

Market data governance as knowledge management: managing temporality via rule versioning, provenance and auditability

Oliver Njeru (ORCID 0009-0008-6611-3813) and Paul Okanda (ORCID 0000-0001-5215-4368)
School of Science and Technology, United States International University – Africa, Nairobi, Kenya
Corresponding author: Oliver Njeru, onjeru@usiu.ac.ke

Published in *VINE Journal of Information and Knowledge Management Systems*, Emerald Publishing Limited.
Version of record: <https://doi.org/10.1108/VJIKMS-03-2026-0132>

Received 9 March 2026; revised 17 May 2026 and 13 August 2026; accepted 9 September 2026.

Cite as: Njeru, O. and Okanda, P. (2026), “Market data governance as knowledge management: managing temporality via rule versioning, provenance and auditability”, *VINE Journal of Information and Knowledge Management Systems*, Vol. ahead-of-print No. ahead-of-print. <https://doi.org/10.1108/VJIKMS-03-2026-0132>

Licence. This author accepted manuscript is deposited under a Creative Commons Attribution Non-commercial 4.0 International (CC BY-NC) licence. This means that anyone may distribute, adapt, and build upon the work for non-commercial purposes, subject to full attribution. If you wish to use this manuscript for commercial purposes, please visit Marketplace.

Licence terms: <https://creativecommons.org/licenses/by-nc/4.0/>

About this version. The pages that follow are the manuscript as accepted for publication, unchanged. The published article was copy-edited and typeset by Emerald and differs in style and wording in places; please cite the published version. Points in this manuscript that the published article supersedes:

- The manuscript refers to a supplementary file and states that the package “will be released publicly under a permanent identifier on acceptance”. The analysed file set, the validation harness and the claim-by-claim verification report are archived at <https://doi.org/10.5281/zenodo.22734804> (Apache-2.0).
- Keywords as published: Knowledge management, Market data governance, Design science research, Auditability, Provenance, Boundary objects. The manuscript’s sixth keyword reads “equity markets”.
- Figure 1: the Evaluation activity is covered in Sections 4.5 and 5.2; the figure reads “Sections 4.7, 5.2”.
- Figure 2: the field names are `effective_from` and `effective_to`; the figure prints a stray backslash before each underscore.

Declarations, as published.

Competing interests: Oliver Njeru is the founder of Black Eternum Capital. The company did not fund or commission this work. The authors declare no other competing interests.

Funding: This research received no grant from any funding agency in the public, commercial or not-for-profit sectors.

Data availability: No real or licensed market data is used or redistributed; all data in the artefact is synthetic and generated by a seeded script included with it. The exact file set analysed, the validation harness and a report mapping each claim to the file and line range evidencing it are openly available at <https://doi.org/10.5281/zenodo.22734804> under the Apache-2.0 licence.

Market Data Governance as Knowledge Management: Managing Temporality via Rule Versioning, Provenance, and Auditability

Abstract

Purpose: This paper asks how organisations hold governance knowledge stable enough to execute reliably yet revisable as policies and conventions change. It develops a temporality-focused conceptual elaboration of knowledge management (KM) for executable governance, reading market data rules as boundary objects whose validity is bounded in time.

Design/methodology/approach: The study is a retrospective analysis of an existing design artifact – a Python repository for an equity trading ecosystem, built by the authors for an earlier project – from which design knowledge is abstracted using design science research principles [Hevner et al., 2004, Gregor and Hevner, 2013], a documented coding protocol, and analytic generalisation [Yin, 2018].

Findings: The artifact carries governance knowledge in effective-dated fee regimes resolved deterministically, explicit session and settlement conventions, schema exemplars and fixtures without a validation gate, and reporting treated as an audit output. It supports a lightweight provenance analogue, not formal provenance, answering “what rule applied when” through inspectable configuration and deterministic resolution. Five artifact-level validations pass and are independently reproducible.

Originality/value: Boundary-object theory treats interpretive flexibility as the source of an object’s coordinating power. Executable governance inverts this: flexibility is removed at execution and preserved at review. Naming that inversion, and the design knowledge following from it, is the contribution.

Research limitations/implications: One artifact, analysed by those who built it, can support claims about how executable governance may be *structured*, not about adoption or effectiveness. Propositions are theoretically derived and stated with falsification conditions.

Practical implications: Treat fee schedules, calendars, schemas and reporting templates as versioned executable controls, and their defaults as governance decisions requiring review.

Keywords: knowledge management; market data governance; design science research; auditability; provenance; equity markets

Article classification: Research paper

1 Introduction

Equity markets are information-dense socio-technical systems. Price discovery, best execution, risk control and regulatory compliance depend on the integrity of market data and, just as much, on

the operational knowledge that gives that data meaning: which fee regime applied in which period, which holidays define a valid trading day, how stale prices are handled, how a vendor’s fields should be read. Failures in this layer are rarely dramatic. They are detected late, after the affected outputs have been consumed downstream, and their cost is paid in rework and mistrust rather than in a visible incident [Redman, 1998].

What makes the layer difficult is not that its knowledge is hard to write down. Fee schedules, calendars and schema conventions are eminently writable. The difficulty is that what is written down is true only for a period. A brokerage cap is set by a statutory instrument that will be replaced; a settlement lag reflects a market structure that will be reformed; a vendor field means one thing before a migration and another after it. Governance must therefore hold two commitments at once: it must stabilise shared meaning so that everyone applying a rule reaches the same answer, and it must stay revisable, so that adopting a new rule does not cost the ability to reconstruct and defend what the old one required. We call this the stability–change tension, and take it to be the central problem market data governance poses for KM.

Knowledge management is the natural lens. KM research holds that organisational knowledge exists in tacit and explicit forms and that the systems built around it should support its creation, transfer and application [Alavi and Leidner, 2001, Nonaka and Takeuchi, 1995]; market data governance is precisely a function that captures operational knowledge, validates it and disseminates it in a form systems can apply. What existing work leaves under-specified is the temporal structure of that knowledge once it becomes executable, and Section 2 shows that each adjacent literature supplies one part of the answer and stops short of the others: codification explains externalisation but not the standing of a superseded version; boundary-object theory explains coordination across communities but assumes an object open to interpretation; data governance specifies authority but not replay mechanics; and temporal data management supplies those mechanics as a database technique rather than as an account of organisational knowledge. The gap is their intersection: governance knowledge that is at once codified, time-bounded, machine-executed and organisationally contested.

1.1 Contribution

We offer a *temporality-focused conceptual elaboration* of KM in executable governance settings. We claim an elaboration rather than an extension of KM theory: we do not replace codification, organisational memory or boundary objects with new constructs, but specify what happens to one established construct when the artifact carrying it becomes executable and its validity time-indexed. Section 2.2 develops the resulting inversion, in which an object coordinates by being unambiguous when applied and negotiable when reviewed. Three supporting pieces of design knowledge follow, each anchored in the artifact: five design principles, five artifact-level consequences that we verify directly, and five organisational propositions stated with the conditions under which they should hold and the observations that would falsify them. The case is the evidentiary anchor for that design knowledge, not a contribution in itself.

1.2 Research questions

The gap yields three questions, in the order the paper answers them. If governance knowledge must be both stable and revisable, we first need to know by what means an artifact holds both commitments at once (**RQ1**: how do market data governance artifacts stabilise shared knowledge while enabling change over time?). That requires identifying the mechanisms which make temporal structure explicit rather than implicit, and asking what they do and do not make auditable (**RQ2**). Finally, to be useful beyond one setting those mechanisms must be stated as transferable design knowledge and as claims others can test (**RQ3**).

1.3 Scope

The artifact also contains a multi-agent reinforcement learning (RL) training stack. This paper does not analyse learning algorithms; its subject is the governance layer – policy expression, reference data, quality controls, audit outputs – surrounding them. Section 2 sets out the theoretical architecture; Section 3 states what was done, by whom, and to which version of the artifact; Section 4 reports what the artifact contains; Section 5 moves from evidence to principles and propositions and makes the chain between them explicit; Sections 6–8 discuss, bound and conclude.

2 Theoretical Background

This section builds one theoretical architecture rather than a survey, and the hierarchy among its parts is worth stating before the parts themselves. Boundary-object theory is the principal explanatory foundation, because the artifacts we study coordinate interpretation across roles; the elaboration we offer is an amendment to it, and everything else is in support. Three literatures play defined supporting parts. Codification research supplies the process account of how operational knowledge becomes explicit and reusable, but stops short of what happens to a superseded version. Work on *ba* and on classification supplies the conditions under which a codified object stays interpretable and contestable, and the reason its defaults are not neutral. Data governance, quality and provenance research supplies the domain vocabulary for what such artifacts are for, and the outcome constructs against which they are judged. One further literature, valid-time data management, supplies the mechanism of time-indexed validity without any account of it as organisational knowledge. Each is used for that job and no other, so that a single technical mechanism is not relabelled several times over.

2.1 Codifying operational knowledge

KM research treats knowledge as a strategic resource and asks how organisations convert what is known into forms that can be shared and applied [Alavi and Leidner, 2001]. In production data environments the hard part is not discovery but stabilisation: an operational rule must mean the same thing to the analyst reading it, the engineer implementing it and the system applying it.

Codification achieves this, but is never terminal, since codified knowledge is continually recombined as contexts change [Nonaka and Takeuchi, 1995, Nonaka and von Krogh, 2009].

Temporality is where codification and organisational memory meet, and is treated thinly in both. Superseding a rule must not erase it: an organisation that cannot say what its rule required in 2015 cannot settle a dispute about a trade booked in 2015. The requirement is not that history be archived but that the historical rule remain *applicable* – that the past can be re-run under the rule that governed it. We call this replayability, and it distinguishes a versioned governance artifact from a document with an archive behind it.

2.2 Boundary objects, and what happens when they execute

Star and Griesemer [1989] showed that cooperation across communities with different aims requires not consensus but shared objects plastic enough to be read locally and robust enough to hold a common identity, and Carlile [2002] sharpened this by distinguishing boundaries at which knowledge must be transferred, translated or transformed. Governance artifacts are boundary objects in this sense: a fee schedule is read by a compliance officer as an obligation, by an engineer as a computation and by a trader as a cost.

Time has entered this literature from the other side. Yakura [2002] analyses timelines as *temporal* boundary objects: artifacts whose subject matter is time and whose coordinating power comes from interpretive flexibility, since groups holding incompatible assumptions about duration can still work from one chart. Chang et al. [2013] extend this to megaproject schedules. In both, the artifact *depicts* time and remains open to reading.

The artifacts studied here occupy the opposite position. Their subject matter is not time; their *validity* is indexed by time, and a resolver selects among versions without human interpretation. Ambiguity is designed out at the point of application, so that a given date yields the same rule on every run, and designed back in at the point of review, where the window, the label and the defaults can be challenged before the next version takes effect. This is what we mean by a time-bound executable boundary object, and the inversion is the substance of our elaboration: coordinating power comes not from flexibility at the moment of use but from the guarantee of its absence, paired with preserved contestability elsewhere. The mechanism is not itself novel – valid-time modelling has been standard in temporal databases for decades [Snodgrass, 1999] – but it has been treated as a data-management technique rather than as a way of holding knowledge across a boundary that is temporal rather than communal.

2.3 Keeping codified rules interpretable

An executable rule that no one can question is not governance. Nonaka and Konno [1998] describe *ba* as the shared context in which knowledge is created and activated, and note that knowledge separated from that context loses the meaning that made it usable. Repositories, configuration files and review workflows can serve as a *cyber ba*: a shared, inspectable place where assumptions are externalised and, through repeated use, internalised. Bowker and Star [1999] supply the

counterweight: classifications are never neutral, their defaults quietly determining what counts as normal. In executable governance a default value, a fallback branch and an unimplemented hook are therefore governance positions rather than implementation details, and the artifact must keep them legible if review is to function at all.

2.4 Governing data through artifacts

Data governance research specifies decision rights, stewardship and accountability [Khatri and Brown, 2010, Otto, 2011, Abraham et al., 2019], and Tallon et al. [2013] reframe the information artifact itself as the unit of governance, which licenses reading a configuration file as an object of KM rather than as infrastructure. Data quality research supplies the consumer’s standpoint, quality being multi-dimensional and defined by fitness for use [Wang and Strong, 1996, Pipino et al., 2002], and Chengalur-Smith et al. [1999] show that the availability of quality information itself changes decisions. Provenance research supplies the vocabulary of derivation history [Simmhan et al., 2005, Buneman et al., 2001], with standards work aiming at interoperability [W3C Provenance Working Group, 2013]. Full derivation graphs remain rare in practice; Section 5.1 keeps them apart from the lighter analogues most operations run. In regulated institutions, expectations that governance be operational rather than aspirational are explicit [Basel Committee on Banking Supervision, 2013].

2.5 Why market data is a demanding setting

Market data concentrates the tension. Prices and reference data arrive from several vendors with non-identical conventions, so meaning must be reconciled before it is applied [Tallon et al., 2013]. Calendars and settlement lags differ across markets and change over time, making them governed reference data rather than implementation detail, since a misaligned timestamp corrupts analytics silently. Brokerage caps and statutory levies are defined by named instruments with commencement dates, so a rule valid for a period must stay available for replay long after it is superseded. Identifiers and corporate actions shift beneath the data. And consumers act on the quality signals they receive [Chengalur-Smith et al., 1999, Redman, 1998], while supervisory expectations make “what rule applied when” a routine question [Basel Committee on Banking Supervision, 2013].

2.6 Synthesis

Governance succeeds when knowledge is not only documented but reviewable, executable and replayable, and no single literature accounts for all four at once. The remainder develops their intersection using an artifact in which all four are present.

3 Method

3.1 What this study is

This is a *retrospective analysis of an existing design artifact* from which design knowledge is abstracted. It is informed by design science research (DSR) but does not report a DSR cycle: we did not identify a governance problem, specify objectives, build an artifact to meet them and evaluate it in an organisation. The artifact existed first, built for another purpose; we analysed it afterwards and reconstructed the objectives it turns out to satisfy. We use the DSR literature in two ways and no others – [Gregor and Hevner \[2013\]](#) to position what kind of knowledge we claim, and the activity vocabulary of [Peffer et al. \[2007\]](#) and [Hevner et al. \[2004\]](#) to discipline the abstraction from instance to principle. In that typology the output is nascent design theory. Figure 1 maps the analysis onto that vocabulary and marks where it departs.

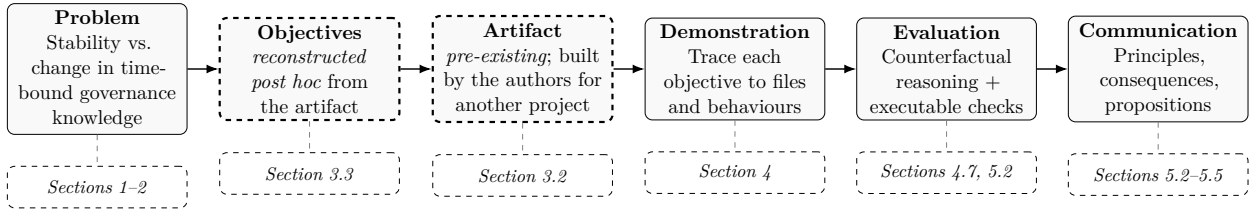


Figure 1: The analysis mapped onto the activity vocabulary of design science research [[Peffer et al., 2007](#)], with the corresponding manuscript sections below. The two boxes with heavy dashed borders mark where this study departs from a design science cycle: the artifact was not built to meet stated objectives, and the objectives were reconstructed from the artifact afterwards. Source: Authors’ own work.

3.2 The artifact and its provenance

The artifact is a Python repository implementing an equity trading ecosystem for a frontier market: data ingestion, feature construction, an execution environment with fees and price limits, and reporting. *The authors wrote it.* It was built for an earlier, separate project on multi-agent RL for portfolio allocation under trading frictions, and the mechanisms analysed here were built as engineering necessities of that project. They were not designed to answer this paper’s questions, and no governance objective was written down before they were built. The six objectives of Section 3.3 are therefore reconstructions: they name what the artifact does, not what it was told to do.

Selection was purposive and, to be plain, opportunistic: we chose an artifact to which we have complete access, including access to intent that a reader of the code alone could not recover. That access is the study’s principal asset and its principal threat, and Section 3.5 treats it as both. The artifact is nonetheless information-rich in ways independent of our access [[Yin, 2018](#)]: it carries heterogeneous governance objects that must stay stable yet change; its temporal structure is substantive, encoding four regulatory periods with commencement dates; and it contains visible incompleteness. Artifacts that advertise their boundaries are more useful for theorising than polished

ones, because they show where designers stopped.

Version boundary. The analysis covers one fixed state of the repository. Exactly the modules, configuration and fixtures analysed – and no other part of the original repository – are available for inspection as a supplementary file accompanying this submission, with the validation harness of Section 4.5 and a report mapping each claim in Section 4 to the file and line range evidencing it. The harness is the only component written for this paper. The package will be released publicly under a permanent identifier on acceptance.

3.3 Objectives reconstructed from the artifact

Read as an answer, the artifact implies six objectives for executable governance knowledge: keep policy expression separate from policy execution; make validity windows explicit; resolve rules deterministically; make data contracts inspectable; make defaults visible and contestable; and emit truthful audit artifacts, including known incompleteness. Section 4.5 evaluates the artifact against them. They appear before the findings because that is the order in which they can be understood, not because they preceded the design.

3.4 Coding protocol

Each governance-relevant file is treated as a unit of evidence and coded in four steps. A unit is *included* if it materially encodes, resolves, validates or reports on a governance-relevant property. It is *excluded* if it relates exclusively to learning algorithms, optimisation or model architecture, which fall outside the scope of Section 1.3; build and developer tooling is excluded unless it encodes reproducibility controls. Each included unit is *categorised* by its dominant role among policy object, reference data, control, and audit artifact, with cross-coding recorded where it serves more than one. Its behaviour is then *abstracted*: what it does that makes it governance-bearing, such as selecting deterministically by date, bounding a value, or emitting a stable output schema. Those behaviours are what Section 5 generalises.

The four categories derive from the literatures of Section 2.4: policy and control from data governance and risk-data principles [Khatri and Brown, 2010, Basel Committee on Banking Supervision, 2013], reference data from data management practice, audit artifact from the provenance and quality literatures [Simmhan et al., 2005, Wang and Strong, 1996]. They were refined by re-coding boundary cases; the calendar, first coded as reference data, is also a control when used to validate trade dates, and is recorded as both.

3.5 Reflexivity, traceability and validity

Coding was performed by the authors, who are also the artifact’s designers. This creates two risks: charitable reading, where a designer credits the artifact with intent the code does not carry, and retrospective coherence, where scattered engineering decisions are narrated as a design programme

they never were. Three countermeasures follow. Every finding cites a file a reader can open, and the inspection package exists so the check does not depend on our summary. We report mechanisms that are present but unused as prominently as those that work, since these are what a charitable reading would smooth away. And we state that the objectives are post hoc reconstructions. What we cannot do is compute inter-coder reliability: there is one artifact and one coding team, and traceability substitutes for it imperfectly.

Validity threats are addressed accordingly. *Construct* validity rests on mapping each concept to a named operationalisation; *internal* validity on counterfactual reasoning about alternative designs (Section 5.2) and, for artifact-level claims, on executable checks; *external* validity is analytic rather than statistical, generalising to principles and propositions that transfer to comparable settings [Yin, 2018, Gregor and Hevner, 2013], with Section 7 stating where transfer should fail.

3.6 Evidence and accuracy constraints

Findings are restricted to what repository files directly evidence. The artifact flags its fee values as estimates for academic use; we treat them as examples of governance codification, not as authoritative market facts. Mechanisms present only as intent are reported as incomplete and treated as negative knowledge rather than capability.

4 Findings

This section reports what the artifact contains and what executing it establishes; interpretation is deferred to Section 5. Each subsection cites the module concerned, and the inspection package maps every claim below to a file and line range.

4.1 Effective-dated fee policy

The fee configuration defines trading fee periods, each carrying `effective_from` and `effective_to` dates and a `regulatory_period` label naming the instrument that set it. Four periods are encoded, spanning 2002–2012, 2012–2016, 2016–2022 and 2022 onwards (Table I). Resolution is performed by a small module in which an `is_active` predicate tests the date bounds and `resolve_effective_item` selects the most recent active candidate, so the answer for a date is a function of the date alone. The fee engine loads this configuration, supports both the current nested structure and an earlier flat one, optionally applies day-trading rebates (a mode the main wiring does not enable), and returns an itemised breakdown rather than one opaque number. It bounds the brokerage multiplier against a rate normalised to the regulatory maximum, so misconfiguration cannot produce a charge above the cap. The resolver also excludes entries marked `status:projected`; no such entry exists in the configuration, making this capacity for a change not yet made rather than a rule in force.

Table I: Standard-rate fee periods encoded in the artifact configuration. Values are as defined there and are flagged in the artifact itself as estimates for academic use; they are not authoritative fee schedules. Source: Authors’ own work.

Effective window	Label	Brokerage max (up to 100k)	Brokerage max (over 100k)
2002-01-01 to 2012-08-16	Pre-2012 estimates	0.0178	0.0150
2012-08-17 to 2016-03-10	Legal Notice 88/2012	0.0178	0.0150
2016-03-11 to 2022-08-11	Legal Notice 35/2016	0.0176	0.0136
2022-08-12 to (open)	Legal Notice 135/2022	0.0176	0.0136

4.2 Reference data, contracts and defaults

Three further objects carry governance knowledge. The calendar module fixes the session at 09:00 to 14:00, generates holidays per year from a fixed list with weekend adjustment, and computes settlement by advancing three *trading* days, skipping weekends and holidays; it therefore encodes not only a number – a T+3 lag – but its operational interpretation, which is the part usually left tacit and usually got wrong.

The intraday data contract is communicated by example: a CSV whose header is **date,time,ticker,close**, with **time** an hourly bucket between 09 and 14, accompanied by a generator producing deterministic synthetic bars and a liquidity-tier fixture. There is no validation gate, so the contract is carried by exemplar and exercised by fixture – a weaker guarantee than validation and a stronger one than a comment.

The feature module defines a stable schema of fourteen features per ticker, in a fixed order exposed by an accessor rather than assumed by callers, and its defensive controls are deterministic: missing-data paths yield fixed defaults, predominantly zeros, with liquidity defaulting to the midpoint of 0.5; remaining NaNs become 0.0; values are clipped to $[-3, +3]$ and divided by three, bounding output in $[-1, +1]$. Each is a governance decision – what an absent value shall be taken to mean – expressed as arithmetic.

4.3 Execution constraints, and what is not implemented

The trading environment adds a price-limit clamp of $\pm 10\%$ against the previous day’s close and constructs a separate fee engine per agent. Two boundaries are visible in the same code. First, VAT on brokerage is enabled unconditionally even though a per-agent VAT rate is defined in configuration and carried through the training driver into each agent’s fee settings, where it is never read: the field is plumbed, not honoured. Second, a settlement queue and lag parameter exist, and a fuller queue abstraction exists in the calendar module, but the environment’s settlement hook is an empty stub and the abstraction is never wired in. These are findings rather than defects: a visible stub states a boundary, and a silently omitted one would not.

4.4 Reporting as an audit output

The analysis module converts a run history into summaries and plots, writing a human-readable markdown summary and a machine-readable metrics CSV under stable names with a stable column order, per-agent columns sorted rather than in dictionary order. The separation of the two forms, and the stability of both, is what lets them function as a record rather than a log.

4.5 Artifact-level validation

To evaluate the artifact against the six reconstructed objectives we wrote a harness exercising them directly. Five checks are reported in Table II and all pass. V1 resolves the fee rule on both sides of each of the four regulatory transitions and compares the selected period against the expected label. V2 computes settlement for eight trade dates spanning midweek cases, weekends, holiday clusters and year end, against an independent re-implementation of the advance. V3 regenerates the fixtures and checks the output column by column against the documented schema. V4 passes ten thousand values, containing 10% NaNs and 15% out-of-range entries, through the defensive transform and measures what it did. V5 drives the reporting routine twice on identical input and compares the emitted files byte for byte.

These evaluate the artifact; they are not tests of the propositions in Section 5.4, and nothing about organisational behaviour follows from them. Two boundaries deserve statement. V1’s expected labels are written by hand from the configuration, so it validates the resolver, not the fee data the artifact flags as approximate. V2’s two implementations share one holiday set, so it validates the settlement advance, not the calendar’s fidelity to any real exchange. The harness is deterministic: run in the environment pinned by the repository, and again under substantially newer numerical dependencies, it produced an identical results object, which is the sense in which Table II is independently checkable.

5 From Evidence to Design Knowledge

We now move from what the artifact contains to what can be abstracted from it. Because auditability is the outcome most easily over-claimed, we fix its meaning first.

5.1 Disambiguating auditability

“Auditability” is used loosely enough to hide real differences. Table III separates five capabilities and states what the artifact supports for each. Its position is that of a *lightweight provenance analogue*: it answers “what rule applied when” through inspectable configuration and deterministic resolution, but maintains no derivation graph [Simmhan et al., 2005, W3C Provenance Working Group, 2013] and is not a compliance-grade implementation of any supervisory expectation.

Table II: Artifact-level validation results, produced by the harness in the inspection package. Each check evaluates one of the objectives reconstructed in Section 3.3; none is a test of the propositions in Section 5.4. Source: Authors’ own work.

ID	Check	Objective evaluated	Result
V1	Fee resolver at regime boundaries	Deterministic rule resolution	8/8 boundary dates resolve to the expected period across the four transitions, day before and day after each.
V2	T+3 settlement advance	Explicit validity windows	8/8 trade dates settle correctly across weekends, holiday clusters and year end, against an independent re-implementation.
V3	Schema conformance of fixtures	Inspectable data contracts	Output matches the documented columns, hours, tickers and tier labels; 540 intraday and 3 liquidity rows.
V4	Defensive-transform sanitisation	Visible and contestable defaults	On 10,000 values with 10% NaNs and 15% out-of-range: 10.00% replaced, 15.27% clipped, output in $[-1, +1]$.
V5	Audit-output stability	Truthful audit artifacts	Both files byte-equal across two runs on identical input; CSV header begins with the eight documented columns (25 total).

Table III: Five capabilities commonly gathered under “auditability”, and what the artifact supports for each. Source: Authors’ own work, drawing on [Simmhan et al. \[2005\]](#), [W3C Provenance Working Group \[2013\]](#), [Basel Committee on Banking Supervision \[2013\]](#).

Capability	Working definition	What the artifact supports
Technical reproducibility	The same inputs and code yield the same outputs across runs.	Supported: seeded fixtures and validated byte-stable outputs.
Provenance / lineage	A recorded derivation history of a data product: sources, transformations, agents.	<i>Lightweight analogue only.</i> Effective-dated configuration and itemised outputs permit reconstruction by reading; no derivation graph is maintained.
Audit trail	A persistent, time-ordered record sufficient for after-the-fact review.	Supported: stable summary and metrics files persisted as outputs.
Explainability of rule application	The ability to say why a particular rule applied to a given input.	Supported: deterministic effective-dated selection and itemised cost breakdown.
Regulatory auditability	The ability to satisfy supervisory expectations on data integrity and reporting.	<i>Not claimed.</i> Properties are consistent with such principles [Basel Committee on Banking Supervision, 2013] but this is not a compliance-grade implementation.

5.2 Design principles

Five principles generalise the behaviours of Section 4. Each is stated with the alternative it displaces, since a design principle with no plausible rival is a description.

Pattern A – effective-dated policy objects. Represent policy as versioned objects with explicit validity windows, resolved deterministically by date. The rival edits policy in place: cheaper, and it loses the past, since once the 2016 rates are overwritten a dispute about a 2019 trade cannot be settled from the system that booked it. Effective dating makes change additive and turns review into examination of the delta.

Pattern B – contract-first data interfaces. Define input schemas explicitly, align ingestion to them, and pair each with a fixture so pipeline changes can be exercised without proprietary data. The rival is the artifact’s own practice, a schema communicated by exemplar with no validation gate, and it is instructive because it half-works: the contract is discoverable but drift is undetected, so a vendor’s silent column change becomes a modelling problem rather than an ingestion error.

Pattern C – instrumented defensiveness. Pair imputation, clipping and normalisation with indicators quantifying how much sanitising occurred. The rival is silent robustness, again the artifact’s state: the transform is correct and says nothing, so a rise in missingness is absorbed rather than reported [Chengalur-Smith et al., 1999, Wang and Strong, 1996].

Pattern D – audit-first outputs. Treat summaries, configurations and metrics as persisted artifacts with stable names and schemas, not as logs. The rival is ephemeral output, which makes every retrospective question an archaeology exercise.

Pattern E – explicit negative knowledge. Record in the code where assumptions are simplified or mechanisms unimplemented. The rival is silent omission: a stub that does nothing tells a later reader settlement was considered and deferred, where its absence would let the same reader assume settlement was handled.

5.3 Artifact-level consequences

Each principle has a consequence at the level of the artifact: a property of the built thing, verifiable by inspection or execution, not to be confused with an organisational outcome. Rule selection becomes a reproducible function of the input date (D1, checked by V1); conformance becomes checkable without proprietary data (D2, V3); the cost of robustness becomes quantifiable (D3, V4); two runs become comparable, because outputs are byte-stable on identical input (D4, V5); and boundaries become discoverable without execution (D5). None of these says anything about whether an organisation using the artifact governs better.

5.4 Organisational propositions

The propositions below take that step. They are theoretically derived, not established by the case, and each states the mechanism it posits, the conditions under which it should hold, a moderator,

and what would falsify it. We state falsification conditions because a proposition that cannot fail is not worth testing.

P1 (temporal traceability). *Where policy is encoded as effective-dated objects, retrospective questions about which rule applied are resolved faster and with less dispute than where policy is edited in place.* The mechanism is the elimination of reconstruction work: the answer is looked up rather than rebuilt from archives and recollection. This requires that regimes change and that disputes are adjudicated retrospectively, and regime-change frequency moderates the effect. It fails if such teams show no reduction in time-to-resolution for retrospective queries against comparable teams, controlling for that frequency.

P2 (coordination without translation). *Contract-first interfaces reduce reliance on tacit coordination between data producers and consumers by giving each role an object it can validate independently.* The mechanism is boundary-object translation made explicit [Star and Griesemer, 1989, Carlile, 2002]: schema and fixture let engineer and analyst each verify their own side without negotiating. This requires genuine role separation and is moderated by upstream heterogeneity, contracts earning their cost where vendors disagree. It fails if onboarding time and reconciliation effort are unchanged after contracts are introduced under those conditions.

P3 (signalled quality and upstream repair). *Defensive transformations improve robustness but suppress the signal that would prompt repair; pairing them with quality indicators restores it.* The mechanism is the one Chengalur-Smith et al. [1999] demonstrate for decisions generally: visible quality information changes behaviour and invisible quality information cannot. This requires an upstream owner who can act, and that ownership also moderates it, since indicators without an addressable owner generate noise rather than repair. It fails if instrumented teams detect and repair upstream defects no faster than uninstrumented teams with equivalent ownership.

P4 (audit artifacts as organisational memory). *Persisting stable, structured outputs as first-class artifacts increases an organisation's capacity to review its own past decisions, beyond what the underlying data alone supports.* The mechanism is memory externalised in comparable form: stable schemas make two moments commensurable and unstable ones do not. This requires that outputs be retained and reachable, moderated by schema stability, since a drifting schema degrades the archive it appears to build. It fails if such organisations reconstruct past decisions no better than those retaining only raw inputs and code.

P5 (negative knowledge and calibrated trust). *Explicit documentation of what is not implemented increases safe reuse by bounding the applicability reusers assume.* The mechanism is calibration: a visible boundary prevents the inference of completeness that silence invites [Bowker and Star, 1999]. This requires reuse beyond the original context, moderated by the reuser's distance from the original team, since proximity substitutes for documentation. It fails if reusers of boundary-documented artifacts make as many out-of-scope applications as reusers of otherwise identical artifacts without it.

5.5 The chain from evidence to proposition

Table IV sets out, for each principle, the artifact evidence it rests on, the category that evidence was coded into, the mechanism inferred, the artifact-level consequence with the check establishing it, and the proposition that follows. It is readable in either direction, and reading it backwards is the more useful exercise: it shows exactly where each proposition stops being evidenced and starts being argued, at the boundary between the fifth column and the sixth.

Table IV: The analytical chain, one row per design principle: artifact evidence, the category it was coded into, the mechanism inferred, the artifact-level consequence with the check establishing it, and the proposition that follows. The break between the last two columns is where evidence ends and theoretical derivation begins. Source: Authors' own work.

Evidence	Category	Inferred nism	mecha-	Consequence (check)	Proposition
<code>fees.yaml</code> + <code>resolver.py</code> : four dated regimes, most- recent-active selection	Policy ob- ject; con- trol	Validity is indexed and is a function of the date, so superseded rules retain standing	time- selection	D1 Rule selection is re- producible across runs (V1) → Pattern A	P1 Effective-dated policy resolves retro- spective questions faster and with less dispute than in-place editing
<code>intraday_ prices_ hourly.csv</code> + generator: exemplar schema with fixtures, no validation gate	Reference data; fix- ture	A contract carried by example is discover- able by every role but detects no drift	D2 Conformance is checkable without pro- prietary data (V3) → Pattern B	P2 Contract-first interfaces reduce tacit coordination where roles are separated	
<code>feature_ engineering_ v2.py</code> : fixed defaults, NaN substitution, clip and rescale	Control	Robustness silently converts quality prob- lems into admissible values	D3 Sanitisation rates are measurable and out- put is bounded (V4) → Pattern C	P3 Defensive transforms suppress the repair signal unless paired with indica- tors	
<code>plot_ training_ results.py</code> : stable names, stable sorted column order	Audit arti- fact	Outputs designed as records rather than logs make two mo- ments commensurable	D4 Outputs are byte- stable across runs (V5) → Pattern D	P4 Persisted structured outputs in- crease capacity to review past decisions beyond what data alone supports	
Empty settle- ment hook, unwired queue, unread VAT field, unused <code>projected</code> status	Control; audit arti- fact	Visible incomplete- ness states a boundary that silence would con- ceal	D5 Boundaries are dis- coverable by inspection without execution	P5 Explicit negative knowledge in- creases safe reuse by bounding assumed applicability	

5.6 The conceptual model

Institutional knowledge is externalised into executable boundary objects which hold two opposing forces at once: stability mechanisms (contracts, deterministic resolution, stable output schemas) keep meaning reliable for reuse, while change mechanisms (effective dating, additive configuration, version control) accommodate revision. Figure 2 shows the resolution pattern at the centre of that arrangement. The resulting audit artifacts close a learning loop, letting stakeholders review, contest and revise the objects without losing historical interpretability. That loop is what makes the arrangement a KM system rather than a configuration convention: without the review setting, the mechanisms produce compliance records nobody reads.

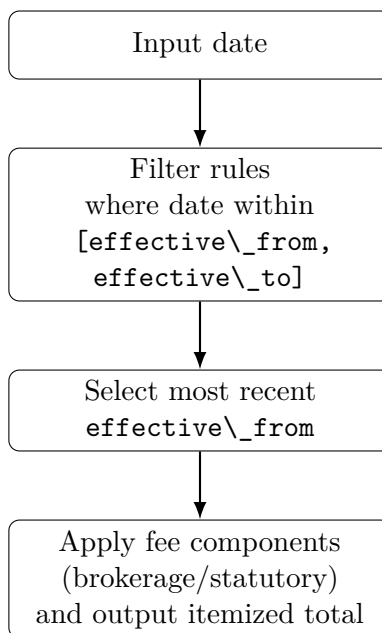


Figure 2: Effective-dated rule resolution pattern implemented by the repository’s resolver and fee engine. Source: Authors’ own work.

6 Discussion

6.1 What the elaboration adds, and what it does not

Three consequences follow from reading governance rules as time-bound executable boundary objects. First, the unit of analysis shifts from the policy to the policy *version*. Codification research has always accepted that codified knowledge is superseded; what changes is that the superseded version retains standing, because the organisation must still be able to act under it. Organisational memory here is not an archive of what was once believed but a live capacity to re-execute it.

Second, the inversion of Section 2.2 identifies a boundary the literature has not addressed. Star and Griesemer’s boundaries lie between communities, and Yakura’s temporal boundary objects still

coordinate communities, using time as their subject. The boundary crossed by an effective-dated rule lies between the present interpreter and a past state of the same organisation, and coordination across it cannot rely on interpretive flexibility because the two sides cannot negotiate: the past cannot be consulted. It relies instead on determinism at execution and preserved contestability at review. We claim this as an elaboration, not a new theory of boundary objects, and whether it generalises beyond executable settings is open.

Third, the neutrality of the shared space cannot be assumed. Defaults, fallbacks and stubs decide what counts as normal [Bowker and Star, 1999], and in an executable artifact they do so automatically and at scale. A cyber *ba* that does not surface its own defaults is not a space for knowledge work but a mechanism that produces answers.

6.2 Trade-offs

None of this makes executable governance unambiguously good, and the artifact shows four ways it can go wrong.

Codification strengthens replay but can weaken situated meaning. Deterministic resolution improves replayability at the cost of making governance into what the repository says rather than what the institution means; extracting a rule from practice reduces ambiguity for machines and increases it for humans facing the exceptional case [Nonaka and Konno, 1998]. Review rituals and recorded rationale are what keep executable governance interpretable.

Robustness can hide defects. Defensive transformation without indicators converts a data-quality problem into a model input: missingness and clipping stop being signals and become decisions taken silently, which is the condition under which drift goes unnoticed [Chengalur-Smith et al., 1999, Redman, 1998].

Auditability can become theatre. Audit-first outputs may be mistaken for complete governance, and this artifact shows the risk in its own body: it emits well-formed reports from a system whose settlement hook does nothing.

Boundary objects concentrate power in defaults. A schema exemplar and a fixed feature order define what is normal downstream [Bowker and Star, 1999], and the plumbed-but-ignored VAT field shows how a governance parameter can be present, visible and inert at once.

These cut against a claim we are careful not to make. Executable artifacts do not automatically reduce dependence on tacit knowledge. They can reduce certain forms of it when accompanied by review practice, recorded rationale, instrumented defaults and stewardship. Without those they substitute one tacit dependence for another: tacit trust in the artifact.

6.3 Practical implications

Governance knowledge is best packaged to serve humans and machines at once, which means a middle path between policy-as-document (readable, not reproducible) and policy-as-code (executable, not reviewable): policy expressed in configuration a non-programmer can read, resolved by small modules that can be tested. Two habits follow: handle policy changes as code changes

are handled, and treat defaults as governance decisions with named owners. Where catalogue and lineage platforms exist these patterns supply the last mile of executability; where they do not, as in many emerging-market institutions, they provide minimum viable traceability inside an ordinary repository workflow.

6.4 Machine learning systems

One application sharpens rather than extends the argument. When models are trained on historical data the governing regime is part of the training conditions, so the unit of analysis becomes the pair of policy version and model version: deploying a model trained under one regulatory period into another without an effective-dated layer reintroduces the drift Pattern A guards against. Silent sanitisation (Pattern C) and invisible stubs (Pattern E) are correspondingly more consequential when absorbed by an opaque model rather than surfacing as an error. We claim nothing about model performance; the point is that the governance layer, not the learning layer, determines whether such systems can be audited at all.

7 Limitations and Future Research

The study rests on one artifact, analysed by the people who built it, and can support claims only about how executable governance may be *structured* – not about adoption, effectiveness, user interpretation, audit outcomes, or reduced reliance on tacit knowledge [Yin, 2018, Gregor and Hevner, 2013]. The reflexive threat of Section 3.5 is mitigated by traceability and an inspectable file set, not eliminated: a reader can check what the code does, but not what we failed to notice in it.

The artifact’s coverage bounds the analysis further. Its fee values are flagged as academic estimates, its settlement processing is a stub, and corporate actions, vendor reconciliation and licensing constraints are absent – each a domain in which temporality behaves differently. A single-market frontier setting also limits transfer: we expect the patterns to hold wherever regimes change and history must be defensible, and to matter least where a stable regime makes replay a non-question.

Testing the propositions requires designs this study cannot supply: multi-site comparison of dispute-resolution outcomes before and after adopting effective-dated configuration (P1); cross-vendor work on onboarding and reconciliation effort under contract-first interfaces (P2); field or quasi-experiments contrasting instrumented and uninstrumented teams on upstream defect detection and time to repair (P3); studies of retained audit artifacts testing reconstruction capability (P4); and longitudinal qualitative work on teams adopting explicit boundary documentation (P5). Comparing machine learning pipelines that share an effective-dated policy layer with those that do not would extend the logic to Section 6.4. Mixed-methods designs combining repository inspection with interviews, audit-log analysis and outcome measurement are the natural instrument, because the gap being closed is exactly the one between artifact-level design knowledge and organisational

evidence.

8 Conclusion

The theoretical insight established here is narrow and, we hope, precise. Governance rules that are both codified and executed behave as boundary objects with an inverted relation to interpretive flexibility: their coordinating power at the moment of application comes from the guarantee that interpretation has been removed, while the flexibility boundary-object theory places in the object is relocated to the review setting where versions are contested. The boundary they span is temporal rather than communal – between an organisation and its own past states – and across it negotiation is impossible, which is why determinism does the work plasticity does elsewhere.

The design knowledge that travels is the five principles of Section 5.2 and the reasons they displace their rivals: version policy rather than edit it, specify contracts rather than exemplify them, measure what defensive transforms do, persist outputs as records rather than logs, and write down what is not implemented. These should transfer wherever rules change and past states must remain defensible, carrying the trade-offs of Section 6.2 with them.

What the case cannot establish is anything about organisations. One artifact, built by its analysts for another purpose, shows how such mechanisms may be structured and what they demonstrate about themselves; it shows nothing about whether teams using them govern better or trust more appropriately. Those claims are stated as propositions with falsification conditions attached, and require multi-site, longitudinal and experimental work before they can be believed. The organisational question is left open, deliberately.

References

- Rene Abraham, Joerg Schneider, and Jan vom Brocke. Data governance: A conceptual framework, structured review, and research agenda. *International Journal of Information Management*, 49: 424–438, 2019. doi: 10.1016/j.ijinfomgt.2019.07.008.
- Maryam Alavi and Dorothy E. Leidner. Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues. *MIS Quarterly*, 25(1):107–136, 2001. doi: 10.2307/3250961.
- Basel Committee on Banking Supervision. Principles for effective risk data aggregation and risk reporting (bcbs 239). Bank for International Settlements, 2013. URL <https://www.bis.org/publ/bcbs239.htm>. Accessed January 2026.
- Geoffrey C. Bowker and Susan Leigh Star. *Sorting Things Out: Classification and Its Consequences*. MIT Press, Cambridge, MA, 1999. ISBN 9780262024617.

- Peter Buneman, Sanjeev Khanna, and Tan Wang-Chiew. Why and where: A characterization of data provenance. In *Database Theory — ICDT 2001*, Lecture Notes in Computer Science, pages 316–330. Springer Berlin Heidelberg, Berlin, Heidelberg, 2001. doi: 10.1007/3-540-44503-X_20.
- Paul R. Carlile. A pragmatic view of knowledge and boundaries: Boundary objects in new product development. *Organization Science*, 13(4):442–455, 2002. doi: 10.1287/orsc.13.4.442.2953.
- Artemis Chang, Caroline Hatcher, and Jai Kim. Temporal boundary objects in megaprojects: Mapping the system with the integrated master schedule. *International Journal of Project Management*, 31(3):323–332, 2013. doi: 10.1016/j.ijproman.2012.08.007.
- I. N. Chengalur-Smith, D. P. Ballou, and H. L. Pazer. The impact of data quality information on decision making: an exploratory analysis. *IEEE Transactions on Knowledge and Data Engineering*, 11(6):853–864, 1999. doi: 10.1109/69.824597.
- Shirley Gregor and Alan R. Hevner. Positioning and presenting design science research for maximum impact. *MIS Quarterly*, 37(2):337–355, 2013. doi: 10.25300/MISQ/2013/37.2.01.
- Alan R. Hevner, Salvatore T. March, Jinsoo Park, and Sudha Ram. Design science in information systems research. *MIS Quarterly*, 28(1):75–105, 2004. doi: 10.2307/25148625.
- Vijay Khatri and Carol V. Brown. Designing data governance. *Communications of the ACM*, 53(1):148–152, 2010. doi: 10.1145/1629175.1629210.
- Ikujiro Nonaka and Noboru Konno. The concept of ‘ba’: Building a foundation for knowledge creation. *California Management Review*, 40(3):40–54, 1998. doi: 10.2307/41165942.
- Ikujiro Nonaka and Hirotaka Takeuchi. *The Knowledge-Creating Company: How Japanese Companies Create the Dynamics of Innovation*. Oxford University Press, New York, 1995. doi: 10.1093/oso/9780195092691.001.0001.
- Ikujiro Nonaka and Georg von Krogh. Perspective—tacit knowledge and knowledge conversion: Controversy and advancement in organizational knowledge creation theory. *Organization Science*, 20(3):635–652, 2009. doi: 10.1287/orsc.1080.0412.
- Boris Otto. Data governance. *Business & Information Systems Engineering*, 3(4):241–244, 2011. URL <https://aisel.aisnet.org/bise/vol3/iss4/6>.
- Ken Peffers, Tuure Tuunanen, Marcus A. Rothenberger, and Samir Chatterjee. A design science research methodology for information systems research. *Journal of Management Information Systems*, 24(3):45–77, 2007. doi: 10.2753/MIS0742-1222240302.
- Leo L. Pipino, Yang W. Lee, and Richard Y. Wang. Data quality assessment. *Communications of the ACM*, 45(4):211–218, 2002. doi: 10.1145/505248.506010.

- Thomas C. Redman. The impact of poor data quality on the typical enterprise. *Communications of the ACM*, 41(2):79–82, 1998. doi: 10.1145/269012.269025.
- Yogesh L. Simmhan, Beth Plale, and Dennis Gannon. A survey of data provenance in e-science. *ACM SIGMOD Record*, 34(3):31–36, 2005. doi: 10.1145/1084805.1084812.
- Richard T. Snodgrass. *Developing Time-Oriented Database Applications in SQL*. The Morgan Kaufmann Series in Data Management Systems. Morgan Kaufmann, San Francisco, CA, 1999. ISBN 9781558604360.
- Susan Leigh Star and James R. Griesemer. Institutional ecology, ‘translations’ and boundary objects: Amateurs and professionals in berkeley’s museum of vertebrate zoology, 1907–39. *Social Studies of Science*, 19(3):387–420, 1989. doi: 10.1177/030631289019003001.
- Paul P. Tallon, Ronald V. Ramirez, and James E. Short. The information artifact in IT governance: Toward a theory of information governance. *Journal of Management Information Systems*, 30(3):141–178, 2013. doi: 10.2753/MIS0742-1222300306.
- W3C Provenance Working Group. PROV-Overview: An overview of the PROV family of documents. W3C Working Group Note, 2013. URL <https://www.w3.org/TR/prov-overview/>. Accessed January 2026.
- Richard Y. Wang and Diane M. Strong. Beyond accuracy: What data quality means to data consumers. *Journal of Management Information Systems*, 12(4):5–33, 1996. doi: 10.1080/07421222.1996.11518099.
- E. K. Yakura. Charting time: Timelines as temporal boundary objects. *Academy of Management Journal*, 45(5):956–970, 2002. doi: 10.2307/3069324. Accessed August 2026.
- Robert K. Yin. *Case Study Research and Applications: Design and Methods*. Sage Publications, Thousand Oaks, CA, 6 edition, 2018. ISBN 9781506336169.