International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

A Hybrid LLM–Graph Architecture for Laboratory Billing Discrepancy and Overpayment Risk Detection

Sai Krishna Puli

Data Integrations & Architecture, LabCorp Drug Development Inc, Loganville, Georgia, USA

ORCID: 0009-0002-7470-5604

ABSTRACT: Mismatches between physician orders, specimen collection records, and billed procedure codes are a persistent and under-examined source of financial leakage in clinical laboratory and microbiology billing. Deterministic rule engines, such as the National Correct Coding Initiative (NCCI) maintained by the Centers for Medicare & Medicaid Services (CMS) to promote correct coding and reduce improper payments, reliably catch exact code-pair combinations that should not be billed together, but they have nothing to say about the free-text, clinically worded order language that microbiology testing routinely produces. We propose a hybrid architecture that pairs a deterministic rule-matching layer and a canonical order-specimen-test-claim linkage graph with a retrieval-constrained large language model (LLM) reconciliation component built to close that gap. The architecture spans seven layers: data ingestion and normalization; a linkage graph in which discrepancies surface as graph anomalies (orphan claims, unbundling, duplicate specimen use, temporal inversion); a Policy, Terminology & Clinical Knowledge Layer spanning coding rules, coverage policies, and clinical knowledge; a hybrid detection engine that combines rule matching, LLM-based semantic reconciliation, and statistical anomaly scoring; a retrieval-augmented generation (RAG) explanation layer that grounds every flagged case in cited rule or guideline text; a human review workflow; and a feedback loop that recalibrates detection thresholds and prompts from auditor decisions. The design rests on one non-negotiable rule: the LLM component never determines a billing code, and it is never asked to. Its narrowest and most consequential task is to answer a single bounded question, whether the observed, performed test semantically corresponds to the ordered test, while a deterministic crosswalk and validation pipeline separately generates every candidate code and finalizes every outcome, which reduces the risk of unsupported code generation while preserving traceability to validated catalog entries. The system is designed to detect laboratory billing discrepancies and prioritize potential overpayment and compliance risk for human review, not to detect fraud: a flag indicates that two or more records disagree in a way that affects payment, and any determination of intent remains a judgment for human reviewers and, where warranted, compliance or legal staff to make. We present this architecture as a design-science contribution, walk through the reconciliation component's prompt and output schema in detail, and discuss what we expect in terms of precision, explainability, and deployment trade-offs. Empirical validation against production claims data has not yet been carried out and is identified as the necessary next step.

KEYWORDS: healthcare billing compliance; retrieval-augmented generation; large language models; graph-based anomaly detection; clinical laboratory informatics; explainable artificial intelligence

I. INTRODUCTION

Billing discrepancies and overpayments cost the United States healthcare system tens of billions of dollars every year; independent surveys attribute losses across the broader category of healthcare fraud, waste, and abuse to somewhere between 3 and 10 percent of total healthcare expenditure (Nabrawi & Alanazi, 2023). Overpayment risk is the more precise frame for the problem this paper addresses: most mismatches between what was ordered, collected, performed, and billed are not evidence of intent to deceive, they are evidence of a data or process discrepancy that happens to result in an incorrect payment, and treating every discrepancy as presumptive fraud invites both false accusations and reviewer fatigue. Clinical laboratory and microbiology billing sits squarely in the exposed part of that spending. A single episode of care produces several loosely coupled records: a physician order, one or more specimen collection events, the tests that were run, and the procedure codes eventually billed. These records come from different systems, at different times, and in most healthcare organizations they are reconciled informally, if at all.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

Two features of this domain make automated discrepancy detection harder than it looks. First, microbiology orders are often written as free text or loosely structured comments rather than a single discrete procedure code, for example "urine culture, reflex to sensitivity if positive." Exact-match rule engines routinely fail on language like this because there is no single code to match against. Second, the relationship between an order, its specimen, the test performed, and the resulting claim line is relational rather than tabular. A discrepancy is usually a missing or duplicated link in a chain of custody, not an outlier value sitting in a single row of a spreadsheet.

We propose an architecture that takes both features head-on. It represents the order, specimen, test, and claim relationship as an explicit linkage graph, so structural discrepancy patterns can be queried directly rather than inferred, and it adds a retrieval-constrained large language model component that reconciles free-text order language against a controlled test catalog without inventing codes that were never billed or performed. Section 2 reviews related work in billing discrepancy and overpayment risk detection, drawing on the adjacent fraud-detection literature, and in retrieval-grounded language models. Section 3 sets out the design methodology used to arrive at the architecture. Section 4 walks through the architecture layer by layer. Section 5 closes with the contribution, its limits, and what empirical validation would need to look like.

II. LITERATURE REVIEW

This paper's focus, detecting billing discrepancies and estimating overpayment risk rather than adjudicating fraud, draws heavily on an adjacent and more established literature that is framed explicitly around fraud detection. That framing difference matters: fraud implies intent, which is a legal and investigative determination that sits outside the scope of an automated system, whereas a discrepancy or overpayment risk flag simply says that two or more records disagree in a way that affects payment. With that distinction in mind, Section 2.1 reviews the fraud-detection literature on its own terms, since the underlying pattern-recognition problem is largely shared, before Sections 2.2 through 2.4 turn to coding-compliance rule systems, retrieval-grounded language modeling, and the explainability concerns that motivate this paper's design.

2.1 Healthcare Claims Anomaly and Fraud Detection

Automated detection of anomalous healthcare claims has moved through several overlapping generations of methods, and situating this paper's architecture against that progression helps place the contribution. The earliest generation is rule-based: fixed logic, most visibly the CMS National Correct Coding Initiative discussed in Section 2.2, flags a claim only when it matches a predefined pattern such as a disallowed code pair. Rule-based systems are fast, fully explainable, and carry none of the risk that comes with a learned model, but they cannot adapt to a pattern nobody has written a rule for yet. A second generation is statistical: Johnson and Nagarur (2016) proposed one of the earlier multi-stage approaches, combining provider-, service-, and claim-level abnormality scores into a single risk measure and passing that measure through a decision-tree threshold, an early example of blending statistical scoring with a simple learned component rather than relying on either alone. A third generation applies general-purpose machine learning directly to claims features: Nabrawi and Alanazi (2023) show that supervised classifiers trained on structured claims data can identify anomalous billing patterns with reasonable accuracy, though they note that interpretability and the scarcity of confirmed labels remain real obstacles in compliance-sensitive settings. A fourth, more recent generation turns to graph-based machine learning, motivated by the observation that billing anomalies, fraudulent or not, tend to be relational rather than something a single row of tabular claims data can reveal; Yoo, Shin, and Kyeong (2023) compare graph neural networks against traditional machine learning for Medicare fraud detection and find that modeling the relationships among providers, beneficiaries, and claims materially improves detection over feature-based approaches alone, a finding that carries over directly to relational overpayment risk detection even though the two problems are conceptually distinct. The current state of the field is best described as hybrid: production systems increasingly compose several of these generations rather than committing to one, pairing deterministic rules for the violations they can express exactly with statistical or learned scoring for everything else, which is the same design principle this paper's own detection engine follows in Section 4.4.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011



Figure 1. Evolution of healthcare claims anomaly and fraud detection methods, from rule-based systems to hybrid pipelines.

2.2 Laboratory Billing and Coding Compliance

Laboratory billing compliance rests on several code systems and rule sets that were designed for different purposes and do not line up cleanly with one another. CPT, maintained by the American Medical Association, and HCPCS Level II, maintained by CMS, identify the procedure that was billed. LOINC, maintained by the Regenstrief Institute, identifies the observation that was ordered or reported. ICD-10-CM, maintained jointly by CMS and the National Center for Health Statistics, identifies the diagnosis that establishes medical necessity for the test (Centers for Disease Control and Prevention, National Center for Health Statistics, n.d.). None of these systems was built with the others in mind, and a discrepancy-detection architecture has to reconcile all three, plus specimen type, panel structure, reflex logic, add-on procedure rules, unit limits, and payer-specific coverage policy, before it can say whether a billed line is actually supported by what was ordered and performed.

The relationship between LOINC and CPT deserves particular attention because it is easy to get wrong in a way that looks reasonable and produces false discrepancy flags. LOINC's own documentation describes the original mapping effort's use case explicitly: LOINC codes were meant for ordering and reporting, CPT codes were meant for billing, and the resulting map runs in one direction, from LOINC codes to CPT billing codes, not the reverse (LOINC, n.d.). That same documentation notes that CPT is often less specific than LOINC, so a single CPT code can correspond to several LOINC codes representing different components, specimens, or result properties bundled under one billable procedure. Treating this relationship as a simple one-to-one lookup, which is the natural first instinct when building a crosswalk table, would silently misrepresent the domain and generate spurious mismatch flags every time a legitimate one-to-many case came up. We therefore treat this explicitly as a design requirement rather than an implementation detail: the crosswalk table introduced in Section 4.1 stores the LOINC-to-CPT relationship as a versioned, directional, one-to-many mapping, records which LOINC components a given CPT code is understood to cover, and never assumes the map can be safely inverted from a CPT code back to a single LOINC code.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

ICD-10-CM diagnosis codes attached to an order serve a different function again: they establish medical necessity rather than identify the test itself, and this distinction turns out to be more consequential than it first appears.

CMS is explicit that NCCI edits are correct-coding edits, not medical-necessity edits, and that the NCCI program performs no medical review and is not a prior-authorization program (Centers for Medicare & Medicaid Services, n.d.-b). Read literally, that statement means an architecture built only around NCCI-style code-pair logic is answering exactly one question, whether the codes on a claim are compatible with each other and cannot answer the separate questions of whether the service was covered for this patient or whether it was the clinically appropriate test to order in the first place. Collapsing these questions into a single rule set is a common source of both false positives and missed discrepancies, so we treat laboratory billing compliance as spanning three distinct policy domains, each with its own rule sources, its own owner, and its own failure mode, and each represented as a separate component of the Policy, Terminology & Clinical Knowledge Layer described in Section 4.3.

Coding compliance. This domain governs whether the codes on a claim are internally consistent with one another, independent of the patient or the payer. NCCI Procedure-to-Procedure edits and Medically Unlikely Edits operate here (Centers for Medicare & Medicaid Services, n.d.-d), alongside modifier logic that can legitimately override a PTP edit when a modifier documents a genuinely distinct service, unit limits on how many times a code can be billed per day, and bundling rules that determine when a panel code already includes a component that would otherwise be billed separately. Add-on codes, which cannot be billed on their own and are only valid alongside a specified primary procedure code, belong here as well. All of these are deterministic, code-level questions that do not require knowing the patient's diagnosis or the payer's coverage policy, which is why the rule-matching layer in Section 4.4 can resolve them without invoking the LLM reconciliation component at all.

Coverage and payment policy. This domain governs whether a covered, correctly coded service is payable for this patient, by this payer, at this rate. Medicare coverage rules are set through National and Local Coverage Determinations, and because LCDs are issued by individual Medicare Administrative Contractors rather than centrally, the same test can be covered in one MAC's jurisdiction and not another's; state Medicaid programs and commercial payers each layer their own coverage policy on top of that, so the same code combination and the same ICD-10-CM diagnosis pairing can be compliant for one payer and non-compliant for another. Payment amount is a related but distinct question: for Medicare, the Clinical Laboratory Fee Schedule sets the rate through a private payor rate methodology that updates on a recurring data-collection cycle (Centers for Medicare & Medicaid Services, n.d.-a), so a payment amount that changes across quarters for the same code is a CLFS update, not an overpayment signal. Ordering-provider enrollment sits in this domain too: CMS requires the ordering or referring provider's National Provider Identifier to appear on clinical laboratory claims and to match an active Medicare enrollment record and denies claims that fail this check regardless of whether the underlying test was medically appropriate (Centers for Medicare & Medicaid Services, n.d.-c). Because this domain varies by payer and by jurisdiction in a way coding compliance does not, the coverage-policies branch of the Policy, Terminology & Clinical Knowledge Layer in Section 4.3 partitions it by payer rather than treating it as one universal rule set.

Clinical appropriateness. This domain governs whether the test that was ordered, or reflexively triggered, was the right test given the clinical picture, a question neither of the first two domains can answer on its own. Panel composition determines which individual component results a single panel code is understood to include, so a DiagnosticReport aggregating several LOINC-coded Result/Observation nodes under one billed ClaimLine is expected behavior, not the unbundling pattern described in Section 4.2. Reflex testing, where a positive result on one test automatically triggers a second test under a standing protocol rather than a new physician order, requires the linkage graph to support conditional edges created after the fact and traced back to that protocol rather than flagged as an order-not-fulfilled discrepancy. Specimen requirements constrain which tests a given collection event can legitimately support, so the Order-Specimen edge has to carry specimen type as an attribute rather than a timestamp alone, and testing algorithms and clinical practice guidelines define the sequences a reasonable clinician would follow, which is the material the RAG explanation generator in Section 4.4 retrieves from when it has to justify why a given sequence of tests was, or was not, appropriate.

None of these three domains, individually or combined, resolves the free-text clinical language that precedes coding in the first place. A physician's order comment such as "urine culture, reflex to sensitivity if positive" must be resolved into the correct LOINC and CPT concepts before any of the checks above can even run, and that resolution step is exactly the gap the LLM reconciliation layer in this paper is built to close.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

2.3 Retrieval-Augmented Generation and Semantic Grounding in Large Language Models

The mechanism we use to close that gap draws on retrieval-augmented generation (RAG). Lewis et al. (2020) show that conditioning a language model's output on passages retrieved from an external, non-parametric knowledge source improves factual accuracy on knowledge-intensive tasks compared with relying on the model's parametric memory alone, an idea that traces back to the transformer architecture that made large-scale language modeling practical in the first place (Vaswani et al., 2017). The retrieval step in most RAG systems is implemented with dense passage retrieval, where queries and candidate passages are embedded into a shared vector space and matched by similarity instead of exact keyword overlap (Karpukhin et al., 2020), which suits the problem of matching loosely worded clinical order text against a structured test catalog reasonably well. Even with retrieval grounding, hallucination is not fully eliminated, and Ji et al. (2023) survey the range of ways generative language models can still produce content that is fluent but unfaithful to the source material, which is why the architecture described here also constrains the model's output format rather than trusting retrieval alone. Beyond reconciliation, general-purpose LLMs have shown they can encode a meaningful amount of clinical knowledge on their own (Singhal et al., 2023), though their performance still falls short of clinicians on harder reasoning tasks, which is one more reason to keep this architecture's LLM component narrowly scoped and advisory rather than autonomous. Interoperability among the source systems that generate order and result data is itself standardized through the HL7 Fast Healthcare Interoperability Resources (FHIR) specification, which defines the ServiceRequest and DiagnosticReport resources this paper's ingestion layer builds on (HL7 International, n.d.).

2.4 Explainability and the Research Gap

A final concern that runs through all this work is explainability. Tjoa and Guan (2021) survey the explainable AI (XAI) literature and note that adoption in high-stakes domains depends less on raw accuracy than on whether a model's output can be inspected and justified after the fact, which is especially true in a regulated billing environment where a wrong flag has real financial and provider-relationship consequences. Taken together, the literature offers two mature but largely separate toolkits: graph- and feature-based machine learning for claims-level anomaly detection, and retrieval-grounded language models for reconciling unstructured text against structured references. Existing program-integrity approaches can identify statistical outliers, but they do not necessarily establish whether an individual billed service is semantically supported by the clinical order, specimen, and performed test that preceded it, and that distinction is exactly what motivates the architecture proposed here. As far as we are aware, no prior work combines an explicit order-specimen-test-claim linkage graph with a retrieval-constrained LLM reconciliation layer specifically for laboratory and microbiology billing discrepancy detection, which is the gap this paper addresses.

III. METHODOLOGY

This work follows a design-science research methodology: the contribution is an artifact, an architecture, built to address a stated practical problem, rather than a statistical evaluation of a trained model against a labeled dataset. We arrived at the architecture in four stages.

First, we formalized the problem from documented billing-compliance requirements, mainly the structure of the CMS NCCI rule set (Centers for Medicare & Medicaid Services, n.d.-d) and the standard data artifacts produced across a laboratory episode of care: the HL7/FHIR order message, the specimen collection record, the microbiology result, and the corresponding institutional claim line. Resolving identity consistently across these artifacts is itself a classic record-linkage problem, and the probabilistic matching approach used in the ingestion layer follows the foundational framework laid out by Fellegi and Sunter (1969).

Second, we designed a layered architecture to separate concerns that need different update cadences and carry different risk tolerances: data normalization, relational linkage, static reference knowledge, active detection, human review, and feedback. This separation follows fairly standard practice in the adjacent fraud- and overpayment-detection system design literature, where deterministic and learned components tend to be composed rather than used on their own, since deterministic rules cannot resolve free text by themselves and learned components alone lack the auditability that regulators expect (Johnson & Nagarur, 2016; Nabrawi & Alanazi, 2023; Yoo et al., 2023).

Third, we specified the LLM reconciliation sub-architecture so that the model is never the component that determines a billing code, following the retrieval-constrained generation pattern of Lewis et al. (2020) and Karpukhin et al. (2020) and then narrowing it further. The LLM touches the pipeline at only two of eight stages, extracting structured clinical entities from free text and later answering a single bounded semantic-correspondence question, whether the observed test matches the ordered test, rather than choosing or ranking a code, while candidate retrieval, terminology validation,

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

CPT/HCPCS crosswalk mapping, and final validation are all deterministic steps that generate and finalize every code independently of the model. It is required to emit a fixed JSON schema rather than free-form text at both LLM stages, so every output is traceable back to a real catalog entry or a specific correspondence judgment rather than something the model produced from memory.

Fourth, we validated the architecture qualitatively by walking through four representative discrepancy patterns, orphan claims, unbundling, duplicate specimen use, and temporal inversion, and checking that each is expressible either as a query over the linkage graph or as a structured comparison inside the reconciliation layer. This walkthrough shows that the architecture covers known discrepancy patterns; it does not amount to an empirical evaluation of detection precision or recall on production data, which we flag in Section 5 as necessary follow-on work.

IV. RESULTS AND FINDINGS

The architecture we propose, shown in Figure 2, is organized into seven layers running from data ingestion through feedback-driven recalibration.

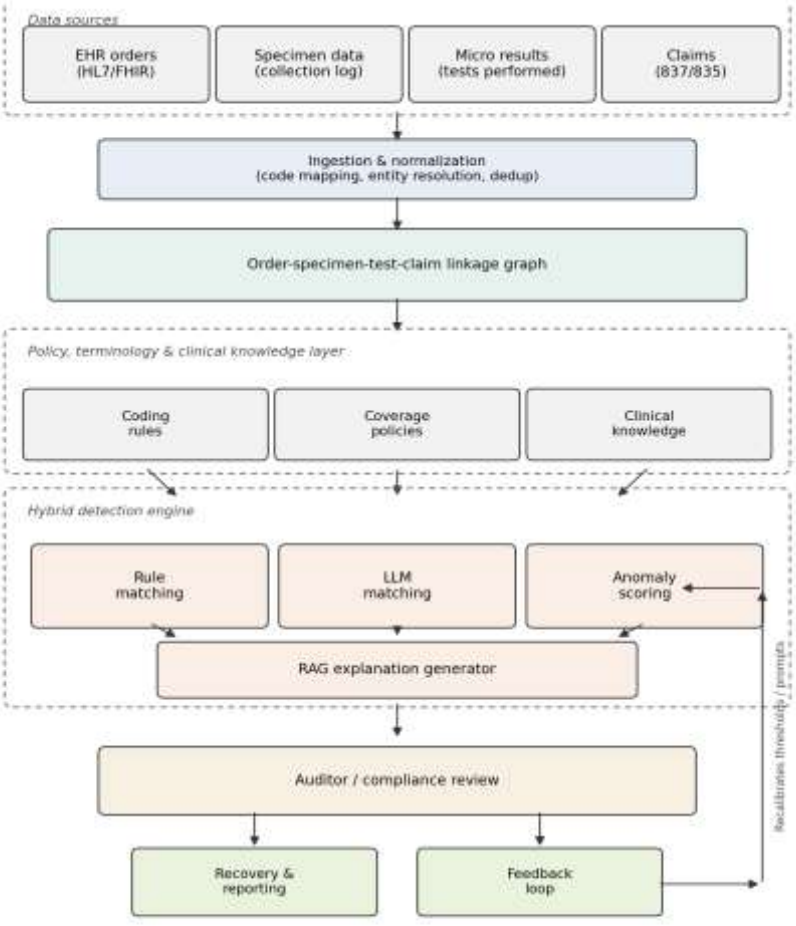


Figure 2. Proposed hybrid architecture for laboratory billing discrepancy detection.

4.1 Data ingestion and normalization

The ingestion layer consumes HL7 v2 order and result messages, FHIR ServiceRequest and DiagnosticReport resources (HL7 International, n.d.), and X12 837/835 claims transactions, and normalizes each into a common schema.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

Every test reference, whether it is an LIS internal code, a free-text order comment, a CPT/HCPCS code, or a LOINC code, gets mapped into a canonical test-concept identifier through a maintained crosswalk table, because a single billed CPT code often corresponds to a panel of distinct LOINC-coded results and the two vocabularies do not line up one to one. Patient and encounter identity are resolved deterministically wherever possible; where deterministic keys are missing, we fall back to probabilistic matching along the lines described by Fellegi and Sunter (1969), and any match produced this way is explicitly flagged as lower confidence downstream.

4.2 Order-specimen-test-claim linkage graph

Each episode of care is modeled as a directed graph. We keep the section's original name because order, specimen, test, and claim remain the backbone of the model, but the graph itself now spans a richer chain than that name alone suggests: Patient, Encounter, Order, Specimen, PerformedTest, Result/Observation, Instrument, DiagnosticReport, ClaimLine, Payer, and ProcedureCode, shown in Figure 3. Patient and Encounter anchor every downstream node to a verified identity and episode of care rather than leaving identity resolution implicit, which matters because the probabilistic matching fallback described in Section 4.1 can otherwise attach a specimen or result to the wrong person. Instrument records which device produced a result, which supports equipment-level audit and quality-control provenance without being treated as a billing signal. DiagnosticReport sits between individual results and the claim because that is how laboratory information systems aggregate observations before billing and separating it from ClaimLine is what lets the unbundling pattern below be stated precisely. Payer and ProcedureCode are broken out as first-class nodes, rather than flat attributes on ClaimLine, specifically so each can carry the full metadata schema from Table 1: a ProcedureCode node's source, version, and effective and expiration dates come directly from the coding-rules branch of the Policy, Terminology & Clinical Knowledge Layer, and a Payer node's edge determines which payer-partitioned branch of the coverage-policies knowledge base in Section 4.3 applies to that claim line.

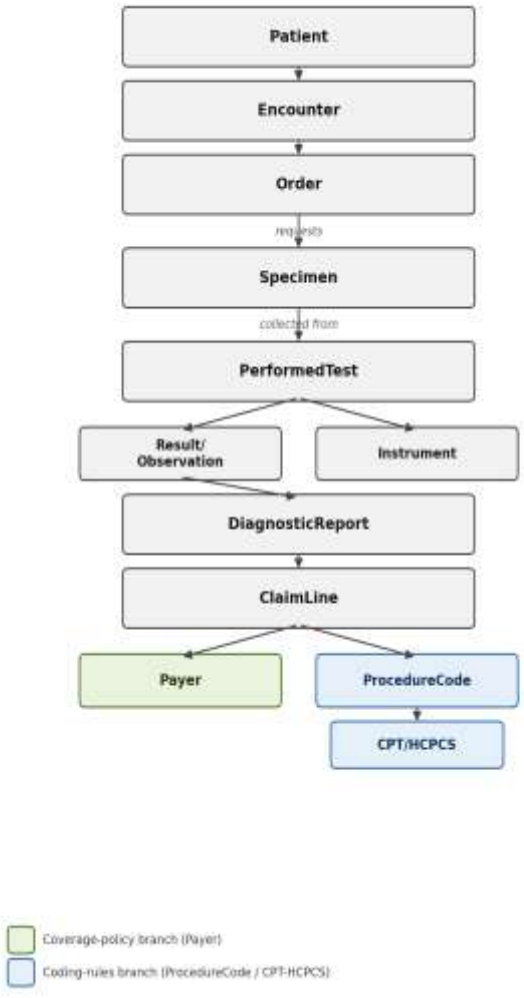


Figure 3. Expanded order-specimen-test-claim linkage graph, from patient identity through to procedure code.

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

The chain runs Patient to Encounter to Order, which requests a Specimen, which is collected from for a PerformedTest. PerformedTest branches into Result/Observation, which continues the chain toward billing, and Instrument, which terminates there as contextual metadata. Result/Observation feeds a DiagnosticReport, which is what a ClaimLine is actually billed against, and ClaimLine in turn branches into Payer and ProcedureCode, with ProcedureCode carrying the specific CPT or HCPCS value. This structure lets discrepancies be expressed as graph patterns instead of row-level anomalies. An orphan claim is a ClaimLine node with no inbound edge from any DiagnosticReport. Unbundling is a single DiagnosticReport node fanning out to multiple ClaimLine nodes whose ProcedureCode values represent components of one panel. Duplicate specimen use is one Specimen node linked to two functionally identical PerformedTest nodes inside a clinically implausible time window. Temporal inversion is any violation of the expected causal ordering from encounter to order to specimen collection to test performance to result to billing. The expanded schema also makes a pattern visible that the four-node version could not express at all: a patient or encounter mismatch, where a DiagnosticReport or ClaimLine traces back through the graph to a different Patient or Encounter than the one the original Order belongs to, which catches cross-patient specimen mix-ups and encounter misattribution rather than only billing-code discrepancies.

4.3 Policy, Terminology & Clinical Knowledge Layer

Following the three-domain distinction set out in Section 2.2, this layer is split into three separately maintained branches rather than one undifferentiated rule set, because each domain has a different authority, a different update cadence, and answers a different question about a flagged claim. We call it a Policy, Terminology & Clinical Knowledge Layer rather than a reference knowledge base because it carries terminology, not only policy: the coding-rules branch is where the CPT/HCPCS and LOINC vocabularies discussed in Section 2.2 actually live alongside the rules that govern them, which keeps code definitions and the rules that reference those codes versioned together rather than drifting apart in two separately maintained systems.

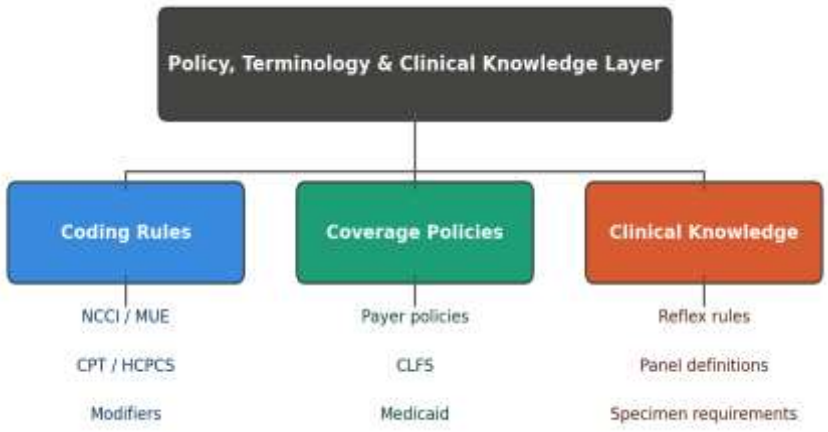


Figure 4. Structure of the Policy, Terminology & Clinical Knowledge Layer.

Coding rules. This branch encodes CPT/HCPCS and LOINC terminology together with NCCI Procedure-to-Procedure edits, Medically Unlikely Edits, and modifier-override logic (Centers for Medicare & Medicaid Services, n.d.-d) as structured, versioned records, each carrying an effective date and a source citation.

Coverage policies. This branch encodes payer policies, Clinical Laboratory Fee Schedule payment amounts (Centers for Medicare & Medicaid Services, n.d.-a), and state Medicaid coverage rules, partitioned by payer and jurisdiction since the same code combination can be compliant under one payer's policy and not another's.

Clinical knowledge. This branch encodes reflex-testing rules, panel definitions, and specimen requirements, for instance the conditions under which a positive culture automatically triggers susceptibility testing without a discrete separate order.

We keep these three branches separate because they come from different authorities and update on different schedules, and all three are used as retrieval context by the reconciliation and explanation layers described next.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

Separating the three branches is necessary but not sufficient. Within each branch, every individual artifact, whether it is a single NCCI edit, a payer's coverage policy, or a reflex-testing protocol, is stored with a standard metadata schema rather than as bare rule text. This is what turns the layer from a rule repository into something an auditor can actually interrogate: a flag is only as defensible as the specific, dated, jurisdiction-scoped source it was checked against, which is exactly the inspectable-and-justifiable standard Tjoa and Guan (2021) identify as the real bar for adoption in high-stakes domains. Table 1 sets out the eight fields every artifact carries.

Field	Purpose
Source	The originating publisher of record, for example CMS, a named Medicare Administrative Contractor, the AMA, Regenstrief (LOINC), or a payer's medical policy unit.
Version	The specific edition or release identifier of the artifact, for example NCCI PTP Edits v31.1 or CPT 2025, so a flag can be traced to the exact rule text that produced it.
Effective date	The date from which the artifact governs claims, distinct from the date it was published or ingested.
Expiration date	The date after which the artifact no longer applies, or an explicit open-ended marker if it is still current, so a superseded rule is never silently confused with a current one.
Jurisdiction	The geographic or administrative scope the artifact applies to, for example a specific MAC's region, a state Medicaid program, or national Medicare.
Payer	The specific payer or payer type the artifact is scoped to, since coverage and payment policy is not universal across payers even for the same code combination.
Authority	The issuing body's regulatory standing, for example national CMS policy versus a MAC's Local Coverage Determination versus a payer's internal medical policy, which determines how a conflict between two applicable artifacts is resolved.
Provenance	The retrieval chain from the original published source document or URL through any internal transformation, so a citation in a flagged case can be traced back to primary source text rather than a paraphrase.

Table 1. Metadata schema attached to every artifact in the Policy, Terminology & Clinical Knowledge Layer.

This schema does the work that lets Section 4.4's RAG explanation generator cite a specific rule rather than an undated general policy statement, and it lets an auditor reconstruct, months after a claim was flagged, exactly which version of which rule applied on the date of service, under which payer's policy, in which jurisdiction. Without it, two artifacts that happen to agree today but diverge next quarter, or a national policy and a MAC-specific override that happen to conflict, are indistinguishable to the detection engine, which is precisely the kind of silent failure a compliance audit cannot tolerate.

4.4 Hybrid detection engine

Rule matching runs the graph pattern queries above and the NCCI/MUE rule set deterministically against every claim line, producing high-confidence, fully explainable flags at low computational cost. Statistical anomaly scoring runs independently of the rule engine, looking for distributional outliers such as provider-level test-ordering rates that diverge from a peer cohort, a task that anomaly detection methods have handled reasonably well in other domains for some time (Chandola et al., 2009).

The LLM matching component is there to handle what the deterministic layer cannot: free-text and loosely coded orders. Its architecture goes one step further than a simple retrieval-then-constrain pattern: the language model is never the component that determines the billing code. Every code that can ever appear in a discrepancy flag is generated and finalized by a deterministic step; the LLM's role is bounded to interpreting clinical language and reconciling meaning against candidates it did not generate itself. Figure 5 lays out the eight-stage pipeline this component actually runs.

International Journal of Engineering & Extended Technologies Research (IJEETR)

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

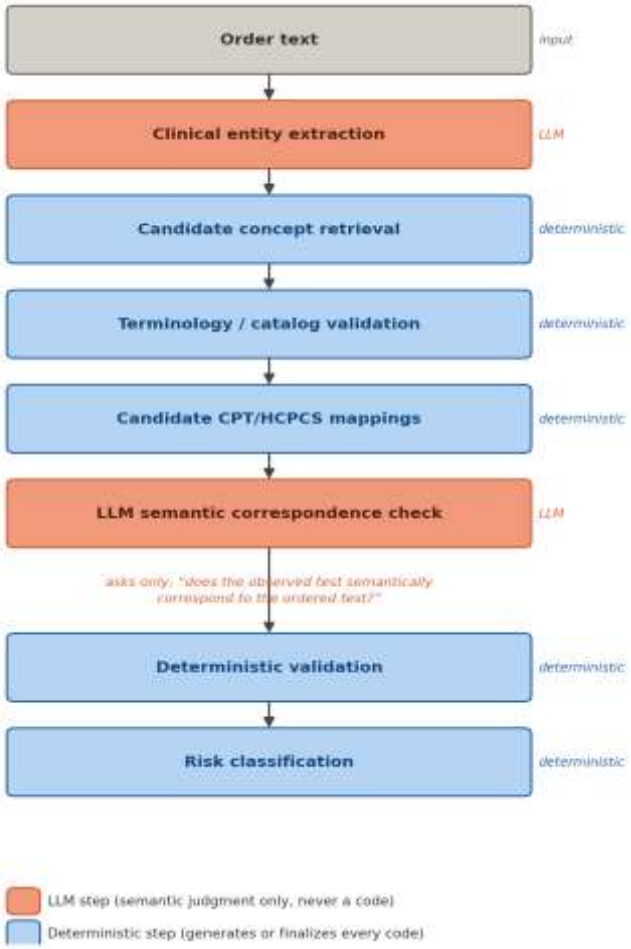


Figure 5. LLM reconciliation pipeline. The LLM is asked only whether the observed test corresponds to the ordered test; it never proposes a billing code.

The order text is first passed to a clinical entity extraction stage, where the LLM's only job is to pull structured clinical entities out of free text, the analyte, specimen, timing, and any reflex condition, without attaching a code of any kind. Candidate concept retrieval then runs as a purely deterministic embedding-similarity search (Karpukhin et al., 2020) over the institution's own test catalog, using those extracted entities as the query; the LLM has no input into which candidates come back. Terminology and catalog validation filters that candidate list against the metadata schema described in Section 4.3, discarding any concept that is expired, superseded, or outside the applicable jurisdiction before the LLM ever sees it. Candidate CPT/HCPCS mappings are then generated by applying the versioned, directional, one-to-many crosswalk introduced in Section 2.2, deterministic table lookups, not model inference, so every CPT or HCPCS code that could possibly be proposed already exists in the crosswalk before the LLM is invoked a second time.

Only at this point does the LLM run again, and it is given exactly one bounded question to answer: does the observed, performed test semantically correspond to the ordered test, not what CPT code should this be. The performed test already has a known identity in the microbiology LIS data; it is not ambiguous and the LLM is never asked to code it. What is uncertain is only whether that performed test matches the clinical intent behind the free-text order, and the model returns a fixed JSON object addressed to that question alone: a categorical correspondence judgment (corresponds, does not correspond, or uncertain), a confidence level, and a short citation-only rationale. It is instructed explicitly never to output a code, a candidate ranking, or a discrepancy classification, and to default to uncertain whenever the order text is ambiguous. A deterministic validation stage then combines three independent signals to reach the actual discrepancy classification: the LLM's correspondence judgment, whether the code that was actually billed appears among the candidate CPT/HCPCS mappings generated deterministically in the prior stage, and the

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

standard specimen-type, NCCI, and panel checks. A does-not-correspond or uncertain judgment is sufficient on its own to route the case to mandatory human review regardless of what the code-level checks show, since a semantic mismatch and a coding mismatch are different failure modes and neither should be allowed to mask the other. Risk classification, the final stage, is itself deterministic, combining the validated outcome with the correspondence confidence into a single ranked flag. Narrowing the LLM's task from code selection to a bounded semantic-correspondence judgment follows the same retrieval-augmented generation logic that grounding reduces hallucination relative to relying on parametric knowledge alone (Lewis et al., 2020), taken one step further: a model asked whether two already-identified things mean the same thing is answering a narrower and more verifiable question than a model asked to pick the correct one from a list, which matters given how easily generative models can still produce fluent but unfaithful output even when their choices are constrained (Ji et al., 2023).

Outputs from all three detection paths converge on a RAG explanation generator, which is restricted to explaining and citing retrieved rule or guideline text rather than introducing new claims of its own, so every flag a human reviewer sees comes with a specific, traceable citation rather than an opaque score.

4.5 Human review, recovery, and feedback

Flagged cases are queued for compliance review, ranked by the product of estimated dollar impact and confidence, and every reviewer decision, whether to uphold, overturn, or request more information, is captured and tagged with the detection layer that produced the flag. This tagging makes per-layer precision tracking possible, which is the main signal used to recalibrate anomaly-scoring thresholds and to refine the LLM matching component's few-shot examples or fine-tuning set over time, closing the loop shown on the right side of Figure 2.

V. CONCLUSION

We have presented a hybrid architecture for detecting billing discrepancies that arise from mismatches among laboratory orders, specimen collection, and microbiology testing. Its central contribution is the combination of an explicit order-specimen-test-claim linkage graph, which makes relational discrepancies queryable instead of implicit, with a retrieval-constrained large language model component that reconciles free-text clinical order language against a controlled catalog without introducing hallucinated billing codes. By requiring that every automated flag, whether it comes from deterministic rules, semantic matching, or statistical anomaly scoring, be accompanied by a citation-grounded explanation, the architecture is built to satisfy the auditability that regulated healthcare billing compliance work demands, and to stay advisory to human reviewers rather than acting on its own. The architecture is scoped deliberately to overpayment risk rather than to fraud: it flags where an order, specimen, test, or claim record disagrees with another, and leaves any determination of intent to the human reviewers and, where warranted, the compliance or legal staff who are equipped to make that judgment.

The main limitation of this work is that it has only been validated through a qualitative walkthrough of representative discrepancy patterns and has not yet been tested empirically against production claims data. Future work should prioritize a pilot deployment that measures precision, recall, and false-positive rate per detection layer against auditor-confirmed outcomes, an evaluation of how sensitive the LLM matching component is to test-catalog coverage and prompt design across institutions, and an assessment of how the feedback loop affects detection performance across successive recalibration cycles.

REFERENCES

1. Centers for Disease Control and Prevention, National Center for Health Statistics. (n.d.). ICD-10-CM. https://www.cdc.gov/nchs/icd/icd-10-cm/index.html
2. Centers for Medicare & Medicaid Services. (n.d.-a). Clinical Laboratory Fee Schedule (CLFS). https://www.cms.gov/medicare/payment/fee-schedules/clinical-laboratory-fee-schedule-clfs
3. Centers for Medicare & Medicaid Services. (n.d.-b). Medicare NCCI FAQ library. https://www.cms.gov/medicare/coding-billing/national-correct-coding-initiative-ncci-edits/medicare-ncci-faq-library
4. Centers for Medicare & Medicaid Services. (n.d.-c). Medicare provider enrollment: Ordering, certifying, and referring providers. https://www.cms.gov/Outreach-and-Education/Medicare-Learning-Network-MLN/MLNProducts/EnrollmentResources/provider-resources/provider-enrolment/Med-Prov-Enroll-MLN9658742.html

|ISSN: 2322-0163| www.ijeetr.com | A Bimonthly, Peer Reviewed, Scholarly Indexed Journal |

| Volume 7, Issue 1, January - February 2025 |

DOI:10.15662/IJEETR.2025.0701011

5. Centers for Medicare & Medicaid Services. (n.d.-d). National Correct Coding Initiative (NCCI) edits. U.S. Department of Health and Human Services. https://www.cms.gov/medicare/coding-billing/national-correct-coding-initiative-ncci-edits
6. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly detection: A survey. ACM Computing Surveys, 41(3), Article 15. https://doi.org/10.1145/1541880.1541882
7. Fellegi, I. P., & Sunter, A. B. (1969). A theory for record linkage. Journal of the American Statistical Association, 64(328), 1183–1210. https://doi.org/10.2307/2286061
8. HL7 International. (n.d.). FHIR (Fast Healthcare Interoperability Resources) specification. https://www.hl7.org/fhir/
9. Ji, Z., Lee, N., Frieske, R., Yu, T., Su, D., Xu, Y., Ishii, E., Bang, Y. J., Madotto, A., & Fung, P. (2023). Survey of hallucination in natural language generation. ACM Computing Surveys, 55(12), Article 248. https://doi.org/10.1145/3571730
10. Johnson, M. E., & Nagarur, N. (2016). Multi-stage methodology to detect health insurance claim fraud. Health Care Management Science, 19(3), 249–260. https://doi.org/10.1007/s10729-015-9317-3
11. Karpukhin, V., Oğuz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., & Yih, W. (2020). Dense passage retrieval for open-domain question answering. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (pp. 6769–6781). Association for Computational Linguistics.
12. Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive NLP tasks. Advances in Neural Information Processing Systems, 33, 9459–9474.
13. LOINC. (n.d.). LOINC and other standards. Regenstrief Institute. https://loinc.org/kb/faq/loinc-and-other-standards
14. Nabrawi, E., & Alanazi, A. (2023). Fraud detection in healthcare insurance claims using machine learning. Risks, 11(9), Article 160. https://doi.org/10.3390/risks11090160
15. Singhal, K., Azizi, S., Tu, T., Mahdavi, S. S., Wei, J., Chung, H. W., Scales, N., Tanwani, A., Cole-Lewis, H., Pfohl, S., Payne, P., Seneviratne, M., Gamble, P., Kelly, C., Babiker, A., Schärli, N., Chowdhery, A., Mansfield, P., Demner-Fushman, D., Agüera y Arcas, B., … Natarajan, V. (2023). Large language models encode clinical knowledge. Nature, 620(7972), 172–180. https://doi.org/10.1038/s41586-023-06291-2
16. Tjoa, E., & Guan, C. (2021). A survey on explainable artificial intelligence (XAI): Toward medical XAI. IEEE Transactions on Neural Networks and Learning Systems, 32(11), 4793–4813. https://doi.org/10.1109/TNNLS.2020.3027314
17. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30, 5998–6008.
18. Yoo, Y., Shin, J., & Kyeong, S. (2023). Medicare fraud detection using graph analysis: A comparative study of machine learning and graph neural networks. IEEE Access, 11, 88278–88294. https://doi.org/10.1109/ACCESS.2023.3305962