violates the architecture.

3.2 Non-Cumulative Derivation Invariant

A cumulative chain uses

$$V_{i,n} = F_n(V_{i,n-1}), \quad (4)$$

which may inherit compression, phase, spectral, or nonlinear errors from prior stages. Living Master instead requires

$$V_{i,n} = \mathcal{R}(O_i, E_n, I_i, C_i, P_{i,n}, T_n). \quad (5)$$

Thus version n may be compared with earlier versions, but is not acoustically derived from them. This does not eliminate source defects or defects introduced by E_n ; it eliminates deterministic inheritance of previous processing decisions.

4 Lifecycle and Version Graph

4.1 States

A candidate passes through the state machine in Table 1. Publication is implemented as an atomic change to a logical alias or manifest pointer; no accepted file is overwritten.

Table 1: Version lifecycle states.

State	Required condition
Generated	Render completed; hashes and engine identity recorded.
Measured	Objective features and policy checks completed.
Reviewed	Listening or expert review completed when policy requires it.
Accepted	Quality, risk, rights, and intent thresholds pass.
Published	Atomic alias points to this accepted version.
Superseded	Another accepted version becomes default; file remains available.
Rejected	Candidate retained with failure evidence or deleted under retention policy.
Revoked	Previously accepted version removed from delivery for rights or safety reasons.

4.2 Candidate-Before-Replacement Rule

An engine update grants permission to create candidates, not authority to replace the default. Let $Q(V)$ be technical-perceptual quality, $R(V)$ risk, $F(V, I_i)$ intent fidelity, and $L(V, V_{i,p})$ unacceptable divergence from the current published version. Define

$$G_{i,n} = Q(V_{i,n}) - Q(V_{i,p}) - \lambda R(V_{i,n}). \quad (6)$$

Promotion requires

$$\begin{aligned} G_{i,n} &> \theta_i, \\ R(V_{i,n}) &< R_{\max}, \\ F(V_{i,n}, I_i) &> \eta_i, \\ L(V_{i,n}, V_{i,p}) &< L_{\max,i}. \end{aligned} \quad (7)$$

Thresholds may depend on genre, rights policy, intended aesthetics, and the confidence of the evaluation data.

4.3 Rollback

Rollback is a metadata transaction: change the published alias from $V_{i,n}$ to an earlier accepted $V_{i,k}$. The origin and both derived files remain unchanged. This makes rollback fast, auditable, and independent of re-rendering.

5 Metadata and Provenance Model

Each work is represented by an append-only manifest whose minimum entities are summarized in Table 2. The model can be serialized as JSON-LD, relational tables, or graph records. PREMIS concepts can represent objects, events, agents, rights, fixity, and significant properties; PROV-O can express that a version was generated by a rendering activity using a specific engine and source.

5.1 Identity and Fixity

Every stored binary object should carry at least one collision-resistant digest. An origin identifier must be independent of storage path. Engine identity should include source revision, model weights, dependency lock, and execution image; a version is reproducible only if both the media source and the rendering environment are identifiable.

5.2 Intent Profile

Creative intent is not reduced to one emotion label. A practical profile may include target foreground/background balance, desired width, dynamic

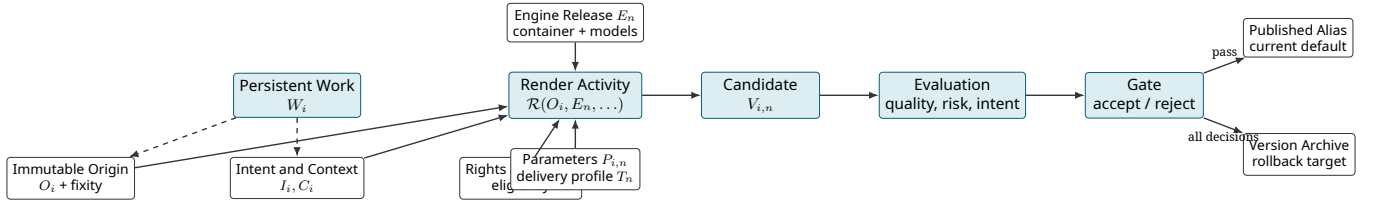


Figure 1: Living Master object and decision flow. The published alias is mutable, but the origin, candidate files, and decision history are append-only.

Table 2: Minimum Living Master metadata entities and representative fields.

Entity	Stable identifier	Representative fields	Relevant standards mapping
Work	work_id	title, creators, rights, intent profile, context, eligibility policy	MPEG-21 Digital Item; PREMIS Intellectual Entity
Origin	origin_id	URI, codec, sample rate, channels, bit depth, duration, checksum, ingest event	PREMIS Object and fixity; OAIS Content Information
Engine release	engine_id	semantic version, source commit, model hashes, dependency lock, container digest, calibration set	PROV Agent/Entity; PREMIS Environment
Render activity	render_id	engine, origin, parameters, delivery profile, start/end time, compute environment, logs	PROV Activity; PREMIS Event
Candidate version	version_id	file hash, parent origin, metrics, risk, intent score, review evidence, state	PREMIS Object; PROV Entity
Gate decision	decision_id	policy version, thresholds, reviewer/agent, pass/fail reasons, timestamp	PROV Activity/Agent; PREMIS Event
Published alias	alias_id	current version, effective interval, territory/device profile, rollback target	MPEG-21 relation and presentation metadata

character, tolerated distortion, vocal intimacy, roughness, temporal contrast, reference notes, and protected traits. Some fields are machine-readable constraints; others remain descriptive evidence for review.

5.3 Provenance Rule

Every version must answer four questions: what source was used, what engine and parameters generated it, what changed, and why it was accepted or rejected. A version without this evidence may be playable, but is not a trustworthy Living Master asset.

6 Catalog Inheritance

Let a platform contain N works. When engine E_{n+1} is released, an eligibility function selects

$$S_{n+1} = \{i \mid \text{eligible}(W_i, E_{n+1}) = 1\}. \quad (8)$$

Eligibility can depend on rights, source availability, audio format, prior failure tags, compute cost, genre support, and creator opt-in. Candidates are scheduled for $i \in S_{n+1}$, evaluated, and promoted independently.

The catalog quality at release n can be represented as

$$Q_{\text{cat},n} = \frac{\sum_{i=1}^N w_i Q(V_{i,p(n)})}{\sum_{i=1}^N w_i}, \quad (9)$$

where $p(n)$ maps each work to its default published version and w_i is a declared catalog weighting policy. The catalog improvement is

$$\Delta Q_{\text{cat},n+1} = Q_{\text{cat},n+1} - Q_{\text{cat},n}. \quad (10)$$

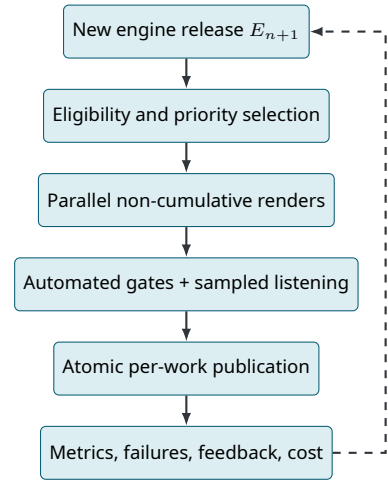


Figure 2: Catalog inheritance cycle. Each work is independently eligible, rendered, gated, and published.

A useful engineering efficiency measure is

$$\text{TDR} = \frac{N_{\text{qualified}} \overline{\Delta Q} R_{\text{reach}}}{C_{\text{engine}} + C_{\text{render}} + C_{\text{review}}}, \quad (11)$$

where R_{reach} is the fraction of listening events served by promoted versions. Equation (11) is an operational planning metric, not a universal economic law.

7 Delivery Profiles and Representation Separation

A Living Master version and a delivery representation are different objects. The accepted version $V_{i,n}$ is the production-level master selected by the gate. A

device- or service-specific representation is

$$A_{i,n,d} = \mathcal{D}(V_{i,n}, T_d), \quad (12)$$

where $\mathcal{D}$ is a delivery transform and T_d is a profile containing codec, sample rate, channel layout, loudness policy, and device constraints. A new encode of the same accepted version does not create a new Living Master generation unless it changes the production interpretation of the work. This distinction prevents routine transcoding from polluting the version history and allows MPEG-21 adaptation or ADM-based delivery metadata to coexist with the longitudinal master lineage.

Delivery representations should still record source-version identity, encoder version, parameters, checksum, and effective policy interval. The delivery resolver may select different representations for headphones, loudspeakers, spatial systems, or bandwidth constraints while preserving a common accepted master identity.

8 Storage and Compute Model

If every candidate were retained indefinitely, storage growth would be unbounded. Let $s(O_i)$ be origin size, $s(V_{i,k})$ retained version size, and m_i the number of retained candidates. Approximate catalog storage is

$$S_{\text{cat}} = \sum_{i=1}^N \left[s(O_i) + \sum_{k=1}^{m_i} s(V_{i,k}) + s(M_i) \right], \quad (13)$$

where $s(M_i)$ is metadata, reports, and logs. Retention policy should distinguish accepted versions, rejected candidates with research value, and disposable intermediate files. Lossless origins and accepted versions receive preservation priority; rejected candidates may be retained as metrics and provenance without retaining full audio when rights and research policy permit.

Catalog rendering is embarrassingly parallel at the work level. For eligible set $\mathcal{S}$, total compute is $C = \sum_{i \in \mathcal{S}} c_i$, while wall-clock time depends on worker capacity, queue policy, model loading, and review latency. Reuse of cached features, deterministic job identifiers, idempotent retries, and content-addressed outputs reduce duplicate work. Storage and compute budgets therefore become explicit release constraints rather than hidden consequences of engine evolution.

9 System Architecture

A reference implementation contains six separable services.

Asset registry. Stores work manifests, origin identifiers, rights, intent, and current aliases.

Immutable object store. Stores origins, candidates, accepted versions, reports, and logs under content-derived identities or equivalent fixity controls.

Render orchestrator. Resolves an engine release and work manifest into an isolated render job. The current Moodify pipeline already separates source input, processed output, and report generation, while treatment records preserve parameters and before/after measurements [8].

Evaluation service. Runs loudness and true-peak checks consistent with ITU-R BS.1770, objective feature extraction, regression checks, and policy-specific risk tests [9]. Subjective comparison may use ITU-R BS.1534 when perceptual differences require controlled listening [10].

Gate and review service. Combines automated thresholds with human review requirements. The experimental Moodify codebase includes a three-gate structure for diagnosis completeness, separation quality, and output compliance, although some production checks remain incomplete [8].

Delivery resolver. Maps a work request and delivery profile to the current accepted version. It must support atomic alias updates, cache invalidation, territory and rights filters, device-specific representations, and immediate rollback.

10 Safety and Failure Modes

10.1 Cumulative Processing

The most direct failure is rendering from the current default rather than the immutable origin. Enforcement should occur at API and storage layers: render jobs accept an `origin_id`, never an arbitrary published-version path, unless an explicitly declared restoration workflow requires otherwise.

10.2 False Improvement

A higher objective score can conflict with artistic intent. The system must retain protected traits and require review when the candidate exceeds divergence limits. A noisy, narrow, dark, or distorted aesthetic can be intentional.

10.3 Engine and Environment Drift

Identical parameters do not guarantee identical audio if dependencies, model weights, codecs, or floating-point environments differ. Reproducibility therefore requires engine manifests and, where necessary, reference renders or tolerance bands.

10.4 Rights and Consent

Re-rendering may be technically possible but contractually prohibited. Creator opt-in, territorial restrictions, derivative-work terms, and revocation must be first-class eligibility inputs. A technical improvement cannot override rights.

10.5 Catalog-Scale Cost

An engine release may make millions of candidate renders economically infeasible. Priority queues should use expected gain, confidence, audience reach, creator value, and cost. Low-confidence classes can be sampled before full deployment.

10.6 Identity and Authenticity

A malicious or mistaken replacement of the origin invalidates the lineage. Fixity verification should run at ingest, before every render, after migration, and periodically under preservation policy. Decision logs should be append-only and access controlled.

11 Validation Protocol

Table 3 defines the minimum validation program before production use.

A first experimental deployment should use a modest catalog, multiple engine releases, and deliberately

Table 3: Architecture validation tests.

Test	Pass condition
Origin im-mutability	Origin digest unchanged across all lifecycle events.
Non-cumulative lineage	Every candidate has direct derivation from origin; no hidden parent-version dependency.
Reproducibility	Re-render within declared sample or metric tolerance using archived engine environment.
Atomic publish	Concurrent readers observe either old or new alias, never a partial state.
Rollback	Previous accepted version restored without audio reprocessing.
Gate determinism	Same evidence and policy version produce same automated decision.
Provenance completeness	All mandatory source, engine, parameter, metric, review, and decision fields present.
Intent preservation	Controlled listening shows no material violation of protected traits.
Catalog simulation	Batch scheduler respects rights, cost ceilings, retries, and per-work isolation.
Disaster recovery	Work manifest, origin, accepted versions, and alias reconstructed from backups.

injected failures. The study should measure reproducibility rate, gate false-positive and false-negative rates, rollback latency, storage growth, render cost, and listener preference under loudness-matched conditions. Longitudinal evaluation is required to test the central claim that an evolving engine can raise catalog quality without eroding identity or trust.

12 Discussion

Living Master separates three concepts that are often conflated: preservation of the work’s origin, generation of new technical representations, and selection of a default playback version. This separation allows technology to evolve without rewriting history. It also changes the unit of audio research value. An improvement to an engine can affect not only future uploads but also eligible historical works, creating a direct connection between research, catalog quality, and user experience.

The architecture also imposes restraint. A new version is not inherently superior, more versions do not automatically create more asset value, and automated processing does not replace creative authority. Version history becomes valuable only when differences are meaningful, provenance is complete, and rollback remains possible.

Living Master is complementary to archival standards rather than a replacement for them. OAIS and PREMIS help preserve authenticity and long-term interpretability; PROV-O helps express lineage; MPEG-21 and ADM help structure multimedia and audio metadata. The proposed layer adds a production lifecycle in which preserved sources can be repeatedly reinterpreted under explicit technical and artistic controls.

13 Implementation Status and Limitations

The Moodify repository provides partial building blocks: separate output generation, analysis and diagnostics, parameterized DSP, treatment records, before/after metrics, and experimental quality gates [8].

It does not yet constitute a production Living Master platform. Missing components include an immutable catalog registry, engine-container archive, complete rights and intent schemas, catalog scheduler, atomic delivery alias service, full gate calibration, production monitoring, and longitudinal listening evidence.

The equations in this paper are design abstractions. Q , R , F , and L require empirical definitions and calibration; catalog weights can encode institutional bias; and engine inheritance can amplify systematic errors if validation is weak. The architecture should therefore be deployed gradually, with creator control, sampled review, conservative promotion thresholds, and transparent version histories.

14 Conclusion

Living Master defines a musical work as a persistent, versioned asset rather than a fixed delivery file. Its central engineering invariants are immutable origin, non-cumulative derivation, candidate-before-replacement, intent-aware quality gating, complete provenance, and reversible publication. These invariants allow historical music to become eligible for future technical improvement without destroying its source or concealing its history. The next step is a reference implementation and longitudinal catalog experiment measuring reproducibility, perceptual benefit, risk, cost, and trust.

15 Reference Manifest and Promotion Transaction

The following compact manifest illustrates the logical minimum for one accepted version. Field names are informative rather than normative.

Table 4: Illustrative work-manifest fragment.

Field	Example semantic content
work_id	Stable work identity independent of files.
origin	Object URI, SHA-256 digest, codec, sample rate, bit depth, ingest event.
intent	Protected traits, target characteristics, references, creator notes, consent scope.
engine	Release ID, source commit, model digests, dependency and container digests.
render	Parameter set, delivery policy, job ID, timestamps, compute environment.
evaluation	Objective metrics, listening evidence, risk flags, intent fidelity, policy version.
decision	Accepted/rejected, reasons, agent, effective time, signature.
published	Alias to accepted version plus rollback target.

An atomic promotion transaction follows this order:

1. verify origin, candidate, report, and engine digests;
2. verify rights, creator consent, gate policy, and review completion;
3. write an append-only decision event referencing all evidence;
4. compare-and-swap the published alias from the expected prior version to the candidate;

5. invalidate delivery caches and open a monitoring interval;
6. automatically restore the prior alias if post-publication safety thresholds fail.

The compare-and-swap condition prevents two concurrent engine jobs from silently overwriting each other's publication decision. The monitoring interval treats publication as reversible until sufficient operational evidence has accumulated.

Conflict of Interest

The author is the founder of Rongjing Wenchuan (Shenzhen) Technology Co., Ltd. and the designer of the Moodify and Jielu Music concepts. The architecture is therefore commercially relevant. No independent validation or production-scale deployment is claimed in this manuscript.

Data and Code Availability

The current Moodify engineering repository is available at <https://github.com/huliye24/moodify-o3is>. The complete Living Master catalog service, metadata schema, and longitudinal evaluation dataset described in this paper have not yet been released because they are not yet implemented at production scale.

AI-Assistance Disclosure

A generative language model was used to assist with manuscript structuring, English drafting, and typesetting. The author supplied the theory, system concepts, repository materials, and research direction, and remains responsible for verifying all technical claims, equations, references, and interpretations before submission.

References

- [1] X. Zhu, "Jielu Living Master Theory (JLMT)," internal technical manuscript, Rongjing Wenchuan (Shenzhen) Technology Co., Ltd., 2026.
- [2] Consultative Committee for Space Data Systems, *Reference Model for an Open Archival Information System (OAIS)*, CCSDS 650.0-M-3, Dec. 2024.
- [3] PREMIS Editorial Committee, *PREMIS Data Dictionary for Preservation Metadata*, Version 3.0, Library of Congress, 2015.
- [4] W3C, "PROV-O: The PROV Ontology," W3C Recommendation, Apr. 2013.
- [5] ISO/IEC, *Information Technology – Multimedia Framework (MPEG-21) – Part 2: Digital Item Declaration*, ISO/IEC 21000-2:2005.
- [6] ISO/IEC, *Information Technology – Multimedia Framework (MPEG-21) – Part 7: Digital Item Adaptation*, ISO/IEC 21000-7:2007.
- [7] ITU-R, *Audio Definition Model*, Recommendation ITU-R BS.2076-3, Feb. 2025.
- [8] X. Zhu, "Moodify-o3is: Explainable AI-music post-processing engineering repository," version inspected July 2026, <https://github.com/huliye24/moodify-o3is>.
- [9] ITU-R, *Algorithms to Measure Audio Programme Loudness and True-Peak Audio Level*, Recommendation ITU-R BS.1770-5, Nov. 2023.
- [10] ITU-R, *Method for the Subjective Assessment of Intermediate Quality Level of Audio Systems*, Recommendation ITU-R BS.1534-3, Oct. 2015.