buildings

MDPI

Article

Semantic Mapping and Cross-Model Data Integration in BIM: A Lightweight and Scalable Schedule-Level Workflow

Tianjiao Zhao [1,*] and Ri Na [2]

[1] Department of Construction Management, College of Engineering & Technology, East Carolina University, Greenville, NC 27858, USA
[2] Department of Civil, Construction, and Environmental Engineering, College of Engineering, University of Delaware, Newark, DE 19716, USA; nari@udel.edu
* Correspondence: zhaot22@ecu.edu

Abstract

Despite the widespread adoption of BIM, information exchange across disciplines remains hindered by heterogeneous structures at the tabular data level, particularly when integrating data across multiple discipline-specific models. Manual mapping, rigid templates, or one-off programming scripts are labor-intensive and difficult to scale, limiting automated querying, cross-model aggregation, and schedule-level analytics. This study proposes a lightweight, workflow-driven approach for semantic normalization and cross-model integration of BIM schedule data, with optional script-supported workflow configuration used only to assist the configuration of deterministic, rule-guided mapping logic, rather than serving as a core analytical method. By introducing a customizable subcategory layer, the workflow enables fine-grained semantic alignment and efficient normalization across diverse schedule datasets, implemented through lightweight Python scripting and rule-guided semantic matching used solely as a supporting mechanism for deterministic field mapping. Using structural, architectural, and HVAC models, we demonstrate a stepwise process including data cleaning, hierarchical classification, consistency checking, batch analytics, and automated computation of cross-model metrics such as opening-to-wall ratios. Sample-based validation confirms the workflow's reliability, achieving semantic mapping agreement rates above 95% and reducing manual processing time by more than 85%. The workflow is readily extensible to other disciplines and modeling conventions, supporting high-throughput data integration for tasks such as design coordination, semantic alignment, RFI reduction, accelerated design reviews, and data-driven decision making. Overall, rather than introducing a new algorithm, the contribution of this work lies in formalizing a reusable, schedule-level workflow abstraction that enables consistent semantic alignment and automated cross-model aggregation without relying on rigid ontologies or training-intensive learning-based models. Any optional tooling used during workflow configuration is auxiliary and does not constitute a standalone learning-based method requiring model training or performance benchmarking. This provides a reusable methodological foundation for scalable, schedule-level BIM data integration and cross-model analytics.

Keywords: Building Information Modeling (BIM); semantic classification; cross-model data integration; schedule-level workflow automation; workflow automation; schedule-level data processing

Check for updates

Academic Editors: Filiberto Chiabrando and Yasser Mohamed

Received: 24 November 2025
Revised: 26 February 2026
Accepted: 23 March 2026
Published: 28 March 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

With the growing adoption of Building Information Modeling (BIM) in the construction industry, there is an increasing need for semantic normalization, batch classification, and—critically—cross-model data integration to support advanced analytics and collaborative workflows. In practice, the way element attributes, parameter definitions, and data fields are structured in BIM models often varies substantially—not only between disciplines such as structural, architectural, and HVAC, but also among different project teams and stakeholders. In this study, the term "schedule" refers to model-derived tabular representations of BIM data (e.g., element attributes and quantities exported from BIM authoring tools), rather than construction time schedules or project scheduling artifacts. Taking Revit (Autodesk, Inc., San Francisco, CA, USA) as an example, this diversity in schedule data structures and semantic conventions creates persistent challenges for data interoperability, schedule-based analytics, and the practical application of BIM data across project phases, frequently resulting in fragmented data silos that limit the full potential of digital construction workflows. In practice, these inconsistencies often manifest as coordination delays and Requests for Information (RFIs) during design and review cycles, where teams must repeatedly resolve ambiguous or mismatched model semantics.

While high-level categories (e.g., Revit's "Category") are formally standardized [1,2], deeper semantic layers such as family and subcategory remain subject to project-specific conventions and modeling habits. The lack of an explicit, standardized subcategory layer often leads to semantic ambiguity, impeding cross-disciplinary analysis and the automation of schedule-driven processes. As projects become more federated and complex, relying solely on a category–family–type structure often fails to provide the semantic granularity required for automation, cross-disciplinary reconciliation, and large-scale analytics. As a result, introducing a flexible, well-defined subcategory layer between category and family can potentially bring analytic clarity and improved data quality, and enable robust semantic mapping and information normalization—critical for meaningful data integration and intelligent querying across large, multidisciplinary BIM environments.

However, existing solutions are often limited. Manual or rule-based classification is often labor-intensive, brittle, and project-specific, while existing learning-based and text-centric approaches primarily address text parsing rather than schedule-level semantic reconciliation or cross-model automation. In practice, these approaches tend to remain at the level of simple text annotation or parameter extraction, lacking the robust, scalable pipelines needed for batch schedule classification and semantic normalization across heterogeneous BIM datasets. Much prior work relies on fixed dictionaries or static mappings, even though real projects require workflows that adapt to evolving classification hierarchies and analytical needs. As a result, there remains a significant gap between current text- or element-level classification tools and the requirements of high-throughput, repeatable BIM data integration in contemporary construction practice.

To address these gaps and to mitigate coordination inefficiencies that stem from inconsistent model semantics, this study proposes and empirically validates a generalizable, workflow-driven methodology for hierarchical semantic classification and schedule standardization in Revit-based BIM models. The approach is intended not as a rigid mapping, but rather as a flexible and adaptable process—supported by lightweight, script-based automation—that practitioners can readily customize or extend for their own models, semantic hierarchies, and integration requirements. This significantly reduces manual coding and repetitive effort compared to traditional case-by-case programming or handcrafted rule sets, while ensuring the process remains transparent, reproducible, and easy to maintain. Semantic matching and mapping logic are implemented using deterministic,

rule-guided techniques (e.g., keyword matching and string normalization), rather than trained machine learning classification models.

Building on these methodological foundations, we further illustrate how semantic normalization and standardized schedule processing enable robust cross-model data integration. By aligning and aggregating data across structural, architectural, and HVAC models, the workflow supports batch queries and analytics that were previously labor-intensive or infeasible—such as automated calculation of opening-to-wall ratios, multi-model statistics, and unified data visualizations. As such, this paper addresses the problem of cross-model schedule integration in BIM, focusing specifically on field alignment across Revit-based schedules from different models (e.g., architectural vs. structural vs. MEP). Cross-model integration in this study is achieved through a two-stage process: discipline-specific schedule normalization followed by automated cross-model aggregation. While broader interoperability challenges involving IFC or other native formats are acknowledged, they are outside the scope of this study.

The primary research objectives addressed by this study are as follows:

1. We formalize a reusable, workflow-level methodological framework for schedule-based semantic normalization and cross-model integration, rather than proposing a new classification algorithm.
2. We conduct a systematic comparison of classification workflows and engineering adaptability for structural, architectural, and HVAC models, highlighting key use cases and boundary conditions for both standardization and automation; These boundary conditions are defined at the schedule abstraction level and include the availability of tabular schedule exports, the presence of minimally interpretable categorical or parametric fields, and manageable levels of naming heterogeneity and schedule field completeness. These conditions define the practical limits within which the proposed workflow can be transferred across tools, projects, and modeling conventions.
3. Most importantly, we quantitatively and qualitatively validate the feasibility and added value of cross-model data integration, including automated computation of cross-disciplinary metrics such as opening-to-wall ratios.

Using the publicly available "Snowdon Towers" Revit sample model [3] as a benchmark, we implement and validate a complete workflow encompassing semantic mapping, data standardization, cross-model schedule integration, and multi-dimensional analytics. The use of a publicly available reference model allows reproducibility and controlled evaluation of the proposed workflow, while acknowledging that real-world deployments may involve greater variability. This is a mixed-use project located in the downtown of a typical small U.S. city, combining commercial, residential, and retail elements. Although demonstrated on the Snowdon Towers sample project, the workflow is model-agnostic; all scripts operate purely on schedule data and thus generalize readily to models following other naming conventions, LOD standards, and classification practices. From a transferability perspective, the applicability of the proposed workflow depends primarily on schedule-level characteristics rather than on specific model geometry. Key influencing factors include the degree of naming heterogeneity, the proportion of non-standard family labels, and the completeness of schedule fields. These factors can be inspected prior to deployment to determine whether the workflow can be applied directly or requires minor customization, thereby supporting informed generalization beyond the case study. The results underscore the practical significance and future potential of automated, workflow-driven BIM data processing for both engineering practice and research in digital construction. By improving data consistency and reducing coordination-related RFIs, the proposed workflow provides tangible value for real-world project teams while establishing a methodological foundation for scalable, workflow-enabled BIM analytics.

While the workflow is demonstrated using Revit-based schedules, the proposed process operates entirely at the schedule abstraction level (CSV-based tabular data), making it inherently extensible to IFC-based or other BIM platforms once equivalent schedule exports are available. Revit is therefore used as a testbed rather than as a methodological dependency.

A detailed investigation of native-to-IFC parameter mapping is acknowledged as an important complementary direction, but is considered outside the scope of this study, which focuses on lightweight, post-export semantic normalization rather than schema-level interoperability.

2. Literature Review

2.1. The Challenge of Heterogeneous BIM Schedules

Although BIM is now widely adopted across the architecture, engineering, and construction (AEC) industry, its schedule data remains heterogeneous, discipline-dependent, and difficult to reconcile. Major models, such as structural, architectural, and MEP models, frequently employ different naming conventions, levels of detail, and schema structures, leading to persistent issues of interoperability, duplicated information, and fragmented data silos. This structural heterogeneity creates major obstacles for cross-model aggregation, automated querying, and downstream analytics—particularly in large, federated projects where multiple consultants contribute data with inconsistent semantics.

Three related issues plague schedule integration in real-world projects: manual reconciliation, field misalignment, and version drift. Manual reconciliation is often necessary to reconcile equivalent fields (e.g., "Structural Material" in a structural model and "Material" in an architectural model) and resolve semantic conflicts. This process is found to be error-prone and time-consuming even in early cross-system heterogeneous integrations. Field misalignment inherits schedules with different data structures for equivalent concepts (e.g., different parameter names for wall heights), a condition Xue et al. encountered in IFC-based schedule management [4]. This necessitates transformations that are rarely reusable in other projects. Version drift adds to these woes when updates to discipline-specific variants render previously established mappings invalid, making integrated analytics such as quantity takeoff or work in progress tracking breakdown [5]. Matarneh et al. [6] observed that these pain points are acute in facility management handovers where schedule data must be reconciled between BIM and external systems (e.g., ERP). While prior work in cross-network fusion [7] and level-aware transformations [8] provides theoretical solutions, they lack lightweight, configurable implementations that scale to schedule-level cross-discipline integration in multi-discipline BIM work environments.

The Industry Foundation Classes (IFC) schema specifies the open BIM data exchange standard. However, native BIM schedule conversion (e.g., schedules in Revit) to IFC and IFC to native BIM schedules often generates interoperability problems. Golabchi and Kamat [9] examined IFC in practice and identified critical limitations in modeling non-geometric data, such as Revit schedule relationships and schedule attributes. Afsari et al. furthered the argument that cloud-based collaboration increases these limitations due to inadequate coverage of IFC's complete schema in commercial BIM applications [10]. Attribute data transfer between various BIM programs based on IFC proved to be erroneous, with data found to be lost for calculated values and custom parameters in data comparison made by Lai and Deng [11]. Steel et al. [12] illustrated these drawbacks to be caused by the "impedance mismatch" between proprietary data models and IFC's generic format, with manual post-processing correction fixes lost and mismatched information. As such, a combination of schedules across different models remains error-prone, especially when

combining architectural, structure, and MEP models with varying detail and diverging naming conventions.

These practical challenges have motivated three major technical approaches: (1) rule-based and template-based classification systems; (2) learning-based classification models and lightweight scripting for batch schedule processing; and (3) ontology or knowledge-graph frameworks. Each provides partial solutions but faces well-documented limitations for schedule-level integration.

2.2. Rule-Based and Template-Driven Classification

Traditional BIM classification relies heavily on predefined rules, templates, or schema mappings. Classification schemes such as Uniclass and OmniClass provide standardized taxonomies to categorize BIM data, including BIM schedules. These schemes are designed to facilitate common naming, grouping, and coding of elements in buildings throughout projects and disciplines in a swift manner. Tang et al. [4] demonstrated, for instance, that automatic BIM object classification with Uniclass applied to IFC data held potential to reduce manual labor in data categorization. Similarly, OmniClass was used in various constructions such as coding urban underground space facilities [13] and asphalt pavement diseases [13]. Nonetheless, Pupeikis et al. [5] discovered significant variances of classes in different systems, rendering cross-project and cross-regional combination of data more complicated than justified. Despite these standards having theoretical foundations for the convergence of semantics, implementation in practice often needs customizations and fragmented mapping that initiates interoperability breakdowns.

Common data exchange structures (e.g., COBie (Construction Operations Building Information Exchange, NBIMS-US V3)) or scripting environments (e.g., Dynamo (Autodesk Dynamo, V2)) allow practitioners to enforce naming conventions or link Revit entities to external databases. For instance, Kuzminykh et al. [14] utilized Dynamo scripts to link Revit model entities with cost databases via Excel, independently classifying for 4D/5D BIM applications.

Although effective for well-bounded use cases, these approaches are fragile: even minor variations in family naming or parameter structures require substantial manual rework. Their lack of adaptability makes them difficult to scale across projects or reuse in highly complex, multi-disciplinary environments, where taxonomies evolve.

2.3. Learning-Based BIM Classification: Advancements and Gaps

Recent efforts apply machine learning, deep learning, and language models to automate BIM element classification. Multi-modal approaches combining geometric and textual features have achieved high accuracy at the element level, including results above 98% for fine-grained component classification [15] and strong ensemble-learning performance for MEP and architectural objects [16]. Graph neural networks (GNNs) further improve modeling of spatial or relational context [17], while invariant signature-based methods support the classification of AEC objects [18], and association rule mining supports automated coding of prefabricated parts [19,20].

Yet, these learning-based models face practical constraints: (1) they require large labeled datasets that are rarely available for project-specific families [21]; (2) they do not generalize well to unseen taxonomies or non-standardized modeling practices [22]; (3) high-level architectures like MMDL or GNNs are too computationally intensive to rule out real-time application in resource-limited environments; and (4) critically, they focus on element-level classification rather than schedule-level semantic alignment.

Thus, most existing element-level, learning-based tools struggle with tasks central to cross-model schedule integration—field normalization, resolving naming conflicts, and supporting flexible, project-adaptive workflows.

2.4. Text-Centric Approaches and Lightweight Scripting for Batch Schedule Processing

Text-centric approaches including natural language processing (NLP) tools have been applied to specifications, regulatory reports, and issue logs. For instance, Pan et al. [23] compared mainstream language models on managing general databases. Xie et al. [24] utilized NLP to relate real-world facilities to BIM objects through the extraction of maintenance requirements from text-based specifications. While this enabled automated labeling, it was not able to cope with the interpretation of context-dependent terms, the common shortcoming of text-centric methods when they are used beyond the context they were trained on.

Other researchers have utilized NLP for regulatory checking for the purposes of compliance. Zhou et al. [25] and Shen et al. [26] created ontology-based systems for the enforcement of safety rules based on BIM and NLP. These uses demonstrated wide potential for structured workflows for the purposes of compliance, but they required extensive predefined taxonomies, limiting adaptability to diverse project needs. Additionally, Nabavi et al. [27] suggested a BIM-based query engine with the utilization of NLP for automated information querying. Nevertheless, there were semantic ambiguities with schedule-specific parameters for their system, such as differences between synonyms such as "Unconnected Height" and "Height." Thus, while text-centric systems excel at document parsing, they provide limited support for reconciling discipline-specific parameter semantics or aligning heterogeneous schedule fields.

Related efforts have also examined schedule-level BIM data structuring and interoperability challenges. For example, recent work published in ITcon investigated the organization and transformation of BIM schedule data to support downstream interoperability and analysis [28]. While such studies highlight the importance of structured schedule data, they primarily focus on specific transformation strategies or standardization mechanisms. In contrast, the present study formalizes a reusable, workflow-driven abstraction for cross-model semantic integration that emphasizes adaptability and repeatability over fixed transformation rules.

In addition to NLP, lightweight scripting libraries such as Dynamo and the Revit API are widely used for BIM automation. However, their scopes remain biased toward geometry rather than semantic normalization. Research by Wang et al. [28] is typical of this trend, where space reasoning—clash detection or multiscale geometry queries—dominates scripting at the expense of schedule field normalization complexity. The lack of built-in support for reconciling similar but differently named attributes (e.g., "Structural Material" vs. "Material") highlights the limitations of current scripting tools. Further, when scripting is embedded in more sophisticated pipelines, as with the embodied GHG emission computation performed by Forth et al. [29], rigid definitions of schemas foreclose on generalizability. Their NLP-facilitated workflow, although successful within the constrained ontology, cannot easily generalize to new or evolving classification schemes. In this way, schedule semantics—particularly when traversing disciplines—still rely heavily on manual interpretation and intervention.

In contrast to method-specific or algorithm-centric solutions, this study positions NLP and AI tools as auxiliary mechanisms to support workflow configuration and semantic mapping, rather than as the core methodological contribution. The primary focus is on establishing a reproducible, schedule-level integration workflow that remains interpretable, adaptable, and lightweight for real-world BIM practices.

2.5. Ontology-Based and Semantic Web Models

Semantic web technologies offer theoretically robust means to achieve semantic alignment through formal domain ontologies such as ifcOWL, BOT, or bcfOWL. IfcOWL ontology, founded on the Industry Foundation Classes (IFC) schema, enables building models to be expressed in the Web Ontology Language (OWL) and to facilitate data integration and reasoning across various domain applications [28,30]. Similarly, ontologies such as the Building Topology Ontology (BOT) and bcfOWL (for BIM collaboration) provide standardized representations for conveying spatial and non-spatial information in buildings [31]. Such ontologies lie at the base of sophisticated applications such as automated compliance checking [32], life cycle assessment (LCA) [33,34], and digital twin integration [35,36]. With the formalization of domain knowledge, ontologies in principle would enable BIM data to be interpreted uniformly across software systems and stakeholders in projects.

However, ontology-based approaches face three practical barriers: (1) high development and maintenance overhead requiring specialized expertise [37,38]; (2) poor adaptability to project-specific taxonomies or evolving parameter structures [34,38] (e.g., road infrastructure [34], indoor environmental quality [39], or building services [40]; and (3) limited applicability at schedule granularity, where semantics diverge widely from standardized ontologies [39,40]. Consequently, while ontologies are powerful for conceptual modeling, they are rarely adopted for day-to-day BIM schedule reconciliation.

2.6. Research Gaps

Beyond algorithmic classification or ontology-driven interoperability, there exists a largely underexplored methodological gap at the schedule abstraction level of BIM data. In practice, many coordination, quantity takeoff, and cross-model analytics tasks operate not on raw geometric entities, but on exported schedule tables that are manually cleaned, reconciled, and restructured in an ad hoc manner. Despite its prevalence in engineering workflows, this schedule-level integration problem has received limited systematic methodological treatment in existing BIM research.

Across these methodological families, a clear gap persists: there is no lightweight, customizable, and workflow-oriented approach that enables multi-level semantic classification and cross-model schedule integration without relying on rigid templates, extensive ontologies, or training-intensive learning-based models.

Specifically: (1) rule-based methods lack scalability, (2) learning-based models lack adaptability and do not address schedule-level fusion, (3) lexical or string-based matching approaches struggle with naming conflicts or aligning heterogeneous schedule fields, and (4) ontologies lack practicality for project-specific semantics.

This study addresses these gaps by introducing a script-driven, workflow-based, subcategory-based semantic normalization workflow that: (1) supports schedule-level cross-model data fusion, (2) scales across disciplines, (3) can be customized without heavy ontologies, (4) reduces manual effort, and (5) establishes a reusable foundation for downstream analytics and automation.

3. Materials and Methods

In the remainder of this paper, the term "schedule data" refers to tabular BIM schedule exports independent of authoring software. Tool-specific terms (e.g., Revit schedules or Revit categories) are used only when describing the case study implementation.

Rather than proposing a new algorithm, the proposed methodology is formulated as a schedule-level semantic integration pipeline that abstracts common operations—data normalization, hierarchical classification, consistency checking, and aggregation—into a reusable workflow applicable across BIM disciplines. To avoid redundancy while pre-

serving methodological clarity, the structural model is used as a representative case to illustrate each major processing stage and core operation in detail, while equivalent workflows and discipline-specific results for the architectural and HVAC models are provided in the Appendices A–C as supporting material. The method enriches existing schedule hierarchies with a customizable subcategory level, enabling granular semantic alignment and extensible query workflows without modifying the original BIM models.

In this study, any references to AI or NLP refer solely to optional external tools used during the initial configuration of mapping rules or script authoring. These tools are not embedded in the executable workflow and do not participate in data processing, classification, or analysis. All operational steps of the proposed methodology are deterministic, rule-based, and fully reproducible.

In addition, semantic mapping is implemented only as a lightweight, rule-guided matching process rather than as a trained machine learning or embedding-based model. Specifically, keyword matching combined with string normalization techniques (e.g., lowercasing, delimiter removal, and abbreviation expansion) is used to identify candidate mappings between schedule fields and subcategory labels. Ambiguous or low-confidence matches are flagged for manual review during the initial configuration and taxonomy validation stage, following a human-in-the-loop strategy. Once validated, the mapping configuration can be reused for subsequent automated runs without further human intervention.

Crucially, this workflow is not a static dictionary or rigid template, but rather an adaptable pipeline designed for real engineering environments with evolving modeling conventions and information needs. Introducing a "subcategory" layer between standard category and family definitions provides the semantic granularity required for robust schedule normalization and downstream analytics. Cross-model integration in this study is achieved through a conceptually two-stage process—discipline-specific schedule normalization followed by automated cross-model aggregation—implemented through a structured sequence of schedule-level operations. The functional stages, inputs, outputs, and abstraction roles of this pipeline are formally defined in Table 1, and their relationships and information flow are illustrated in Figure 1, which provides a platform-agnostic specification of the proposed schedule-level integration workflow. All steps are deterministic and rule-based; optional tooling may assist initial configuration, but does not participate in execution.

Table 1. Formal definition of the schedule-level integration pipeline.

Stage	Input	Operation	Output	Role in Abstraction
Data Collection	Raw schedules	Lossless export	CSV tables	Tool-agnostic input
Field Normalization	CSV tables	Header unification	Standardized tables	Semantic alignment
Hierarchical Classification	Tables + mapping	Category–Subcategory assignment	Classified tables	Semantic structure
Consistency Checking	Classified tables	Conflict detection	Validated mappings	Reliability
Cross-Model Aggregation	Normalized tables	Batch aggregation	Unified analytics	Generalizability

Sections 3.1–3.3 describe a tool-agnostic, schedule-level workflow operating on CSV-based tabular data, while Section 3.4 details the Revit-based case study implementation used for demonstration and validation.

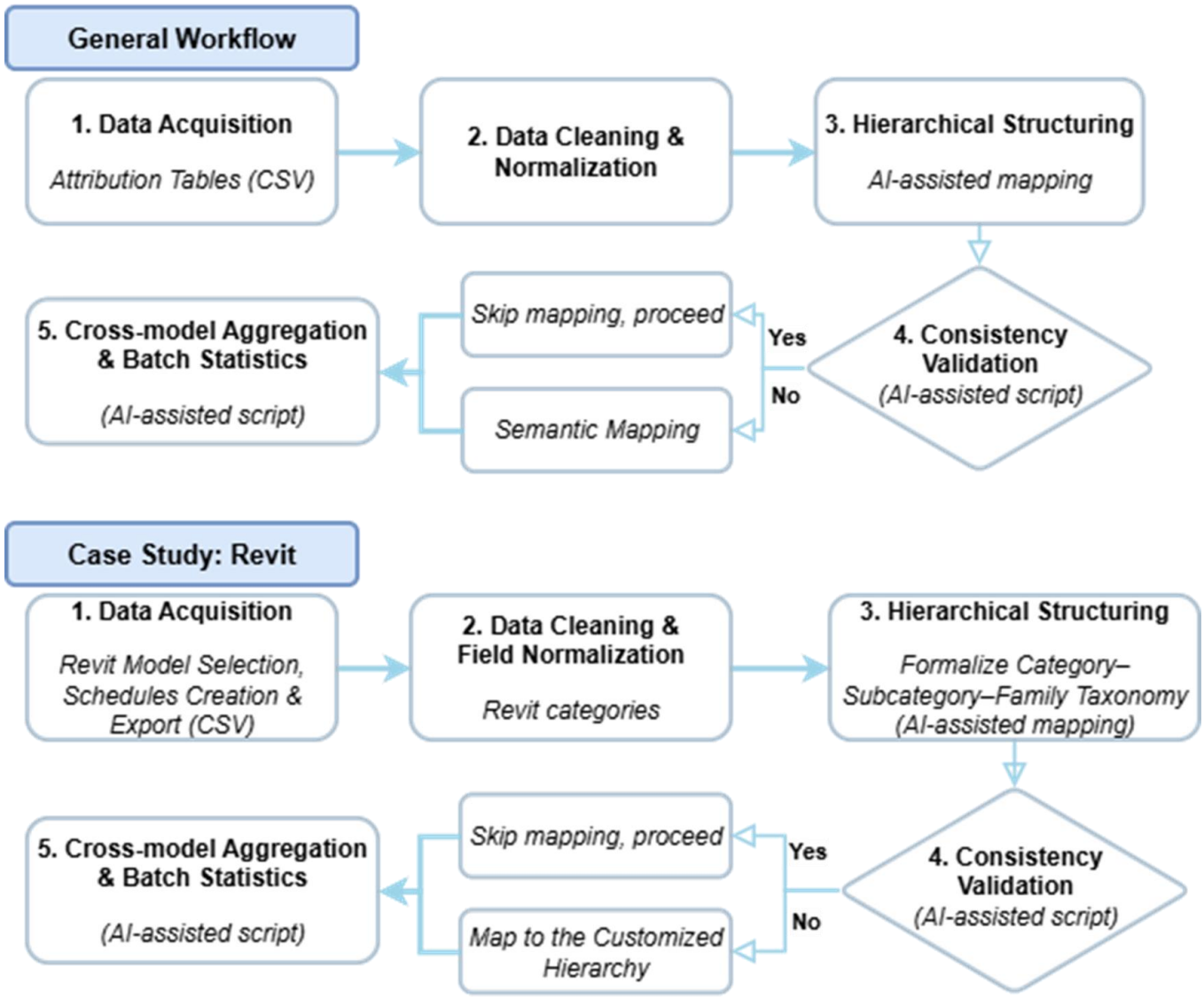


Figure 1. Schedule-level workflow for semantic classification and cross-model BIM schedule integration.

3.1. Schedule-Level Data Collection and Preprocessing (Tool-Agnostic)

This stage corresponds to the input normalization component of the pipeline, operating on tabular BIM schedule data exported in CSV format to establish a consistent analytical baseline independent of the source modeling tool. Rather than documenting tool-specific extraction procedures, this stage is framed to highlight the semantic variability and preprocessing challenges that motivate a schedule-level workflow abstraction. The focus of this stage is comprehensive, lossless data extraction from each discipline, providing a robust foundation for subsequent semantic normalization.

- **Model Selection and BIM Schedule Export**: The study utilizes the official "Snowdon Towers" Revit sample project, selecting representative structural, architectural, and HVAC models (see Figure 2 for representative views of each model). For each discipline, representative schedules are selected to capture the diversity of element families and parameter structures typically encountered in practice. For each model, the primary schedules span multiple element groups (e.g., framing-related structural components and enclosure-related architectural elements). To maximize data utility, all available fields are exported, including core attributes such as Family, Type, Level, Material, Area, and Volume (see Figure 3 for representative examples). Figure 3 presents

the schedules in their native Revit interface to illustrate the structure and variability of raw schedule data prior to semantic normalization. If a standard field (e.g., Material) is absent, analogous fields (such as Structural Material or Frame Material) are included. This inclusive export strategy is intended to expose semantic inconsistencies rather than to prescribe a fixed data schema. Additional discipline-specific fields are retained wherever available, further enhancing the extensibility and downstream applicability of the dataset. Such comprehensive export ensures that downstream semantic mapping and analytics are not constrained by data loss or field omission—a prerequisite for robust, reusable pipelines.

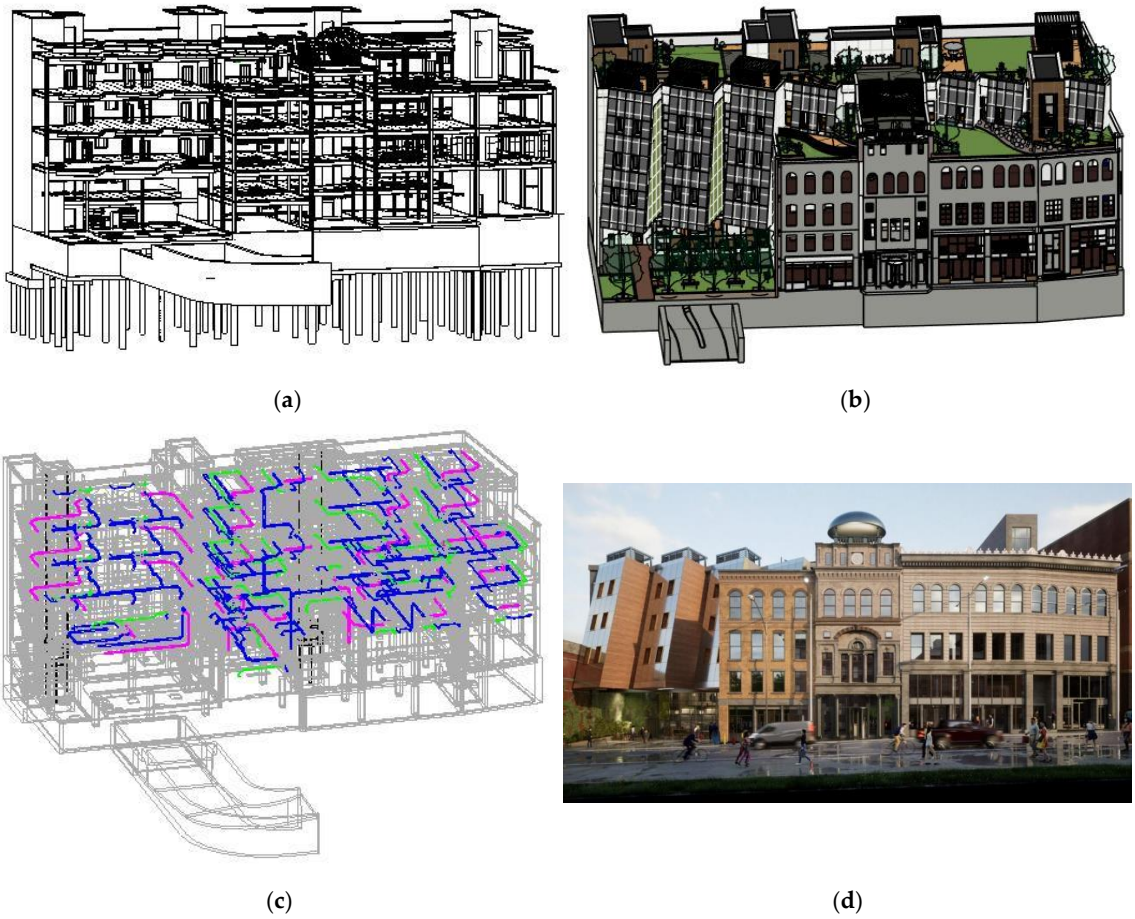
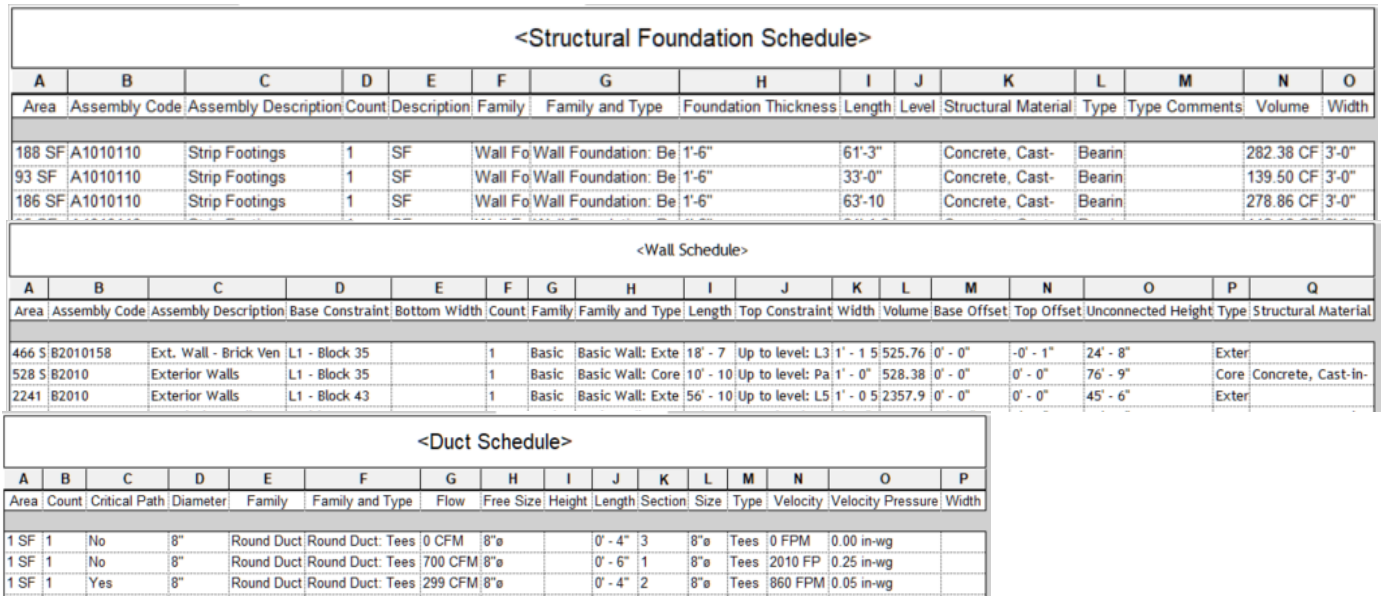


Figure 2. Representative view of the "Snowdon Towers" sample project in Revit, illustrating disciplinary differences in modeling conventions: (**a**) Structural model, (**b**) Architectural model, (**c**) HVAC model, (**d**) Actual building appearance.

- **Data Cleaning**: Raw BIM schedule data are cleaned and preprocessed to remove blanks and duplicates, resulting in comprehensive, discipline-specific datasets. Here, duplicates are defined in semantic rather than syntactic terms, referring to semantically duplicated entries arising from multi-category schedules, linked elements, or repeated family instances across views, rather than identical element IDs exported by Revit. This distinction underscores the need for semantic normalization beyond simple record deduplication.
- **Field Standardization**: Key analytical fields (e.g., Count, Area, Volume, Material) are unified across schedules, with standardized headers and selective field merging (such as "Structural Material/Material" or "Unconnected Height/Height"). Field consolidation is only performed when semantic overlap is clear and no data conflict arises, ensuring data integrity for subsequent cross-table analysis. Rather than en-

forcing rigid schemas, field unification is deliberately conservative and is validated through both automated heuristics and expert review to prevent unintended data conflation. The workflow is fully automated at the data processing and analytics stage, while expert review is only required during initial field standardization or taxonomy definition, and is not needed for repeated runs. This balance between automation and minimal expert intervention reflects the study's focus on repeatable, practice-oriented workflows rather than one-off data preparation. This process combines automated field detection with human judgment to ensure that merged fields retain consistent business meaning and analytical value across models.



<Structural Foundation Schedule>

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Area	Assembly Code	Assembly Description	Count	Description	Family	Family and Type	Foundation Thickness	Length	Level	Structural Material	Type	Type Comments	Volume	Width
188 SF	A1010110	Strip Footings	1	SF	Wall Fo	Wall Foundation: Be	1'-6"	61'-3"		Concrete, Cast-	Bearin		282.38 CF	3'-0"
93 SF	A1010110	Strip Footings	1	SF	Wall Fo	Wall Foundation: Be	1'-6"	33'-0"		Concrete, Cast-	Bearin		139.50 CF	3'-0"
186 SF	A1010110	Strip Footings	1	SF	Wall Fo	Wall Foundation: Be	1'-6"	63'-10		Concrete, Cast-	Bearin		278.86 CF	3'-0"

<Wall Schedule>

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
Area	Assembly Code	Assembly Description	Base Constraint	Bottom Width	Count	Family	Family and Type	Length	Top Constraint	Width	Volume	Base Offset	Top Offset	Unconnected Height	Type	Structural Material
466 S	B2010158	Ext. Wall - Brick Ven	L1 - Block 35		1	Basic	Basic Wall: Exte	18' - 7	Up to level: L3	1' - 1 5	525.76	0' - 0"	-0' - 1"	24' - 8"	Exter	
528 S	B2010	Exterior Walls	L1 - Block 35		1	Basic	Basic Wall: Core	10' - 10	Up to level: Pa	1' - 0"	528.38	0' - 0"	0' - 0"	76' - 9"	Core	Concrete, Cast-in-
2241	B2010	Exterior Walls	L1 - Block 43		1	Basic	Basic Wall: Exte	56' - 10	Up to level: L5	1' - 0 5	2357.9	0' - 0"	0' - 0"	45' - 6"	Exter	

<Duct Schedule>

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
Area	Count	Critical Path	Diameter	Family	Family and Type	Flow	Free Size	Height	Length	Section	Size	Type	Velocity	Velocity Pressure	Width
1 SF	1	No	8"	Round Duct	Round Duct: Tees	0 CFM	8"ø		0' - 4"	3	8"ø	Tees	0 FPM	0.00 in-wg	
1 SF	1	No	8"	Round Duct	Round Duct: Tees	700 CFM	8"ø		0' - 6"	1	8"ø	Tees	2010 FP	0.25 in-wg	
1 SF	1	Yes	8"	Round Duct	Round Duct: Tees	299 CFM	8"ø		0' - 4"	2	8"ø	Tees	860 FPM	0.05 in-wg	

Figure 3. Representative view of the model schedules in Revit illustrating heterogeneity in field structure and naming conventions prior to semantic normalization. The original Revit schedule interface is shown to reflect the native structure and formatting of schedule data prior to export and normalization. Note: Due to the large number of schedule fields, the figure is displayed at reduced scale to fit within the page layout. Column widths are compressed to preserve header visibility, and some cell contents may appear truncated in the printed view but remain readable when magnified. The schedules are exported using a consistent set of available fields across models, including both shared and model-specific attributes.

3.2. Multi-Level Semantic Classification and Normalization

This stage corresponds to the hierarchical semantic mapping component of the pipeline, where discipline-specific schedule data are aligned to a configurable Category–Subcategory–Family structure to achieve semantic consistency across heterogeneous models.

- **Hierarchical Classification Framework**: In this workflow, each discipline's schedules are mapped to a customizable "Category–Subcategory–Family" structure. While the Category follows Revit's official definitions, the Subcategory is defined based on industry practices but always tailored to the project context and the actual data available. This framework is deliberately designed for flexibility—subcategories can be refined or expanded to support project-specific semantics, internal company standards, or evolving industry requirements. In practical implementation, users define the subcategory layer through a lightweight configuration table or a script-based mapping file (e.g., in CSV or JSON format), which can be iteratively refined without modifying the native BIM model. The configuration table lists Families, assigned Subcategories, and optional keywords for NLP-assisted matching. The configuration can be manually edited in Excel or automatically generated via Python, enabling rapid iteration

without modifying the native Revit model. This design choice ensures accessibility for non-programmers while maintaining full script-level flexibility for advanced users. While anchored in Revit's official taxonomy, the subcategory layer is tailored to fit specific project semantics and can be updated as standards evolve—supporting both external interoperability and internal process control. Type is assumed to be uniquely determined by Family and does not require separate mapping. The proposed subcategory taxonomy is informed conceptually by established classification standards such as OmniClass, while remaining intentionally lightweight and project-adaptable rather than prescriptive. Subcategories are designed to be mutually exclusive, sufficiently granular for schedule-level analytics, and stable across project phases. To prevent semantic drift as family definitions evolve, mapping files are versioned and maintained independently from scripts, allowing controlled updates without disrupting downstream analyses.

- **Consistency Checking**: Python scripts are used to systematically check the mapping between Family and Subcategory. If a one-to-one correspondence is detected, the mapping is confirmed; otherwise, batch mapping and additional Subcategory fields are introduced. The consistency-checking step automatically flags any Family–Subcategory pairs that appear in multiple semantic groups, allowing users to resolve conflicts through a simple update to the mapping table rather than modifying the BIM model itself. This enables batch classification across models with varying granularity, ensuring adaptability to different modeling standards. Both automated script generation and manual adjustment are supported. Automated scripts, which may be automatically generated or iteratively refined during configuration, rapidly scan for many-to-many or ambiguous mappings, flagging edge cases for targeted review and drastically reducing manual curation workload.
- **Mapping Implementation**: Python scripts (using pandas and related libraries) automate subcategory assignment in batch mode, enabling efficient and reproducible classification even for large datasets. All subsequent analysis and visualization are based on these standardized categories and subcategories, supporting traceable and reusable classification outputs.
- **Discipline-Specific Handling**:
    - For structural models, substantial manual or automated mapping is required due to non-standard naming conventions.
    - For architectural models, the process relies mainly on merging and mapping, with partial normalization.
    - For HVAC models, Family and Subcategory are typically aligned, requiring only verification with no further processing.

3.3. Batch Statistics, Visualization, and Cross-Model Integration (Tool-Agnostic)

This stage represents the aggregation and query component of the pipeline, in which normalized schedules are merged, analyzed, and queried to enable repeatable cross-model analytics independent of their original disciplinary boundaries. Based on the classified and standardized data, efficient batch statistics and visualization are generated for each discipline using lightweight scripting or spreadsheet-based tools. For models not requiring semantic remapping, statistics such as "Category + Family + Count" or "Type + Material + Count" can be computed directly, whereas mapped datasets support higher-level queries such as "Category + Subcategory + Count" or "Subcategory + Material + Count."

This end-to-end pipeline—from native BIM schedule export to multi-source metric computation—operates without manual intervention between data export and final analysis, ensuring both reproducibility and scalability for large-scale projects.

To further demonstrate the practical value of this automated, extensible workflow, we incorporate a cross-model aggregation scenario. Specifically, a newly generated generic model schedule of openings is directly exported from Revit as a CSV and, without any manual preprocessing or integration into previous combined structural schedules, is seamlessly merged and analyzed alongside other discipline schedules. In this study, openings are defined as doors and windows associated with wall elements, excluding curtain wall panels. Duplicate openings are filtered using unique element identifiers to avoid double counting, and openings not assigned to a level are excluded from level-based calculations. All area values are reported in square feet, consistent with BIM schedule outputs. A spot-check comparison against Revit-calculated values for selected levels confirmed consistency within acceptable tolerance, indicating that no extraction bias was introduced. This fully automated approach allows engineering queries—such as the calculation of opening-to-wall ratios, spatial distributions, or other cross-model metrics—to be performed reproducibly and at scale, without the need for bespoke, case-by-case integration scripts. This end-to-end process exemplifies the full automation of cross-model data integration, enabling key engineering metrics such as opening-to-wall ratios and spatial distributions to be computed efficiently and reproducibly.

The evaluation combines qualitative validation with supplementary quantitative checks—such as mapping agreement with manually validated references and reduced manual processing time—to demonstrate practical efficiency gains. Throughout this study, mapping agreement is assessed based on consistency between automated category–subcategory assignments and manually validated reference mappings.

3.4. Case Study Implementation: Revit Schedules and Execution Environment

This section describes the concrete implementation of the proposed workflow using Revit schedules as a case study and testbed.

- **Toolchain**: Revit 2024 (for schedule export), Excel 2021 (Microsoft Corporation, Redmond, WA, USA) (for initial data cleaning), Python 3.11 (with pandas, matplotlib, etc. for data processing and visualization), and Jupyter Notebook v7 (Project Jupyter, open-source) (for batch scripting and reproducibility).
- **Open Data and Code Reuse**: All automation scripts and classification dictionaries are designed for open sharing and reuse by the research and engineering community.
- To ensure semantic accuracy, randomly sampled entries from each model were manually cross-checked against the automated mappings by two independent reviewers.

4. Results

The following results evaluate one concrete instantiation of the proposed schedule-level pipeline, using a publicly available BIM model to examine how the abstraction behaves under varying degrees of semantic heterogeneity. Rather than documenting implementation steps, the results emphasize observed patterns, limitations, and analytical implications across structural, architectural, and HVAC models.

This section systematically presents the application and effectiveness of the automated classification and standardization workflow across structural, architectural, and HVAC BIM models, as well as the subsequent integration and analysis of data across these disciplines. Through key statistical tables and visualizations, we reveal not only domain-specific classification challenges, improvements in data consistency, and automation of semantic analytics, but also demonstrate the value of fully automated, cross-model data aggregation—enabling advanced engineering queries, such as the computation of opening-to-wall ratios. These findings collectively highlight how model complexity and discipline-specific conventions

shape both the feasibility and benefits of automated classification, providing critical insights for further refinement and broader deployment.

4.1. Structural Model

In this study, from the perspective of the pipeline abstraction, the structural model represents a high-heterogeneity condition, exposing the upper bounds of semantic variability that the workflow is designed to accommodate. Accordingly, results from this model are used to assess whether the workflow can normalize highly fragmented schedule semantics into a form suitable for repeatable downstream analysis, making the structural model a rigorous testbed for the proposed multi-level classification process.

The resulting classification demonstrates that the proposed Category–Subcategory–Family hierarchy is sufficient to impose semantic structure on otherwise inconsistent family naming without requiring modification of the original BIM model.

4.1.1. Semantic Normalization and Enhancement

The initial dataset contains a wide range of family names, many of which lack clear grouping or semantic structure, reflecting substantial schedule-level semantic dispersion (see Table 2). Representative element types (e.g., framing- and slab-related components) illustrate this variability; however, the primary challenge lies in inconsistent semantic patterns rather than in any specific family label.

Table 2. Structural model: Category–Family–Count table illustrating semantic fragmentation prior to normalization.

Category	Family	Count
Column	TOTAL	74
Column	Concrete-Rectangular-Column	13
Column	HSS-Hollow Structural Section-Column	1
Column	SC_Reference Column	20
Column	W Shapes-Column	40
Floor	TOTAL	27
Floor	Floor	27
Foundation	TOTAL	90
Foundation	Foundation Slab	4
Foundation	SFD_Pile_Cap_Rectangle	13
Foundation	SFD_Round	64
Foundation	Wall Foundation	9
Framing	TOTAL	942
Framing	C Shapes	6
Framing	Concrete-Rectangular Beam	35
Framing	K-Series Bar Joist-Angle Web	474
Framing	LH-Series Bar Joist	8
Framing	SF_Steel L Angle	25
Framing	W Shapes	394
Wall	TOTAL	58
Wall	Basic Wall	58
Total		1191

Applying the "Category–Subcategory–Family" three-level framework demonstrates how schedule-level semantic structure can be imposed without altering the original BIM model, enabling consistent cross-family alignment (see Table 3). Each taxonomy table presents representative entries to illustrate the level of abstraction required for cross-model semantic alignment, while complete mappings are provided in Appendix B for transparency and reproducibility. The inclusion of example types is intended to evidence mapping consistency rather than to document all possible classifications. To further demonstrate the

comprehensiveness of the approach, two example types are included, illustrating the classification from Category through Subcategory and Family down to specific Example Types.

Table 3. Representative "Category–Subcategory–Family–Example Type" taxonomy for the structural model (complete table provided in Table A1).

Category	Subcategory	Family	Example Types
Structural Column	Concrete Column	Concrete-Rectangular-Column	—
	Steel Column	W Shapes-Column, HSS	—
	Reference Column	SC_Reference Column	—
Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB18 × 24, CB24 × 51, CB24 × 24, CB24 × 30, CB24 × 39
	Steel Beam	W Shapes	—
	Bar Joist	K-Series Bar Joist, LH-Series Bar Joist	—
	Steel Angle/Channel	SF_Steel L Angle, C Shapes	—
Structural Floor/Slab	Concrete Slab	Floor	—
Structural Foundation	Foundation Slab	Foundation Slab	—
	Wall Foundation	Wall Foundation	—
	Pile Cap	SFD_Pile_Cap_Rectangle	—
	Round Foundation	SFD_Round	—
Structural Wall	Concrete Wall	Basic Wall	Concrete 10", Concrete 12", Concrete 18", Concrete 24"

The batch mapping scripts (representative excerpts shown in Appendix C.1) substantially improve classification consistency and traceability, providing a solid foundation for downstream analytics. Preliminary testing showed that the automated semantic classification greatly reduced manual mapping effort and minimized common misclassification patterns, demonstrating clear efficiency gains on large, heterogeneous datasets.

Figure 4 shows the resulting standardized Revit schedule after automated classification.

2]:

	Auto_Category	Auto_Subcategory	Family	Type	Count	Volume
0	Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB18x24	1	35.95 CF
1	Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB24 x51	2	106.06 CF142.08 CF
2	Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB24x24	16	18.28 CF43.75 CF42.78 CF35.78 CF50.37 CF41.32 ...
3	Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB24x30	14	68.19 CF76.67 CF16.67 CF68.19 CF76.67 CF68.19 ...
4	Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB24x39	2	100.59 CF113.08 CF
5	Structural Wall	Concrete Wall	Basic Wall	Concrete 10"	11	645.19 CF500.57 CF105.31 CF619.10 CF510.76 CF4...
6	Structural Wall	Concrete Wall	Basic Wall	Concrete 12"	19	709.48 CF374.54 CF756.98 CF884.08 CF443.57 CF1...
7	Structural Wall	Concrete Wall	Basic Wall	Concrete 18"	14	773.81 CF537.35 CF251.81 CF231.63 CF126.76 CF2...
8	Structural Wall	Concrete Wall	Basic Wall	Concrete 24"	14	369.75 CF2298.18 CF1020.35 CF841.87 CF120.52 C...

]:

Figure 4. Structural model: Python classification results example shown in the Jupyter Notebook environment. The original output interface is presented to illustrate the structure of automated classification results generated by the workflow.

4.1.2. Statistical Visualization

The distribution of element counts by category reveals structural modeling biases that are only observable after semantic normalization, underscoring the analytical value of schedule-level classification (see Figure 5, "Sum of Count by Category" and Figure 6, "Sum of Count by Family"). Element count statistics are not intended to directly represent physical quantities for continuous elements such as walls, slabs, or foundations. Instead, they are used to reveal semantic distribution patterns, modeling bias, and data consistency across categories and families, which are necessary prerequisites for reliable quantity takeoff, cross-model aggregation, and downstream BIM analytics. The results indicate that the structural system is dominated by the "Framing" category, particularly the "K-Series Bar Joist–Angle Web" and "W Shapes" families.

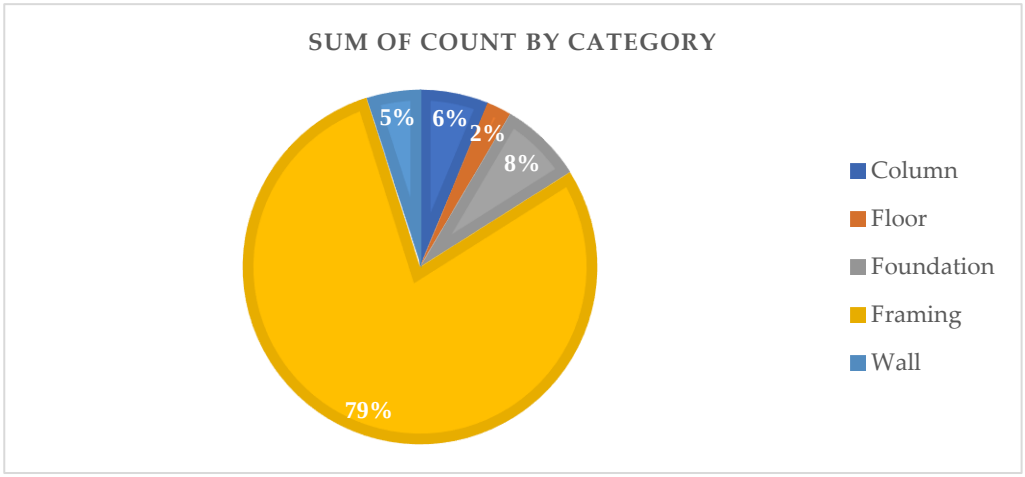


Figure 5. Structural model: Sum of count by category.

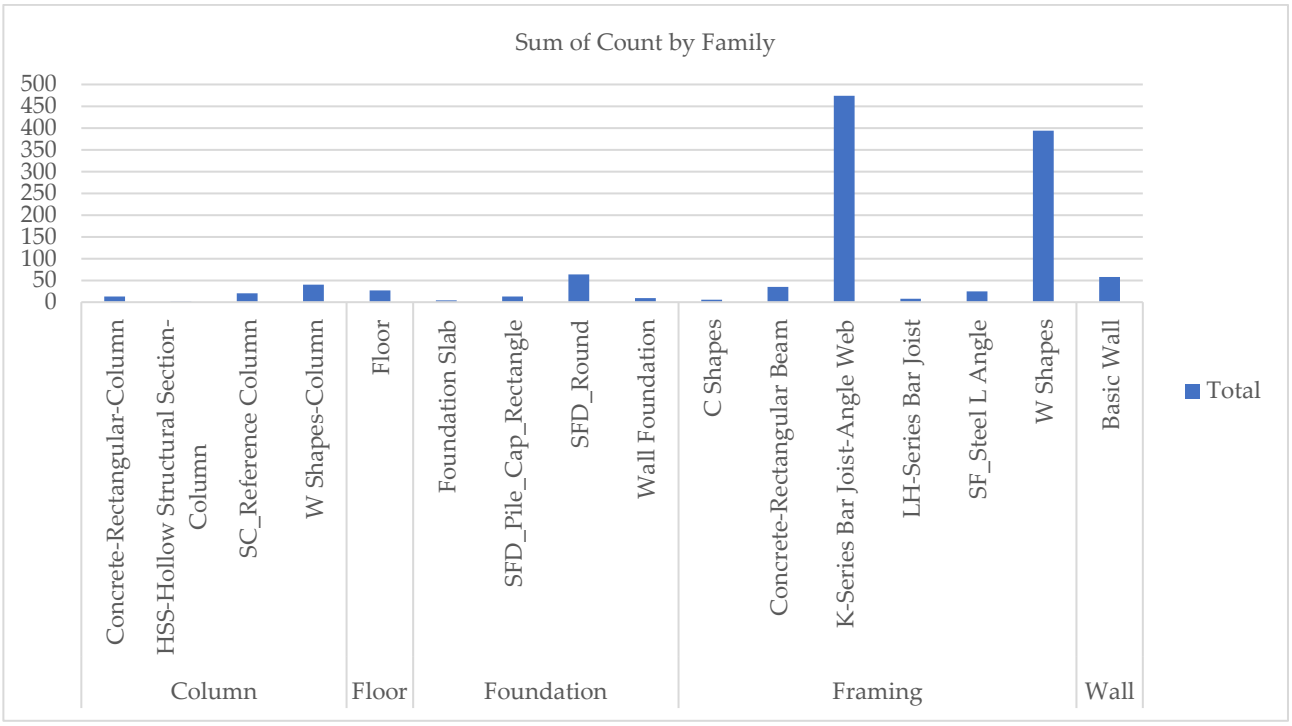


Figure 6. Structural model: Sum of count by family.

4.1.3. Automated Type Distribution Analysis

Using "Concrete-Rectangular Beam" and "Basic Wall" as examples, Python scripts are employed to automate multi-type classification, retrieval, quantitative analysis, and visualization, where type-level statistics serve to verify classification completeness and consistency rather than to replace quantity-based metrics. This demonstrates both the efficiency and scalability of the workflow, supporting applications such as structural variant analysis and standardized component recommendation. The type-wise distribution bar charts are shown in Figures 7 and 8.

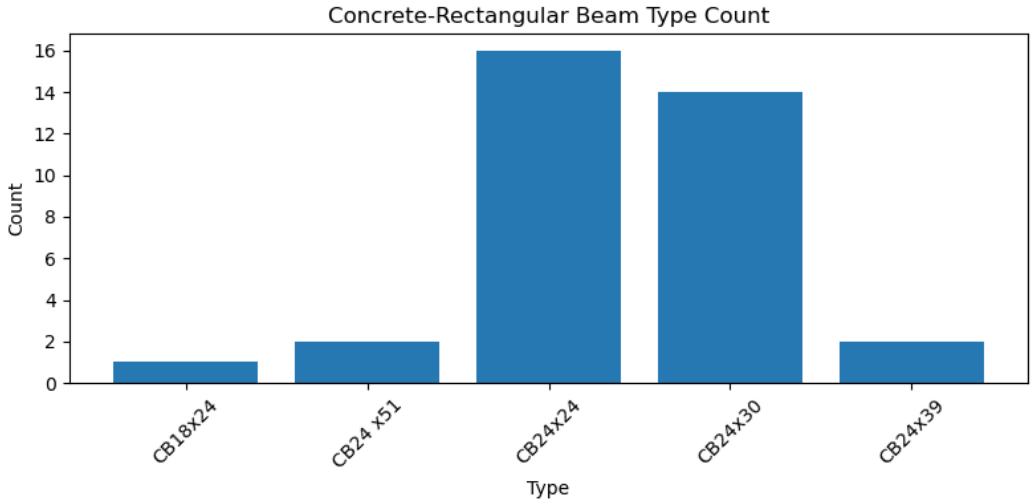


Figure 7. Count by type example: Concrete-Rectangular beam.

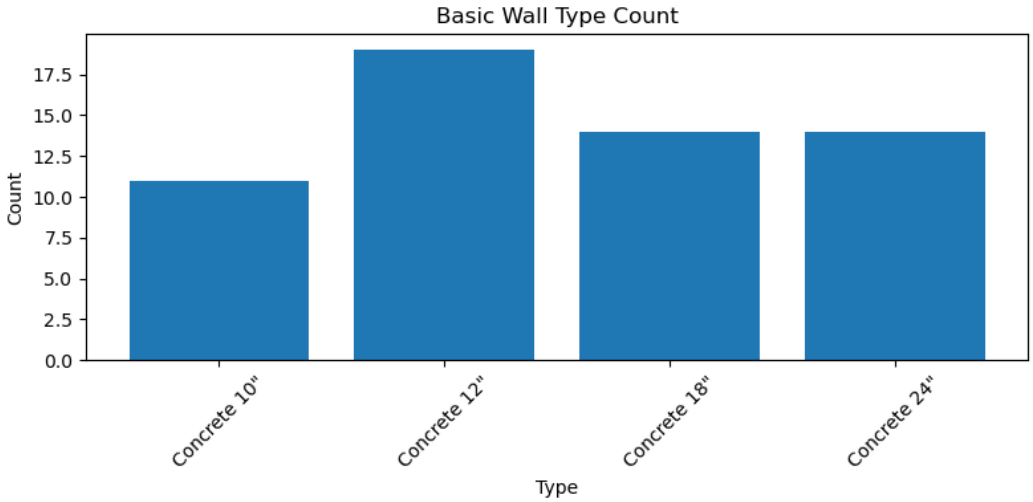


Figure 8. Count by type example: Basic wall.

4.2. Architectural Model

In contrast, the architectural model corresponds to a moderate-heterogeneity condition, where family naming is relatively standardized but semantic inconsistency arises primarily from field proliferation and parameter variation across schedules. This indicates that, in partially standardized models, workflow effectiveness depends more on field-level normalization than on family-level remapping (see Table A2). After applying the proposed workflow, consolidated and harmonized schedule attributes enable more detailed spatial and material analyses, demonstrating the workflow's ability to support cross-disciplinary data fusion in cases where semantic issues are driven by parameter heterogeneity rather than naming inconsistency.

4.2.1. Batch Classification Results

Rather than presenting step-by-step procedures, the following results summarize how different sources of semantic inconsistency are addressed through field-level normalization and batch classification.

For the architectural model, schedules from multiple categories (Walls, Doors, and Windows) were first consolidated to support cross-category analysis and unified processing. Key overlapping fields, such as "Structural Material/Material" and "Unconnected Height/Height," were standardized and merged to improve data compatibility and analytical flexibility (see Table 4). All architectural schedules were then cleaned, aggregated, and consolidated into a master table using Excel, and Category- and Family-level statistics were computed via pivot tables to establish a consistent analytical baseline.

Table 4. Architectural model: Field mapping & merging table.

Unified Field Name	Original Fields from Schedules	Explanation
Structural Material/Material	Walls: Structural Material; Doors/Windows: Material	Unified to represent the main material for all categories
Unconnected Height/Height	Walls: Unconnected Height; Doors/Windows: Height	Unified to represent the height/vertical dimension for all

To enable more granular semantic querying, a four-level classification structure—Category, Subcategory, Family, and Type—was introduced, with representative example types included to support downstream automation (Table A3). Automated batch processing of the resulting mapping relationships was implemented using Python scripts (representative excerpts provided in Appendix C.2), enabling multidimensional statistical analysis of material and spatial distributions. Supplementary visualizations of the classification outputs, including category–subcategory–count and subcategory–material–count summaries, are provided in Appendix B (Tables A4 and A5), and example type-level statistics are shown in Figure A1.

4.2.2. Materials and Spatial Distribution

As shown in Table A6, "Metal Stud Layer" and "Metal Furring" are by far the most frequent materials within the wall category, reflecting the dominance of lightweight steel framing in this building's interior partitions. For doors and windows, steel ("STL") and aluminum ("ALUM") are most common, consistent with commercial construction standards. This material breakdown facilitates efficient quantity takeoff and supports further sustainability analysis.

Table A7 summarizes the floor-wise distribution of architectural elements. Doors and windows are distributed across levels L1–L5, with higher counts on typical floors. Wall elements, recorded without explicit floor assignment in this schedule, reflect standard Revit modeling and scheduling practices. These spatial statistics are valuable for construction planning and cost estimation.

See Figure 9 for a pie chart showing the proportion of main materials.

4.2.3. Automated Statistics and Visualization

Figure 10 illustrates the distribution of elements across architectural subcategories, highlighting the dominance of certain subtypes (e.g., "Basic Wall") within the project. This automated classification assists in identifying modeling biases or areas for potential design optimization.

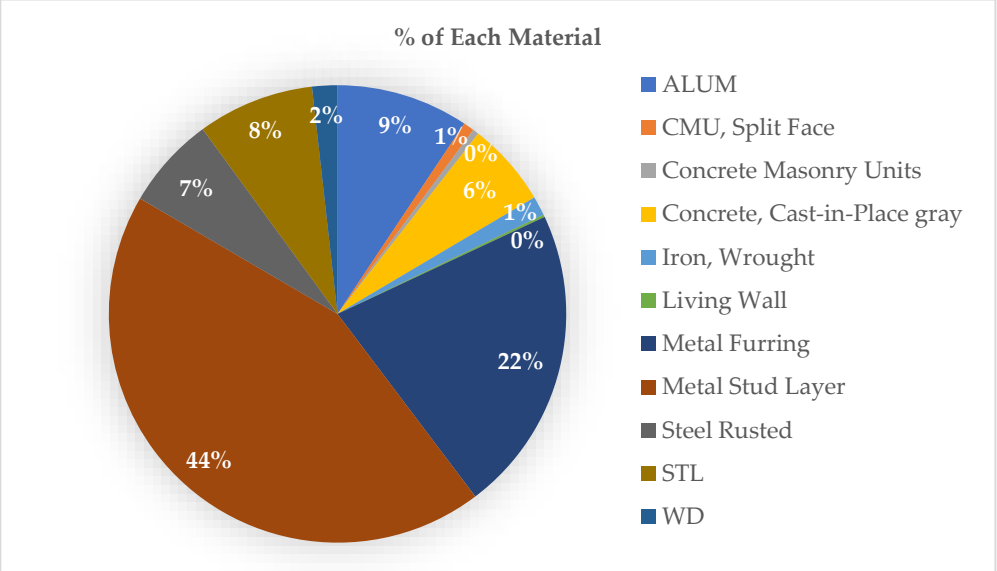


Figure 9. Architectural model: % of each material (pie chart).

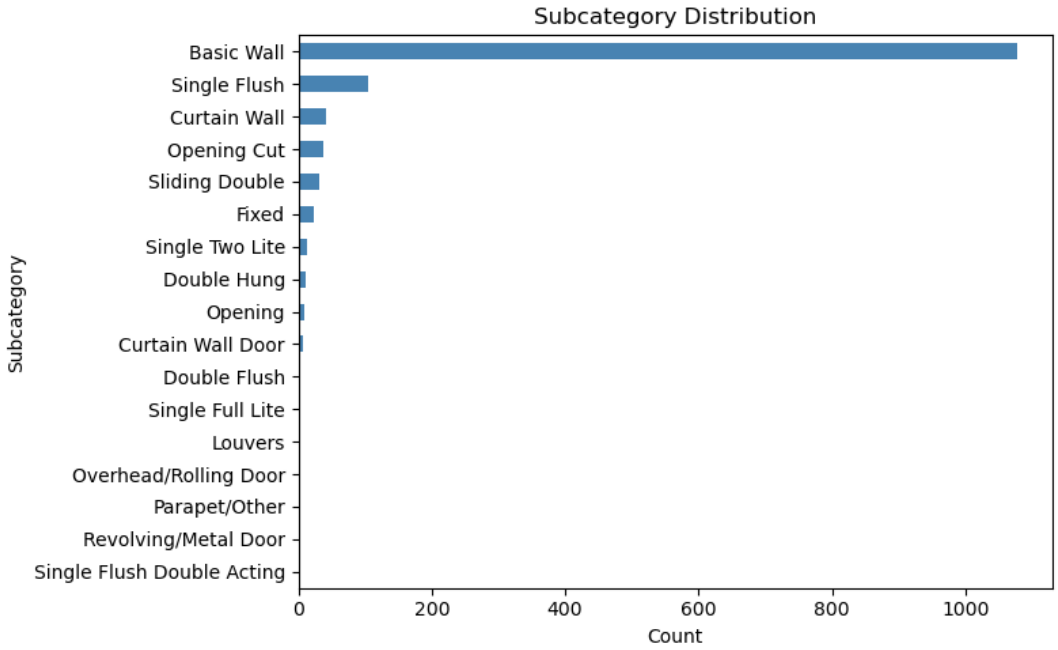


Figure 10. Architectural model: Subcategory distribution (bar chart).

Figure 11 presents a stacked bar chart of material usage by subcategory, offering granular insight into how different materials are allocated among architectural families. This detailed breakdown is essential for contractors and designers aiming to optimize structural-material compatibility and procurement strategies.

4.3. HVAC Model

The HVAC model represents a low-heterogeneity condition, allowing the pipeline to verify semantic consistency and bypass unnecessary normalization. The results confirm that the workflow can automatically detect cases where semantic normalization is not required, thereby avoiding redundant processing. This absence of variation is uncommon in structural and architectural models, underscoring the discipline-specific nature of modeling conventions and explaining why standardized MEP taxonomies often require minimal semantic normalization. This case further validates the workflow's adaptability—automating

only where needed and transparently bypassing redundant processing in highly structured domains. Such a standardized taxonomy in MEP domains signals a maturing industry consensus and lowers the barrier for broader adoption of automated BIM workflows.

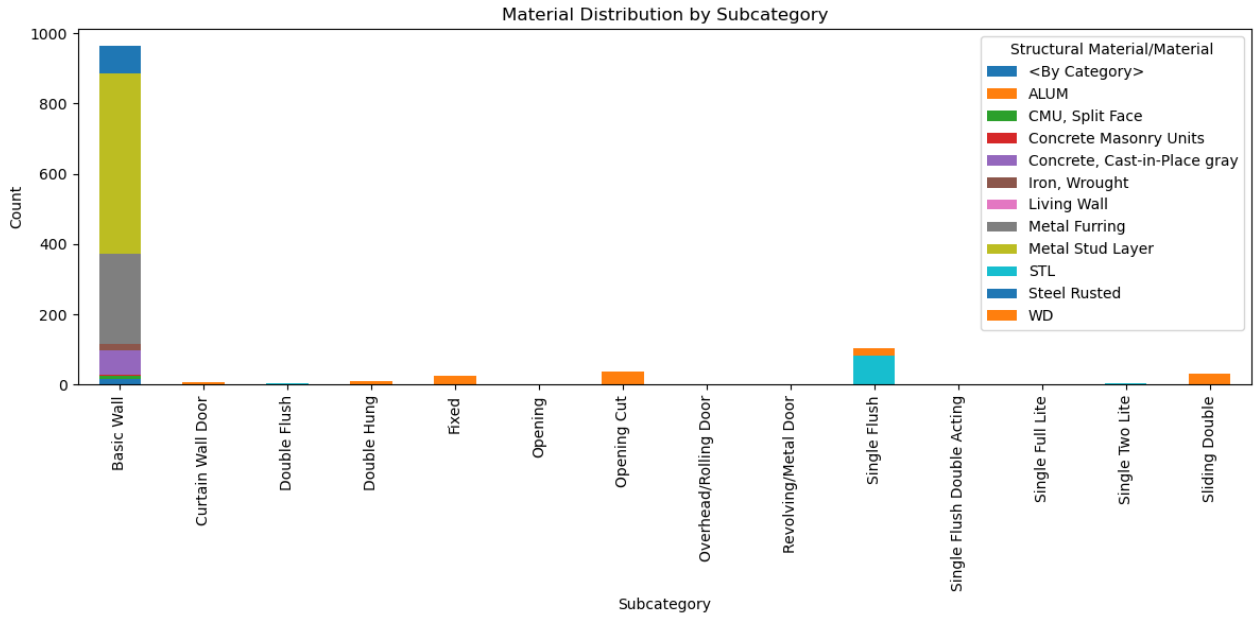


Figure 11. Architectural model: Material distribution by subcategory (stacked bar chart).

A summary table of the HVAC hierarchical structure (Category–Family–Type–Count) is provided in Table A8, and the Category–Subcategory–Family–Example Types mapping is reported in Table A9. Consistency was further verified using an automated Family–Subcategory check (Appendix C.3), which confirmed one-to-one mapping and detected no ambiguous or many-to-many pairs; therefore, no additional normalization was required. The type distribution is summarized in Figure 12.

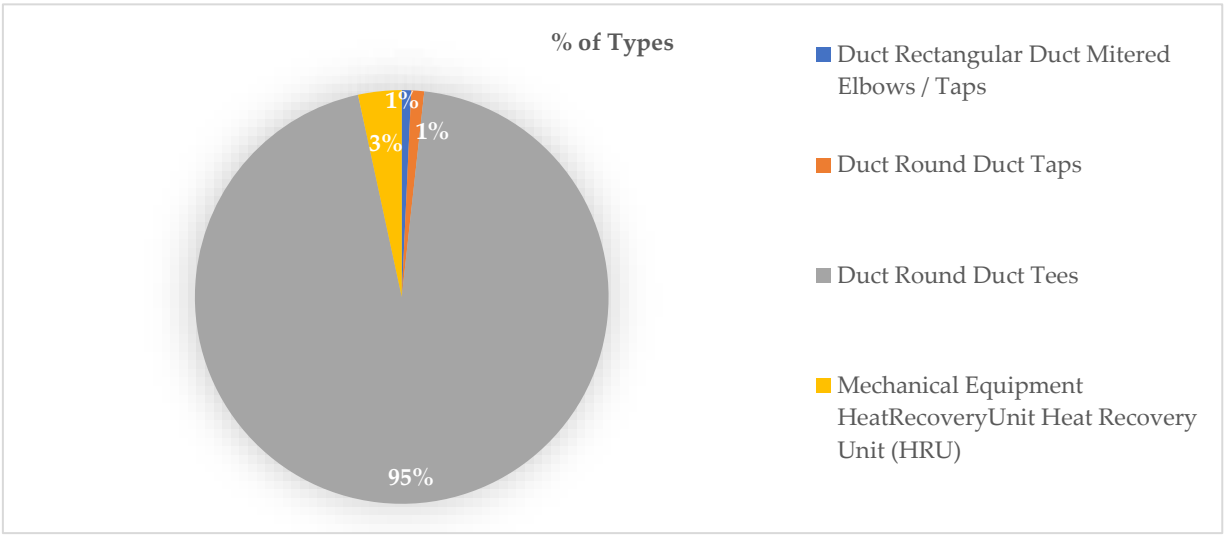


Figure 12. HVAC model: % of types (pie chart).

4.4. Sample-Based Validation of Semantic Mapping Agreement and Processing Efficiency

To avoid misinterpretation of quantitative results as predictive or algorithmic performance metrics, this study adopts a sample-based validation protocol focused on agreement with human-reviewed reference mappings rather than accuracy in a machine learning sense. Validation is performed to assess the internal consistency and reliability of the proposed

deterministic workflow under realistic modeling conditions, not to evaluate generalization or predictive capability.

To evaluate the reliability of the proposed workflow under realistic modeling conditions, a sample-based validation was conducted across three representative BIM models. For each model, a random subset of Revit schedule entries was manually checked by two independent reviewers to confirm the correctness of automated category–subcategory mapping. The samples were randomly selected across different floors and element types to ensure representation of heterogeneous modeling patterns, reducing bias and providing a realistic approximation of typical project variability.

For each discipline-specific model, a random subset of schedule entries was selected for validation. Sampling was stratified across different element categories and building levels to capture representative semantic variation. Each sampled entry was independently reviewed by two domain-informed reviewers, and discrepancies were resolved through discussion to establish a consensus reference mapping. While not intended as an inter-rater reliability study, this procedure provides a pragmatic check on workflow consistency under realistic project conditions.

The validation results, summarized in Table 5, indicate a high level of agreement between automated mappings and human-validated classifications across all three models, supporting the reliability of the workflow under realistic modeling conditions.

Table 5. Sample-based validation of semantic mapping agreement with human-reviewed reference classifications (non-predictive).

Model	Records Checked	Mapping Agreement Rate (%)	Averaged Manual Time (min)	Automated Time (min)	Effort Reduction (%)
Structural	200	>95	90	8	~91
Architectural	150	>96	60	7	~88
HVAC	50	≈100	20	2	~90

Here, "mapping agreement" is defined as the proportion of sampled schedule entries for which the automated category–subcategory assignment matches an independently reviewed human reference classification. This metric reflects internal workflow consistency rather than predictive accuracy or generalization performance.

Although sample-based, these results indicate that the workflow produces consistent and reliable semantic mappings once a mapping configuration is established.

4.5. Cross-Model Query and Automatic Aggregation

To further demonstrate the practical value of automated classification and data standardization, we implemented a cross-model statistical query scenario—a typical pain point in engineering practice. Such scenarios are frequently encountered in real-world projects, including façade opening management, net opening ratio verification, and coordination between architectural, structural, and MEP teams. Specifically, we used the generic model schedule (openings) and the structural model schedule (walls) to automatically compute, for each building level, the total opening area, wall area, and the opening-to-wall area ratio ("opening rate") (representative excerpt shown in Appendix C.4). Such cross-model analytics, once a major bottleneck in design coordination and compliance checks, can now be routinely scripted and updated as project data evolves, and are significantly accelerated through the proposed script-driven Python workflow, reducing manual coordination effort and improving overall efficiency.

The cross-model aggregation results demonstrate that, once schedule semantics are normalized, discipline-specific datasets can be merged and queried without bespoke inte-

gration logic, which is a key indicator of workflow generalizability. Figure 13 presents the results of the cross-model aggregation, listing, for each floor, the number of openings, total opening area, wall count, wall area, and the calculated opening rate. The results highlight the power of semantic normalization and batch scripting in integrating and analyzing heterogeneous BIM data sources.

Level	Opening_Count	Opening_Area_in2	Wall_Count	Wall_Area	Opening_Ratio	Opening_Ratio_%
Elevator Pit	NaN	NaN	16.00	842.00	NaN	NaN
L1_35_Low	NaN	NaN	1.00	4067.00	NaN	NaN
L1_37_Med	NaN	NaN	1.00	6239.00	NaN	NaN
L1_43_High	6.00	103,622.00	4.00	4920.00	0.15	14.63
L2	5.00	8846.00	1.00	4279.00	0.01	1.44
L2 TOS	2.00	0.00	NaN	NaN	NaN	NaN
L3	5.00	15,132.00	3.00	9432.00	0.01	1.11
L3 TOS	9.00	0.00	NaN	NaN	NaN	NaN
L4	5.00	15,132.00	2.00	10,365.00	0.01	1.01
L4 TOS	11.00	0.00	NaN	NaN	NaN	NaN
L5	6.00	27,530.50	3.00	9612.00	0.02	1.99
L5 TOS	11.00	0.00	NaN	NaN	NaN	NaN
M1	2.00	3726.00	1.00	899.00	0.03	2.88
M1 TOS	6.00	0.00	NaN	NaN	NaN	NaN
Parapet 2	NaN	NaN	5.00	725.00	NaN	NaN
Parking	7.00	48,834.00	30.00	11,985.00	0.03	2.83
R2	7.00	15,132.00	1.00	7060.00	0.01	1.49
R2 TOS	9.00	0.00	1.00	804.00	0.00	0.00
Top of Footing	NaN	NaN	39.00	1001.00	NaN	NaN

Figure 13. Cross-model level-wise opening and wall statistics (with opening ratio).

5. Discussion

From a methodological perspective, the primary contribution of this study is the formalization of a schedule-level semantic integration pipeline that separates semantic normalization from model geometry, enabling repeatable cross-model analytics without reliance on rigid schemas or learning-based algorithms. It is important to emphasize that the contribution of this study lies at the workflow and methodological abstraction level, rather than in the development of new learning-based or algorithm-centric methods. The discussion therefore focuses on how schedule-level semantic normalization and structured cross-model integration can support repeatable analytics and engineering workflows, rather than on algorithmic performance or predictive intelligence. Empirical application to three representative models offers several areas for further discussion:

5.1. Semantic Classification Complexity and Automation Challenges Across BIM Models

Rather than treating the three discipline-specific models as isolated case studies, this work intentionally leverages their differing degrees of semantic heterogeneity as controlled test conditions for the proposed pipeline. The structural, architectural, and HVAC models, respectively, represent high-, medium-, and low-heterogeneity scenarios, allowing the workflow's robustness, adaptability, and boundary conditions to be evaluated without introducing additional datasets.

The structural model posed the greatest challenge for mapping and batch semantic classification, due to highly heterogeneous family naming and the lack of standardization. By introducing a three-level "Category–Subcategory–Family" semantic hierarchy, the automated workflow enabled rapid normalization of complex, multi-dimensional data—laying a foundation for downstream analytics and data-driven analytics, standardized cost estimation, and design optimization.

The architectural model, though partially standardized, featured a proliferation of fields with inconsistent naming and a strong need for field consolidation. Results demon-

strate that flexible field unification and configurable classification mapping are prerequisites for cross-disciplinary and cross-project BIM big data analytics. This approach can be extended to other highly customized architectural subsystems, such as doors, windows, curtain walls, and finishes.

The HVAC model illustrated a high degree of industry standardization in classification, confirming the effectiveness of automated consistency checking in reducing manual normalization and enhancing data reusability. This case reflects the trend toward "pre-classified" modeling and semantic mapping in MEP domains and provides a reference for future industry-standard automated verification and semantic validation. Our findings suggest that upfront standardization or adoption of robust mapping workflows is essential for large-scale, federated BIM projects seeking to leverage automation. Integrating such strategies early in the modeling process can dramatically reduce downstream manual effort, enhance data quality, and maximize the benefits of digital construction workflows. In practical design coordination, improved semantic alignment also reduces Requests for Information (RFIs) and minimizes coordination clashes, since ambiguities in element categorization are resolved upstream—leading to fewer interpretation errors and faster interdisciplinary approvals.

5.2. Methodological Innovations and Application Extensions

The study's introduction of classification consistency checking and lightweight adaptive batch normalization supports both industry standards and project-specific customization, significantly improving the efficiency of semantic data processing for BIM. The principle of "batch mapping + consistency assessment + redundancy minimization" forms a universal toolchain for enterprise-level BIM operations, smart construction site data integration, and digital twins. The reduction in manual mapping effort—often the single largest barrier to data-driven BIM—directly supports enterprise digital transformation and opens new avenues for downstream analytics and data-driven model development at scale. By using script-driven automation to perform semantic classification, the workflow not only accelerates project cycles but also helps eliminate persistent 'information silos' between disciplines, enabling more effective data sharing and holistic project analysis.

Automated classification and multi-dimensional analytics improve BIM data interoperability and reusability, laying the groundwork for emerging research in learning-enabled extensions built upon structured BIM data, design pattern recognition, material flow analysis, and carbon footprint estimation. Compared with manual or template-based normalization, the proposed workflow achieved up to an order-of-magnitude efficiency gain, effectively eliminating most manual mapping and field alignment work. This aligns with recent automation advances reported in Liu et al. [15] and Utkucu et al. [16], though our focus lies on scalable schedule-level integration rather than element-level deep learning classification. The ability to perform fine-grained automated classification in structural and architectural models further enables downstream applications such as intelligent recommendations and knowledge graph construction.

To further clarify the methodological positioning of the proposed approach, Table 6 provides a structural comparison between the proposed schedule-level integration pipeline and commonly adopted alternatives, including rule-based scripting, ontology-driven methods, and learning-based classification approaches. Rather than benchmarking algorithmic performance, the comparison highlights differences in abstraction level, adaptability, determinism, and suitability for cross-disciplinary schedule integration.

Notably, the ability to automate cross-model data aggregation—such as the integration of opening and wall schedules for real-time opening ratio calculation—represents a significant advance over conventional case-by-case querying. This cross-domain capability

is enabled by semantic normalization at both family and subcategory levels, ensuring that disparate BIM datasets can be seamlessly merged, queried, and visualized. The validation results (Table 5) confirm the workflow's reliability, achieving semantic mapping accuracies above 95% and over 85% average time savings across disciplines—substantially reducing the manual overhead that traditionally limits data-driven BIM analysis. From a project-management standpoint, these efficiency gains translate into measurable downstream value, potentially lowering coordination costs and shortening review cycles in multi-disciplinary BIM environments. The approach demonstrated here provides a scalable foundation for high-throughput engineering analyses, risk evaluation, and data-driven construction management.

Table 6. Comparison of schedule-level integration approaches.

Aspect	Rule-Based Scripts	Ontology-Based	Learning-Based	Proposed Pipeline
Schedule-level focus	Partial	Limited	Rare	Yes
Cross-discipline reuse	Low	Medium	Low	High
Training required	No	No	Yes	No
Adaptability	Low	Low	Medium	High
Determinism	High	High	Low	High

5.3. Limitations and Future Work

This study's main limitations include: (1) the classification dictionary, field consolidation, and mapping process are not yet fully automated, with some scenarios still requiring manual review and adjustment; and (2) only representative sample-based benchmarking was conducted, so further quantitative validation on larger, more diverse datasets remains an important direction for future work. The flexible nature of the proposed pipeline, however, makes such large-scale benchmarking feasible, and future studies should extend testing to large projects and multiple BIM platforms. While the current validation used representative sample-based verification rather than full-scale benchmarking, the observed performance improvements—especially in mapping agreement and time efficiency—provide a realistic indication of practical gains achievable with lightweight script-driven automation. More fundamentally, practical deployment in real-world projects faces additional complexity—including heterogeneous data sources, integration with legacy systems, and the need for robust data governance protocols. In addition, broader adoption will require industry-wide data standardization initiatives and effective dissemination of open-source toolkits to practitioners.

Beyond these practical limitations, the proposed pipeline operates under clearly defined boundary conditions: availability of tabular schedule exports, minimally interpretable categorical or parametric fields, and the ability to define a project-appropriate semantic granularity (e.g., a customizable subcategory layer) that bridges high-level categories and model-specific families, as well as manageable levels of naming heterogeneity. These conditions define the scope within which the abstraction is transferable across projects, tools, and disciplines, and clarify that the contribution lies in workflow formalization rather than universal semantic resolution.

Future research should extend the workflow's capabilities by developing ML-based classification recommendations built on structured schedule data, automated label-learning mechanisms, and cross-project rule-library sharing. Incorporating graph-structured information—such as spatial adjacency or system connectivity—could further enhance semantic reasoning. No supervised or pre-trained ML model was employed for classification; future work may explore data-driven learning approaches. In parallel, deeper integration with real-time construction data would enable closed-loop analytics, supporting

automated code compliance checking, digital twin calibration, and dynamic, web-based BIM environments. These directions represent practical next steps for embedding the proposed workflow within enterprise-scale BIM ecosystems and advancing toward fully automated, continuously updating semantic data pipelines.

6. Conclusions

This study proposed and empirically validated an automated semantic classification and data standardization workflow for multi-disciplinary BIM models, capable of accommodating multi-level semantic hierarchies and heterogeneous data sources. The method enables on-demand, batch selection and quantitative analysis of any element Family or Type after automated classification—providing a reusable data pipeline for refined construction management and cross-project benchmarking. By structuring schedule data into consistent semantic layers, the workflow establishes a necessary data foundation for downstream quantity takeoff, cross-model aggregation, and automated engineering analysis. Rather than introducing a new algorithm, the contribution of this work lies in formalizing a reusable, schedule-level workflow abstraction that addresses a commonly encountered but under-theorized integration problem in BIM practice.

Based on extensive experiments with structural, architectural, and HVAC Revit models, our results demonstrate that the workflow:

- Rapidly achieves high-efficiency mapping and normalization between Family and industry-aligned subcategory semantics, enabling consistent identification of elements required for downstream quantity takeoff and cross-model computation;
- Flexibly adapts to varying levels of modeling standardization and complexity, supporting repeatable quantity- and area-based analytics across disciplines without manual rework;
- Sample-based testing across representative BIM models shows that the workflow consistently achieves semantic agreement rates above 95% with human-validated mappings, demonstrating reliable schedule-level normalization. In addition, over 85% average time savings were observed compared with manual normalization, directly reducing the effort required to prepare BIM data for quantity takeoff and coordinated analysis;
- Supports automated consistency checking and decision-based normalization, reducing manual effort and aligning with emerging trends in standardized industry modeling;
- Provides a foundational integrated data layer and methodological support for downstream applications, including script-driven BIM analytics, intelligent recommendations, and cross-platform data integration.

This approach is generalizable, scalable, and readily applicable in engineering practice, offering an effective pathway for the automation and intelligent mining of BIM data, as well as for standardizing digital construction processes. This capability directly supports real-world engineering tasks such as quantity takeoff validation, façade opening control, and interdisciplinary design coordination, which traditionally require extensive manual reconciliation across models. The proposed workflow not only enables semantic normalization and automation within single-discipline BIM models, but also supports robust, high-efficiency cross-model data retrieval and aggregation—a critical requirement for next-generation digital construction and smart facility management. By enhancing cross-disciplinary semantic alignment and reducing the frequency of semantic conflicts and RFIs, the workflow directly supports more seamless collaboration between designers, engineers, and contractors. Overall, it provides a robust foundation for semantic analytics and integrated construction management, serving as a stepping stone toward scalable, data-driven digital construction ecosystems.

Going forward, this methodology offers immediate value for industry practitioners seeking to streamline schedule-based workflows and opens new research avenues in scalable workflow-driven BIM analysis, automated code compliance, and semantic data integration for digital twins. From a practical perspective, the workflow lowers the barrier for engineers and construction managers to reuse BIM schedule data for quantitative analysis, reducing reliance on manual spreadsheets and ad hoc scripts. Its adoption can accelerate high-quality digital transformation in the construction sector, enabling both engineering empowerment and academic innovation.

Author Contributions: Conceptualization, T.Z. and R.N.; methodology, T.Z.; software, T.Z.; validation, T.Z. and R.N.; formal analysis, T.Z.; investigation, T.Z. and R.N.; resources, R.N.; data curation, T.Z.; writing—original draft preparation, T.Z.; writing—review and editing, T.Z. and R.N.; visualization, T.Z.; supervision, T.Z.; project administration, T.Z. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data and code supporting this study are publicly available at https://github.com/TZ-code-code/bim-semantic-mapping (accessed on 22 March 2026). The repository contains executable Python scripts, versioned semantic mapping files, example schedule CSVs, and documentation. Environment specifications, including Python version and library dependencies (e.g., pandas and matplotlib), are provided via a requirements.txt file to support reproducibility.

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

Appendix A

```
examples = df[df["Family"].isin(["Door-Opening", "Door-Passage-Single-Two_Lite"])]

type_examples = examples.groupby(["Family", "Type"]).rize().reset_index(name="Count")
print(type_examples)
print(type-Ssamples)
```

```
                      Family                Type  Count
0                Door-Opening          32" x 84"      8
1                Door-Opening          60" x 84"      1
2  Door-Passage-Single-Two_Lite  34" x 84" (90 MIN)      3
3  Door-Passage-Single-Two_Lite  36" x 84" (90 MIN)      2
4  Door-Passage-Single-Two_Lite  36" x 89" (90 MIN)      8
```

Figure A1. Python-based retrieval and counting of example types after automated classification. The combined script-and-output view is retained to illustrate the typical form of executable workflow results.

Appendix B

Table A1. Full taxonomy for the Structural model (Category–Subcategory–Family–Example Type).

Category	Subcategory	Family	Example Types
Structural Column	Concrete Column	Concrete-Rectangular-Column	CC24 × 24
	Steel Column	W Shapes-Column, HSS	W10 × 49, W14 × 109, W18 × 119, W8 × 31; HSS6 × 6 × 5/8
	Reference Column	SC_Reference Column	SC_Reference Column

Table A1. *Cont.*

Category	Subcategory	Family	Example Types
Structural Beam/Framing	Concrete Beam	Concrete-Rectangular Beam	CB18 × 24, CB24 × 51, CB24 × 24, CB24 × 30, CB24 × 39
	Steel Beam	W Shapes	W12 × 26, W14 × 30, W16 × 26, W18 × 55, W18 × 86, W30 × 235, W36 × 232, W8 × 10
	Bar Joist	K-Series Bar Joist, LH-Series Bar Joist	16K2, 16K6, 20K6; 18LH04
	Steel Angle/Channel	SF_Steel L Angle, C Shapes	_Size; C8 × 11.5
Structural Floor/Slab	Concrete Slab	Floor	1 1/2 Metal Roof Deck NO Concrete, Concrete 10, Concrete 12″, Concrete 6, NW Concrete on Metal Deck
Structural Foundation	Foundation Slab	Foundation Slab	FNDN-36
	Wall Foundation	Wall Foundation	Bearing Footing—36″ × 18″
	Pile Cap	SFD_Pile_Cap_Rectangle	36 × 36 × 24 Pile_24, 36 × 36 × 24 Pile_30
	Round Foundation	SFD_Round	24 × 24 × 20′, 24 × 24 × 25′, 30 × 30 × 25′
Structural Wall	Concrete Wall	Basic Wall	Concrete 10″, Concrete 12″, Concrete 18″, Concrete 24″

Table A2. Architectural Model: Category–Family–Count Table highlighting field-driven semantic heterogeneity.

Category	Family	Count
Door	TOTAL	141
Door	Door-Curtain-Wall-Double-Storefront	4
Door	Door-Curtain-Wall-Single-Storefront	2
Door	Door-Exterior-Revolving-Full Glass-Metal	1
Door	Door-Opening	9
Door	Door-Overhead-Rolling	1
Door	Door-Passage-Double-Flush	3
Door	Door-Passage-Single-Flush	105
Door	Door-Passage-Single-Flush-Dbl_Acting	1
Door	Door-Passage-Single-Full_Lite	2
Door	Door-Passage-Single-Two_Lite	13
Wall	TOTAL	1120
Wall	Basic Wall	1077
Wall	Curtain Wall	42
Wall	Parapet Cap Bandstand	1
Window	TOTAL	106
Window	Schematic Opening Cut	38
Window	Window-Double-Hung	9
Window	Window-Double-Hung-Double	1
Window	Window-Fixed	20
Window	Window-Fixed-Transom	4
Window	Window-Louvers	2
Window	Window-Sliding-Double	32
Total		1367

Table A3. Full taxonomy for the Architectural Model (Category–Subcategory–Family–Example Type).

Category	Subcategory	Family	Example Types
Wall	Basic Wall	Basic Wall	Block 37 Pilaster, Block 39 Pilaster, Chase—GWB & Metal Stud 2 1/4″, Chase—GWB & Metal Stud 3 1/8″, Chase—GWB & Metal Stud 4 1/4″, Chase—GWB & Metal Stud 6 5/8″, Core—Concrete 10″, Core—Concrete 12″, Core—Concrete 18″, Exterior—10 1/4″ Rainscreen w Insultation on Metal Stud, Exterior—12 5/8″ Rainscreen w Insultation on Metal Stud, Exterior—12 5/8″ Rainscreen w Insultation on Metal Stud (Alley), Exterior—13 5/8″ Rainscreen (Two Sides) w Insultation on Metal Stud, Exterior—14 5/8″ Rainscreen w Insultation on Metal Stud, Exterior—5 3/8″ Rainscreen w Insultation (for Concrete Core Walls), Exterior—9 7/8″ EIFS on Mtl. Stud, Exterior—GWB on Metal Stud w Insulated Panel 8 7/8″, Foundation—24″ Concrete, Furring—GWB & Channels 1 7/8″, Generic—4″, Generic—6″, Interior—4 7/8″ Partition (1-h), Interior—4 7/8″ Partition (NR), Interior—5 1/2″ Partition (1 h Smoke), Interior—6 1/8″ Partition (2-h), Interior—7 1/4″ Partition (1 h), Interior—7 1/4″ Partition (NR), Interior—8 1/2″ Partition (2-h), Interior—CMU 8″, Living Wall, Party—6 1/8″ (2-h), Planter Wall—Steel Plate 1/2″, Retaining—Split Face CMU, Soffit—Beam Wrap (Both Sides), Soffit—Beam Wrap 10 1/2″ (Two Sides), Soffit—Beam Wrap 11 1/4″ (One Side), Soffit—Beam Wrap 9 1/4″ (One Side), Soffit—Beam Wrap Large (Both Sides), Solar Wall (L5), Solar Wall (Multi-Level), Solar Wall (Parapet), Trellis Member
	Curtain Wall	Curtain Wall	_Not Defined, Block 37 Storefront, Block 39 Storefront, Block 41 Storefront, Block 43 Storefront, Glazing Wall—Stair, Solar Panels, Storefront
	Parapet/Other	Parapet Cap Bandstand	Parapet Cap Bandstand

Table A3. *Cont.*

Category	Subcategory	Family	Example Types
Door	Curtain Wall Door	Door-Curtain-Wall-Double-Storefront, Door-Curtain-Wall-Single-Storefront	Door-Curtain-Wall-Double-Storefront; Door-Curtain-Wall-Single-Storefront
	Revolving/Metal Door	Door-Exterior-Revolving-Full Glass-Metal	78″ × 108″
	Opening	Door-Opening	32″ × 84″; 60″ × 84″
	Overhead/Rolling Door	Door-Overhead-Rolling	Access Door
	Double Flush	Door-Passage-Double-Flush	72″ × 84″
	Single Flush	Door-Passage-Single-Flush	26″ × 84″, 36″ × 84″, 36″ × 84″ (180 MIN), 36″ × 84″ (20 MIN), 36″ × 84″ (45 MIN), 36″ × 84″ (60 MIN), 36″ × 84″ (90 MIN), 36″ × 84″ (Cores), 36″ × 84″ (Exterior), 36″ × 84″ (Roof), 36″ × 96″, 36″ × 96″ (Exterior), 42″ × 96″ (Exterior)
	Single Flush Double Acting	Door-Passage-Single-Flush-Dbl_Acting	36″ × 96″
	Single Full Lite	Door-Passage-Single-Full_Lite	36″ × 96″ Storefront
	Single Two Lite	Door-Passage-Single-Two_Lite	34″ × 84″ (90 MIN), 36″ × 84″ (90 MIN), 36″ × 89″ (90 MIN)
Window	Opening Cut	Schematic Opening Cut	Schematic Opening Cut
	Double Hung	Window-Double-Hung, Window-Double-Hung-Double	50″ × 98″; 74″ × 98″
	Fixed	Window-Fixed, Window-Fixed-Transom	39″ × 76″ (Solar Wall); 50″ × 80″
	Louvers	Window-Louvers	22″ × 24″
	Sliding Double	Window-Sliding-Double	50″ × 60″

Note: Abbreviations used in the table follow common building and BIM naming conventions. “GWB” denotes gypsum wall board; “Mtl.” denotes metal; “CMU” denotes concrete masonry unit; and “EIFS” denotes exterior insulation and finish system. The notation “w” indicates “with,” and dimensional values (e.g., 4 7/8″, 10 1/4″) represent nominal wall thicknesses in inches.

Fire-resistance ratings are indicated in parentheses using either hours or minutes. “1-h” and “2-h” denote one-hour and two-hour fire-rated assemblies, respectively, while “MIN” (e.g., 20 MIN, 45 MIN, 90 MIN) denotes fire-resistance duration in minutes, commonly used for door assemblies. “NR” indicates no fire-resistance rating, and labels such as “Smoke” denote assemblies with additional smoke-resistance requirements.

Terms such as "Chase," "Core," "Furring," "Soffit," and "Rainscreen" refer to functional or construction-specific wall configurations as defined in standard architectural practice. "Curtain Wall" and "Storefront" denote glazing systems typically used for exterior envelope assemblies.

"Generic" and "Not Defined" indicate placeholder or non-standardized family definitions within the BIM model. Dimensions listed for doors and windows (e.g., $36'' \times 84''$) represent nominal width and height.

All names are preserved as defined in the original BIM model to maintain consistency with the source data.

Table A4. Python classification results (Category–Subcategory–Count).

Category	Subcategory	Count
Door	Curtain Wall Door	6
Door	Double Flush	3
Door	Opening	9
Door	Overhead/Rolling Door	1
Door	Revolving/Metal Door	1
Door	Single Flush	105
Door	Single Flush Double Acting	1
Door	Single Full Lite	2
Door	Single Two Lite	13
Wall	Basic Wall	1077
Wall	Curtain Wall	42
Wall	Parapet/Other	1
Window	Double Hung	10
Window	Fixed	24
Window	Louvers	2
Window	Opening Cut	38
Window	Sliding Double	32

Table A5. Python classification results (Subcategory–Structural Material/Material–Count).

Subcategory	Structural Material/Material	Count
Basic Wall	<By Category>	15
Basic Wall	CMU, Split Face	9
Basic Wall	Concrete Masonry Units	5
Basic Wall	Concrete, Cast-in-Place gray	70
Basic Wall	Iron, Wrought	16
Basic Wall	Living Wall	2
Basic Wall	Metal Furring	255
Basic Wall	Metal Stud Layer	514
Basic Wall	Steel Rusted	77
Curtain Wall Door	ALUM	6
Double Flush	STL	3
Double Hung	ALUM	10
Fixed	ALUM	24
Opening	STL	1
Opening Cut	ALUM	38
Overhead/Rolling Door	STL	1
Revolving/Metal Door	STL	1
Single Flush	STL	83
Single Flush	WD	21
Single Flush Double Acting	STL	1
Single Full Lite	STL	2
Single Two Lite	STL	5
Sliding Double	ALUM	32

Table A6. Architectural model: Material distribution by category.

Category	Material	Count
Door	TOTAL	141
Door	ALUM	6
Door	STL	97
Door	WD	21
Door	(blank)	17
Wall	TOTAL	1120
Wall	<By Category>	15
Wall	CMU, Split Face	9
Wall	Concrete Masonry Units	5
Wall	Concrete, Cast-in-Place gray	70
Wall	Iron, Wrought	16
Wall	Living Wall	2
Wall	Metal Furring	255
Wall	Metal Stud Layer	514
Wall	Steel Rusted	77
Wall	(blank)	157
Window	TOTAL	106
Window	ALUM	104
Window	(blank)	2
Grand Total		1367

Table A7. Architectural model: Category–level distribution table.

Category	Level	Count
Door	TOTAL	141
Door	L1—Block 35	8
Door	L1—Block 37	16
Door	L1—Block 43	10
Door	L2	16
Door	L3	24
Door	L4	29
Door	L5	20
Door	Parking	8
Door	R2	10
Wall	TOTAL	1120
Wall	(blank)	1120
Window	TOTAL	106
Window	L2	15
Window	L3	33
Window	L4	33
Window	L5	21
Window	M1	4
Grand Total		1367

Table A8. HVAC model: Category–Family–Count table.

Category	Item	Count
Duct	TOTAL	1053
Duct	Rectangular Duct	8
Duct	Mitered Elbows/Taps	8
Duct	Round Duct	1045
Duct	Taps	11
Duct	Tees	1034
Mechanical Equipment	TOTAL	37
Mechanical Equipment	HeatRecoveryUnit	37
Mechanical Equipment	Heat Recovery Unit (HRU)	37
Grand Total		1090

Table A9. HVAC model: Category–Subcategory–Family–Example types table.

Category	Subcategory	Family	Example Types
Duct	Rectangular Duct	Rectangular Duct	Mitered Elbows/Taps
Duct	Round Duct	Round Duct	Taps, Tees
Mechanical Equipment	Heat Recovery Unit	HeatRecoveryUnit	Heat Recovery Unit (HRU)

Appendix C

Note: Due to space limitations, only representative excerpts of the scripts are provided below. Full Python scripts will be made publicly available upon publication.

Appendix C.1. Structural Model

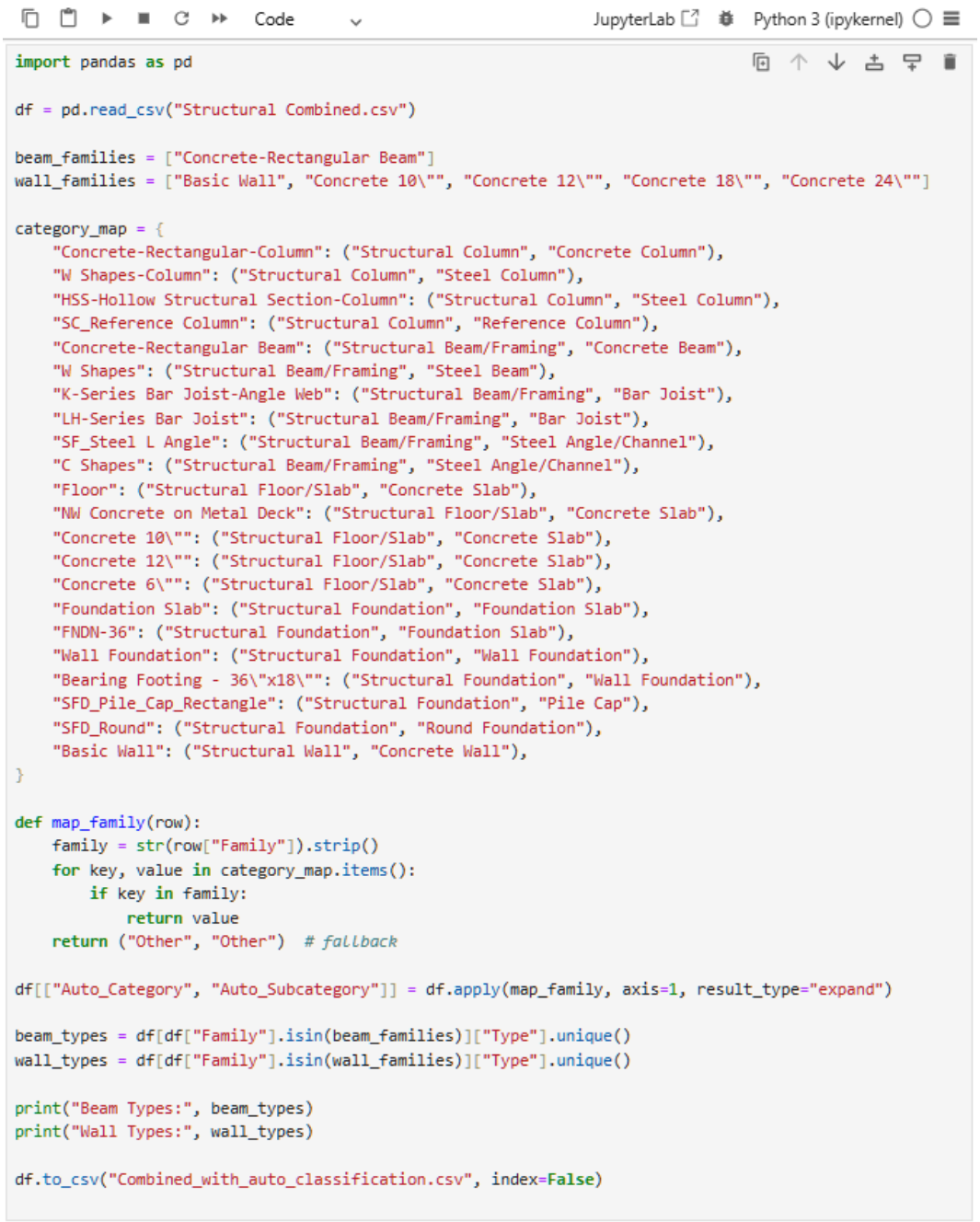


Figure A2. Structural model script (representative excerpt).

Appendix C.2. Architectural Model

```python
import pandas as pd

df = pd.read_csv("Architectural Combined.csv")

subcategory_map = {
    "Basic Wall": "Basic Wall",
    "Curtain Wall": "Curtain Wall",
    "Parapet Cap Bandstand": "Parapet/Other",
    "Door-Curtain-Wall-Double-Storefront": "Curtain Wall Door",
    "Door-Curtain-Wall-Single-Storefront": "Curtain Wall Door",
    "Door-Exterior-Revolving-Full Glass-Metal": "Revolving/Metal Door",
    "Door-Opening": "Opening",
    "Door-Overhead-Rolling": "Overhead/Rolling Door",
    "Door-Passage-Double-Flush": "Double Flush",
    "Door-Passage-Single-Flush": "Single Flush",
    "Door-Passage-Single-Flush-Dbl_Acting": "Single Flush Double Acting",
    "Door-Passage-Single-Full_Lite": "Single Full Lite",
    "Door-Passage-Single-Two_Lite": "Single Two Lite",
    "Schematic Opening Cut": "Opening Cut",
    "Window-Double-Hung": "Double Hung",
    "Window-Double-Hung-Double": "Double Hung",
    "Window-Fixed": "Fixed",
    "Window-Fixed-Transom": "Fixed",
    "Window-Louvers": "Louvers",
    "Window-Sliding-Double": "Sliding Double"
}

df["Subcategory"] = df["Family"].map(subcategory_map)

df.to_csv("Architectural_with_subcategory.csv", index=False)

df.head()
```

Figure A3. Architectural model script.

```python
pivot_cat_subcat = df.pivot_table(index=["Category", "Subcategory"], values="Count", aggfunc="sum")
display(pivot_cat_subcat)

pivot_subcat_material = df.pivot_table(
    index=["Subcategory", "Structural Material/Material"],
    values="Count",
    aggfunc="sum"
)
display(pivot_subcat_material)
```

Figure A4. Python classification script for "Subcategory–Structural Material/Material–Count" pivot table.

```python
import matplotlib.pyplot as plt

subcat_count = df.groupby("Subcategory")["Count"].sum().sort_values()
subcat_count.plot(kind='barh', figsize=(8,5), color="steelblue")
plt.xlabel("Count")
plt.title("Subcategory Distribution")
plt.tight_layout()
plt.show()

pivot_plot = df.pivot_table(
    index="Subcategory",
    columns="Structural Material/Material",
    values="Count",
    aggfunc="sum",
    fill_value=0
)
pivot_plot.plot(kind="bar", stacked=True, figsize=(12,6))
plt.ylabel("Count")
plt.title("Material Distribution by Subcategory")
plt.tight_layout()
plt.show()
```

Figure A5. Python classification script for "Material Distribution by Subcategory" stacked bar chart.

Appendix C.3. HVAC Model

```python
import pandas as pd

encodings = ["utf-8-sig", "gbk", "cp936", "latin1"]

for enc in encodings:
    try:
        df = pd.read_csv("HVAC Combined.csv", encoding=enc)
        print(f"Success! Encoding used: {enc}")
        break
    except Exception as e:
        print(f"Failed with encoding: {enc}, error: {e}")

subcategory_map = {
    "Rectangular Duct": "Rectangular Duct",
    "Round Duct": "Round Duct",
    "HeatRecoveryUnit": "Heat Recovery Unit",
}
df["Subcategory"] = df["Family"].map(subcategory_map)

unique_fam_subcat = df.groupby("Family")["Subcategory"].nunique()
print(unique_fam_subcat)
```

```
Failed with encoding: utf-8-sig, error: 'utf-8' codec can't decode byte 0xf8 in position 12548:
invalid start byte
Failed with encoding: gbk, error: 'gbk' codec can't decode byte 0xf8 in position 12548: illegal
multibyte sequence
Failed with encoding: cp936, error: 'gbk' codec can't decode byte 0xf8 in position 12548: illega
l multibyte sequence
Success! Encoding used: latin1
Family
HeatRecoveryUnit    1
Rectangular Duct    1
Round Duct          1
Name: Subcategory, dtype: int64
```

Figure A6. HVAC model: consistency validation—Python code and output screenshot.

Appendix C.4. Cross-Model Query and Automatic Aggregation

```
open_summary = gm.groupby('Level').agg(
    Opening_Count=('Count', 'sum'),
    Opening_Area_in2=('Open_Area_in2', 'sum')
).reset_index()

# Structural: robust area extraction
def extract_total_area(area):
    if pd.isna(area): return 0.0
    area = str(area)
    numbers = re.findall(r'[\d,]+(?=\s*SF)', area)
    total = sum([float(num.replace(',', '')) for num in numbers])
    return total

if 'Area' in struc.columns:
    struc['Area_num'] = struc['Area'].map(extract_total_area)
else:
    struc['Area_num'] = 0

if 'Level' in struc.columns:
    wall_summary = struc.groupby('Level').agg(
        Wall_Count=('Count', 'sum'),
        Wall_Area=('Area_num', 'sum')
    ).reset_index()
else:
    wall_summary = pd.DataFrame()

if not wall_summary.empty:
    result = pd.merge(open_summary, wall_summary, on='Level', how='outer')
else:
    result = open_summary

# Add Opening Ratio %
result['Opening_Ratio'] = result['Opening_Area_in2'] / (result['Wall_Area'] * 144)
result['Opening_Ratio_%'] = (result['Opening_Ratio'] * 100).round(2)

cols = ['Level', 'Opening_Count', 'Opening_Area_in2', 'Wall_Count', 'Wall_Area', 'Opening_Ratio', 'Opening_Ratio_%']

cleaned = result[cols].dropna(subset=['Opening_Count', 'Wall_Area'], how='all')
print(cleaned.to_string(index=False, float_format="%.2f"))
cleaned.to_excel('Levelwise_Opening_Wall_Summary_with_Ratio.xlsx', index=False)
```

Figure A7. Cross-model query and automatic aggregation script.

References

1. Family Categories (Revit 2024 Help). Autodesk. Available online: https://help.autodesk.com/view/RVT/2024/ENU/?guid=GUID-48B14613-5BFC-4E4D-B1DF-80B85EC8DFF3 (accessed on 1 July 2025).
2. MEP Categories and Subcategories (Revit 2024 Help). Autodesk. Available online: https://help.autodesk.com/view/RVT/2024/ENU/?guid=GUID-848D90BD-9CAC-426F-9087-CC7BFC97DBBC (accessed on 1 July 2025).
3. Revit Sample Project Files. Autodesk. Available online: https://help.autodesk.com/view/RVT/2024/ENU/?guid=GUID-61EF2F22-3A1F-4317-B925-1E85F138BE88 (accessed on 1 July 2025).
4. Tang, S.; Bito, T.; Shide, K. Automatic Classification of BIM Object Based on IFC Data Using the Uniclass Classification Standard. *Buildings* **2025**, *15*, 2347. [CrossRef]
5. Pupeikis, D.; Navickas, A.A.; Klumbyte, E.; Seduikyte, L. Comparative Study of Construction Information Classification Systems: CCI versus Uniclass 2015. *Buildings* **2022**, *12*, 656. [CrossRef]
6. Matarneh, S.T.; Danso-Amoako, M.; Al-Bizri, S.; Gaterell, M.; Matarneh, R. Developing an interoperability framework for building information models and facilities management systems. In *Proceedings of the Creative Construction Conference 2018—Proceedings, Ljubljana, Slovenia, 30 June–3 July 2018*; Budapest University of Technology and Economics: Budapest, Hungary, 2018; pp. 1018–1024. [CrossRef]
7. Li, X.T.; Zhang, H.W.; Liu, Z.Z.; Qu, W.Y.; Wu, Z.H.; Wang, L. Application of BIM coding and example modelling for typical asphalt pavement diseases based on Omni Class. *J. Phys. Conf. Ser.* **2021**, *2044*, 012153. [CrossRef]

8. Yang, B.; Liu, B.; Xiao, J.; Zhang, B.; Wang, Z.; Dong, M. A novel construction scheduling framework for a mixed construction process of precast components and cast-in-place parts in prefabricated buildings. *J. Build. Eng.* **2021**, *43*, 103181. [CrossRef]
9. Golabchi, A.; Kamat, V.R. Evaluation of Industry Foundation Classes for Practical Building Information Modeling Interoperability. In Proceedings of the 30th International Symposium on Automation and Robotics in Construction and Mining, Montreal, CA, Canada, 11–15 August 2013. [CrossRef]
10. Afsari, K.; Eastman, C.; Shelden, D. Building Information Modeling data interoperability for Cloud-based collaboration: Limitations and opportunities. *Int. J. Archit. Comput.* **2017**, *15*, 187–202. [CrossRef]
11. Lai, H.; Deng, X. Interoperability Analysis of IFC-Based Data Exchange Between Heterogeneous Bim Software. *J. Civ. Eng. Manag.* **2018**, *24*, 537–555. [CrossRef]
12. Steel, J.; Drogemuller, R.; Toth, B. Model interoperability in building information modelling. *Softw. Syst. Model.* **2012**, *11*, 99–109. [CrossRef]
13. Zhang, M.; Lu, B. A Study on Encoding of Urban Underground Space Facilities Based on Omniclass. In *Proceedings of the 18th Conference of the Associated Research Centers for the Urban Underground Space*; Lecture Notes in Civil Engineering; Wu, W., Leung, C.F., Zhou, Y., Li, X., Eds.; Springer Nature: Singapore, 2024; Volume 471, pp. 357–362. [CrossRef]
14. Kuzminykh, A.; Kukina, A.; Bardina, G. 4D and 5D Design Processes Automation Using Databases, Classification and Applied Programming. In *Robotics, Machinery and Engineering Technology for Precision Agriculture; Smart Innovation, Systems and Technologies*; Shamtsyan, M., Pasetti, M., Beskopylny, A., Eds.; Springer: Singapore, 2022; Volume 247, pp. 667–675. [CrossRef]
15. Liu, H.; Gan, V.J.L.; Cheng, J.C.P.; Zhou, S. Automatic Fine-Grained BIM element classification using Multi-Modal deep learning (MMDL). *Adv. Eng. Inform.* **2024**, *61*, 102458. [CrossRef]
16. Utkucu, D.; Ying, H.; Wang, Z.; Sacks, R. Classification of architectural and MEP BIM objects for building performance evaluation. *Adv. Eng. Inform.* **2024**, *61*, 102503. [CrossRef]
17. Austern, G.; Bloch, T.; Abulafia, Y. Incorporating Context into BIM-Derived Data—Leveraging Graph Neural Networks for Building Element Classification. *Buildings* **2024**, *14*, 527. [CrossRef]
18. Wu, J.; Akanbi, T.; Zhang, J. Constructing Invariant Signatures for AEC Objects to Support BIM-Based Analysis Automation through Object Classification. *J. Comput. Civ. Eng.* **2022**, *36*, 04022008. [CrossRef]
19. Xu, Z.; Xie, Z.; Wang, X.; Niu, M. Automatic Classification and Coding of Prefabricated Components Using IFC and the Random Forest Algorithm. *Buildings* **2022**, *12*, 688. [CrossRef]
20. Xiang, X.; Ma, Z.; Li, J. Automated Classification and Coding for Bim Components Based on Applicable Big Data. In *Proceedings of the Creative Construction Conference 2023*; Budapest University of Technology and Economics: Keszthely, Hungary, 2023; pp. 7–20. [CrossRef]
21. Xiao, M.; Chao, Z.; Coelho, R.F.; Tian, S. Investigation of Classification and Anomalies Based on Machine Learning Methods Applied to Large Scale Building Information Modeling. *Appl. Sci.* **2022**, *12*, 6382. [CrossRef]
22. Petrochenko, M.; Nedviga, P.; Kukina, A.; Strelets, K.; Sherstyuk, V. Machine learning model for the BIM classification in IFC format. *Mag. Civ. Eng.* **2024**, *17*, 12602. [CrossRef]
23. Pan, Y.; Yu, H.; Zhao, T.; Sun, J. Can Language Models Enable In-Context Database? *arXiv* **2024**, arXiv:2411.01807. [CrossRef]
24. Xie, Q.; Zhou, X.; Wang, J.; Gao, X.; Chen, X.; Chun, L. Matching Real-World Facilities to Building Information Modeling Data Using Natural Language Processing. *IEEE Access* **2019**, *7*, 119465–119475. [CrossRef]
25. 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]
26. Shen, Q.; Wu, S.; Deng, Y.; Deng, H.; Cheng, J.C.P. BIM-Based Dynamic Construction Safety Rule Checking Using Ontology and Natural Language Processing. *Buildings* **2022**, *12*, 564. [CrossRef]
27. Nabavi, A.; Ramaji, I.; Sadeghi, N. An Artificial Intelligence-Based Framework for Automated Information Inquiry from Building Information Models Using Natural Language Processing and Ontology. In *Computing in Civil Engineering 2023*; American Society of Civil Engineers: Corvallis, OR, USA, 2024; pp. 381–389. [CrossRef]
28. Wang, J.; Gao, X.; Zhou, X.; Xie, Q. Multi-scale Information Retrieval for BIM using Hierarchical Structure Modelling and Natural Language Processing. *J. Inf. Technol. Constr.* **2021**, *26*, 409–426. [CrossRef]
29. Forth, K.; Abualdenien, J.; Borrmann, A. Calculation of embodied GHG emissions in early building design stages using BIM and NLP-based semantic model healing. *Energy Build.* **2023**, *284*, 112837. [CrossRef]
30. Oraskari, J.; Törmä, S.; Yalcinkaya, M.; Singh, V.; Beetz, J. Sharing Linked Building Data in a Peer-to-Peer Network: ifcOWL Meets InterPlanetary File System. *J. Comput. Civ. Eng.* **2024**, *38*, 04023037. [CrossRef]
31. Schulz, O.; Oraskari, J.; Beetz, J. bcfOWL: A BIM collaboration ontology. In Proceedings of the 9th Linked Data in Architecture and Construction Workshop—LDAC2021, Luxembourg, 11–13 October 2021.
32. Li, X.; Yang, D.; Yuan, J.; Donkers, A.; Liu, X. BIM-enabled semantic web for automated safety checks in subway construction. *Autom. Constr.* **2022**, *141*, 104454. [CrossRef]

33. Sobhkhiz, S.; Taghaddos, H.; Rezvani, M.; Ramezanianpour, A.M. Utilization of semantic web technologies to improve BIM-LCA applications. *Autom. Constr.* **2021**, *130*, 103842. [CrossRef]
34. Zhao, T. Semantic Representation of Road Infrastructure Information. Ph.D. Thesis, Marquette University, Milwaukee, WI, USA, 2022.
35. Shi, J.; Pan, Z.; Jiang, L.; Zhai, X. An ontology-based methodology to establish city information model of digital twin city by merging BIM, GIS and IoT. *Adv. Eng. Inform.* **2023**, *57*, 102114. [CrossRef]
36. Niknam, M.; Zhao, T.; Karshenas, S. A Semantic Model for Pavement Management Data. In *Computing in Civil Engineering 2021*; American Society of Civil Engineers: Orlando, FL, USA, 2022; pp. 729–736. [CrossRef]
37. Aminu, E.F.; Oyefolahan, I.O.; Abdullahi, M.B.; Salaudeen, M.T. A Review on Ontology Development Methodologies for Developing Ontological Knowledge Representation Systems for various Domains. *Int. J. Inf. Eng. Electron. Bus.* **2020**, *12*, 28–39. [CrossRef]
38. Alharbi, R.; Tamma, V.; Grasso, F. Characterising the Gap Between Theory and Practice of Ontology Reuse. In Proceedings of the 11th Knowledge Capture Conference, Virtual Event, 2–3 December 2021; pp. 217–224. [CrossRef]
39. Donkers, A.; Yang, D.; De Vries, B.; Baken, N. Semantic Web Technologies for Indoor Environmental Quality: A Review and Ontology Design. *Buildings* **2022**, *12*, 1522. [CrossRef]
40. Pauen, N.; Schlütter, D.; Frisch, J.; van Treeck, C. TUBES System Ontology: Digitalization of Building Service Systems. In Proceedings of the 9th Linked Data in Architecture and Construction Workshop (LDAC 2021), Luxembourg, 11–13 October 2021; pp. 43–54.

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 not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.