

DFBI 2026 aims to encourage the international exchange of innovative ideas between researchers from academia and industry. In addition to knowledge dissemination, the conference offers a valuable platform for professional networking, particularly benefiting university professors, graduate students, and postdoctoral researchers.

Research Article

ET-ACC: A Conceptual Framework for Explainable and Trustworthy Automated Compliance Checking in the AEC Industry

Youssef Senousy¹, Franco Cheung², Ogerta Elezaj³, Edlira Vakaj⁴

Birmingham City University^{1,2,3,4}

Correspondence: youssef.senousy@mail.bcu.ac.uk

Copyright: Copyright: © 2026 by the authors.

DFBI is an open-access proceedings distributed under the terms of the Creative Commons Attribution 4.0 International License (CC BY 4.0). View this license's legal deed at <https://creativecommons.org/licenses/by/4.0/>



Abstract

Automated Compliance Checking (ACC) is now a central activity in the Architecture, Engineering, and Construction (AEC). It supports design assurance, regulatory compliance, and digital delivery from early design through to review. Yet many current ACC systems remain opaque. Complex rule sets and AI-assisted reasoning often produce results that are difficult to inspect or explain, so transparency, interpretability, and user trust remain limited. This paper introduces ET-ACC, a multi-layer conceptual framework for Explainable and Trustworthy Automated Compliance Checking. The framework evaluates BIM and IFC models against pre-formalised, machine-readable rules. It is organised into five connected layers—Data, Rule, Reasoning, Provenance, and Explanation—supported by a cross-cutting Human-in-the-Loop and Trust Evaluation component. The Rule Layer manages computable rules and logic structures. The Reasoning Layer combines symbolic reasoning with AI-based support. The Provenance Layer records how data, rules, and reasoning steps relate, which enables complete traceability. The Explanation Layer then turns these traces into role-aware, human-readable outputs for designers, regulators, and other stakeholders. A Human-in-the-Loop and Trust Evaluation component strengthens accountability through expert review and structured transparency assessment. The framework is built on design-science methodology and literature across ACC, explainable AI, and compliance modelling. It offers a structured foundation for next-generation systems that place interpretability and auditability at their core. Full implementation is planned as future work. The framework sets a clear blueprint for a practical compliance-checking platform, referred to as ACE-X, that will operationalise the ET-ACC architecture and support more transparent, safe, and accountable digital building regulation processes and automated design assessment workflows across diverse project types and scales.

Keywords: Automated Compliance Checking, Explainable AI, IFC-based BIM, Symbolic-AI Reasoning, Provenance Modelling, Human-in-the-Loop.

Highlights

- A five-layer ET-ACC framework with a cross-cutting Human-in-the-Loop and Trust Evaluation component, operationalised through ACE-X.
- A practical bridge between symbolic checking and learning.
- A concrete engineering lesson on trustworthiness.

Cite this article:
Lastname, F. M., & Lastname, F. M.
(Year). Title of the article. Title of the
Journal, Volume (Issue), Page range.
<https://doi.org/10.66408/dfbi.2026.26>
366

1 Introduction

Building projects operate under dense regulatory constraints, and design teams invest substantial effort in interpreting requirements and preparing evidence for approval. Many jurisdictions now link BIM adoption to permitting workflows, and recent reviews describe BIM-based e-permit ecosystems as an active research and practice area with growing interest in automation and AI integration (Sun & Kim, 2026). This shift increases pressure on automated compliance checking (ACC) systems to deliver not only faster checks but also defensible outcomes that stakeholders can inspect and reuse across project stages.

ACC research has progressed from early rule-based checking to workflows that combine BIM data, rule languages, and knowledge representations. Recent studies show a clear move towards semantic pipelines that translate regulatory statements into machine-readable structures and then run checks on BIM-derived representations (Pauwels et al., 2024; Jia et al., 2025). At the same time, newer work integrates language models and deep learning with ontologies to reduce manual effort in rule interpretation and mapping (Chen et al., 2024). These directions reflect an important trend, yet a key barrier remains across the ACC literature. Many systems still struggle to provide explanations that stay faithful to rules, data, and intermediate reasoning steps. Recent exploratory work on reasoning-focused machine learning suggests that compact models with explicit intermediate reasoning states may support more inspectable structured-task inference (Jolicoeur-Martineau, 2025).

Opacity appears in two common ways. Rule translation from natural language into executable constraints often involves hidden choices. The chain from regulation text to rule encoding to model query is not always explicit. Studies continue to report difficulty in representing complex constraints, particularly geometric and context-dependent clauses, and this limits both automation coverage and interpretability (Pauwels et al., 2024). Even when systems run successfully, outputs often stop at pass or fail, with little traceability of the data and logic that produced the decision. Newer ACC frameworks call explicitly for transparency and trust features such as traceable decision paths and structured human checkpoints, which shows that explainability is a core requirement for reliable adoption rather than an optional add-on (Purushotham et al., 2026).

Traceability becomes harder when compliance pipelines rely on multiple representations and tools. IFC models are parsed, transformed into graphs or ontologies, checked through rule engines, and then summarised in reports. Each conversion can lose meaning unless the system records what was used and how it was processed. Provenance research in construction already frames trustworthy data exchange as a core need, and BIM provenance models show that recording activities and exchanges supports the reconstruction and validation of decisions across stakeholders (Celik et al., 2023). For ACC, this points to the need for structured links between inputs, such as IFC elements and properties, applied rules, and reasoning outputs.

This paper addresses this gap through ET-ACC, a layered framework for explainable and trustworthy ACC, and through ACE-X, a partial operationalisation of the framework up to the Reasoning Layer. The conceptual design places data extraction, rule execution, and reasoning into connected layers that can later support provenance capture and role-aware explanations.

In ACE-X, the system parses IFC models, constructs a normalised building graph, executes compliance rules, and generates reasoning outputs that explain failures and propose fixes. The current implementation also integrates a trainable component, referred to as a Tiny Recursive Model (TRM), to

support learning from compliance patterns and to expose reasoning-oriented outputs, with explicit handling of training data completeness to avoid misleading performance artefacts.

2 Conceptual Foundations and Prior Work in Automated Compliance Checking

Recent digital building permitting research conceptualises automated compliance checking as a socio-technical capability embedded within broader regulatory workflows. Rather than operating as a standalone verification tool, ACC is increasingly positioned between regulatory interpretation, information availability in BIM submissions, and the verification procedures applied by permitting authorities. Noardo et al. (2022) provide a comprehensive synthesis of Digital Building Permit initiatives and illustrate how ACC functions as one component within interconnected permitting ecosystems.

Within this context, explainability is discussed in the literature as the ability to justify compliance outcomes through inspectable links between model evidence and computational checks. Regulator-facing studies indicate that adoption of ACC systems depends not only on technical correctness, but also on clarity of terminology, consistency of interpretation, and a balanced distribution of automation and human oversight. These concerns are reported directly through interviews with regulators and ACC specialists (Fuchs et al., 2025).

Provenance and traceability are similarly framed as decision reconstruction capabilities. They support questions such as which model facts were used, which rules were applied, and how intermediate evaluations contributed to a compliance outcome. Digital permitting research emphasises that machine-readable submissions and structured information requirements are prerequisites for repeatable and auditable checks across stakeholders. Heiß et al. (2025) discuss this requirement through extensions to information delivery structures that support permitting processes.

Taken together, these studies show that while ACC is increasingly embedded in digital permitting workflows, existing approaches still treat explainability, provenance, and traceability as secondary or external concerns. They are discussed as adoption conditions or reporting requirements, but they are rarely modelled as integral components of the compliance checking process itself. As a result, current systems struggle to provide compliance outcomes that can be reconstructed, audited, and justified across organisational and regulatory boundaries.

2.1 Existing ACC Frameworks and Technical Paradigms

Recent work on automated compliance checking can be grouped into three main technical strands. The first strand uses open standards to specify information requirements and to check submission quality. IDS-based workflows formalise the expected content of IFC exchanges and allow automatic verification of whether required entities and properties are present. De Mendonça and Ferreira (2024) apply IDS to accessibility rules and discuss the benefits and limits of this standards-led route. Fischer et al. (2024) extend IDS towards compliance tasks such as escape route analysis and fire resistance checks, where relationships between elements and filtering logic become essential. These studies show that IDS can support parts of compliance checking, yet they still offer limited support for reasoning traces that connect results back to regulatory intent or intermediate evaluations.

The second strand centres on semantic consistency and constraint formalisation around IFC representations. Research on ifcOWL examines how IFC schema constraints and validation logic can be translated into forms that support automated checking, which strengthens reproducibility and improves confidence in model-derived facts. Wang et al. (2024) study the conversion and validation of IFC “WHERE” rules, which links directly to the reliability of the facts used in downstream checking pipelines. This line of work strengthens semantic integrity, but it gives less attention to how reasoning steps are made visible or reconstructed once validation has completed.

The third strand proposes system-level architectures that combine compliance logic with knowledge graphs and governance mechanisms. Tao et al. (2024) present a blockchain-enabled common data environment that uses knowledge-graph structures to check BIM metadata compliance. Their work illustrates how compliance obligations extend beyond geometry into collaboration and information management, yet explanation and lineage remain largely implicit within the system’s operational logic.

Alongside these strands, recent reviews discuss the role of machine learning in ACC and the trade-offs between rule-grounded precision and learning-based coverage. Alnuzha and Bloch (2025) survey machine learning use in automated compliance-related workflows and highlight both where learning supports scalability and where it increases validation and accountability demands.

2.2 Open Challenges and Research Opportunities in ACC

Recent studies point to three gaps that persist, even where standards and semantic checking pipelines have become more mature.

First, many workflows improve rule execution or increase information completeness, but they still fail to support decision reconstruction that links regulatory intent, formal rule structure, and the specific model evidence used during execution. Regulator-focused work identifies this as a practical barrier to adoption, since authorities need to validate how a decision was reached, not just the outcome (Fuchs et al., 2025).

Second, standards such as IDS raise the quality and consistency of what gets submitted, yet compliance results still depend on how checking logic is assembled, how exceptions are managed, and how relationships between elements are evaluated. Recent IDS extensions show progress, but they also expose the difficulty of capturing contextual constraints in a controlled and reusable form (Fischer et al., 2024).

Third, LLM-assisted ACC pipelines introduce a distinct traceability problem. They can help with mapping and alignment between regulatory language and BIM data, but they add processing steps that must be checked and tied back to authoritative sources. Recent work on LLM-based model alignment highlights this tension and argues for explicit validation layers around language-based components (Saluz et al., 2025).

Taken together, these gaps are unlikely to be addressed by simply extending rule execution engines or attaching reporting modules after the fact. They reflect a structural limitation in many current ACC architectures, where reasoning, lineage, and explanation are not treated as core outputs of the compliance process.

2.3 ET-ACC: Conceptual Framework Overview

In response to these structural limitations, ET-ACC is proposed as a layered conceptual framework that embeds reasoning, provenance, and explanation directly within the compliance checking architecture rather than treating them as auxiliary functions. ET-ACC addresses these gaps by structuring ACC as connected layers that separate concerns while keeping links explicit between them. The Data and Rule layers support-controlled inputs and computable checks. The Reasoning layer supports structured justification of outcomes. Across these five layers, a Human-in-the-Loop and Trust Evaluation component provides oversight, expert review, and structured trust assessment.

This paper operationalises ET-ACC up to the Reasoning Layer through ACE-X, which provides a concrete base for later provenance capture and role-aware explanation generation.

3 Methodology

This study follows a design science research methodology. The work addresses a practical weakness in automated compliance checking by designing and instantiating an artefact that makes compliance outcomes inspectable. The research yields two connected outputs. ET-ACC is the conceptual framework that structures explainable and trustworthy automated compliance checking as a layered process. ACE-X is a partial operational instantiation that implements core layers and prepares interfaces for provenance capture and explanation delivery.

The methodology ties conceptual choices to executable behaviour. This alignment allows the study to examine how traceability and explanation arise during compliance execution rather than being attached as post-processing. The current scope focuses on architectural coherence and reasoning transparency, supported by a working prototype and structured runtime artefacts.

3.1 ET-ACC as the Methodological Framework

ET-ACC acts as the organising structure of the methodology and guides both system design and analysis. The framework models compliance checking as connected layers that separate responsibilities while keeping explicit links between inputs, rules, intermediate evaluations, and outputs. It comprises five layers: Data, Rule, Reasoning, Provenance, and Explanation, together with a cross-cutting Human-in-the-Loop and Trust Evaluation component. The layered structure and information flows are shown in Figure 1.

The Data Layer prepares controlled BIM-derived inputs. The Rule Layer holds machine-readable rules that are linked to their originating regulatory statements. The Reasoning Layer produces structured justifications that connect outcomes to evaluated conditions and evidence. The Provenance Layer records how model facts, rules, and reasoning steps relate. The Explanation Layer turns these records into stakeholder-facing accounts. Human oversight and trust evaluation provide checkpoints for inspection, correction, and refinement. In ACE-X, the Data, Rule, and Reasoning layers are implemented directly, while provenance and explanation are prepared through logs, stable identifiers, and interfaces that support later extension without reworking the pipeline.

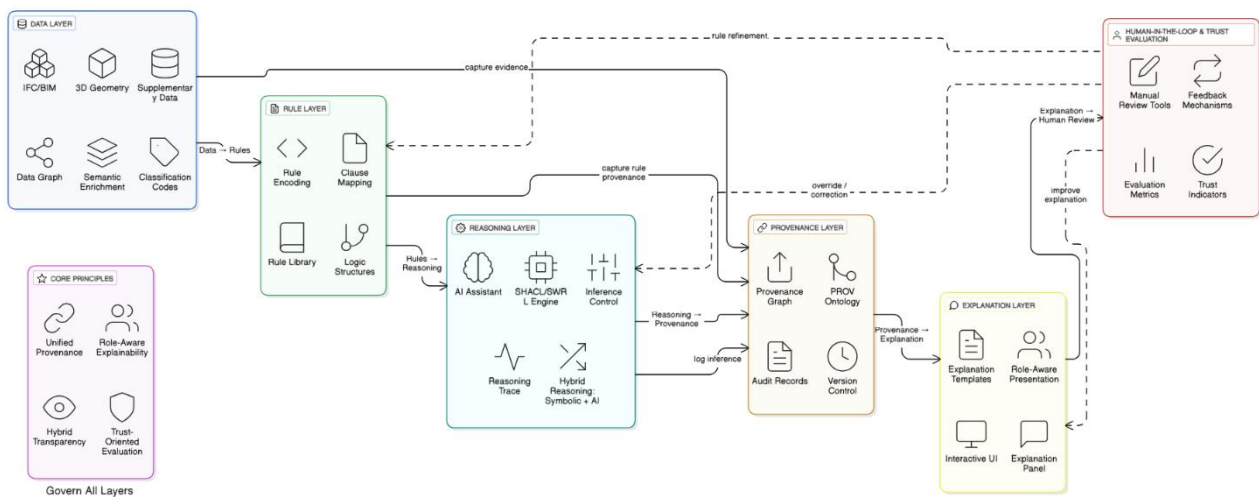


Figure 1. ET-ACC layered framework for explainable and trustworthy automated compliance checking

3.2 ACE-X Compliance and Reasoning Pipeline

ACE-X operationalises the ET-ACC logic as an end-to-end pipeline from IFC ingestion to reasoning outputs. The process starts with ingestion of IFC-based BIM models. The system parses IFC entities and extracts relevant geometric and semantic properties, then converts them into a normalised building data graph that represents elements, their properties, and relationships. This unified graph becomes the authoritative input for downstream processing and preserves references to original IFC entities to support evidence traceability.

Compliance checking then evaluates the graph against a set of pre-formalised, machine-readable rules. In this paper, pre-formalised means that relevant regulatory clauses were encoded into structured rule definitions before system execution. Rules specify applicability, conditions, thresholds, and required relations, and each rule is linked to its originating regulatory statement through clause mapping so that regulatory intent remains visible during execution. Rule execution produces deterministic outcomes and records the data elements accessed and the conditions evaluated.

The Reasoning stage builds on these outputs by generating reasoning traces that capture intermediate evaluation steps, dependencies, and affected elements. Symbolic reasoning provides explicit evaluation paths, and controlled AI-based support is used to assist interpretation and explanation without overriding rule outcomes. Runtime artefacts, including rule evaluation logs, reasoning traces, and version identifiers, are captured to prepare provenance-ready records that can support later decision reconstruction and explanation generation.

3.3 Illustrative Worked Example of End-to-End ACE-X Execution

To make the end-to-end behaviour of ACE-X more concrete, consider an IFC model containing an accessible door element, D-017, with a clear width of 780 mm. During ingestion, the Data Layer parses the IFC model, extracts the relevant door properties, normalises the width value to millimetres, resolves its spatial context, and records the element in the canonical building graph using its IFC GlobalId as the primary reference. The Rule Layer then applies the ADA_DOOR_MIN_CLEAR_WIDTH rule to accessible doors only. In this case, the rule compares the extracted width against the configured minimum threshold of 815 mm, so the condition $780 \geq 815$ evaluates to false. The system therefore records a fail result together with the rule identifier, regulatory reference, evaluated value, threshold, and affected

element identifier. The Reasoning Layer then classifies the case as a dimensional violation, generates both short and detailed explanations, and produces tiered remediation guidance ranging from a simple source-model adjustment to a broader redesign response. This example illustrates how ACE-X preserves a traceable chain from IFC-derived evidence, through rule evaluation, to structured reasoning and stakeholder-facing remediation output.

3.4 Reasoning-Oriented Learning with the Tiny Recursive Model

ACE-X integrates a compact learning component, referred to as the Tiny Recursive Model, to support reasoning by learning patterns from prior compliance checks and exposing inspectable reasoning-oriented outputs. Its operational flow is shown in Figure 2. Training data links three sources: compliance results, IFC-derived element features, and rule specifications. These inputs are combined into feature vectors that retain explicit semantic meaning, including context, element, and rule feature sets.

The model uses compact layers with attention and controlled expansion to keep inference behaviour inspectable. During inference, it produces an advisory label and confidence score at each step, and an iterative refinement loop controls progression through a step counter and explicit convergence checks, as depicted in Figure 2. Outputs include the advisory label, confidence value, number of reasoning steps, and convergence status. Rule execution remains authoritative, and the model outputs support prioritisation, pattern surfacing, and explanation support rather than determining compliance decisions.

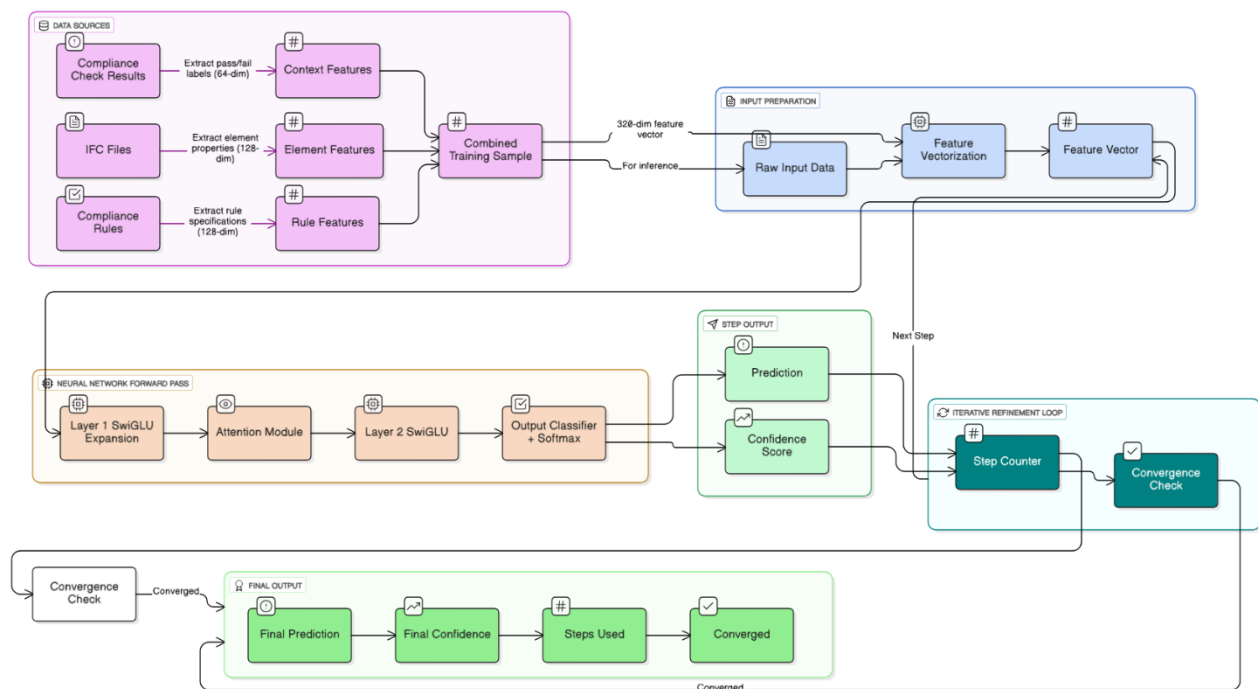


Figure 2. Tiny Recursive Model operational flow within ACE-X.

3.5 System Architecture and Human Oversight

ACE-X is implemented as a web-based application that separates processing services from user-facing inspection. The backend provides services for IFC parsing, graph construction, rule execution, reasoning trace generation, and runtime logging. The frontend supports interaction with models and results and provides views for inspecting failing elements, rule references, and reasoning artefacts. This

separation allows the pipeline to remain stable while supporting stakeholder-facing inspection and review, as illustrated in Figure 3.

The system maps ET-ACC components to implementable responsibilities. Data ingestion and graph construction support the Data Layer. Rule execution services support the Rule Layer. Reasoning services support trace generation and structured justification. Logging and version identifiers support provenance preparation, and interfaces expose artefacts needed for explanation delivery. Human-in-the-loop oversight is treated as a structured checkpoint. Experts can inspect explanations, refine rules, and review reasoning artefacts, and these actions can be captured to support trace completeness and trust evaluation.

Methodological limitations follow from the conference scope. The implementation prioritises architectural coherence, reasoning transparency, and provenance readiness over large-scale statistical validation. Full automation of provenance reasoning and extended empirical evaluation across broader building types and regulatory regimes is reserved for subsequent work.

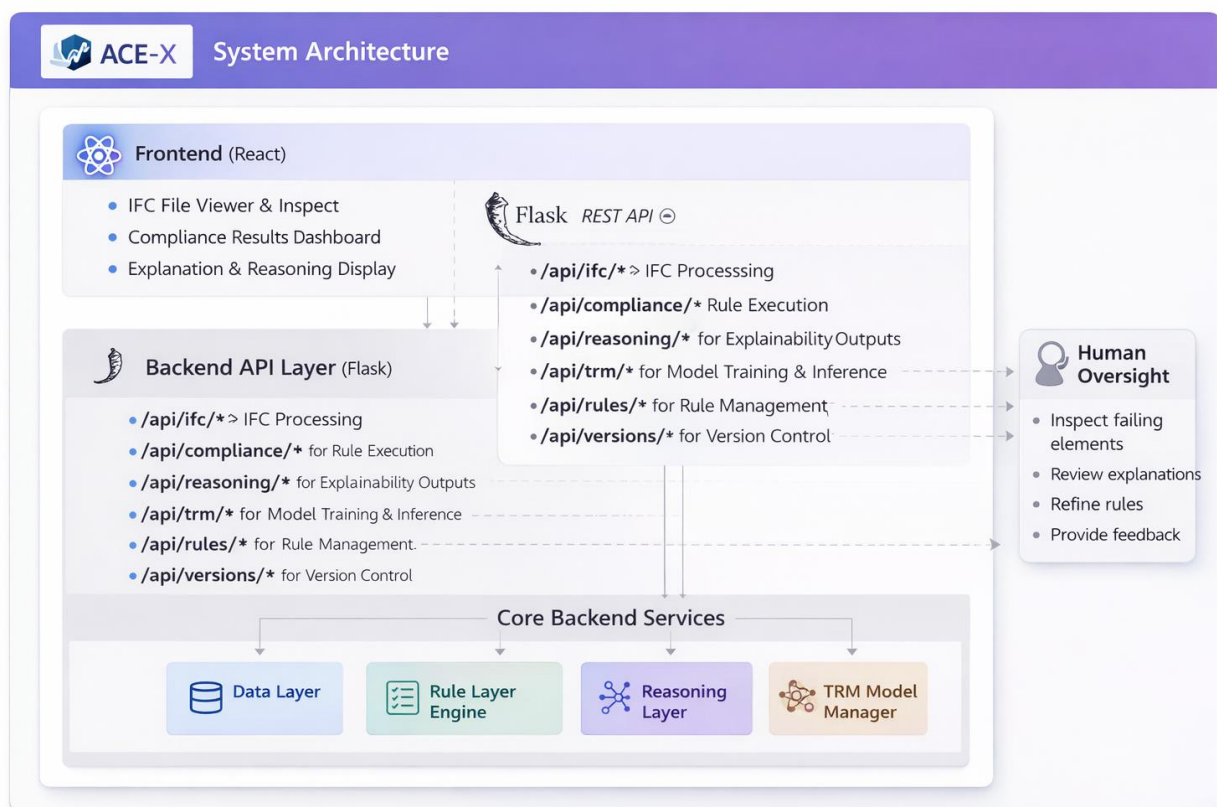


Figure 3. High-level Web Application System Architecture of ACE-X

4 Implementation

4.1 Data Layer

The Data Layer executes a structured set of validation and quality assurance procedures during the processing of IFC building models. It evaluates data integrity, extractability, and physical plausibility, whereas regulatory compliance is handled in subsequent stages of the pipeline.

During ingestion, each IFC file is loaded via the *ifcopenshell.open()* interface. Files that cannot be parsed raise an explicit loading exception and are excluded from further processing. For successfully loaded models, the system records the detected IFC schema version, such as IFC4 or IFC2x3, to support traceability and downstream compatibility assessment.

Element extraction targets core building entities, with checks tailored to each element type. For spatial entities, the process verifies the presence of a valid *GlobalId*, and spaces lacking this identifier are removed from downstream processing. Storey assignment is obtained through traversal of the spatial containment hierarchy. Floor area values are extracted from quantity property sets, prioritising *NetFloorArea* and *GrossFloorArea* where available. When explicit storey information is absent, the system attempts to infer it through alternative containment relationships.

Door entities are processed under comparable constraints. A valid *GlobalId* is required for inclusion. Door dimensions are normalised to millimetres using a heuristic conversion rule, where values less than or equal to ten are interpreted as metres and scaled accordingly. Spatial connectivity is resolved through *IfcRelSpaceBoundary* relationships, linking doors to adjacent spaces. Boundary type and boundary side attributes are validated to confirm semantic consistency. The end-to-end sequence of extraction, validation, and enrichment activities within the Data Layer is shown in Figure 4.

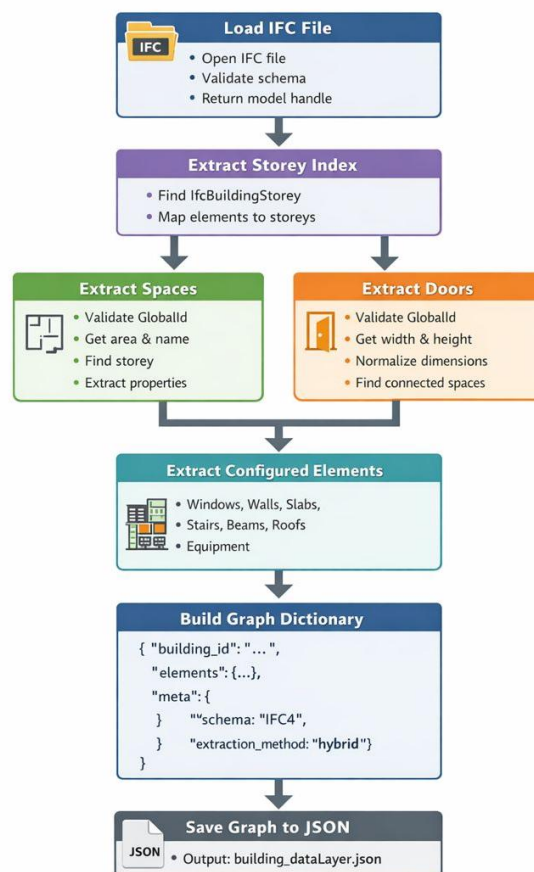


Figure 4. Sequence of Data Layer

For other building elements, the system extracts property sets using guarded access patterns that preserve execution continuity. Extraction failures are captured and logged without interrupting processing. Complex IFC attribute structures, including wrapped values, nominal values, and nested representations, are serialised into JSON-compatible formats to support downstream consumption.

The extraction procedure applies defensive serialisation logic to transform IFC-specific data types, such as enumerations and wrapped values, into standard Python and JSON representations. When optional utility modules are unavailable, the system returns empty property sets rather than raising runtime errors. Numeric properties are subject to controlled type coercion, and values that cannot be converted to floating-point form are explicitly set to null.

A dedicated validation stage performs basic plausibility checks to detect anomalous or corrupt data values. These checks target physical reasonableness rather than compliance. For doors, width values are expected to fall between 100 and 5000 millimetres, and height values between 500 and 5000 millimetres, with the presence of a name attribute also verified. For spaces, extracted areas are validated against a range of 0.1 to 100,000 square metres, and type consistency is enforced across properties.

Validation outcomes are reported at the property level, indicating pass or fail status, identifying missing mandatory attributes, detecting invalid data types, and flagging values outside physically plausible ranges. The Data Layer also generates summary statistics to support transparency and debugging, including counts of extracted spaces and doors, tracking of elements with successfully populated dimensional attributes, and records of partial extraction cases. These metrics support diagnosis of incomplete or degraded model inputs. Representative system outputs and user-facing views of the Data Layer implementation are shown in Figure 5.

The implementation follows a set of guiding principles. Element-level failures do not halt overall processing, allowing partially usable models to progress through the pipeline. Optional data absence is handled through controlled fallback mechanisms. Data validation remains separate from regulatory compliance logic, which is confined to the Rule Layer. All extraction and validation issues are recorded at debug logging level to support reproducibility and detailed post-hoc inspection.

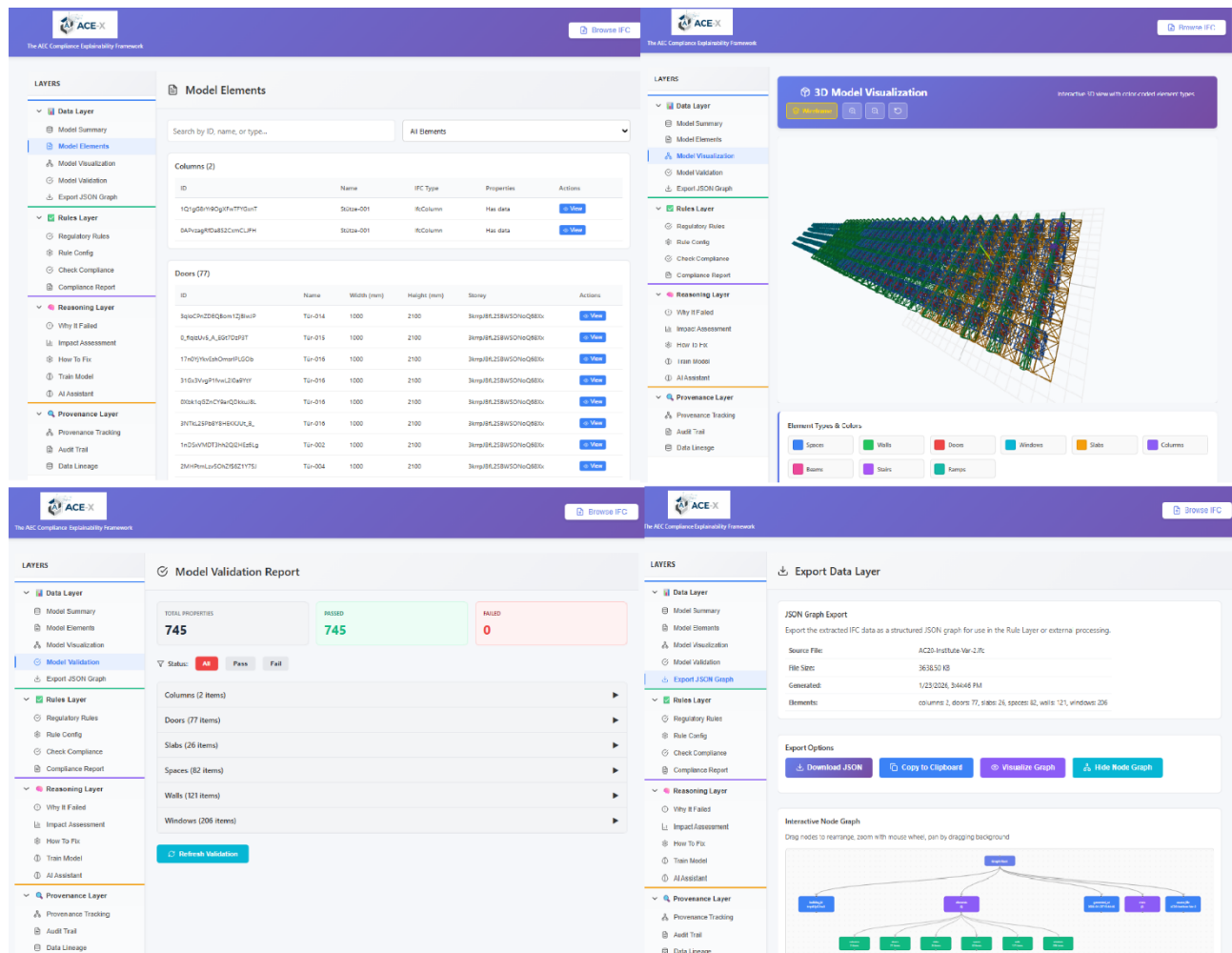


Figure 5. Screenshots of the Prototype Implemented in Data Layer

4.2 Rule Layer

The Rule Layer evaluates regulatory compliance by applying formalised rules to the building graph produced by the Data Layer. It checks element-level and space-level constraints in a way that supports auditability and later explanation, and it retains the contextual information needed for reasoning components downstream.

Rules are specified using a consistent, configuration-driven JSON schema. Each rule captures the target IFC class and applicability filters, a parameterised condition, provenance metadata that links the rule to its source clause, and template-based explanation messages for pass and fail outcomes. Listing 1 provides an excerpt of this rule specification structure. During execution, each evaluation produces a structured rule result object that records both the decision and its regulatory context. A rule result includes a unique rule identifier and name, the target element type and identifier, the outcome status, and a severity level that expresses criticality. It also stores a concise message, the regulatory reference,

and an extensible details field that preserves the evaluated values and related metadata required by the Reasoning Layer.

Listing 1. Abbreviated JSON rule specification schema used in the Rule Layer.

```
{
  "rules": [
    {
      "id": "ADA_DOOR_MIN_CLEAR_WIDTH",
      "target": { "ifc_class": "IfcDoor", "selector": { "filters": [ ... ] } },
      "condition": {
        "lhs": { "source": "qto", "qto_name": "Qto_DoorBaseQuantities", "quantity": "ClearWidth", "unit": "mm" },
        "op": ">=",
        "rhs": { "source": "parameter", "param": "min_clear_width_mm" }
      },
      "parameters": { "min_clear_width_mm": 815 },
      "provenance": { "regulation": "2010 ADA Standards", "section": "404.2.3", "source_link": "..."},
      "explanation": { "short": "...", "on_fail": "...", "on_pass": "..."}
    }
  ]
}
```

Rules are executed by a central rule engine that iterates over the configured rule set and evaluates each rule against the building graph. The execution flow is summarised in Figure 6. Before evaluation, the engine performs basic structural checks to confirm that required element categories exist in the graph. It then evaluates rules sequentially to preserve deterministic behaviour and traceability. Runtime exceptions are captured and logged, and the engine can continue under a lenient execution mode when needed. This behaviour supports strict validation workflows, while also allowing exploratory runs over imperfect models without losing the full execution trace.

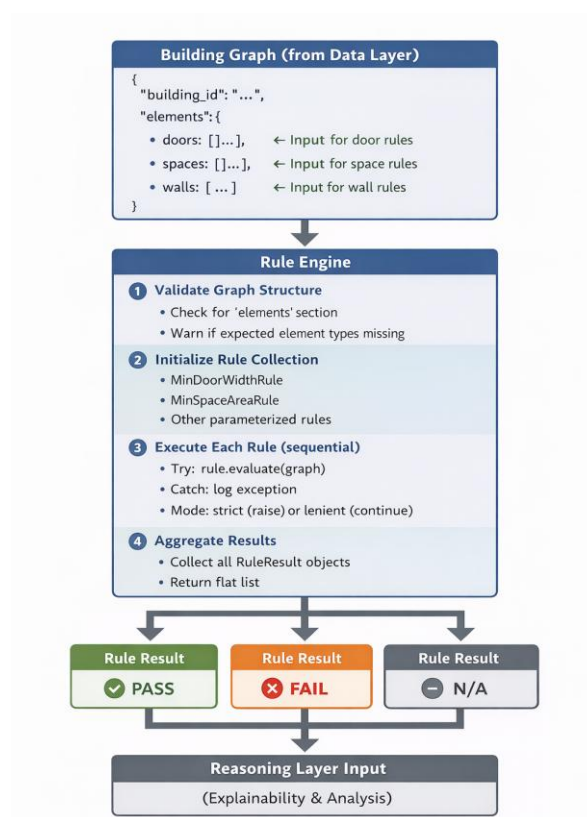


Figure 6. Sequence of Rule Layer

The rule base is organised by target element category, with separate modules for doors, spaces, and other building elements. Each rule encapsulates its evaluation logic and operates on the relevant subset of the graph. This modular organisation supports incremental extension of the rule set and reduces coupling between unrelated checks. The implementation also supports multi-source value extraction. Regulatory quantities may be derived from property sets, quantity take-off definitions, or direct element attributes. When preferred sources are missing, the system applies controlled fallback strategies to alternative sources, improving robustness against incomplete or inconsistent IFC inputs. Extracted values are then assessed through explicit comparison logic that preserves the evaluated left-hand value, the threshold parameter, and the comparison operator.

Rules are loaded dynamically from structured definitions and validated before execution to confirm internal consistency and required parameters. This design allows the rule set to evolve through configuration changes without altering the core engine. Evaluation outputs are aggregated into a flat collection that preserves element-level outcomes while supporting higher-level analysis across rules and element categories. Representative user-facing views and example outputs from the Rule Layer prototype are shown in Figure 7.

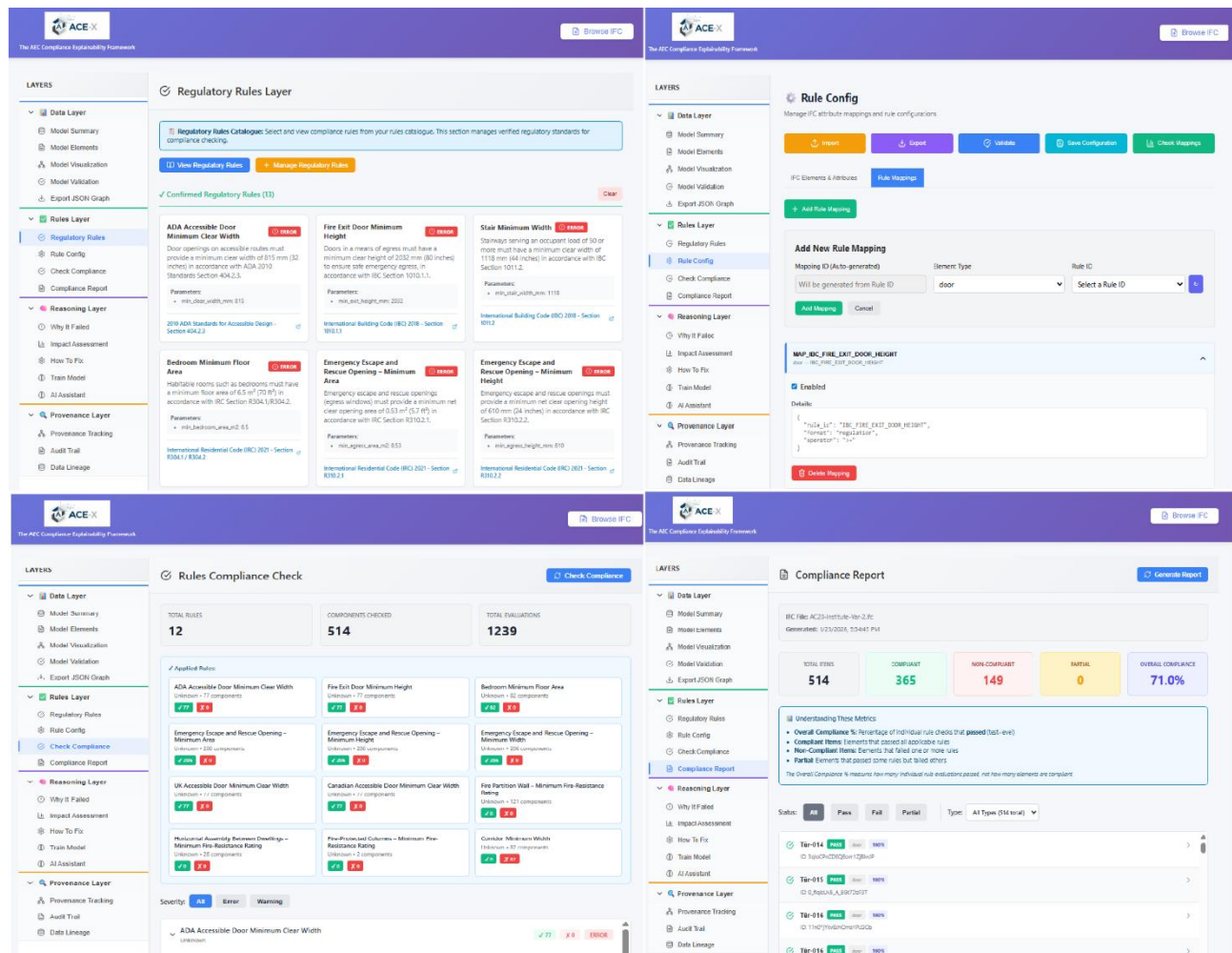


Figure 7. Screenshots of the Prototype Implemented in Rule Layer

The Rule Layer follows a small set of operational principles. Missing or incomplete data results in an explicit non-applicable outcome rather than an execution failure. Rules remain pluggable through a shared abstraction to promote consistent behaviour across heterogeneous checks. Each result retains contextual metadata to support explanation and audit review. Severity levels distinguish critical violations from advisory and informational findings, and regulatory references and source tracking support traceable outcomes suitable for governance and downstream inspection.

4.3 Reasoning Layer

The Reasoning Layer provides the system's main analysis capability. It coordinates rule execution outputs, interprets non-compliance, and produces structured results that support inspection and downstream explanation. The layer is implemented as a set of modular components, each responsible for a defined reasoning function within the compliance evaluation pipeline.

The Reasoning Engine serves as the coordination unit for all reasoning activities. It manages interactions among the layer components, loads regulatory and project-specific rule sets at runtime, and integrates with the Rules Version Manager to preserve consistency and traceability across executions. It also exposes a unified API that supports systematic analysis and aggregation of compliance failures across rules and element categories.

The Failure Explainer interprets and classifies detected non-compliance instances. It groups failures into categories such as missing property, dimensional violation, range violation, and value violation. For each case, it generates a concise one-line explanation and a more detailed multi-line description that captures context needed for review. It associates failures with explicit regulatory references, for example IBC 2018 §1010.1.1, and extracts relevant contextual data, including affected elements, actual versus required values, and measurement units, to support transparent inspection.

The Impact Analyzer quantifies the practical implications of non-compliance. It measures scope by reporting the total number of affected elements, breakdowns by element type, and the estimated proportion of the model impacted. It also produces severity distributions across error, warning, and informational outcomes, and identifies the most frequently affected elements and the most violated rules. These outputs support prioritisation during remediation and review.

The Recommendation Engine translates analytical findings into remediation guidance. It applies a tiered strategy that ranges from low-effort fixes to higher-effort interventions that address systemic issues. Recommendations are configuration-driven and instantiated from predefined templates stored in external configuration files. Template variables, such as element identifiers and measured values, are populated dynamically using project-specific data. Each recommendation includes indicative estimates of effort, resource requirements, and associated cost to support remediation planning. The interaction and sequencing of these reasoning components are summarised in Figure 8.

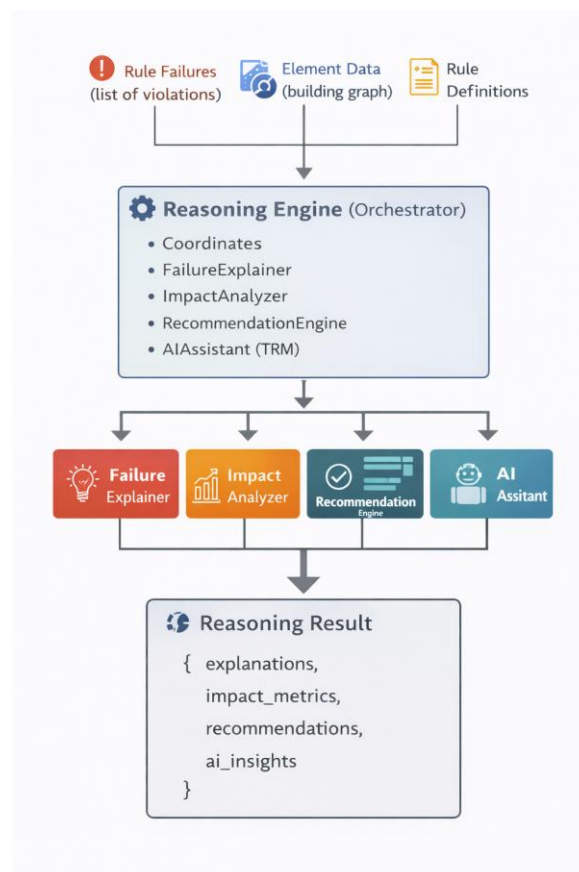


Figure 8. Workflow of Reasoning Layer

The Reasoning Layer also integrates an AI-based assistant based on a Tiny Recursive Model (TRM) to complement symbolic reasoning with data-driven inference. The assistant constructs feature vectors that combine element attributes, rule characteristics, and failure metadata, then processes these vectors using a lightweight neural model to generate pass or fail predictions accompanied by confidence scores. To preserve inspectability, it generates a structured multi-step reasoning trace that exposes intermediate inference states and supports examination of the decision path.

Across the layer, structured data models standardise inputs and outputs to maintain interoperability between components. These models represent detailed failure explanations, quantified impact metrics, individual recommendations and grouped recommendation sets, and aggregated reasoning results that encapsulate the end-to-end analysis outcome. Representative system outputs and user-facing views of the Reasoning Layer prototype are shown in Figure 9.

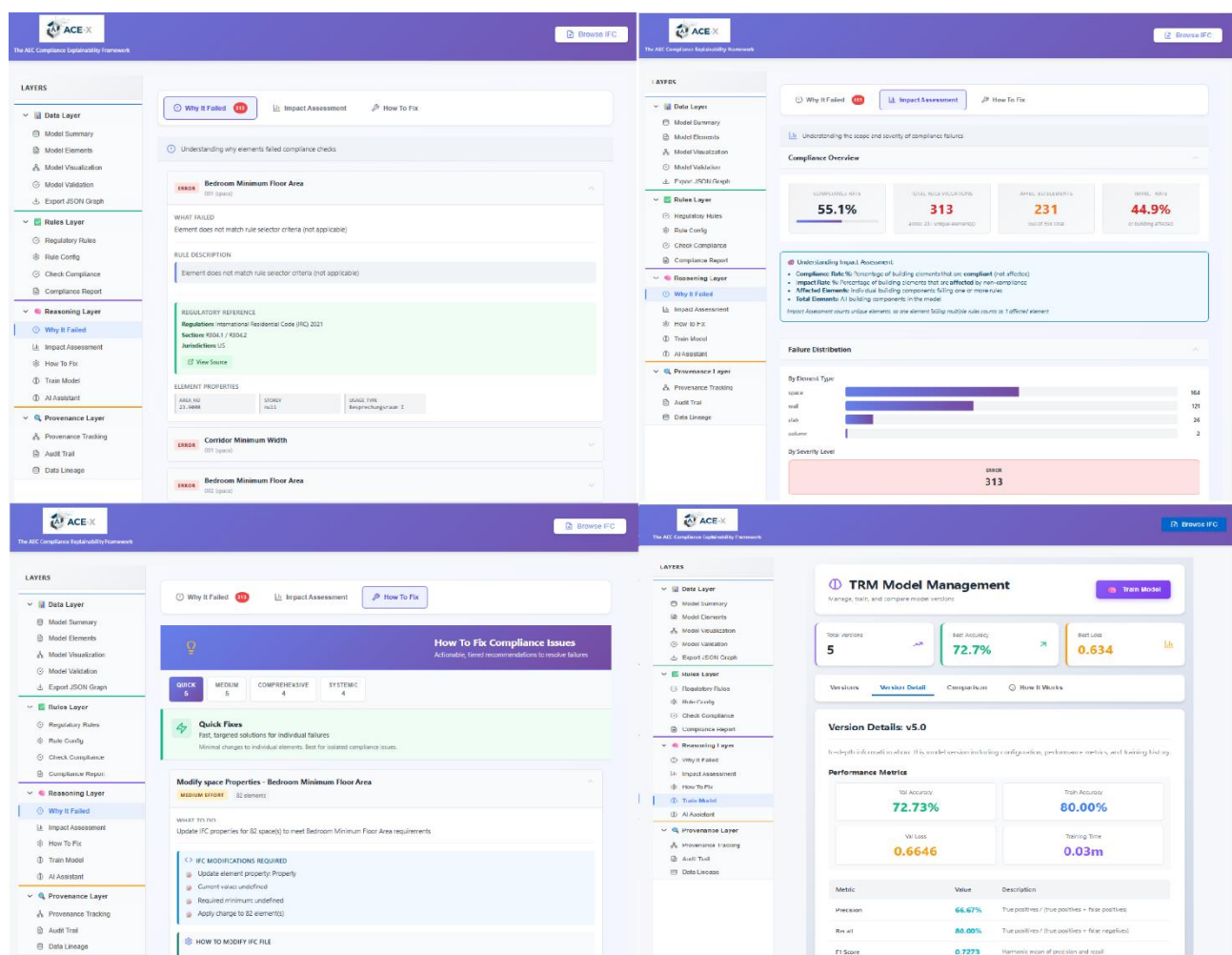


Figure 9. Screenshots of the Prototype Implemented in Reasoning Layer

4.4 Future Directions for Provenance, Explanation, and Human Oversight

Future extensions of the Provenance Layer can target finer-grained traceability of compliance outcomes. One direction is to model decision graphs that make dependencies explicit across extracted data, applied rules, and intermediate reasoning steps. Stronger lineage support can record where each value came from, how it changed as it moved across layers, and any associated confidence or quality indicators. Rule evolution can be treated as a first-class provenance concern through versioned tracking

of regulatory logic, which supports retrospective comparison of outcomes as codes change over time. Provenance can also cover the explanations themselves by recording why a given explanation was selected, which reasoning path produced it, and which model version or configuration influenced the final narrative.

The Explanation Layer can develop into a set of modes that match different stakeholder questions. These modes can include short summaries for rapid screening, side-by-side comparisons across design options, explanations grounded in regulatory intent, and counterfactual “what-if” analyses that test the sensitivity of outcomes to design changes. Accessibility support can be strengthened through multilingual output, text-to-speech, and formats designed for users with accessibility requirements. Interactive presentation, such as drill-down views, three-dimensional inspection, and heat-map overlays, can support comprehension by linking text explanations to spatial evidence. Personalisation can tailor the level of detail and terminology to user role and expertise. Quality-focused work can add measures of clarity and completeness, alongside confidence indicators and alternative explanations that expose uncertainty rather than hiding it.

Future work can also extend human-in-the-loop functions through structured feedback that targets specific components and captures contextual annotations. This feedback can support active learning by improving explanation templates, rebalancing reasoning priorities, and assessing whether recommended fixes reduce recurring failures. Trust measurement can be formalised through explainability trust scores, rule- and element-level confidence signals, and longitudinal tracking of how trust changes with repeated use. Collaborative review can support multi-user validation, disagreement management, and approval workflows. Operational monitoring can add usage analytics, controlled A/B testing, and quality dashboards to guide iteration. Expert escalation can route ambiguous cases to specialists while storing their decisions as reusable knowledge. A complete audit and accountability layer can then provide end-to-end traceability for governance reporting, systematic error analysis, and embedded user training and support.

4.5 Implementation Availability

The ACE-X prototype developed to support this study, including source code, configuration files, and example rule definitions, is available at: <https://github.com/ysenousy/ACC-Explainability-AEC>. As shown in Table 1, ACE-X currently operationalises the Data, Rule, and Reasoning layers, while the Provenance and Explanation layers, together with formal trust evaluation, remain the main areas for the next implementation phase.

ET-ACC layer / component	Role in the framework	Current status in ACE-X	Evidence in current prototype	Current limitation	Planned extension
Data Layer	Prepares controlled BIM-derived inputs through IFC parsing,	Implemented	IFC ingestion, schema detection, element extraction, dimensional	Performance depends on IFC completeness and consistency; unit handling	Strengthen unit-aware processing, improve schema-robust extraction, and

	extraction, validation, and enrichment.		normalisation, plausibility checks, summary statistics, and debug logging are operational.	remains heuristic and fragile when metadata are incomplete.	expand support for additional element classes and richer semantic enrichment.
Rule Layer	Stores and executes machine-readable compliance rules linked to regulatory clauses.	Implemented	Configuration-driven JSON rules, dynamic loading, modular rule execution, explicit NOT_APPLICABLE handling, severity levels, and regulatory references are operational.	Coverage remains bounded by the current rule base; contextual and more complex constraints remain harder to encode and validate consistently.	Expand the rule library, improve handling of contextual and relational constraints, and strengthen clause-to-rule traceability and rule versioning.
Reasoning Layer	Produces structured justifications, impact analysis, remediation guidance, and reasoning-oriented outputs.	Implemented	Failure classification, explanatory outputs, impact analysis, recommendation generation, and TRM-supported advisory reasoning are operational.	Explanation quality is uneven across rule types; confidence scores are not yet empirically calibrated; training data remain limited in scale and diversity.	Improve explanation coverage, calibrate confidence estimates, broaden training data, and extend reasoning support across more building types and jurisdictions.
Provenance Layer	Records links among model facts, rules, reasoning steps, and outputs to support reconstruction and auditability.	Partially prepared, not fully implemented	Logs, stable identifiers, runtime artefacts, and provenance-ready interfaces are already captured in the current architecture.	End-to-end provenance capture is not yet implemented as a first-class operational layer.	Add decision graphs, lineage tracking, rule-evolution history, and explicit provenance records covering data sources, transformation

					s, and reasoning paths.
Explanation Layer	Converts technical traces into stakeholder-facing, role-aware, human-readable accounts.	Partially prepared, not fully implemented	The current system provides structured explanatory outputs within the Reasoning Layer, but not yet a full dedicated Explanation Layer.	Explanations are still limited in personalisation, modality, accessibility, and interaction depth.	Add role-aware explanation modes, counterfactual and comparison views, multilingual and accessibility features, and richer visual drill-down linked to spatial evidence.
Human-in-the-Loop and Trust Evaluation component	Provides expert oversight, inspection, correction, and structured trust assessment across the framework.	Partially operational as a cross-cutting checkpoint	The architecture supports expert review of explanations, rule refinement, and inspection of reasoning artefacts.	Trust scoring and formal evaluation procedures are not yet fully defined or validated.	Add structured feedback capture, trust metrics, disagreement handling, escalation workflows, collaborative review, and longitudinal monitoring of trust and use.

Table 1. Consolidated summary of ET-ACC layers, implementation status in ACE-X, current limitations, and planned extensions.

5 Key Findings and Challenges

5.1 Key findings

- **The layered design is stable and testable.** Phase 1–2 demonstrates that the Data, Rule, and Reasoning layers operate as independent modules yet integrate through clear inputs and outputs. The Data layer extracts multiple element classes from IFC, the Rule layer evaluates a non-trivial rule set against the graph, and the Reasoning layer coordinates explanation, impact analysis, recommendations, and TRM support.

- **The system remains operational under imperfect IFC content.** Extraction routines skip malformed entities, and rule evaluation uses an explicit NOT_APPLICABLE outcome when required fields are absent. This behaviour preserves availability on real models and yields partial results rather than hard failure.
- **Behaviour is controlled through configuration rather than code.** Rule definitions, thresholds, and recommendation templates are externalised in JSON. This design supports jurisdiction changes and reduces regression risk from frequent code edits.
- **The graph representation is a useful intermediate artefact.** The JSON building graph normalises IFC heterogeneity into a consistent structure that can be consumed by rule evaluation, reasoning, and feature extraction. It also supports extension to new element types without redesign of the core pipeline.
- **Rule-informed feature engineering is functioning.** The feature pipeline combines element, rule, and context signals into a fixed-size vector, which creates a concrete interface between symbolic checks and TRM learning.
- **TRM outputs are suitable for triage.** The model returns a binary compliance prediction with a confidence score and a short stepwise trace, which supports escalation of low-confidence cases and structured inspection of model behaviour.
- **Recommendations are produced at multiple effort levels.** The recommendation engine generates tiered remediation guidance, which links compliance findings to practical interventions with different scopes and implementation costs.
- **The investigation confirms a data-governed failure mode.** The fixed-accuracy pattern arises when dimensional data collapses to defaults, driving feature vectors towards constant values. This links model behaviour directly to extraction and enrichment quality rather than to model design.

5.2 Key challenges

- **IFC incompleteness and inconsistency remain the dominant technical risk.** Missing dimensions, weak spatial relationships, and inconsistent identifiers occur in real submissions across schemas, which increases reliance on fallbacks and raises variance in downstream outputs.
- **Unit handling is fragile.** Heuristic conversions between metres, centimetres, and millimetres can fail on boundary values and are hard to validate when unit metadata is not preserved in the graph.
- **Training data volume and coverage are insufficient.** The TRM dataset remains small and narrow across building types and jurisdictions, which constrains generalisation and increases sensitivity to distribution shift.
- **Explanation coverage is uneven across rule types.** Numeric threshold rules map well to templates, yet complex checks and user-defined rules often fall back to generic text, reducing diagnostic usefulness.
- **Confidence scores are not empirically calibrated.** Softmax probabilities are treated as confidence without a demonstrated link to observed correctness rates, which complicates threshold setting for acceptance and review.

6 Discussion

This study set out to address a persistent limitation in automated compliance checking, namely the difficulty of producing compliance outcomes that remain inspectable, reconstructable, and trustworthy across technical and organisational boundaries. The results demonstrate that treating reasoning and traceability as core architectural concerns, rather than as reporting add-ons, leads to more stable and transparent compliance pipelines.

The implementation of ACE-X shows that a layered design can support this objective in practice. The Data, Rule, and Reasoning layers operate as independent modules with clear responsibilities, yet they remain tightly connected through explicit inputs and outputs. This separation supports robustness under imperfect IFC submissions, a condition frequently reported in real permitting workflows. The system's ability to continue operation despite missing or malformed elements, while making these limitations visible through explicit outcomes and logs, aligns with regulator-facing concerns raised in recent ACC studies.

A key observation concerns the role of intermediate representations. The normalised building graph serves not only as a technical convenience but as a stabilising artefact that reduces IFC heterogeneity and supports consistent rule evaluation, reasoning, and feature extraction. This reinforces arguments in the literature that graph-based or structured intermediate models are central to scalable and auditable ACC pipelines. In this work, the graph also becomes the point at which traceability can later be anchored, since it preserves links to original IFC identifiers while remaining independent of schema-specific complexity.

The results also highlight the importance of data governance in learning-assisted compliance systems. The fixed-accuracy behaviour observed during early experiments did not arise from model design weaknesses but from feature collapse caused by missing dimensional data. Once this issue was identified and corrected, model behaviour varied meaningfully across buildings. This finding underscores a broader point. Learning components in ACC systems amplify upstream data quality issues, and opaque performance metrics can mask such failures if feature construction is not explicit and inspectable. The study therefore reinforces the need to align explainability requirements with data preparation and feature engineering, not only with model outputs.

The integration of the Tiny Recursive Model illustrates how compact learning components can complement symbolic checking without undermining rule authority. By positioning the model as advisory, and by exposing confidence values and stepwise reasoning traces, the system avoids shifting compliance decisions away from formal rules. This design responds directly to concerns in the literature regarding accountability and validation when machine learning is introduced into regulatory workflows. At the same time, the current results confirm that training data coverage remains a limiting factor, and that confidence values require calibration before they can support operational thresholds.

Several challenges remain evident. Rule execution cost grows with model size and rule count, which constrains interactive use in large buildings. Unit handling and heuristic conversions remain fragile where IFC models lack explicit unit metadata. Explanation quality varies across rule types, with complex or contextual checks proving harder to express through templates. Most importantly, provenance

capture is not yet implemented as a first-class layer, which limits reproducibility and governance readiness despite architectural preparation.

These findings also carry broader implications for practice and governance. For digital building permitting, the framework suggests that ACC systems are more likely to be accepted when compliance outcomes can be traced back to explicit model evidence, rule logic, and intermediate reasoning steps, rather than being presented as opaque pass-or-fail decisions. For practitioners and regulators, this supports more accountable review processes, clearer identification of modelling deficiencies, and better alignment between automated checks and expert judgement. More broadly, for AI-assisted compliance systems, the study highlights that trust depends not only on prediction quality, but also on transparency, auditability, and the ability to preserve human oversight in cases of ambiguity or limited data quality.

These limitations do not undermine the core contribution of the study. Instead, they clarify where architectural decisions succeed and where further work is required. The results suggest that explainable automated compliance checking is less a problem of adding interpretive text and more a problem of designing pipelines in which decisions can be reconstructed from first principles.

7 Conclusions

This paper introduced ET-ACC, a conceptual framework for explainable and trustworthy automated compliance checking and presented ACE-X as a partial operational instantiation of the framework. The work demonstrates that explainability, reasoning transparency, and traceability can be embedded within compliance architectures through deliberate layering and explicit interfaces between data, rules, and reasoning processes.

The implementation confirms that automated compliance checking can remain operational under real-world IFC imperfections while still exposing the limits of the input data and the logic applied. The study shows that structured reasoning outputs, impact analysis, and tiered recommendations can be generated in a deterministic and inspectable manner when rule execution and reasoning are treated as connected stages rather than isolated functions.

A central contribution lies in the analysis of learning-assisted compliance checking. The identification of a data-governed failure mode, where missing dimensions caused feature collapse and misleading accuracy, provides a concrete lesson for the design and evaluation of hybrid symbolic–learning systems. The results demonstrate that model behaviour in such systems is inseparable from extraction quality and feature semantics, and that explainability must extend into data preparation and training workflows.

While the current implementation does not fully realise the Provenance and Explanation layers, it establishes a clear architectural path for doing so without restructuring the pipeline. This supports future extensions that can capture lineage, rule evolution, and decision reconstruction in a systematic way. The work also provides a foundation for empirical evaluation across broader building types, regulatory regimes, and user roles.

In summary, the study argues that trustworthy automated compliance checking depends less on increasing automation depth and more on making compliance processes legible. By aligning conceptual design with executable artefacts, ET-ACC and ACE-X offer a structured step towards compliance systems that can be inspected, justified, and governed with confidence.

References

1. Sun, H., & Kim, I. (2026). Global advancements in BIM-based building e-permit system adoption: a review. *Journal of Civil Engineering and Management*, 32(1), 1-20. <https://doi.org/10.3846/jcem.2026.24912>
2. Purushotham, N., Kailashnath, C., & Mutis, I. (2026). Framework for automated building code compliance checking to improve transparency, trust, validation, and design interpretation. *Automation in Construction*, 181, 106598. <https://doi.org/10.1016/j.autcon.2025.106598>
3. Pauwels, P., van den Bersselaar, E., & Verhelst, L. (2024). Validation of technical requirements for a BIM model using semantic web technologies. *Advanced Engineering Informatics*, 60, 102426. <https://doi.org/10.1016/j.aei.2024.102426>
4. Jia, L., Chen, M., Chen, C., & Jin, Y. (2025). A Semantic Web and IFC-Based Framework for Automated BIM Compliance Checking. *Buildings*, 15(15), 2633. <https://doi.org/10.3390/buildings15152633>
5. Chen, N., Lin, X., Jiang, H., & An, Y. (2024). Automated Building Information Modeling Compliance Check through a Large Language Model Combined with Deep Learning and Ontology. *Buildings*, 14(7), 1983. <https://doi.org/10.3390/buildings14071983>
6. Jolicoeur-Martineau, A. (2025). Less is More: Recursive Reasoning with Tiny Networks. *arXiv*. <https://arxiv.org/abs/2510.04871>
7. Celik, Y., Petri, I., & Barati, M. (2023). Blockchain supported BIM data provenance for construction projects. *Computers in Industry*, 144, 103768. <https://doi.org/10.1016/j.compind.2022.103768>
8. de Mendonça, E. A., & Ferreira, S. L. (2024). Use case for Automated Code Compliance Checking of Accessibility Rules in BIM models using the IDS standard. In *Proceedings of the 41st International Conference of CIB W78*. <https://itc.scix.net/paper/w78-2024-21>
9. Noardo, F., Guler, D., Fauth, J., Malacarne, G., Mastrolembro Ventura, S., Azenha, M., Olsson, P.-O., & Senger, L. (2022). Unveiling the actual progress of Digital Building Permit: Getting awareness through a critical state-of-the-art review. *Building and Environment*, 213, 108854. <https://doi.org/10.1016/j.buildenv.2022.108854>
10. Fuchs, S., Fauth, J., Boden, M., & Amor, R. (2025). The Challenge of Automated Compliance Checking: a Regulatory View. In *Proceedings of the 2025 European Conference on Computing in Construction*. <https://doi.org/10.35490/EC3.2025.264>
11. HeiB, M., Becker, S., & Rüppel, U. (2025). Extension of the Information Delivery Specification for BIM-based Checking in Data-driven Building Permit. In *Proceedings of the 2025 European Conference on Computing in Construction*. <https://doi.org/10.35490/EC3.2025.210>
12. Fischer, S., Urban, H., Schranz, C., Loibl, P., & van Berlo, L. (2024). Extending Information Delivery Specifications for digital building permit requirements. *Developments in the Built Environment*, 20, 100560. <https://doi.org/10.1016/j.dibe.2024.100560>
13. Wang, C., Zhang, L., & Yan, W. (2024). Enhancement and validation of ifcOWL ontology based on Shapes Constraint Language (SHACL). *Automation in Construction*, 160, 105293. <https://doi.org/10.1016/j.autcon.2024.105293>
14. Tao, X., Wu, Z., Xu, Y., Zheng, C., Fang, Y., Das, M., Liu, H., Gong, X., & Cheng, J. C. P. (2024). Smarter smart contracts for automatic BIM metadata compliance checking in blockchain-enabled common data environment. *Advanced Engineering Informatics*, 62, 102627. <https://doi.org/10.1016/j.aei.2024.102627>
15. Alnuzha, M., & Bloch, T. (2025). The role of machine learning in automated code checking – a systematic literature review. *Journal of Information Technology in Construction*, 30, 22-44. <https://doi.org/10.36680/j.itcon.2025.002>
16. Saluz, U., Baimuratov, I., & Geyer, P. (2025). semio: Large-language-model-based building-information-model alignment for automatic-compliance-checking - towards closing the gap between model authoring and model checking for kit-of-parts architecture. In A. Moreno-Rangel & B. Kumar (Eds.), *EG-ICE 2025: AI-Driven Collaboration for Sustainable and Resilient Built Environments Conference Proceedings*. <https://doi.org/10.17868/strath.00093288>

Disclaimer/Publisher's Note

The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and do not reflect the views of the Architecture, Buildings, Construction and Cities (ABC2) Journal and/or its editor(s). DFBF Journal and/or its editor(s) disclaim any responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.