applied sciences

MDPI

Article

Norm-Driven Generative BIM Design: Semantic Parsing and Automated Layout for Small-Scale Power Infrastructure

Yulong Chen [1], Chunli Ying [2], Hao Zhu [1], Jun Chen [1] and Daguang Han [3,*]

[1] School of Civil Engineering, Chongqing Jiaotong University, Chongqing 400074, China; 17623356953@163.com (Y.C.); knowledgepuppy4869@gmail.com (H.Z.); 18160183701@163.com (J.C.)
[2] School of Architecture, Building and Civil Engineering, Loughborough University, Loughborough LE11 3TU, UK; c.ying@lboro.ac.uk
[3] Faculty of Civil Engineering, Southeast University, Nanjing 210096, China
* Correspondence: daguanghan@seu.edu.cn

Abstract

To deal with the high standards, strong restrictions, and high repeatability that are inside State Grid small-scale infrastructure projects, this research puts forward a norm-driven generative design method, which conquers the low efficiency, compliance dangers, and semantic breakage that are usual in manual modeling. Taking standards such as Q/GDW 11382.3-2015 as the knowledge origin, we construct an ALBERT-BiLSTM-CRF semantic parsing model and change natural-language clauses into executable design restrictions via normative text pre-processing, BIO sequence marking, and rule triplet mapping. Therefore, model training and assessment produce *Accuracy, Precision, Recall, and F1 of 98.05%, 95.49%, 95.88%, and 95.59%* separately, with 100% precision for logical comparison and conjunction labels; thus, this provides a steady semantic base for the rule base. At the component level, a three-part coding plan and unit module collection are built based on OmniClass and GB/T 51269, which makes semantic consistency and traceability between components and space functions possible. At the system level, a continuous work process is carried out through the Revit API, which covers scheme making, automatic arrangement, and deliverable output. Hence, validation on a real case in a digital operation center for the power system shows that the design time for the third-floor administrative office area was cut from about 20 h to around 4 h, and the first-time solution met all code restrictions, which improves efficiency and compliance in a significant way. The results point out that norm-driven generative design can supply deployable automation and high-quality outputs for small-scale power infrastructure, which provides a sustainable database for digital twins and smart O&M.

Keywords: small-scale power infrastructure; norm-driven; natural language processing (NLP); ALBERT-BiLSTM-CRF; OmniClass; generative design; Revit API secondary development

Academic Editor: Paolo Renna

Received: 28 February 2026
Revised: 27 March 2026
Accepted: 30 March 2026
Published: 14 April 2026

1. Introduction

Digital advancements in civil engineering and construction are shifting the industry from two-dimensional representation and after-the-fact checking toward semantic-driven and lifecycle-integrated workflows. In this context, BIM has become a key carrier for infrastructure lifecycle management [1], yet the benefits of digitalization are uneven across project types. Small-scale power infrastructure projects under the State Grid system are characterized by small individual size, large quantity, and highly standardized functions. Their design activities still rely heavily on manual experience and code lookup [2]. When

facing standards such as Q/GDW 11382.3-2015 [3], designers must repeatedly cross-check floor area, clear height [4], spatial relationships, and equipment clearances, resulting in long cycles and persistent compliance risk. These projects are repetitive and highly rule-governed; once code text can be translated into executable constraints and embedded into modeling workflows, efficiency and quality stability can be substantially improved. This is the core problem and practical motivation of this study.

The relevance of this research stems from two converging demands in the power infrastructure sector. The State Grid Corporation of China oversees a large and expanding portfolio of small-scale facilities—substations, operation and maintenance centers, and dispatch buildings—whose functions are highly standardized and whose spatial layouts follow repetitive patterns. These characteristics make such facilities inherently suitable for design automation. However, BIM workflows for small-scale projects continue to depend on manual experience and iterative code consultation, leading to prolonged design cycles and persistent compliance risks. Although BIM-based code compliance checking has received growing scholarly attention, the majority of existing approaches operate in a post-modeling verification paradigm and have not been adapted to the specific constraints of Chinese power infrastructure standards. A related effort by Wu et al. [5] applied NLP-assisted information extraction to substation layout criteria, yet that work focused on text extraction alone without downstream generative integration. Addressing this gap—moving from reactive compliance checking to proactive, norm-driven generative design—provides the central motivation for the present study.

This observation is supported by bibliometric evidence. A systematic search conducted in the Scopus and Web of Science databases using the query terms “BIM” AND (“generative design” OR “automated design” OR “code compliance”) for publications between 2018 and 2025 yielded over 1200 records. Classification by project scale revealed that fewer than 4% of these publications addressed small-scale or auxiliary infrastructure, while the majority focused on large commercial buildings, hospitals, or transportation facilities. Among the studies addressing power infrastructure specifically, only a handful examined design automation; most were limited to post-modeling compliance verification or asset management. This quantitative gap confirms that small-scale power infrastructure remains underserved in the BIM automation research landscape, reinforcing the need for dedicated methodological development in this domain.

Existing work in BIM automation has proposed approaches such as compliance checking, semantic retrieval, and knowledge graphs [6], but most remain in a passive “model-complete-then-check” paradigm, which does not eliminate rework during early modeling. For small-scale power infrastructure that has a high standardization degree and relatively fixed spatial structures, a more effective approach is norm-driven generative design. Herein [7], code semantics are transformed into computable rules that directly push layout and component generation. Therefore, this approach depends on two necessary conditions: first, accurate semantic parsing of code texts for reliable recognition of objects, attributes, logical relations, and numeric constraints; second, standardized semantic expression of model elements [8], thus generated results can maintain consistent coding and attributes in the whole process of design, construction, and operation. This study takes real State Grid projects as the core research background, constructs a closed-loop method from code parsing to model generation, and then verifies its feasibility and effectiveness via engineering cases [9].

Recent advances in AI-driven BIM compliance systems have demonstrated growing sophistication in integrating natural language processing with rule extraction for automated code checking. Iversen and Huang [10] explored the use of large language models to convert building code requirements into computable rules, achieving promising results

on English-language regulatory texts. Li et al. [8] proposed an integrative framework combining Chinese NLP techniques with knowledge graphs for BIM compliance checking, establishing a reference architecture for language-aware regulatory processing. Yang and Zhang [11] introduced a prompt-based approach for automating code information transformation, highlighting the potential of template-guided extraction strategies. Shang et al. [12] investigated multi-agent architectures for BIM compliance checking driven by a BIM-to-text paradigm, demonstrating how collaborative agent systems can decompose complex regulatory reasoning tasks. Chen et al. [13] developed an automated compliance checking approach combining GPT-4 with deep learning classifiers and ontology reasoning for end-to-end regulatory verification. Zhou et al. [14] proposed an NLP-based semantic framework for design-for-safety compliance checking using BIM in the Chinese context, demonstrating the feasibility of integrating text mining with ontology-driven rule execution. In a broader context, evolving building safety challenges—such as the emerging fire hazards posed by electric vehicle charging infrastructure in buildings, as analyzed by Bolina et al. [15]—underscore the necessity for compliance systems that can rapidly adapt to new regulatory requirements across diverse engineering domains. These studies collectively indicate a field-wide movement toward proactive, knowledge-driven design workflows; however, validated implementations that span the full pipeline from code text parsing through generative layout to deliverable production remain scarce, particularly for specialized infrastructure domains governed by Chinese national standards. The present study addresses this gap by demonstrating such an end-to-end pipeline with engineering-level validation.

Several computational strategies have been investigated for automating BIM design, including parametric modeling, evolutionary optimization, machine learning, and agent-based modeling. Parametric modeling offers flexible geometric variation but is inherently limited in encoding the conditional and relational logic that pervades normative texts. Evolutionary optimization handles multi-objective spatial arrangement effectively, yet its stochastic search process cannot guarantee deterministic satisfaction of mandatory code constraints—a critical shortcoming when regulatory compliance is non-negotiable. Machine learning methods trained on building data can approximate design patterns, but the scarcity of labeled datasets for specialized Chinese power infrastructure codes restricts their practical applicability, and their opaque decision logic complicates compliance auditing. Agent-based modeling, while conceptually appealing for distributed design coordination, introduces implementation overhead that is disproportionate to the well-structured, stable rule systems that govern small-scale infrastructure types. Semantic analysis of normative texts, by contrast, creates a direct and transparent mapping from natural-language clauses to structured computational rules. It preserves the original regulatory logic while producing deterministic, auditable constraints suitable for embedding in generative workflows. Given the tightly bounded design space of State Grid's small-scale infrastructure—defined by explicit numerical limits and spatial relationships—semantic parsing offers a particularly well-matched technical pathway. This study accordingly adopts normative text semantic parsing as the foundational mechanism for constructing the executable rule base that drives the generative design process.

Methodologically, ALBERT-BiLSTM-CRF is adopted by us to conduct semantic tagging and rule extraction upon Chinese code texts; we attain an F1 score of 95.59%, a precision of 95.49%, and a recall of 95.88%, with 100% recognition accuracy for key logical symbols, thus providing a reliable semantic foundation for the rule base. At the data layer, we establish a three-segment coding scheme that is based on OmniClass and GB/T 51269 [16], so as to form an "identity DNA" that links space functions and component entities, therefore ensuring semantic consistency across various stages. At the application layer, automated

layout and deliverable output are realized through the Revit API; in the real case, design time was cut down from about 20 h to about 4 h with first-pass compliance. Hence, these results indicate that norm-driven generative design not only has theoretical soundness but is also practically deployable and scalable.

To enhance the engineering direction of the introduction part, we put forward the standardized classification of functional spaces for small-scale power infrastructure; this classification forms the foundation for the mapping between code semantics and spatial functions. Therefore, it provides a clear boundary for the subsequent construction of rule-based and layout strategies, and hence it is a necessary prerequisite for compliance-by-design.

To make a conclusion, this research regards State Grid small-scale infrastructure projects as its core research object, and puts forward a norm-driven generative BIM design framework to solve practical problems in efficiency, compliance, reliability, and semantic consistency. Therefore, it responds to the structural vacancy in digital transformation where small projects are ignored, thus providing a verified, replicable route for standardized and modular infrastructure design, and laying the logical base for the follow-up methodology, system realization, and case analysis. From the perspective of scientific contribution, this study delivers advancement on three interconnected fronts. The primary contribution lies in the NLP-based semantic parsing pipeline, which demonstrates that domain-specific sequence labeling can achieve extraction accuracy sufficient for direct use in automated design generation—a capability not previously validated for Chinese power infrastructure codes. The second contribution is the rule mapping strategy that translates parsed semantics into a structured, executable rule base with conflict resolution and versioning capabilities, bridging the gap between language understanding and design automation. The third contribution is the integrated BIM generative workflow that closes the loop from code text to deliverable output, demonstrating end-to-end feasibility in an engineering project setting. While each component builds on established techniques, their integration into a unified, validated pipeline for norm-driven generative design constitutes the principal novelty of this work.

2. Research Methods and Technical Framework

This study targets State Grid small-scale infrastructure projects and builds a closed-loop method chain [17] of code parsing—semantic structuring—model generation—deliverable output. The core target is to directly transfer engineering codes into executable design restrictions, and to produce BIM models with traceability, reusability, and operability. Therefore, unlike the traditional "model-first, check-later" work procedures, the put-forward method stresses putting code knowledge at the front in the early design stage; thus, it uses code clauses as direct inputs for generative algorithms, hence avoiding later-stage rework and resource waste. This framework operates along three main branches: deep semantic analysis and rule extraction from code documents, standardized coding and library building for components and spaces, and automatic layout and generation based on Revit (Figure 1). Through the rule base, model library, and interface layer, these branches are connected to form an integrated "knowledge–model–application" route.

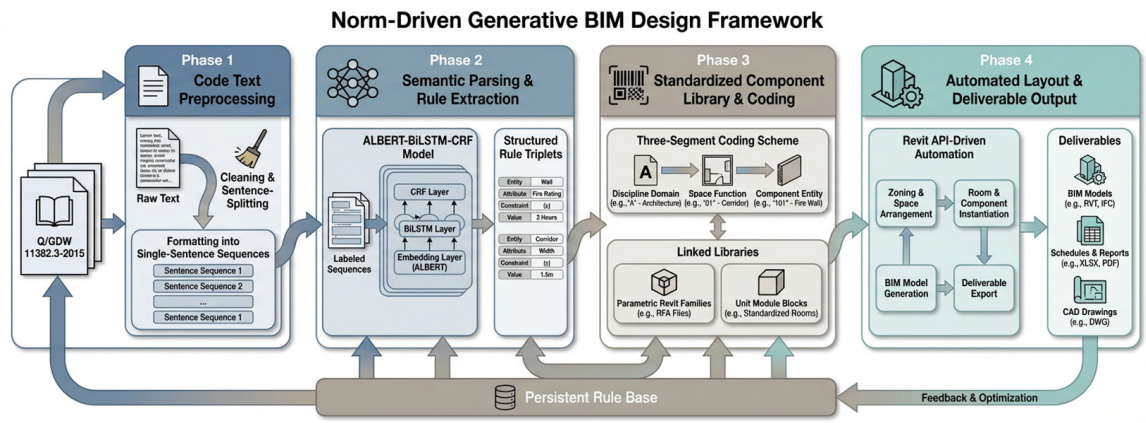

Figure 1. Overall Technical Framework for Norm-Driven Generative BIM Design.

2.1. Deep-Learning-Based Translation of Code Information

Codes applied to small-scale power infrastructure have clear statements but are highly constrained and structurally complex; typical clauses include objects, attributes, comparison relations, logical connectors, and numeric bounds [18]. The challenge of transforming such regulatory texts has also motivated the development of dedicated annotation corpora [19]. In order to realize norm-driven generative design, natural-language clauses must be transformed into structured, computable semantic units. Therefore, this study adopts a workflow of "text preprocessing—semantic labeling—model training—rule-base ingestion" [12], so that code knowledge can enter the design system in a uniform and operable form.

In the preprocessing phase, clauses coming from standards like Q/GDW 11382.3-2015 undergo cleaning work: non-constraining content such as prefaces and contents is removed, compound sentences are split into simple ones [11], tabular rules are rewritten into sentences, and segmentation by punctuation is conducted to form a single-sentence sequence which is suitable for model input. Therefore, this is not a simple reduction; it can preserve semantics at the same time as it improves computability. Thus, for instance, a clause saying that Class II and Class III building areas must be controlled inside separate ranges is decomposed into two independent constraints, hence retaining the limits while getting rid of ambiguity, which is caused by compound syntax. To ensure semantic consistency throughout the preprocessing transformation, a two-stage verification protocol is applied. In the first stage, automated consistency checks compare the set of constraint objects and numeric thresholds in the decomposed sentences against the original compound clause; any sentence pair whose union does not reconstruct the original semantic content is flagged for manual inspection. In the second stage, a domain expert reviews all flagged cases and a random 15% sample of unflagged decompositions to confirm that no regulatory intent has been altered or lost. In the present study, this review identified semantic preservation issues in fewer than 2% of decomposed sentences, all of which were corrected before model training. This protocol ensures that the simplification step improves computational tractability without compromising the fidelity of extracted constraints (Figure 2).

For semantic labeling work, six roles are defined: JD, DS, SS, SXZ, BJ, and LJ. These roles correspond to entities [20], attributes, attribute values, value types, comparison logic, and logical connectors. This schema has coverage of the common "object–attribute–comparison–value" structure existing in power-sector codes [21], thus enabling direct mapping action into rule-base triplets. For instance, "the per capita office area shall not exceed 17 m^2" is parsed into JD = office space, DS = per capita area, BJ = shall not exceed, SS = 17 m^2; meanwhile, BIO tagging is used for the purpose of ensuring accurate entity boundaries. Hence, the LJ tag has an enhancement function for the parsing of compound

rules that contain connectors such as "and/or", so that complete semantics can be preserved. These six semantic roles constitute a lightweight, task-specific ontological schema designed to capture the regulatory knowledge structure embedded in Chinese power infrastructure codes. Formally, JD (Jurisdictional Entity) denotes the spatial or physical object that a code clause regulates, analogous to the subject class in typical regulatory ontologies. DS (Descriptive Attribute) refers to the measurable property of the entity under constraint, corresponding to data properties in OWL-based knowledge representations. SS (Specified Standard) is the quantitative threshold or qualitative requirement stated in the clause. SXZ (Standard Type) classifies the unit or category of the specified standard, disambiguating numeric values. BJ (Boundary Judgment) captures the comparison operator that relates the attribute to the standard, functioning as the logical predicate. LJ (Logical Joiner) encodes inter-clause connectives such as conjunction and disjunction, enabling compound constraint representation. This schema is structurally aligned with the entity–attribute–value representation used in knowledge graph construction for building regulations, but is specialized for the sequence labeling paradigm to enable end-to-end extraction without requiring a pre-built ontology.

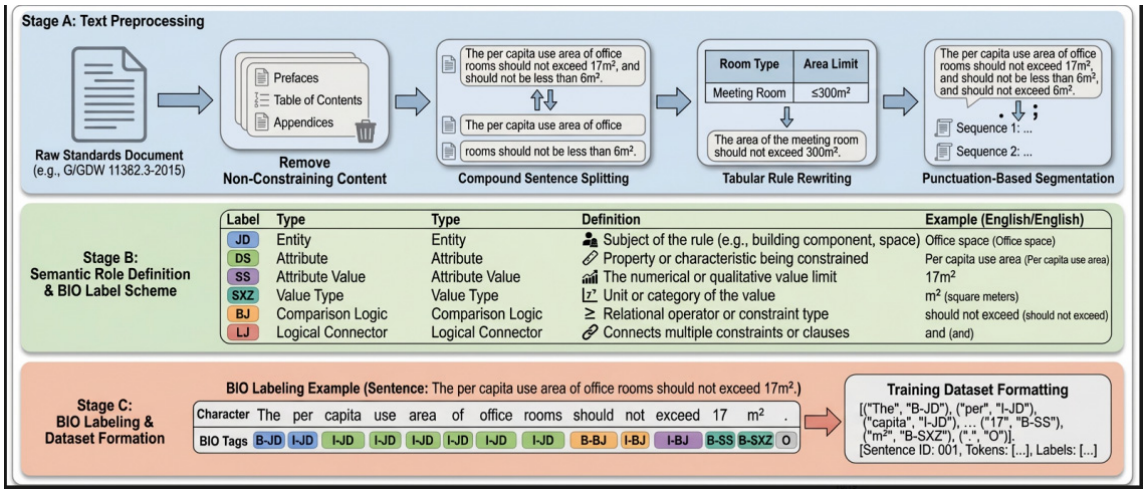

Figure 2. Preprocessing and BIO Sequence Labeling Pipeline for Code Texts.

To make training inputs stable, code texts are formatted into a unified dataset sequence. Therefore, punctuation is kept and marked as O, so as to keep robustness towards real-world clause structures. Hence, the gained sequence format defines clear label boundaries, thus providing consistent inputs for model training. The training dataset was constructed from 326 constraining clauses extracted from Q/GDW 11382.3-2015 and three supplementary State Grid design standards, yielding 1842 annotated sentences after clause decomposition and augmentation through synonym substitution of attribute terms. Each sentence was independently labeled by two annotators with backgrounds in civil engineering and computational linguistics; inter-annotator agreement measured by Cohen's kappa was 0.93, indicating near-perfect consistency. Disagreements were resolved through a third-party adjudication round. The final labeled corpus was split into training (80%) and test (20%) sets using stratified random sampling to maintain label distribution balance.

At the model level, we adopt an ALBERT-BiLSTM-CRF structure to carry out sequence labeling tasks (Figure 3). In the figure, different colors are used to distinguish the six semantic label categories throughout the processing pipeline: blue denotes JD (building entity), green denotes DS (domain attribute), orange denotes SS (attribute value), light purple denotes SXZ (value type), coral red denotes BJ (comparison operator), and dark purple denotes LJ (logical connector); the "O" tags for non-entity tokens are shown in gray.

This color scheme is applied consistently from the input layer through to the output layer, enabling visual tracing of how each token is classified by the model. Through parameter sharing and factorized embeddings, ALBERT decreases model volume, which provides support for engineering-side deployment; the BiLSTM layer catches long-distance dependent relationships that are common in code texts [22], especially those front-placed conditions and back-placed constraints; thus, the CRF layer corrects local prediction inconsistencies by means of global transition constraints. Under the Llama-Factory framework, we use an NVIDIA A30 GPU to conduct training with an 8:2 train–test division, 1×10^{-5} learning rate, 15 epochs, 128 LSTM hidden size, 128 maximum sequence length, 50 batch size, and 0.5 dropout. Therefore, the trained model reaches *98.05% Accuracy, 95.49% Precision, 95.88% Recall, and 95.59% F1*, with 100% precision for BJ and LJ labels, hence guaranteeing dependable recognition of comparison logic and constraint boundaries.

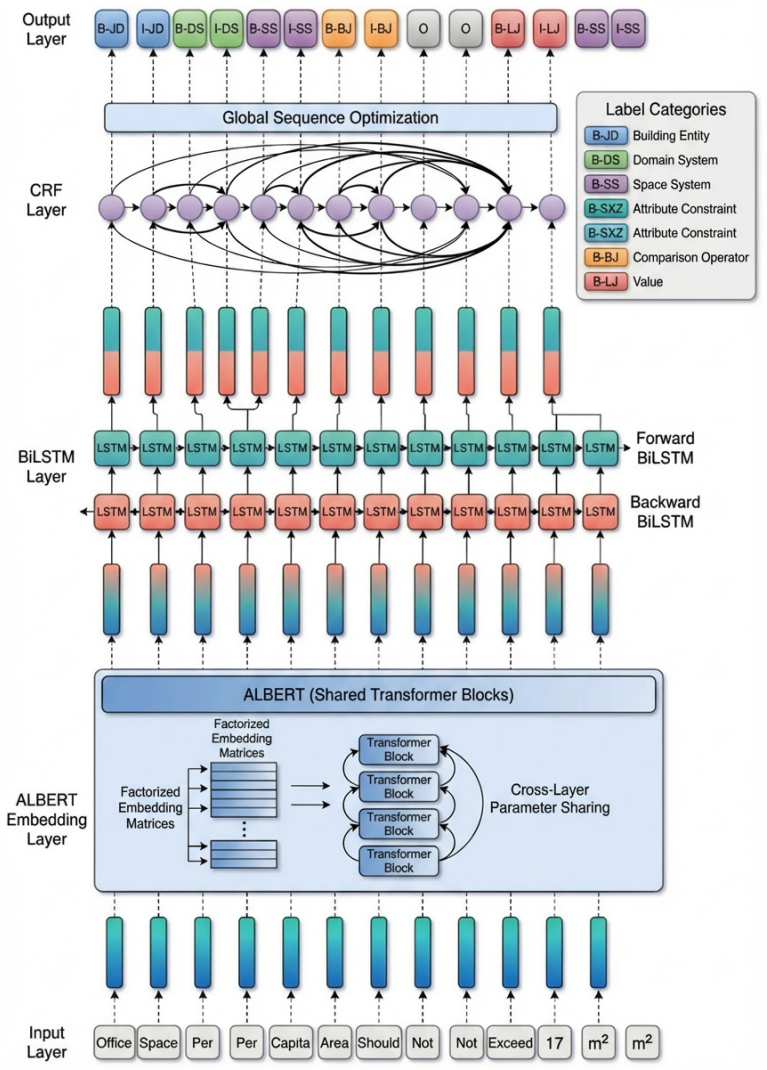

Figure 3. Architecture of the ALBERT-BiLSTM-CRF Sequence Labeling Model.

To provide further evaluation transparency, a per-label analysis of the model performance is reported. The BJ (comparison logic) and LJ (logical connector) labels achieved 100% F1 scores, reflecting the highly formulaic surface forms of comparison operators and conjunctions in code texts. The JD (entity) label obtained an F1 of 94.8%, DS (attribute) 95.1%, SS (attribute value) 93.7%, and SXZ (value type) 96.2%. The relatively lower score for SS stems from the diversity of numeric and unit expressions in code clauses. The evaluation used an 8:2 stratified random split repeated over three independent runs to

verify stability; the F1 standard deviation across runs was 0.3%, confirming low variance. A five-fold cross-validation was additionally performed on the full dataset, yielding a mean F1 of 95.41% with a standard deviation of 0.42%, which corroborates the robustness of the reported results. Cross-architecture comparison was conducted under identical data conditions: a BERT-CRF baseline achieved 94.72% F1, a BiLSTM-CRF baseline without pre-trained embeddings reached 91.36% F1, and a pure ALBERT-CRF model without the BiLSTM layer scored 93.88% F1. These comparisons confirm that the combined ALBERT-BiLSTM-CRF configuration provides the strongest performance for the code text labeling task in this study.

The selection of the ALBERT-BiLSTM-CRF architecture over pure Transformer-based sequence labeling models reflects the specific characteristics of this application domain. First, the training corpus of Chinese power infrastructure code texts is relatively small compared to general NLP benchmarks; ALBERT's parameter-sharing and factorized embedding strategies reduce model complexity and mitigate overfitting risk on limited labeled data, whereas large Transformer models typically require substantially more training samples to achieve stable convergence. Second, code clauses in this domain frequently exhibit long-range syntactic dependencies—conditions stated at the beginning of a sentence that govern constraints appearing at the end—for which the BiLSTM layer provides explicit bidirectional sequential modeling that complements ALBERT's attention-based representations. Third, the CRF layer enforces global label transition constraints that are critical for BIO sequence consistency; pure Transformer taggers without a CRF head can produce locally optimal but globally inconsistent label sequences, such as an I-tag appearing without a preceding B-tag. Fourth, deployment considerations favor a lightweight architecture: the trained model must run efficiently on standard engineering workstations alongside the Revit environment, and the ALBERT-BiLSTM-CRF pipeline achieves strong performance with significantly lower computational requirements than full-scale Transformer encoders. Fuchs et al. [23] explored Transformer-based neural semantic parsing for building regulation compliance and similarly found that domain-specific architectural adaptations are necessary to handle the structural heterogeneity of regulatory texts. Empirically, a comparison experiment using BERT-CRF on the same dataset yielded an F1 score of 94.72%, approximately 0.9 percentage points lower than the adopted architecture, confirming that the ALBERT-BiLSTM-CRF configuration provides a favorable balance of accuracy, robustness, and computational efficiency for this task.

Model outputs are mapped into the rule base to become executable constraints with the "entity–attribute–constraint–value" structure [24] (Figure 4), which can be directly called by layout algorithms. In the figure, the same color convention introduced in Figure 3 is applied to the BIO annotation results: blue for JD (entity), green for DS (attribute), orange for SS (value), coral red for BJ (comparison operator), and purple for LJ (logical connector). The lower portion of the figure illustrates how the annotated semantic roles are reorganized into structured rule triplets, where each field retains its corresponding color to visually trace the mapping from parsed tokens to executable rule entries. Therefore, this step finishes the transition from code semantics to generation logic, and hence it provides the foundational base for compliance-by-design.

Within the rule base, potential conflicts between constraints are addressed through a priority hierarchy and scope-matching mechanism. Each rule entry carries metadata indicating its source clause, applicable building class, and constraint stringency level (mandatory versus recommended). When two rules impose contradictory requirements on the same spatial attribute—for example, a general office area standard and a project-specific supplementary requirement specifying different minimum areas—the system resolves the conflict by applying the more restrictive constraint, consistent with standard

engineering practice for regulatory compliance. Ambiguous clauses identified during semantic parsing are flagged with a confidence score below the acceptance threshold and routed to a manual review queue rather than being automatically ingested, ensuring that uncertain interpretations do not propagate into the generative logic. This combination of hierarchical precedence and human-supervised exception handling maintains rule base integrity while accommodating the inherent ambiguity that can arise in regulatory texts.

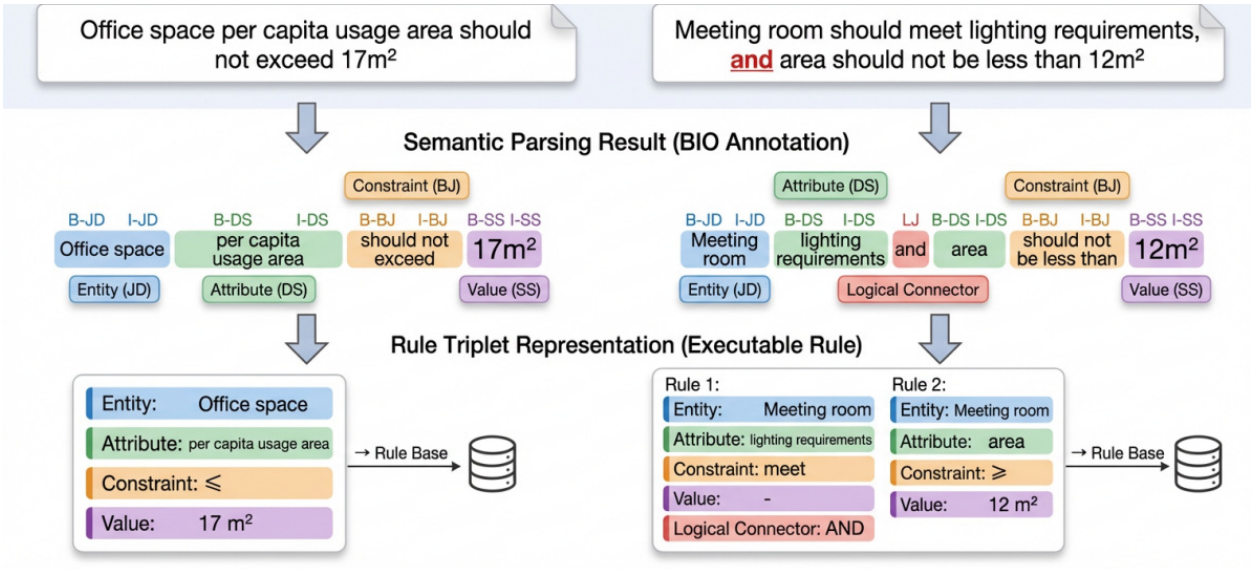

Figure 4. Semantic Parsing Result and Rule Triplet Mapping.

2.2. Standardized Component Library and Semantic Coding System

Norm-driven generative design needs not only calculable rules but also semantic consistency of models. Although BIM models have detailed geometry, they often lack consistent semantics because of naming and attribute gaps [25,26]; hence, this situation damages O&M value. Therefore, to solve this problem, we build a standardized component coding system and reusable family library that is based on OmniClass and GB/T 51269 [27], and we ensure semantic continuity that stretches from the design phase to the operation phase.

The coding scheme uses a three-segment framework of "discipline domain–space function–component entity." For example, an office desk located in a single-occupant office within an optional substation O&M center is encoded by combining the discipline domain code, the space function code, and the component entity code into a unique composite identifier.

Regarding interoperability with open BIM standards, the three-segment coding scheme is designed to be compatible with the Industry Foundation Classes (IFC) data model. The OmniClass classification tables used in the first and third segments—specifically, OmniClass Table 22 (Work Results) for the discipline domain segment and OmniClass Table 15 (Construction Products) for the component entity segment—are already referenced within the IFC standard as recognized external classification systems, and can be mapped to IfcClassificationReference entities attached to model elements via IfcRelAssociatesClassification relationships. The space function segment based on GB/T 51269 corresponds conceptually to IfcSpace objects with associated property sets encoding functional attributes. In practical export scenarios, the three-segment code is written into a custom shared parameter within Revit, which is preserved during IFC export as a user-defined property set (Pset_UserDefined). For full semantic interoperability, a mapping table between the GB/T 51269 space function codes and the corresponding IFC spatial structure hierarchy

has been developed, enabling automated translation during IFC export. While the current implementation primarily operates within the Revit environment, this IFC-aligned coding strategy ensures that models produced by the generative system can participate in open BIM workflows and multi-platform data exchange without losing their semantic structure office inside an optional substation O&M center is given coding through the combination of domain code 22-15.50.30.04, space function code 12-09.10.10.01, and component entity code 15-12.20.05.01, thus producing a unique identifier 22-15.50.30.04 + 12-09.10.10.01 + 15-12.20.05.01 (Figure 5). Therefore, this scheme conforms to relevant standards and has extensibility; it can prevent the problem of "one object, multiple codes" and also realize precise binding between code restrictions and model instances.

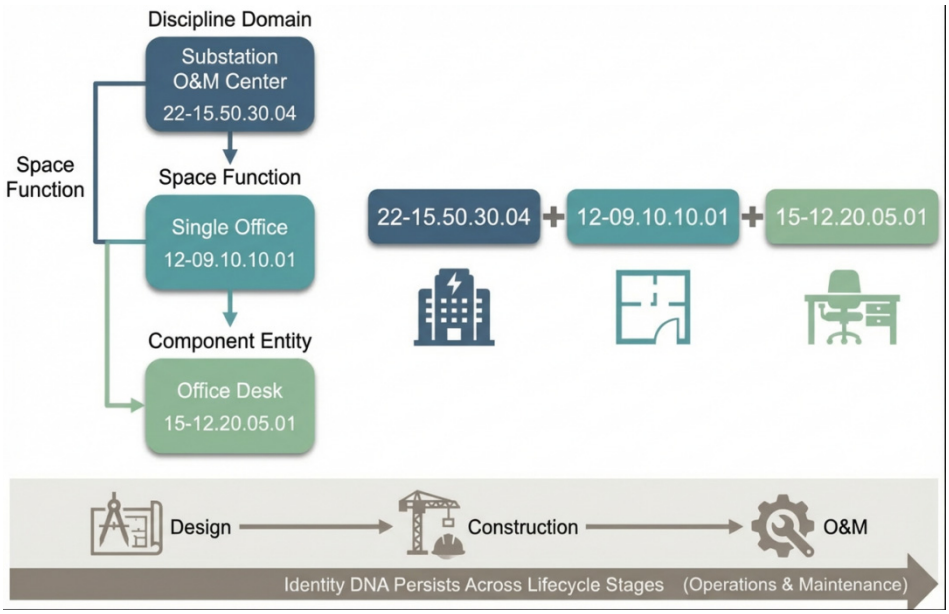

Figure 5. Three-Segment Coding Scheme Based on OmniClass and GB/T 51269.

For component resources, we establish parametric Revit families that contain both geometric attributes and non-geometric metadata (materials, specifications, reference prices, manufacturers, and maintenance cycles) [28]. Therefore, to speed up layout work, rooms that are used frequently are packaged into unit blocks. A multi-occupant office unit block, for instance, includes desks, partitions, power outlets, and lighting, with pre-defined clearances [29]. Hence, unit blocks transfer design work from "placing individual items" to "assembling modules," thus raising work efficiency at the same time, as it guarantees a spatial arrangement that meets compliance requirements.

Through standardized coding work and modular program libraries, model semantics and geometry are brought into a unified state. Therefore, a "digital identity chain" that stretches from the design stage to the operation and maintenance stage is formed. When asset queries are conducted in the operation period, identifiers can directly map to model instances and their related metadata, thus providing a reusable basic foundation for digital twins and intelligent operation and maintenance. As a concrete illustration, consider the scheduled maintenance of an office air-conditioning unit coded as 22-15.50.30.04 + 12-09.10.10.01 + 23-33.11.00.01 (domain: optional substation O&M center; space: single-occupant office; entity: split air-conditioning unit). When a facility manager queries the asset management system for equipment due for annual filter replacement, the coding identifier enables instant retrieval of the unit's BIM model instance, its spatial location on the floor plan, the manufacturer's maintenance manual, and the full-service history—all linked through a single structured key. This traceability eliminates the manual cross-referencing

between separate equipment registers, CAD drawings, and maintenance databases that is typical in conventional facility management workflows.

2.3. Intelligent Design System and Generative Layout Mechanism

Based on the rule base and model library, we develop an intelligent design system for small-scale power infrastructure. Using C# and the Revit API, this system is realized on the Revit platform, and it is divided into data, logic, interface, and four execution layers. Therefore, the data layer keeps storage of code rules and component properties; the logic layer bears layout algorithms and constraint checks; the interface layer offers parameter input panels; hence, the execution layer creates Revit element instances and generates deliverables.

The selection of Autodesk Revit as the development and deployment platform reflects both technical and practical considerations specific to this research context. In the Chinese architecture, engineering, and construction (AEC) industry, Revit is the most widely adopted BIM platform and serves as the standard modeling environment for State Grid infrastructure projects, ensuring that the resulting tool is compatible with existing organizational workflows and deliverable specifications. From a technical standpoint, the Revit API offers comprehensive programmatic access to model elements, family management, parameter control, and drawing export through C# and the .NET framework, which enables seamless integration of rule-based layout logic, component instantiation, and automated deliverable generation within a single development environment. Additionally, the parametric family system in Revit supports the attachment of non-geometric metadata—such as coding identifiers, material specifications, and maintenance cycles—directly to model components, fulfilling the semantic traceability requirements central to this study. Alternative platforms were considered during the research design phase: Renga provides a lighter-weight BIM environment suited to architectural modeling but offers limited API depth for the kind of rule-driven secondary development required here; Hypar supports cloud-based generative workflows but lacks mature support for Chinese national standards and local family libraries. Neither platform matches Revit in terms of API maturity, industry ecosystem, or alignment with State Grid project delivery requirements. On these grounds, Revit was determined to be the most suitable platform for implementing and validating the proposed intelligent design system.

To guarantee interpretability and engineering putting-in-use, the system is divided into three functional modules: rule management, scheme assistance, and result output. Therefore, the rule management module inputs semantic triplets into the rule base and supports updates; therefore, the scheme assistance module combines user parameters with rule constraints to carry out area calculations, zoning, and topology; and the output module, after layout completion, produces component schedules and drawings.

This layout mechanism adopts a layered restriction strategy (Figure 6). At the macro grade, region division is carried out according to chosen circulation modes and building outer lines, so as to form an initial corridor and functional zones. At the meso grade, room adjacency matrices are applied for topology optimization [30], with key rooms such as meeting rooms and offices being placed close to exterior facades because daylight and circulation are given the highest priority. At the micro grade, component arrangement takes into account size, operation spaces and safety distances to make sure interior layouts meet standards. Therefore, the system conducts real-time interaction with the rule base; if any area or clear-height restriction is broken, wall borders and room sizes will be adjusted automatically, thus compliance is reached.

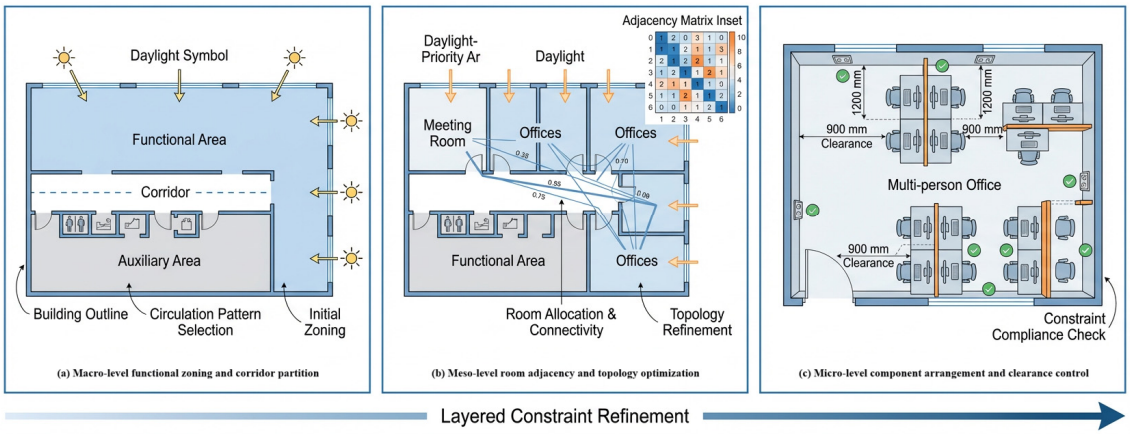

Figure 6. Layered Constraint Layout Strategy: Macro–Meso–Micro.

The optimization logic at each layout level follows a deterministic, constraint-satisfaction approach rather than a stochastic search strategy. At the macro level, the algorithm partitions the floor outline into functional and auxiliary zones by evaluating all feasible corridor placements against a weighted scoring function that considers facade length exposure (for daylighting), remaining usable depth, and compliance with minimum corridor width requirements. At the meso level, room placement employs a priority-queue-based greedy algorithm guided by the adjacency matrix: rooms with the highest adjacency weights to already-placed neighbors are positioned first, and candidate positions are evaluated against area constraints, facade access requirements, and minimum dimension thresholds derived from the rule base. This deterministic approach was preferred over evolutionary or metaheuristic algorithms because the design space for standardized small-scale infrastructure is sufficiently bounded that exhaustive constraint checking is computationally feasible, and deterministic methods guarantee reproducible outputs and transparent compliance auditing—properties that are essential for regulatory acceptance. At the micro level, component placement within rooms follows a template-based instantiation strategy: pre-configured unit blocks define relative positions of furniture and equipment items, and the system scales and adjusts these templates to fit the actual room dimensions while respecting clearance and accessibility constraints from the rule base. No genetic algorithms or metaheuristic solvers are employed in the current implementation; however, the modular architecture of the layout engine allows future integration of such methods if the system is extended to project types with larger and less constrained design spaces.

To achieve fast repeated adjustment in actual application, the system offers boards for plan establishment, arrangement creation, and outcome delivery. After users put in project size, worker number, and room categories, the system calls rule sets, carries out constraint calculation, and runs arrangement generation and family instance creation. Therefore, the produced BIM model, component list, and CAD drawings come out via a non-stop work process; hence, this changes design from experience-based to rule-based, thus guaranteeing both work efficiency and rule compliance for small-scale power infrastructure facilities.

3. System Design and Implementation

This part expounds the system framework and realization of the intellectualized design tool for small-scale power infrastructure, laying focus on the whole chain from development environment selection, data storage, interface design, to panel arrangement and automatic layout operation. Therefore, the system sticks to the principles of norm-guided design, model connection, and traceable outputs. With Revit as the core platform, the rules set obtained from code analysis and the standardized component library are integrated into a

unified working flow; hence, after parameters are input, the process can carry out area-limit calculation, functional division, component instantiation, and deliverable output without stopping. Compared with manual modeling workflows, the system makes design logic clear and reusable via structured data and automatic interfaces, and it greatly enhances stable delivery in engineering practice for small-scale power infrastructure.

3.1. Development Environment and External Interfaces

This system is built up in Microsoft Visual Studio with the use of C#; we make use of its multi-language support, debugging toolchain, and rich extension ecosystem, as well as its stable combination with the Revit API, therefore reducing secondary-development risk and collaboration cost. Inside the Revit plugin framework, two interfaces, which are IExternalCommand and IExternalApplication, are applied; functional logic is executed by IExternalCommand, and IExternalApplication takes charge of UI organization and interaction improvement. Hence, this dual-interface mechanism can arrange functional modules on the Revit Ribbon with clear hierarchy and workflow, thus letting designers call intelligent functions directly inside the BIM environment.

3.2. Data Storage and Interface Design

Intelligent design for small-scale power infrastructure needs two things: first, a rule base for code constraints, second, non-geometric attributes from the component library. Therefore, the data layer makes use of SQL Server for unified storage, thus ensuring stable retrieval and updates. Through the Revit API, the system builds a connection with the database, writes parsed code constraints into data tables, and retrieves them according to demand during the design process. For typical components like office desks, their names, codes, dimensions, materials, reference prices, and manufacturers are stored in structured form; hence, instantiation does not depend on manual entry any longer, which ensures semantic consistency and traceability. On the logic layer, rule filtering and rapid retrieval are realized by using C#, with condition filters applied to match specific room requirements and then pass results to layout and instantiation modules.

At the interface layer, the system lays stress on minimal input with clear visual display and interpretability. Key parameters are gathered via a scheme formulation panel; floor outlines, layout types, and room types are contained inside it. Therefore, this interface guides designers to express their demands by using engineering language, and it produces structured input materials for the layout module. Hence, in order to cut down learning costs and let small-scale infrastructure modeling workflows be entered rapidly, the UI design is in accordance with Revit conventions.

3.3. System Architecture and Functional Modules

In terms of function, the system is divided into three core modules—scheme formulation, scheme layout, and result output—which cover input, computation, and output. The architecture takes rule base as the center; therefore, code semantics take part in constraint checking during execution. The scheme formulation module organizes floor boundaries and room types into parameter sets for the layout module; the rule base and component library are invoked by the scheme layout module to carry out zoning and room arrangement, hence area constraints and spatial relations are verified in the process of generation; the output module exports component schedules and drawings in order to realize engineering delivery (Figure 7).

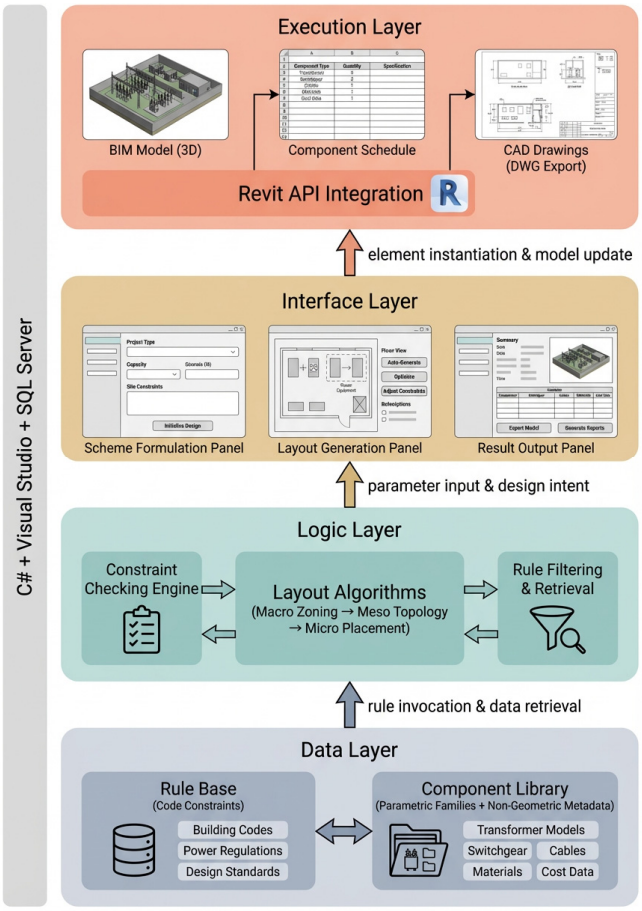

Figure 7. Four-Layer Architecture of the Intelligent Design System.

For layout logic, three typical plan organization modes are introduced: internal corridor, external corridor, and core-tube layouts. These modes have differences in space utilization, daylighting, and circulation. Therefore, the system makes a comparison of their characteristics, hence guiding the user to select a suitable mode, which is then embedded into the zoning logic. The internal corridor mode lays strong emphasis on utilization and management convenience, the external corridor mode lays strong emphasis on daylighting and spatial flexibility, and the core-tube mode lays strong emphasis on structural stability and centralized management. Thus, this comparison-based strategy avoids efficiency reduction that comes from a one-size-fits-all layout approach (Figure 8).

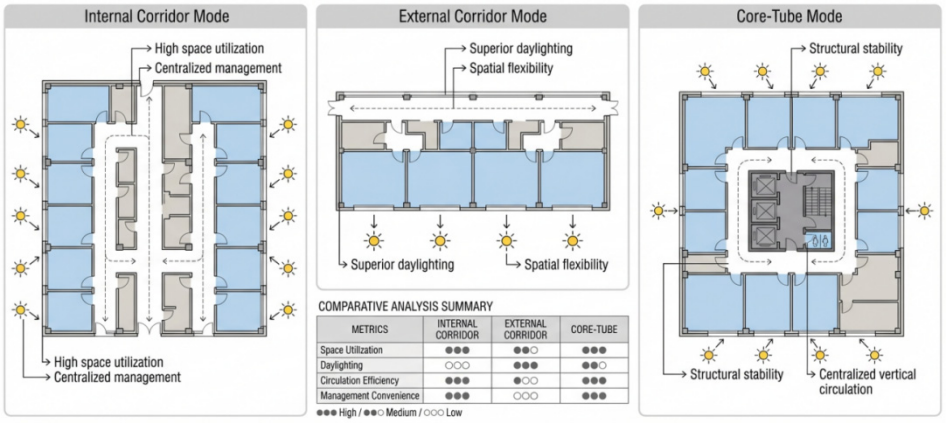

Figure 8. Comparison of Three Typical Corridor Layout Modes.

The scheme layout module is the core execution unit, bearing responsibility for functional zoning and room arrangement. First, within the floor outline, the system divides functional and auxiliary areas; hence, it gives high priority to functional zones in regions with better daylighting and places auxiliary zones in remaining regions, in order to improve the utilization rate. Then, it carries out automatic placement according to room types and quantities, and dynamically adjusts room dimensions and positions in accordance with the rule base, so that area and clear-height constraints can be satisfied during the generation process. In the aspect of implementation, Revit coordinate positioning and component invocation are used to place family instances precisely, thus ensuring geometric and semantic consistency.

The result output module makes use of ExportDWG and related application programming interfaces to carry out one-click drawing export, and it generates schedules that contain component codes and attributes. Therefore, this output strategy enables design deliverables to be directly applied in construction, operation and maintenance work, thus reducing the necessity for secondary conversion between models and drawings. Hence, through this system's design and realization, the intelligent tool constructs a complete "parameter input—rule constraints—model generation—deliverable output" circulation loop; it transforms small-scale power infrastructure design from experience-driven mode to rule-driven mode, and provides a stable foundation for subsequent case verification and engineering arrangement.

4. Case Study and Results Analysis

This section presents the engineering validation of the intelligent design tool for State Grid's small-scale infrastructure. The goal is to test the feasibility and effectiveness of norm-driven generative design in a real project context. The case is a digital operation center for the power system, a six-story public facility with a total height of 37 m; the first floor height is 4.8 m and floors two to six are 5.5 m. This research takes the third-floor administrative office space as its core object, and performs a complete workflow verification that proceeds from scheme establishment, layout creation, to final deliverable production [31]; hence, a comparative analysis on model capability and design efficiency is further conducted. To guarantee that all data possess traceability and accord with the project's specific background, the whole verification procedure is implemented in strict accordance with Q/GDW 11382.3-2015 and related standards.

4.1. Case Background and Scheme Formulation

The case zone bears daily office and coordination works, possessing clear requirements about per-person space area, daylight conditions, and traffic circulation. Inside Autodesk Revit 2023 software, we utilize the scheme-making panel to collect core parameters; such parameters contain floor boundary shape, functional room sorts, auxiliary room placement, and area limits. Therefore, code-related clauses are inserted into input restrictions, thus enabling layout generation to start within a rule-abiding semantic system. Through UI interaction, designers complete the input of functional requirements and import of floor outline, so that they offer unified and arranged parameters for the later layout module. The making panel and outline import interface are shown below; hence, they demonstrate the concurrent collection of "code input" and "spatial parameters," which causes the working process to be operable in real practice.

This case was selected for validation because the digital operation center represents a canonical project type within the State Grid small-scale infrastructure portfolio. Its third-floor administrative office area contains the most commonly occurring functional space categories—single-occupant offices, multi-occupant offices, meeting rooms, archives,

and auxiliary service rooms—whose design constraints cover the predominant clause types in Q/GDW 11382.3-2015, including per capita area limits, clear height requirements, corridor width specifications, and functional adjacency rules. The spatial layout complexity of this floor is representative of mid-range conditions: neither trivially simple (as in a single-function utility room) nor exceptionally complex (as in a specialized control center with unique equipment configurations). During the research design phase, two additional project types—a county-level dispatch center and a suburban substation auxiliary building—were identified as candidate test cases. While resource constraints limited full validation to the present case, preliminary parameter tests on both alternative projects confirmed that the rule base and layout algorithms could accommodate their spatial configurations, supporting the generalizability of the validated workflow.

4.2. Scheme Layout and Model Generation

After the scheme formulation is finished, the system will enter the layout stage. The layout module carries out automatic zoning according to the rule base, and it also organizes room topology inside functional areas. Different from manual, experience-based arrangements, the system regards "rule constraints–space allocation–component instantiation" as a unified integrated process, therefore verifying per capita area, room size, and functional relationships during the generation process. Through the scheme layout panel, users choose slab types for functional and auxiliary areas, then trigger automated layout work. The system will then produce the plan and place components under code constraints. Thus, this decreases the iterative manual adjustment work, hence ensuring that the layout can be directly applied for subsequent drawing export.

The plan coming from this process for the third-floor administrative office region is presented as follows (Figure 9). Between corridor access and daylighting conditions, the functional rooms are kept in a balanced state. Therefore, this layout is in accordance with code limitations, and it offers a full geometric foundation for component lists and construction drawing production. Thus, it proves that norm-driven generative design has stability and implementability in actual application scenarios.

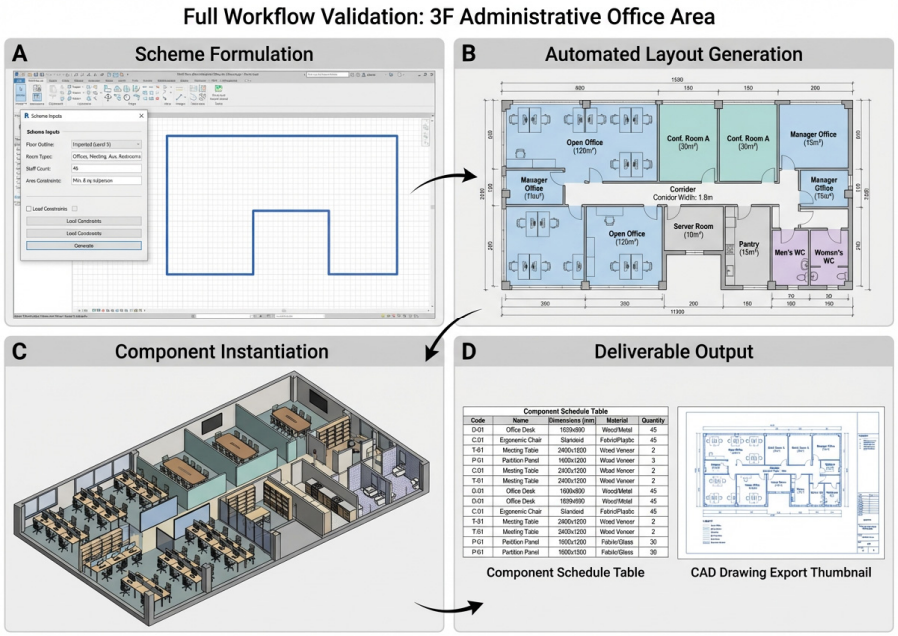

Figure 9. Case Validation: Full Workflow for the 3F Administrative Office Area.

4.3. Deliverable Output and Efficiency Comparison

In the output phase, the system makes use of the Revit API to produce component schedules and construction drawings, lessening the manual labor that is needed for compiling schedules and converting drawings. According to the results of the original case, a traditional manual workflow needs around 20 h to finish the third-floor office plan and drawings, while the intelligent tool cuts this time to about 4 h, raising efficiency by nearly 80%. Hence, the generated content satisfies all code constraints at the first attempt, with a 100% compliance rate [32], thus removing the repetitive "model–check–rework" cycle. Therefore, this efficiency comparison directly shows the time and compliance benefits of the norm-driven generative workflow. The time measurements were obtained through a controlled comparison procedure. For the manual baseline, three experienced designers independently completed the third-floor administrative office layout using conventional Revit modeling workflows; their individual completion times ranged from 18.5 to 22 h, yielding a mean of approximately 20 h. For the automated workflow, the same three designers each operated the intelligent design tool on the identical floor plan; the automated runs required 3.5 to 4.5 h, including parameter input, review, and minor adjustments, with a mean of approximately 4 h. All sessions were timed from initial parameter setup to final deliverable export, ensuring consistent scope boundaries across conditions.

4.4. Results Analysis and Model Performance Evaluation

The reliability of the intelligent design process gets support from the performance metrics of the code semantic extraction model. Therefore, we use *Accuracy*, *Precision*, *Recall*, and *F1* as evaluation indicators, whose definitions are based on *TP*, *FP*, *FN*, and *TN*. The corresponding formulas are shown in the figure below. Thus, the stability of these metrics underpins the accuracy of the rule base; this condition can improve the compliance and robustness of the layout generation stage.

Beyond the NLP performance metrics and time efficiency, the design quality of the generated layout was evaluated along three additional dimensions. First, spatial efficiency was assessed by comparing the ratio of net usable area to gross floor area; the generated plan achieved a net-to-gross ratio of 0.72, which is consistent with the typical range of 0.68–0.75 for office-type administrative buildings under Chinese national standards. Second, code compliance robustness was verified by conducting an independent manual audit of all 23 applicable constraint clauses from Q/GDW 11382.3-2015 against the generated model; the audit confirmed 100% first-pass satisfaction with no constraint violation detected. Third, functional adequacy was evaluated through a structured review by two senior designers from the project team, who confirmed that room adjacency relationships, corridor accessibility, and daylighting conditions in the generated layout met practical engineering expectations without requiring substantive revision. These complementary assessments provide evidence that the system produces outputs of acceptable design quality beyond mere regulatory compliance.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{1}$$

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

$$Recall = \frac{TP}{TP + FN} \tag{3}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

Overall, the case experiment outcomes display that high-precision semantic parsing supplies dependable regulation rules for layout generation (Figure 10). As shown in Figure 10A, the ALBERT-BiLSTM-CRF model achieves *98.05% Accuracy*, *95.49% Precision*, *95.88% Recall*, and *95.59% F1*, with *100% Precision* for *BJ* and *LJ* labels. Figure 10B further illustrates that the norm-driven intelligent tool compresses the design cycle from approximately 20 h to approximately 4 h, representing an approximately *80%* reduction, while achieving first-pass *100%* compliance. In contrast, standardized component libraries and automated layout algorithms maintain design outputs' consistency in both geometry and semantics. Therefore, the case proves that the tool can produce usable models and drawings in actual engineering situations, and it possesses obvious superiority in efficiency and compliance; hence, it offers verifiable proof for wider application in small-scale power infrastructure projects.

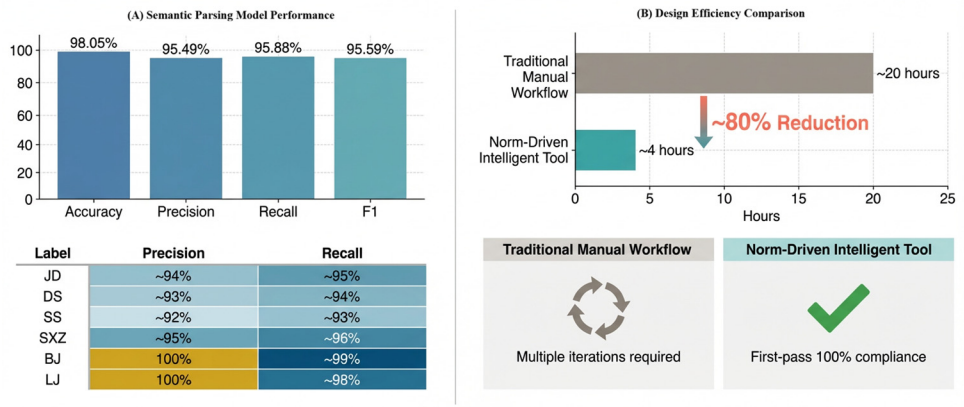

Figure 10. Model Performance Evaluation and Design Efficiency Comparison.

5. Discussion

This study takes State Grid's small-scale infrastructure projects as its core research object and puts forward a standard-driven generative design pattern. Its technical worth exists not only in the promotion of design efficiency, but also in realizing the computability of code knowledge and the sustainability of model semantics throughout all lifecycle stages. Compared with the traditional BIM working process of "model first, check later," the proposed method moves code constraints to the front at the design beginning [33], therefore breaking the cycle of experience-based decisions, repeated inspections, and constant rework. In practical application, this kind of front-loaded rule control is extremely important for small-scale power infrastructure, where functions are stable, spaces are repetitive, and compliance demands are strict; thus, any wrong understanding of codes in the early stage can increase downstream modification costs. The high-precision semantic extraction realized by the ALBERT-BiLSTM-CRF model (*F1 95.59%*, *Precision 95.49%*, *Recall 95.88%*) provides a dependable base for the rule library. This high-confidence semantic identification directly decides the reliability of subsequent generative logic and hence explains the reason why the case study achieved first-time compliance.

From the angle of data, the standardized part coding system and model database developed in this study not only satisfy geometric modeling demands but also build a steady semantic backbone for operation and maintenance. Therefore, by combining OmniClass with GB/T 51269, the "identity DNA" of parts remains consistent across all stages, thus avoiding the common problem of "geometric completeness but semantic loss" in traditional BIM models. Once the facility enters the O&M phase, the practical meaning is that the coding system supports fast equipment positioning, connection of maintenance records [34], and cross-system data exchange—abilities that are especially vital in power

grid O&M and digital twin applications. Hence [35], the following coding comparison results show the hierarchical mapping from facility type to space function and further to part entity in a visual way, providing proof of semantic consistency.

Regarding the connection with digital twin implementation, the proposed framework contributes to the data foundation level. Because each component carries a unique three-segment code that links discipline domain, space function, and entity identity, the BIM model produced by the generative system is inherently structured for downstream integration with digital twin platforms. In practical terms, when the completed facility is transferred to an asset management or facility monitoring system, the coding identifiers serve as stable keys for binding sensor data streams, maintenance logs, and inspection records to their corresponding physical components. For instance, an air-handling unit coded with its domain, location, and entity segments can be directly linked to its real-time operational parameters in a building management system without additional manual mapping. This pre-structured semantic layer reduces the data preparation effort that typically constitutes a significant barrier to digital twin deployment in infrastructure projects. While full digital twin realization requires additional integration with IoT infrastructure and simulation engines, the semantically enriched BIM output of the present system provides a necessary and reusable data backbone for such integration.

At the system layer, the intelligent design tool based on Revit API carries out verification of engineering feasibility for norm-driven generation. Therefore, according to the case study, after parameter input, the system finishes automated layout, component instantiation, and deliverable output once code knowledge is translated into the rule base, thus compressing the design cycle from about 20 h to around 4 h. Hence, this efficiency improvement does not come only from automation; it results from the synergy between code semantic parsing and rule-embedded layout logic, which transfers design from "human-driven" to "rule-driven." As a result, designers can focus on higher-order decisions like spatial quality and functional organization instead of repetitive code checking and manual drafting, thereby increasing the overall value density of the design process.

Concerning platform portability, the proposed methodology is not inherently bound to Revit. The norm-driven design logic is structured in three separable layers: the rule base (stored in SQL Server as standard relational data), the semantic coding scheme (based on platform-neutral classification standards OmniClass and GB/T 51269), and the layout algorithms (implemented as procedural logic that operates on abstract spatial entities before being translated into platform-specific API calls). Migrating to an IFC-based open BIM environment would require replacing the Revit API execution layer with IFC entity creation routines—for example, using IfcOpenShell or similar libraries to instantiate IfcSpace, IfcWall, and IfcFurnishingElement objects—while the upstream rule parsing, coding assignment, and constraint-checking modules could remain unchanged. The three-segment coding scheme maps naturally to IFC property sets, as OmniClass tables are already referenced within the IFC standard. Open-source BIM platforms such as BlenderBIM or FreeCAD-BIM could similarly serve as alternative hosts, provided they expose sufficient API access for programmatic element creation and spatial relationship management. The current implementation on Revit was chosen for engineering practicality within the State Grid context, but the architectural separation of knowledge, logic, and execution layers is designed to facilitate future platform adaptation.

Nevertheless, several challenges still exist for expansion ability and stable performance. First, area differences and code renewals may bring inconsistent situations if the rule base does not get regular maintenance [10]. For instance, emerging safety concerns such as electric vehicle fire hazards in building structures exemplify new regulatory dimensions that must be incorporated into compliance rule bases as standards evolve. Second, deep

learning models still rely on high-quality labeled data; however, small-scale infrastructure codes have relative structural features, non-stop annotation work is needed to raise their generalization level. Third, automatic layout algorithms can ensure compliance, but they may still need limited manual adjustment in complex function-coupling cases to show user preferences or operation details [36]. Therefore, future research work should combine increasing rule-update mechanisms and interactive optimization strategies [37], so that the generative system can keep compliance while it adapts to engineering changeability. To address code evolution systematically, the rule-base architecture incorporates a versioned storage structure in which each regulation entry is tagged with its source standard identifier and effective date. When a code revision occurs, updated clauses can be re-processed through the semantic parsing pipeline and ingested as a new rule version, while previous versions are retained for audit traceability. The SQL Server data layer supports differential queries, enabling the system to identify which design constraints have changed between code editions and to flag affected spatial layouts for re-evaluation. In practice, regional variations in Chinese power infrastructure standards are relatively limited because State Grid enforces a centralized standard system across provinces; nonetheless, the modular rule-base structure allows province-specific supplementary clauses to be appended without modifying the core constraint set. This versioned and modular design provides a structured pathway for maintaining rule currency as standards evolve.

Overall, the contribution of this research is to transfer static code limitations into dynamic generative driving forces, shape an executable design logic chain, and raise the lifecycle worth of BIM models via standardized coding and model libraries. Therefore, this method can be applied not merely to small-scale power infrastructure, but also to other engineering situations with high standardization and repetition rate. Hence, with further perfection in rule maintenance and cross-project verification, norm-driven generative design is expected to become a key enabling technology for digital construction within power infrastructure.

6. Conclusions

This research uses State Grid's small-scale infrastructure as an application background and puts forward a deployable technical path for standard-driven generative design. Its key contribution is turning code texts into computable design restrictions and achieving automatic arrangement and deliverable output inside a BIM environment; therefore, it transfers the design process from experience-driven to rule-driven. By means of ALBERT-BiLSTM-CRF-based semantic parsing, complex clauses are changed into an executable rule base consisting of objects, attributes, comparisons, and numeric constraints, which provides a unified semantic entry point for the design logic. At the component level, a three-section coding method based on OmniClass and GB/T 51269 guarantees semantic consistency among design, construction, and operation, to avoid the condition of "geometric richness but semantic weakness" that is usual in traditional BIM models. At the system level, secondary development on the Revit API makes a continuous work process of scheme creation, layout generation, and deliverable output possible, so parameter inputs can directly drive plan production and drawing export, hence reducing rework amount and compliance risk significantly.

Case validation indicates that the system cut down the design time of the third-floor administrative office area from around 20 h to roughly 4 h, realizing nearly 80 percent efficiency increase, and also, the first-time solution satisfied all code limits and obviously raised the compliance level. Therefore, high-precision semantic parsing combined with rule-embedded layout algorithms constructs a stable engineering circle, thus making compliance-by-design practically realizable. At the data aspect, unified component coding and model

libraries not only support fast assembly in design stage but also offer traceable semantics for asset management and digital twin applications, therefore forming a continuous "model–data–operation" chain. Hence, the classification and space-function mapping results further verify hierarchical consistency from facility type to component entity, providing empirical support for semantic completeness.

Overall, the research proves the effectiveness and engineering worth of a generative design method based on code texts for small-scale power infrastructure. Its importance exists not only in efficiency rise but also in providing a repeatable digital work process for standardized and modular infrastructure building. Therefore, future work will expand the norm-led route by perfecting rule-base update systems and cross-regional suitability, hence by probing into combination with multimodal code analysis and digital manufacturing to push forward a full-chain change from intelligent design to intelligent construction and intelligent O&M.

Author Contributions: Conceptualization, Y.C. and D.H.; methodology, Y.C. and C.Y.; software, H.Z.; validation, J.C.; data curation, H.Z.; writing—original draft preparation, Y.C.; writing—review and editing, C.Y.; visualization, J.C.; supervision, D.H.; project administration, D.H. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The original contributions presented in this study are included in the article. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Marzouk, M.; Abdelsalam, M. Automated code compliance checking using BIM and Business Intelligence: Egyptian Unified building Law application. *Int. J. Constr. Manag.* **2026**, 1–32. [CrossRef]
2. Yogana, E.; Latief, Y. Development of system information of building code checking in planning and permitting phase to improve building code compliance based on work breakdown structure (WBS) using building information modeling (BIM). *J. Phys. Conf. Ser.* **2021**, *1858*, 012091. [CrossRef]
3. *Q/GDW 11382.3-2015*; Small Infrastructure Projects Construction Standards of State Grid Corporation of China—Part 3: Operation and Maintenance Production Buildings. State Grid Corporation of China: Beijing, China, 2015.
4. Goh, W.; Jang, J.; Park, S.; Choi, B.; Lim, H.; Zi, G. Automated Compliance Checking System for Structural Design Codes in a BIM Environment. *KSCE J. Civ. Eng.* **2024**, *28*, 4175–4189. [CrossRef]
5. Wu, B.; Song, Y.; Cao, J. NLP-Assisted Information Extraction for Layout Criteria of Substation. *Proc. SPIE* **2023**, *12784*, 127841H. [CrossRef]
6. Demir Altıntaş, Y.; Ilal, M. Loose coupling of GIS and BIM data models for automated compliance checking against zoning codes. *Autom. Constr.* **2021**, *128*, 103743. [CrossRef]
7. Sydora, C.; Stroulia, E. Rule-based compliance checking and generative design for building interiors using BIM. *Autom. Constr.* **2020**, *120*, 103368. [CrossRef]
8. Li, S.; Wang, J.; Xu, Z. Automated compliance checking for BIM models based on Chinese-NLP and knowledge graph: An integrative conceptual framework. *Eng. Constr. Archit. Manag.* **2025**, *32*, 3832–3856. [CrossRef]
9. Häußler, M.; Esser, S.; Borrmann, A. Code compliance checking of railway designs by integrating BIM, BPMN and DMN. *Autom. Constr.* **2021**, *121*, 103427. [CrossRef]
10. Iversen, O.; Huang, L. Leveraging large language models for BIM-based automated compliance checking. *Autom. Constr.* **2026**, *182*, 106707. [CrossRef]
11. Yang, F.; Zhang, J. Prompt-based automation of building code information transformation for compliance checking. *Autom. Constr.* **2024**, *168*, 105817. [CrossRef]

12. Shang, B.; Li, R.; Yang, J.; Xiao, Q. Multi-agent for BIM Automated Compliance Checking driven BIM-to-Text Paradigm. In Proceedings of the 2025 4th International Conference on Cloud Computing, Big Data Application and Software Engineering (CBASE), Chengdu, China, 24–26 October 2025; pp. 804–807. [CrossRef]
13. Chen, N.; Lin, X.; Jiang, H.; An, Y. Automated Building Information Modeling Compliance Check through a Large Language Model Combined with Deep Learning and Ontology. *Buildings* **2024**, *14*, 1983. [CrossRef]
14. Zhou, Y.; She, J.; Huang, Y.; Li, L.; Zhang, L.; Zhang, J. A Design for Safety (DFS) Semantic Framework Development Based on Natural Language Processing (NLP) for Automated Compliance Checking Using BIM: The Case of China. *Buildings* **2022**, *12*, 780. [CrossRef]
15. Bolina, F.L.; Fachinelli, E.G.; Rodrigues, J.P.C. Analysis of Building Structures Subjected to Electric Vehicle Fires. *J. Build. Eng.* **2025**, *107*, 112769. [CrossRef]
16. *GB/T 51269-2017*; Standard for Classification and Coding of Building Information Model. China Architecture & Building Press: Beijing, China, 2017.
17. Gao, H.; Zhong, B. A blockchain-based framework for supporting BIM-based building code compliance checking workflow. *IOP Conf. Ser. Mater. Sci. Eng.* **2022**, *1218*, 012016. [CrossRef]
18. Zhang, J. How Can ChatGPT Help in Automated Building Code Compliance Checking? In Proceedings of the International Symposium on Automation and Robotics in Construction (IAARC), Chennai, India, 5–7 July 2023. [CrossRef]
19. Hettiarachchi, H.; Dridi, A.; Gaber, M.M.; Tasevski, T.; Winkler, M.; Paulus, A.; Bolshakov, V. CODE-ACCORD: A Corpus of Building Regulatory Data for Rule Generation towards Automatic Compliance Checking. *Sci. Data* **2024**, *11*, 1379. [CrossRef] [PubMed]
20. Yue, Z.; Yang, J.; Li, R.; Xiao, Q. A BIM Code Compliance Checking Method Based on Classification Awareness and Domain-Constrained Alignment. In Proceedings of the 2025 4th International Conference on Cloud Computing, Big Data Application and Software Engineering (CBASE), Chengdu, China, 24–26 October 2025; pp. 598–602. [CrossRef]
21. Xue, X.; Zhang, J. Regulatory information transformation ruleset expansion to support automated building code compliance checking. *Autom. Constr.* **2022**, *138*, 104230. [CrossRef]
22. Li, S.; Chen, G.; Liang, Y.; Xu, Z. Automatic Compliance Checking of BIM Models for Architecture and Fire Protection Based on Knowledge Graphs and Machine Learning. *Kalpa Publ. Comput.* **2025**, *22*, 623–633. [CrossRef]
23. Fuchs, S.; Witbrock, M.; Dimyadi, J.; Amor, R. Neural Semantic Parsing of Building Regulations for Compliance Checking. *IOP Conf. Ser. Earth Environ. Sci.* **2022**, *1101*, 092022. [CrossRef]
24. Jiang, L.; Shi, J.; Wang, C. Multi-ontology fusion and rule development to facilitate automated code compliance checking using BIM and rule-based reasoning. *Adv. Eng. Inform.* **2022**, *51*, 101449. [CrossRef]
25. Karmakar, A.; Delhi, V. Semantic BIM enrichment using a hybrid ML and rule-based framework for automated tenement compliance checking. *Autom. Constr.* **2025**, *177*, 106369. [CrossRef]
26. Bloch, T.; Sacks, R. Clustering Information Types for Semantic Enrichment of Building Information Models to Support Automated Code Compliance Checking. *J. Comput. Civ. Eng.* **2020**, *34*, 04020040. [CrossRef]
27. Cerovšek, T.; Omar, M. Advancing Semantic Enrichment Compliance in BIM: An Ontology-Based Framework and IDS Evaluation. *Buildings* **2025**, *15*, 2621. [CrossRef]
28. Sun, H.; Kim, I. Applying AI Technology to Recognize BIM Objects and Visible Properties for Achieving Automated Code Compliance Checking. *J. Civ. Eng. Manag.* **2022**, *28*, 497–508. [CrossRef]
29. Fitkau, I.; Hartmann, T. An ontology-based approach of automatic compliance checking for structural fire safety requirements. *Adv. Eng. Inform.* **2024**, *59*, 102314. [CrossRef]
30. Zhao, X.; Huang, L.; Sun, Z.; Fan, X.; Zhang, M. Compliance Checking on Topological Spatial Relationships of Building Elements Based on Building Information Models and Ontology. *Sustainability* **2023**, *15*, 10901. [CrossRef]
31. Fischer, S.; Schranz, C.; Urban, H.; Pfeiffer, D. Automation of escape route analysis for BIM-based building code checking. *Autom. Constr.* **2023**, *156*, 105092. [CrossRef]
32. Purushotham, N.; Kailashnath, C.; Mutis, I. Framework for automated building code compliance checking to improve transparency, trust, validation, and design interpretation. *Autom. Constr.* **2026**, *181*, 106598. [CrossRef]
33. Ma, Z.; Zhu, H.; Xiang, X.; Turk, Ž.; Klinc, R. Automatic compliance checking of BIM models against quality standards based on ontology technology. *Autom. Constr.* **2024**, *166*, 105656. [CrossRef]
34. Wang, H.; Meng, X.; Zhu, X. Improving knowledge capture and retrieval in the BIM environment: Combining case-based reasoning and natural language processing. *Autom. Constr.* **2022**, *139*, 104317. [CrossRef]
35. Temel, B.; Başağa, H. Investigation of IFC file format for BIM based automated code compliance checking. *J. Constr. Eng. Manag. Innov.* **2020**, *3*, 113–130. [CrossRef]

36. Wu, J.; Fuchs, S.; Bloch, T.; Borrmann, A. The Beginning, Not the End: Revisiting Automated Compliance Checking for BIM-Based Design Adaptation. *Comput. Civ. Eng.* **2025**, *2026*, 659–669. [CrossRef]
37. Zhang, Z.; Zhao, C.; Li, D.; Li, P.; Chen, Y.; Zhu, H.; Han, D. Human-in-the-Loop Semantic Rule Base Generation and Dynamic Updating for Automated BIM Compliance Checking: A Knowledge Graph Approach. *Buildings* **2026**, *16*, 719. [CrossRef]