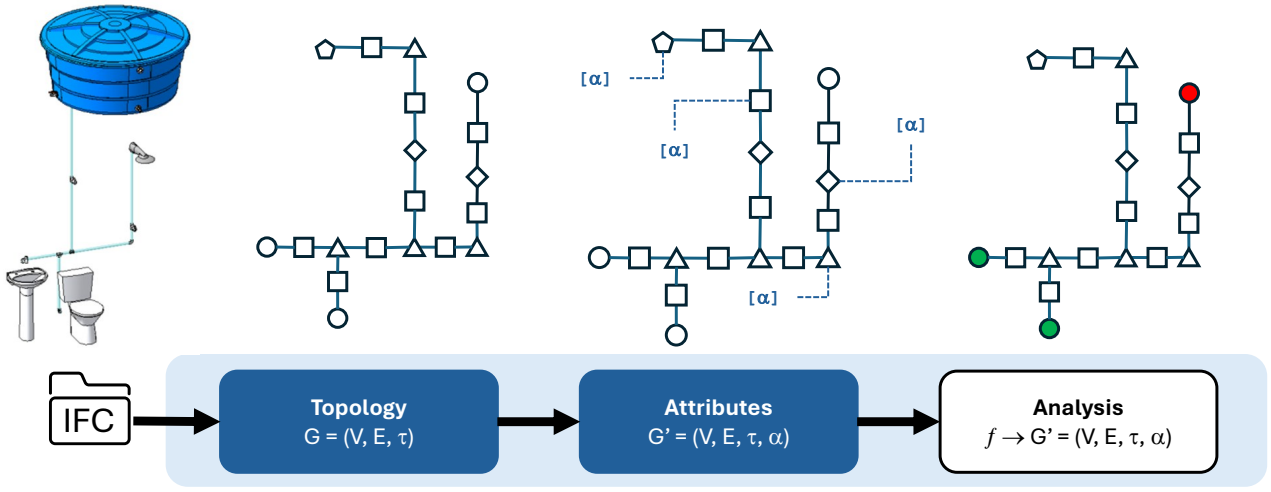
Graphical Abstract

IFC-based graph representations for cross-domain engineering analysis

A.B. Müller, F. R. Corrêa



Framework

Design Science Research (DSR)

Demonstration	Evaluation
Synthetic model	Commercial software, hand calculation
Real-world residential project	Commercial software → Máx. |Δ| = 0.20 mWG

Highlights

IFC-based graph representations for cross-domain engineering analysis

A.B. Müller, F. R. Corrêa

- Three-layer IFC framework extracts graphs from BIM for engineering analysis
- Port-based topology extraction is deterministic and discipline-independent
- Proposed taxonomy classifies topology, attribute, and analysis complexity types
- Evaluated on real residence project against commercial software (max. $|\Delta| = 0.20\,\mathrm{mWG}$)
- Survey ($n = 83$) confirms BIM-analysis disconnect among practitioners

IFC-based graph representations for cross-domain engineering analysis

A.B. Müller[a,*], F. R. Corrêa[a]

[a]*Department of Construction Engineering, Escola Politécnica da Universidade de São Paulo (Poli-USP), Av. Prof. Almeida Prado, 83, São Paulo, 05508-070, SP, Brazil*

ARTICLE INFO

Keywords:
Building information modelling (BIM)
Industry foundation classes (IFC)
Computer aided engineering
Graph-based analysis
Plumbing systems

ABSTRACT

Building information models encode topology and properties needed for engineering calculation, yet existing workflows require each discipline to independently reconstruct analytical representations. This paper proposes a three-layer framework for deriving graph-based engineering analysis directly from IFC models: (i) a Topology Layer extracts system connectivity from port-based relational structures; (ii) an Attributes Layer maps geometric and semantic properties to analysis input parameters; and (iii) an Analysis Layer applies domain-specific computations to the enriched graph. The framework draws an analogy with Automated Rule Checking, separating model preparation from rule execution. Classification taxonomies for topology supply, attribute supply, and analysis complexity characterise applicability across building service disciplines. The framework is instantiated through *ifc-hydro*, an open-source implementation for drinking-water analysis, and demonstrated on synthetic and real-world models evaluated against commercial software and hand calculation, with a maximum deviation of 0.20 mWG, attributable to externally supplied parameter differences, particularly internal diameter values and rounding conventions.

1. Introduction

The increasing adoption of Building Information Modelling (BIM) has substantially expanded the availability of structured digital information throughout the design and delivery of buildings and infrastructure [1, 2, 3]. Leveraging these available assets, there has been growing interest in using BIM data to automate engineering analyses, reducing manual re-modelling, and improving consistency between design intent and computational evaluation [4, 5, 6]. Despite this interest, BIM-based workflows still treat analytical models as secondary artifacts (Section 2.1), manually reconstructed from design models and tailored to specific tools or disciplines.

A broad class of engineering analyses share this data structure: hydraulic pressure verification [7], duct pressure drop verification in HVAC systems, and voltage drop calculation in electrical circuits all involve cumulative scalar computation along network paths from source to terminal.

In each, a discrete network of interconnected components (a graph) is the abstraction which allows the analysis of the behavior of the system. Geometry provides parameters (lengths, elevations, cross-sectional areas), but the analysis itself operates on nodes, connections, and paths. Yet current workflows require each discipline to independently reconstruct this network representation from the design model [8, 9, 10], a process that is error-prone, difficult to audit, and not reusable across domains.

The Industry Foundation Classes (IFC) standard, widely adopted as the open format for BIM data exchange, already encodes much of the information required for these analyses: geometric representations, semantic classifications of building service components, and relational structures that explicitly describe physical connectivity between them.

Yet the dominant mode of IFC use in practice remains data exchange, transferring model geometry and metadata between authoring and coordination tools, rather than transforming model content into representations directly suitable for engineering computation [11, 12, 13]. This constitutes the core limitation addressed in this work: IFC is treated as an exchange format rather than as an analytical substrate with embedded topology and engineering semantics.

Existing approaches to extracting analytical representations from BIM models have largely operated from a geometric perspective, reconstructing topology from three-dimensional solids rather than from the relational structures already present in the schema [14, 15]. Other works have derived connectivity representations through rule-based inference [10, 7], but remain domain-specific or tightly coupled to particular extraction strategies.

This paper proposes a framework for deriving graph-based engineering analysis directly from IFC models, targeting the native relational structures of the schema rather than geometric inference. The framework is organized into three modular layers:

1. A **Topology Layer** that extracts system connectivity from IFC port-based relationships to construct an undirected graph representing the physical network;
2. An **Attributes Layer** that retrieves geometric and semantic properties from IFC entities and maps them to the input parameters required by the target analysis;
3. An **Analysis Layer** that applies domain-specific rules to the enriched graph, computes engineering quantities and produces conformance results.

The first two layers are discipline-independent: the IFC port-based connectivity paradigm is consistent across all MEP disciplines, and the property extraction mechanisms operate on shared schema structures.

*Corresponding author:
andrebmuller@usp.br (A.B. Müller); fabiano.correa@usp.br (F.R. Corrêa)
ORCID(s): 0009-0006-4833-2858 (A.B. Müller); 0000-0001-9742-3971 (F.R. Corrêa)

This separation draws a structural analogy with Automated Rule Checking (ARC), where model preparation is distinguished from rule execution [16]; in the proposed framework, topology extraction and attribute enrichment serve as model preparation, and domain-specific computation serves as rule execution.

The framework is developed following a Design Science Research (DSR) approach [17], in which prescriptive knowledge is produced through the construction and evaluation of an artifact. The framework's applicability is demonstrated through *ifc-hydro* [18], an open-source implementation for the automated analysis of drinking-water systems conforming to Brazilian standards.

The case study is chosen because it exercises all three framework layers under realistic conditions: explicit IFC connectivity constructs define the system topology, geometric and physical properties parametrize the hydraulic equations, and the analysis module implements flow estimation, head loss calculation, and available pressure verification.

The framework is tested on both a synthetic model and a real-world Single Family Residence (SFR), with results validated against commercial software and manual calculation. While the demonstration is domain-specific, the framework architecture is designed for generalisability: Section 3.6 maps the three layers across plumbing, HVAC (Heating, Ventilation, and Air Conditioning), electrical, and structural disciplines, identifying the topology, attribute, and analysis classes that each domain requires.

The contributions of this work are threefold:

- The proposed framework enables a general approach to handling interoperability between design and engineering analysis. Contrasting previous ad hoc implementations, this graph-based framework allows engineering and performance calculations from open building information models without complex geometric reconstruction or spatial inference.
- A three-layer modular structure is proposed (topology, attributes, analysis) in which the first two layers are discipline-independent and reusable, while only the analysis layer requires domain-specific instantiation.
- The applicability of the framework is demonstrated in a drinking-water systems implementation, *ifc-hydro*. The graph-based approach, tested in both a synthetic and a real-world Single Family Residence (SFR) model, correctly calculates flow, pressure drop and available pressure from the IFC.

The remainder of this paper is structured as follows. Section 2 reviews the research background, including a brief characterization of current practice of engineering for drinking-water systems, the interoperability challenges, the IFC data schema, query languages for building information models, graph-based engineering analysis, and the parallel with ARC. Section 3 presents the framework, formalising the three layers and their generalisation across disciplines. Section 4 describes the *ifc-hydro* case study, reporting results and discussing framework behavior. Section 5 concludes with a synthesis of contributions, limitations, and directions for future work.

2. Research background

2.1. Current practice characterization

To complement the literature analysis with empirical evidence from current practice, a web-based questionnaire was distributed to practitioners involved in the design of drinking-water systems in Brazil. The instrument covered five themes: professional profile, calculation methods, modelling and documentation tools, BIM uses and interoperability, and workflow bottlenecks. It was distributed through professional networks and university contacts between January and February of 2026, yielding 83 valid responses. Given the absence of a census of plumbing system designers in Brazil, the population is estimated at roughly 12,000–20,000 active practitioners, derived from the approximately 400,000 registered civil engineers in the CONFEA/CREA system [19], of whom 3–5% are estimated to work in MEP design, with plumbing systems as a primary or regular activity, consistent with the multi-disciplinary profile reported by [20]. The sample constitutes a convenience sample and the results are reported descriptively as a practice characterization rather than as population-level estimates.

The respondent profile skews toward early- and mid-career practitioners: 30% reported fewer than 5 years of experience, 42% between 5 and 10 years, and 24% between 10 and 20 years. The residential sector was the most common area of activity, cited by over 80% of respondents either alone or in combination with other sectors. This profile is consistent with the broader practitioner demographics reported by AltoQi [20], where nearly half of respondents fell in the 25–34 age bracket.

Three findings from the survey are directly relevant to the framework proposed in this paper:

First, BIM adoption is high but disconnected from engineering analysis. Among respondents, 77% use Autodesk Revit as a design tool and 69% routinely deliver IFC models as project outputs. BIM adoption for documentation (88%), quantity extraction (87%), and 3D coordination (81%) is widespread, yet only 49% reported using BIM for engineering analysis (hydraulic calculations). The dominant calculation tool remains the spreadsheet: 88% of respondents use proprietary Excel spreadsheets for sizing, compared to 30% who use specialized software and 31% who use authoring tool plug-ins, these last two largely overlapping with the 49% who reported using BIM for engineering analysis. This confirms the fragmentation of the workflow described by [21] and [5]: practitioners model systems in BIM-capable software and export IFC, but perform engineering calculations in a separate and disconnected environment that requires manual data transfer.

Second, the same professional typically both models and calculates. 63% of respondents reported that the same person is responsible for both modelling and calculation, with an additional 23% indicating that this depends on the project.

This means the disconnect between BIM modelling and engineering analysis is not an organizational boundary between teams but a tool-level gap: the same practitioner must switch between a modelling environment and a calculation environment, re-entering geometric and topological data that already exists in the model.

Third, rework from data re-entry is the second-most cited workflow bottleneck. Respondents were asked to identify their main bottlenecks from a predefined list. Late design changes led the ranking (70% of respondents), followed by rework from manual data re-entry (61%), schedule pressure (53%), and incomplete BIM libraries/objects (37%). Interoperability failures (export/import problems) and lack of property standardization were cited by 23% and 21%, respectively. The prevalence of data re-entry rework, in a context where most respondents already produce BIM models and IFC deliverables, underscores the practical demand for automated pathways from IFC models to engineering analysis, which comprises a central objective of the proposed framework.

Regarding information exchange quality, respondents rated the interoperability experience on a 5-point Likert scale (1 = very problematic, 5 = flows well), with an option to indicate which information exchange item the respondent didn't perform in their professional practice. IFC export from authoring software received the most favorable rating (mean 4.1, s.d. 1.1, n = 73), suggesting that file generation is not the primary barrier. Property standardization (mean 3.2, s.d. 1.2, n = 76) and import into calculation/coordination tools (mean 3.6, s.d. 1.1, n = 66) received lower scores, indicating that the challenge lies not in producing IFC files but in ensuring that the information they contain is structured and complete enough for downstream analysis, a finding consistent with the attribute supply limitations discussed in Section 2.3.

These survey findings are country-specific and subject to the limitations inherent in convenience sampling: the respondents are likely more digitally engaged and BIM-literate than the broader population of plumbing system designers, given the distribution channels used. The reported rates of BIM adoption and IFC delivery may therefore overestimate national averages. However, this bias strengthens the motivation for developing the framework: if practitioners who already use BIM and export IFC still rely predominantly on spreadsheets for engineering calculation and report data re-entry as a major bottleneck, the gap between BIM and engineering analysis persists even among the most digitally mature segment of the profession.

2.2. Interoperability in engineering analysis

The integration of Computer Aided Design (CAD) with Computer Aided Engineering (CAE) tools has been a long-standing objective in the Architecture, Engineering and Construction (AEC) industry. Early integration efforts focused primarily on transferring geometric data from drafting environments to specialized analytical solvers. As the industry transitioned to BIM, this goal expanded significantly. BIM promises a unified digital representation capable of supporting semantically enriched multidisciplinary workflows across the building lifecycle, from design to operation [1].

Two broad strategies have emerged to bridge BIM authoring and engineering analysis. The first relies on proprietary Application Programming Interfaces (APIs) exposed by specific authoring platforms, such as the Revit API or visual programming environments like Dynamo, to extract model data and feed it into external calculation routines [5, 22]. This is, in fact, the dominant commercial model: most specialized MEP analysis tools integrate with authoring platforms through proprietary connectors or embedded plug-in architectures. These approaches can achieve high data fidelity within their native ecosystem, since they access the authoring tool's internal object model directly. However, they produce workflows that are structurally coupled to a single vendor's platform: a Dynamo-based hydraulic sizing script cannot be transferred to an ArchiCAD environment without complete re-implementation, and the resulting analyses are opaque to participants outside that vendor's ecosystem.

The second strategy operates through open, vendor-neutral exchange standards, principally the Industry Foundation Classes (IFC), developed by buildingSMART International and published as ISO 16739 [23]. IFC-based workflows decouple the analytical pipeline from any specific authoring tool, enabling the same framework to process models regardless of their origin. This openness, however, introduces its own challenges: semantic information may be lost or inconsistently mapped during export [24, 25], different authoring tools produce structurally divergent topology in IFC files for equivalent systems [26], or the degree to which models are populated with properties required for engineering computation varies substantially in practice [27]. These challenges are particularly acute in vertical interoperability (transitions from design to analysis where semantic mismatches run deeper than in horizontal exchanges between modelling tools), and even established commercial implementations exhibit element disconnections and incomplete property transfer [28]. The IFC schema and its particularities are further discussed in Section 2.3.

Achieving seamless interoperability thus remains a persistent challenge.

The remainder of this section examines the state of BIM-to-analysis interoperability across disciplines (Section 2.2.1) and specifically for plumbing systems as a case study (Section 2.2.2).

A review of the studied references with their corresponding interoperability strategy and key limitations is summarized in Table 1.

2.2.1. Interoperability across engineering disciplines

A review of recent literature reveals that interoperability solutions are predominantly developed within isolated disciplinary silos.

An examination of interoperability from BIM to analysis across HVAC [29, 6, 30], energy modelling [31, 32, 33],

Table 1
Non-exhaustive literature synthesis of BIM-to-analysis interoperability.

Discipline	References	Interoperability strategy	Key limitation
HVAC	[29]; [6]; [30]	gbXML- and IFC-based	Incomplete semantic transfer
Energy	[31]; [32]; [33]	gbXML- and IFC-based	Insufficient IFC schema coverage
Electrical	[22]; [34]	IFC-based / Proprietary (API)	Platform-dependent extraction
Building Automation Systems	[35]; [36]	IFC-based	Insufficient IFC schema coverage
Plumbing	[5]; [21]; *ifc-hydro* (this work)	Proprietary (Dynamo) / Custom data model / IFC-based	Independent data model construction (prior work) [5, 21]
Fire Safety	[37]	IFC-based	Solver-specific geometry reconstruction
Structural	[24]	IFC-based	Incomplete connectivity transfer
Geotechnical	[38]	IFC-based	Insufficient IFC schema coverage
BIM–GIS	[39]	IFC-based	Geometric representation mismatch

electrical [22, 34], building automation [35, 36], plumbing [5, 21], fire safety [37], structural [24], and geotechnical disciplines [38], and BIM and GIS integration [39] reveals a consistent pattern: each interoperability solution constitutes a vertical pipeline built independently for a specific analytical goal.

For instance, approaches in HVAC and energy modelling focus on extracting specific properties for energy simulation and automating analysis [6, 29], yet even workflows that define dedicated Model View Definitions (MVDs) for IFC-based simulation report persistent data loss in material properties, HVAC connections, and space boundary relationships during export from commercial authoring tools [33, 32], a gap that recent open-source automation frameworks can mitigate through template-based enrichment but not eliminate [31]. Critically, these energy-focused pipelines derive their analytical structure from spatial constructs such as space boundaries and thermal zones rather than from the port-based relational structures already embedded in the IFC schema. Structural workflows struggle with geometric interpretation and element disconnections despite neutral links [24]. Building automation and geotechnical engineering integrations frequently rely on custom MVDs or extensive schema extensions to bridge semantic gaps [35, 38]. Similarly, electrical systems interoperability progresses through either conceptual frameworks for specific simulations such as photovoltaic modelling [34] or proprietary application programming interfaces for automated load estimation [22]. Fire safety analysis presents a parallel challenge, where IFC-to-simulation pipelines require substantial geometric restructuring to conform building information to the rectilinear mesh constraints of the target solver [37].

2.2.2. Plumbing systems as a case study

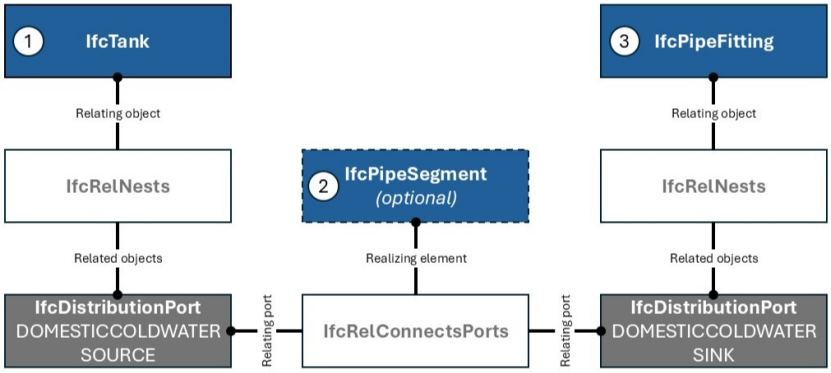
Prior to the present work, no independent studies were identified that apply IFC-based interoperability to plumbing system analysis. The two cases here described further illustrate the fragmentation of interoperability pipelines and serve as the primary baseline for the framework proposed in this study.

In the context of domestic drinking water and sewerage systems, Atencio et al. [5] developed a parametric workflow integrating regulatory calculation procedures into a partially automated design process in a closed interoperability environment. Their methodology combines visual programming and algorithms based on external spreadsheets to automate pipe sizing and compliance checks according to national Chilean standards. While this workflow improves efficiency relative to conventional calculation methods, it relies entirely on external computation loops and data is extracted directly from the model and its geometric representation.

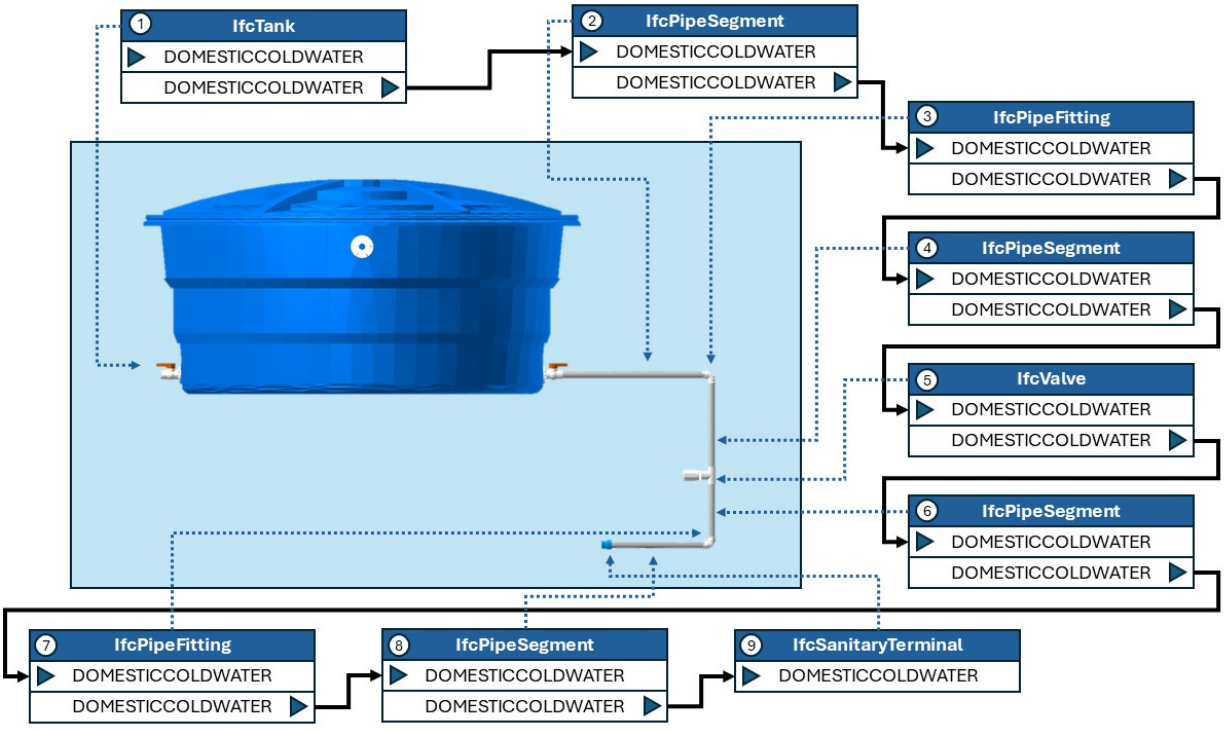
An earlier and more comprehensive effort is the LicA system [21], which performs automated code-checking of domestic water networks against Portuguese regulations using a Breadth First Search algorithm over a node-and-segment graph. However, LicA operates on a proprietary SQL-based data model developed independently of IFC; a retrospective schema comparison found that only 61% of its entities had direct IFC counterparts, with the remainder lacking required property sets or having no correspondence at all. This gap illustrates the recurring pattern whereby domain-specific analytical tools construct parallel data models rather than leveraging the relational structures already embedded in the IFC schema.

2.3. IFC as a semantic data model

The Industry Foundation Classes (IFC) standard provides an open vendor-neutral data schema for representing building information models [40]. Unlike geometry-only



(a) Detailed IFC Port Connection with optional realizing element. Adapted from [41].



(b) Summarized IFC Port Connection with optional realizing elements. Adapted from [41].

Figure 1: IFC Port Connectivity concept.

exchange formats, IFC encodes buildings as structured collections of semantically typed objects (walls, slabs, pipes, ducts, terminals, and so on) together with their attributes, material assignments, spatial containment, and mutual relationships [27]. This semantic richness is what makes IFC the best candidate data source not only for visualization and coordination, but also for automated engineering analysis.

The IFC schema is organised in a four-layer architecture: a Resource Layer providing low-level data types (geometry, materials, quantities, units); a Core Layer centred on `IfcRoot` and its principal extensions for products, processes, and controls; an Interoperability Layer containing shared element classes used across multiple disciplines; and a Domain Layer with specialised entities for architecture, structural engineering, HVAC, plumbing and fire protection, electrical distribution, and other domains [41]. This layered design draws on object-oriented programming principles (class inheritance, polymorphism, and composition) formalised through the EXPRESS data definition language specified in ISO 10303-11 [42].

For engineering analysis of building service systems, two structural features of the IFC schema are particularly relevant: the object definition inheritance hierarchy and the port connectivity concept (Represented in Figure 1).

The entity inheritance hierarchy organises building service components into functional superclasses under `IfcDistributionElement`, which branches into `IfcDistributionFlowElement` and `IfcDistributionControlElement`. Under `IfcDistributionFlowElement` plumbing, HVAC, and electrical components are further distinguished as `IfcFlowSegment` (pipes, ducts, cable segments), `IfcFlowFitting` (elbows, tees, reducers), `IfcFlowTerminal` (sanitary terminals, air terminals), `IfcFlowController` (valves, dampers), `IfcFlowStorageDevice` (tanks), `IfcFlowMovingDevice` (pumps, fans), `IfcEnergyConversionDevice` (transformers, boilers, chillers), `IfcFlowTreatmentDevice` (filters, duct silencers) and `IfcDistributionChamberElement` (sump, trenches, manholes).

Each of these is further branched into subtypes (`IfcFlowFitting` has a subtype `IfcPipeFitting`) and carries a `PredefinedType` attribute that encodes its functional role, for instance, `BEND`, `JUNCTION`, or `TRANSITION` for `IfcPipeFitting`, providing machine-readable semantic classification without relying on naming conventions or free-text descriptions. This classification is consistent across MEP disciplines: a duct elbow and a pipe elbow share the same superclass and predefined-type mechanism, differing only in their domain-specific subtypes.

The port connectivity concept is equally critical. IFC does not encode relationships as simple foreign-key references between entities; instead, it represents relationships as first-class objects, each carrying its own attributes and semantics [28]. Physical connectivity between building service components is encoded through two complementary relationship types. `IfcRelNests` associates a set of `IfcDistributionPort` objects with their parent component, establishing which connection points belong to which physical element. `IfcRelConnectsPorts` then links pairs of ports on different components, explicitly declaring that two elements are physically joined. This two-level mechanism: port-to-component association plus port-to-port connection, provides a complete, machine-readable representation of system topology that is independent of geometric proximity or spatial inference.

These interconnected components are then logically aggregated into functional networks using the `IfcDistributionSystem` entity and the `IfcRelAssignsToGroup` relationship. Rather than utilizing a deep hierarchy of subclasses, `IfcDistributionSystem` relies on a `PredefinedType` attribute to encode its specific domain, such as `DOMESTICCOLDWATER`, `DOMESTICHOTWATER`, or `AIRCONDITIONING`, while its sole subtype, `IfcDistributionCircuit` represents localized sub-networks

This distinction between explicit and implicit topology in IFC is consequential for automated analysis. Explicit topology, encoded through the port-based relationships described above, can be traversed algorithmically to reconstruct the connectivity graph of a building service network without any geometric computation. Implicit topology, by contrast, must be inferred from geometry. Previous work has employed geometric proximity inference [10, 9] to recover connectivity from models where relational data was absent or unreliable. While effective as a fallback, geometric inference can be computationally expensive in large models, sensitive to tolerance parameters, and prone to false connections in densely routed MEP systems.

Beyond topology, the IFC schema provides multiple pathways for associating quantitative properties with building elements. Direct attributes are defined within entity specifications; standard property sets (`Pset_PipeSegmentCommon`, `Pset_ValveCommon`, etc.) collect discipline-specific parameters as key–value pairs; and geometric representations, defined by the `IfcShapeRepresentation` entity, where an inherited attribute (`RepresentationType`) describes the geometric model used for the shape representation (most commonly SweptSolid, CSG, or Brep), from which engineering parameters such as cross sectional areas can be derived. However, the degree to which these properties are populated in practice varies substantially.

Solihin et al. [43] formalised this quality concern through a classification of IFC data quality dimensions: completeness, correctness, meaningfulness, and conformance, and demonstrated that even models exchanged between certified applications frequently exhibit missing properties, inconsistent entity typing, or geometry integrity issues.

Noardo et al. [27] inspecting IFC models produced by practitioners, found that few of the surveyed models achieved sufficient semantic completeness across all assessed dimensions (entity consistency, material assignment, attribute population, and spatial structure) to support automated processing without manual intervention.

These findings underscore a practical tension in IFC-based automation: while the schema provides expressive structures for encoding the information required by engineering analysis, the data actually present in exported models is often incomplete or inconsistent. A recent study [40] demonstrated this explicitly for the `IfcProject` representation context, showing that default export configurations from commercial software fail to populate several semantically relevant attributes despite their availability in the schema, a gap they attributed jointly to modeller omissions and to absent fields in the software interface.

For the framework proposed in this paper, this tension manifests as a distinction between properties that can be extracted directly from the IFC model and those that require derivation from geometric representations, contextual inference from adjacent elements, or supplementation from external lookup tables. Section 3 formalises this distinction through a classification of topology and attribute supply types.

2.4. Query languages for building information models

Retrieving specific subsets of information from large, highly structured building models is a well-recognised challenge in BIM research. As models grow in complexity and the number of contributing disciplines increases, the need for formal mechanisms to select, filter, and extract relevant data becomes critical. Over the past two decades, several query

language approaches have been developed to address this need, ranging from adaptations of general-purpose database languages to domain-specific formalisms designed explicitly for the IFC data model.

General-purpose query languages such as SQL, XQuery, and SPARQL can be applied to IFC data once the schema is mapped to their respective data models [44]. However, each mapping introduces non-trivial transformations. SQL requires the IFC schema to be flattened into relational tables, a process complicated by the deep inheritance hierarchies and objectified relationships characteristic of EXPRESS-based models [45]. XQuery operates on ifcXML representations and can traverse hierarchical structures, but the verbose nesting of IFC's XML serialisation results in queries that are difficult to compose and interpret. SPARQL, designed for RDF triple stores, can traverse the graph structure of IFC data naturally, but requires an intermediate conversion to RDF or OWL representations [45, 44]. A common limitation across these languages is that they are not tailored to the IFC domain; users must possess detailed knowledge of the underlying schema mapping in addition to the query language syntax itself.

To lower this barrier, domain-specific query languages have been proposed. The Partial Model Query Language (PMQL), introduced by Adachi [46], was among the earliest, offering select, update, and delete operations on IFC model servers with support for recursive graph traversal through its cascade operator. More recently, BIMQL [45] was developed as an open, platform-independent language for IFC model repositories. BIMQL introduces domain-specific shortcuts that abstract away the complex navigation of objectified relationships, for example, retrieving properties that are indirectly linked to entities through multiple IFC relationship instances. Its plugin mechanism supports extensibility, including natural-language keyword resolution through the buildingSMART Data Dictionary [47], storey-based filtering, and custom operators.

A parallel line of research has pursued spatial query languages that extend the filtering capabilities to geometric and topological predicates. Borrmann and Rank [15, 48] and Borrmann et al. [49] developed a 3D spatial query language providing metric, directional, and topological operators. Daum and Borrmann [14] subsequently improved the computational performance of topological predicates by introducing boundary-representation-based algorithms, enabling queries against models containing thousands of elements. These operators were integrated into the Query Language for Building Information Models (QL4BIM), available in both textual and visual notations [44]. The visual variant, vQL4BIM, employs a data-flow programming paradigm based on relational algebra operators, and was shown to be more accessible to non-programmer end users such as architects and engineers than equivalent textual queries in SQL or tQL4BIM.

More recently, graph database technologies have been applied to IFC querying. These applications will be discussed in detail in Section 2.5, along with other important developments of graph data formats in building information modelling and engineering.

Despite their diversity in paradigm and expressiveness, these query languages are fundamentally retrieval-oriented. Their purpose is to answer questions about the model: selecting objects that satisfy certain type, attribute, spatial, or relational conditions, and to return subsets of existing model data. The output of a query is a filtered view of the original information. Engineering analysis, however, requires a fundamentally different operation. The task is to transform the model into a computational representation.

The concepts defined by query languages for BIM are used extensively in the proposed frameworks second layer, which extracts the MEP system's attributes and is further described in Section 3.4.

2.5. Graph-based engineering analysis

Graph representations have a long history in the built environment, though their application to BIM-based engineering analysis is still in its onset.

In architectural research, space syntax [50] established that graph-theoretic measures (connectivity, integration, and depth) could capture non-trivial properties of spatial configurations relevant to circulation, wayfinding, and social interaction. By abstracting rooms as nodes and their adjacency as edges, space syntax enabled quantitative comparison of spatial layouts without requiring detailed geometric computation.

Within the BIM domain, efforts to construct graph representations from IFC models have focused predominantly on topological queries and information retrieval. Recently, graph databases have gained traction as a storage and querying paradigm for BIM data. Ismail et al. [51] extracted building knowledge from IFC models into a Neo4j graph database, demonstrating performance advantages over relational databases for traversal-intensive queries.

Zhu et al. [52] developed GQL4BIM, a graph query tool that leverages the Cypher query language to retrieve complex relational patterns (including spatial structure, space boundaries, and accessibility) from IFC data represented as property graphs. Extending this line of investigation, Zhu et al. [53] systematically examined the internal structure of IFC-Graph and proposed a five-level hierarchy (IfcGraphLoD) that enables agile generation of graphs tailored to specific application needs.

These studies confirm that the IFC relationship structure through entities is well suited to graph representation and that such representations enable powerful information retrieval. However, the resulting graphs are designed for querying and data management rather than for quantitative engineering computation.

In MEP systems, several studies have demonstrated that graph representations can support both qualitative and quantitative reasoning about system behaviour. Wang et al. [54] proposed a framework for detecting logical relationships in MEP systems by extracting component connectivity from Revit via its API and storing it in a Neo4j graph database.

Their approach treats point-based instances (fittings, equipment) as nodes and curve-based instances (pipes, ducts) as edges, then applies subgraph matching against knowledge graphs encoding design rules. Betweenness centrality was used to identify critical paths within air-conditioning subsystems.

Xiao et al. [10] constructed logic chains of MEP systems from BIM models using both parametric and non-parametric spatial analysis. Their approach is notable for its dual extraction strategy: when IFC port connectivity is available, spatial relationships are extracted directly from the relational structure; when such information is absent, boundary-representation intersection algorithms serve as a fallback. The resulting connection tables are completed as directed graphs using identification rules, a domain-specific query language that formalises upstream and downstream relationships between MEP component types, and a breadth-first traversal assigns logic levels to each component. Applied to a real project with 763 components across 15 MEP subsystems, the approach achieved an average accuracy exceeding 80%. Importantly, the errors that did occur were predominantly attributable to incomplete or incorrect model data rather than to algorithmic limitations, underscoring the sensitivity of graph-based approaches to the quality of the underlying BIM model, a limitation that any IFC-dependent framework must address.

Han et al. [9] approached MEP graph construction from a geometric standpoint, proposing a directed representative graph model that combines spatial representation with flow direction. Their method converts MEP elements into triangular meshes, iteratively extracts boundary points to identify representative cross-section centres, and connects these centres to produce representative edges that capture the spatial trajectory of each element. Flow direction is then obtained from IFC port semantics, yielding a directed graph suitable for monitoring visualisation.

These different strategies for topology generation anticipate the topology supply classification formalised in Section 3.3.

Graph representations of building data are also increasingly being combined with machine learning techniques. Teclaw et al. [55] employed deep graph convolutional neural networks to validate IFC properties (such as stair riser counts, wall areas, and slab areas) by segmenting 3D mesh elements where each mesh face serves as a graph node with adjacency-based edges. Pan et al. [56] extended query languages to the operations phase by integrating large language models (LLMs) with graph-based digital twins, achieving high-accuracy natural language interaction with graph databases containing over 40,000 building nodes and room-level apartment layouts. Wang et al. [57] applied graph neural networks (GNNs) to room classification tasks on apartment layouts represented as graphs derived from BIM models. These works show that graph-based representations of building data can serve as substrates for data-driven inference.

2.6. Automated Rule Checking (ARC)

Automated Rule Checking (ARC) refers to the systematic application of computable rules to BIM models in order to verify compliance with regulatory, contractual, performance, or best-practice requirements. The output of ARC systems typically consists of structured indicators of compliance, non-compliance, or required revision. Rules may address universal building regulations, such as municipal codes, as well as discipline-specific guidelines or internal design standards [58].

ARC feasibility rests on BIM's semantically rich, object-oriented data environment that enables computational rule interpretation when sufficiently explicit and organized. Retrieving the relevant subsets of information from such models is a prerequisite for any automated check, drawing on the query language capabilities discussed in Section 2.4.

Methodologically, ARC workflows follow four stages: rule interpretation translates normative natural-language texts into formal logical expressions; model preparation ensures the required information is present and structured, often requiring derivation of implicit properties or construction of secondary representations; rule execution validates attributes, detects conflicts, or evaluates geometric and relational conditions against the encoded logic; and result reporting provides auditable outputs with explicit clause references to support traceability [58]. Complementing this procedural structure, Solihin and Eastman [16] classify rule complexity into four categories:

1. Rules that operate directly on explicit model data;
2. Rules that require simple attribute derivations (e.g., computed distances or orientations);
3. Rules that depend on extended data structures and more complex inferential reasoning;
4. Rules which are solution-oriented and require performance verification, often involving simulation or analytical demonstration of compliance.

This taxonomy is significant because it informs the selection of implementation strategies, rule languages, and supporting computational infrastructures.

In contrast to this coherent methodological and infrastructural ecosystem, interoperability for engineering analysis remains largely characterized by discipline-specific transformation pipelines. While numerous studies demonstrate the feasibility of extracting information from IFC models to support simulation, sizing, or performance evaluation, these efforts typically culminate in bespoke analytical adapters tailored to a single domain or toolchain. Unlike ARC, which has converged around structured rule interpretation frameworks, reusable formal languages and even practical implementations (Solibri Model Checker [59] and CORENET X [60]), engineering analysis workflows frequently reconstruct ad hoc analytical representations from scratch.

It is important to note, however, that ARC and engineering analysis occupy fundamentally different phases of the design process. ARC is a verification activity: it evaluates an existing model against encoded regulatory, contractual,

or best-practice constraints, typically at delivery or approval stages. Engineering analysis, by contrast, is a design-phase activity: it computes system performance to produce the design itself. While both rely on BIM's semantically rich data environment, ARC presupposes a completed model to check, whereas engineering analysis operates during design development, when the model is still being shaped by the results of computation.

Nonetheless, this boundary is not absolute: recent work has begun to extend ARC beyond design-stage compliance into construction and manufacturing, particularly in the context of off-site prefabrication. Solihin et al. [61], for instance, adapt semiconductor Manufacturing Rule Check (MRC) methodologies to assess prefabricated bathroom unit producibility directly from IFC models, demonstrating that rule-based verification can target downstream constructibility concerns as well.

The framework formalised in the following section takes inspiration from the field of ARC to address the methodological gap found in the interoperability in engineering analysis domain.

3. The framework

3.1. Research approach

The framework is developed following the Design Science Research (DSR) methodology [17], which structures the production of prescriptive knowledge through artifact construction and evaluation [62]. DSR is appropriate here because the research objective is not to describe an existing phenomenon but to design a reusable solution for a class of engineering problems: the automated extraction and analysis of network-structured building systems from IFC models.

The DSR process followed five stages. Awareness was informed by a practitioner survey (Section 2.1) and literature review (Sections 2.2–2.6), which identified the disconnect between BIM adoption and engineering analysis as a persistent workflow bottleneck. Suggestion produced the three-layer framework architecture described in the following subsections. Development resulted in an open-source prototype, *ifc-hydro* [18]. The demonstration applied the prototype to both a synthetic model (Section 4.2), evaluated against results of a commercial software and hand calculations, and a real-world project (Section 4.3), evaluated against a commercial software.

A single-domain case study is sufficient for demonstration under DSR since the framework's architecture ensures domain independence by construction. The Topology Layer operates on IFC connectivity constructs shared across all MEP disciplines; the Attributes Layer is parametrized by reusable property mappings; and the Analysis Layer is modular by design (Figure 2). Section 3.6 formalises this generalisability through a classification of topology, attribute, and analysis types across building service disciplines.

3.2. Framework overview

The three-layer architecture introduced in Section 1 is formalised here. The Topology Layer produces a graph, the Attributes Layer enriches this graph into an attributed graph by extracting and mapping component properties. The Analysis Layer operates a function on the attributed graph to compute domain-specific engineering quantities.

Figure 2 illustrates these three layers and its underlying graph model.

3.2.1. Graph model

The framework's mathematical substrate is a labelled graph in the sense of Harary [63].

A graph $G = (V, E)$ consists of a finite set V of vertices and a set $E \subseteq \{\{u, v\} : u, v \in V,\ u \neq v\}$ of edges, each an unordered pair of distinct vertices. The graph is simple: no self-loops and no multi-edges.

A connected graph with no cycles is a tree; a tree on n vertices has exactly $n-1$ edges. A rooted tree is a tree with a distinguished vertex $r \in V$ designated as the root. In a rooted tree, there is exactly one path from r to every other vertex, a property that is fundamental to the analyses addressed by this framework.

To encode engineering semantics, two functions are defined on G:

- A **vertex type function** $\tau : V \to T$, where T is a finite set of component types (e.g., {source, segment, fitting, controller, terminal}). This function partitions V into disjoint subsets: $V_{\text{source}} = \tau^{-1}(\text{source})$, $V_{\text{terminal}} = \tau^{-1}(\text{terminal})$, and $V_{\text{intermediate}} = V \setminus (V_{\text{source}} \cup V_{\text{terminal}})$.
- A **vertex attribute function** $\alpha : V \to \mathbb{R}^d$, which assigns to each vertex a vector of d engineering properties (e.g., length, internal diameter, elevation, loss coefficient, demand weight). The dimension d and the interpretation of each component are domain-specific.

The type function τ and the attribute function α are formalised by the Topology Layer (Section 3.3) and Attributes Layer (Section 3.4), respectively; their engineering interpretation is detailed in those sections.

The attributed graph $G' = (V, E, \tau, \alpha)$ produced is thus a simple, labelled, vertex-attributed graph.

When G' is a tree rooted at a source vertex $r \in V_{\text{source}}$ (`IfcTank` or `IfcBoiler` for example), the set of root-to-leaf paths $\mathcal{P} = \{P_1, P_2, \ldots, P_m\}$, where each P_j terminates at a distinct leaf in V_{terminal}, constitutes the complete set of paths on which engineering analysis (Section 3.5) operates.

3.3. Topology layer

The Topology Layer transforms the implicit connectivity information in an IFC model into an explicit graph $G = (V, E)$.

Its input is a parsed IFC file; its output is a simple undirected graph where vertices represent building service components and edges represent physical connections between them.

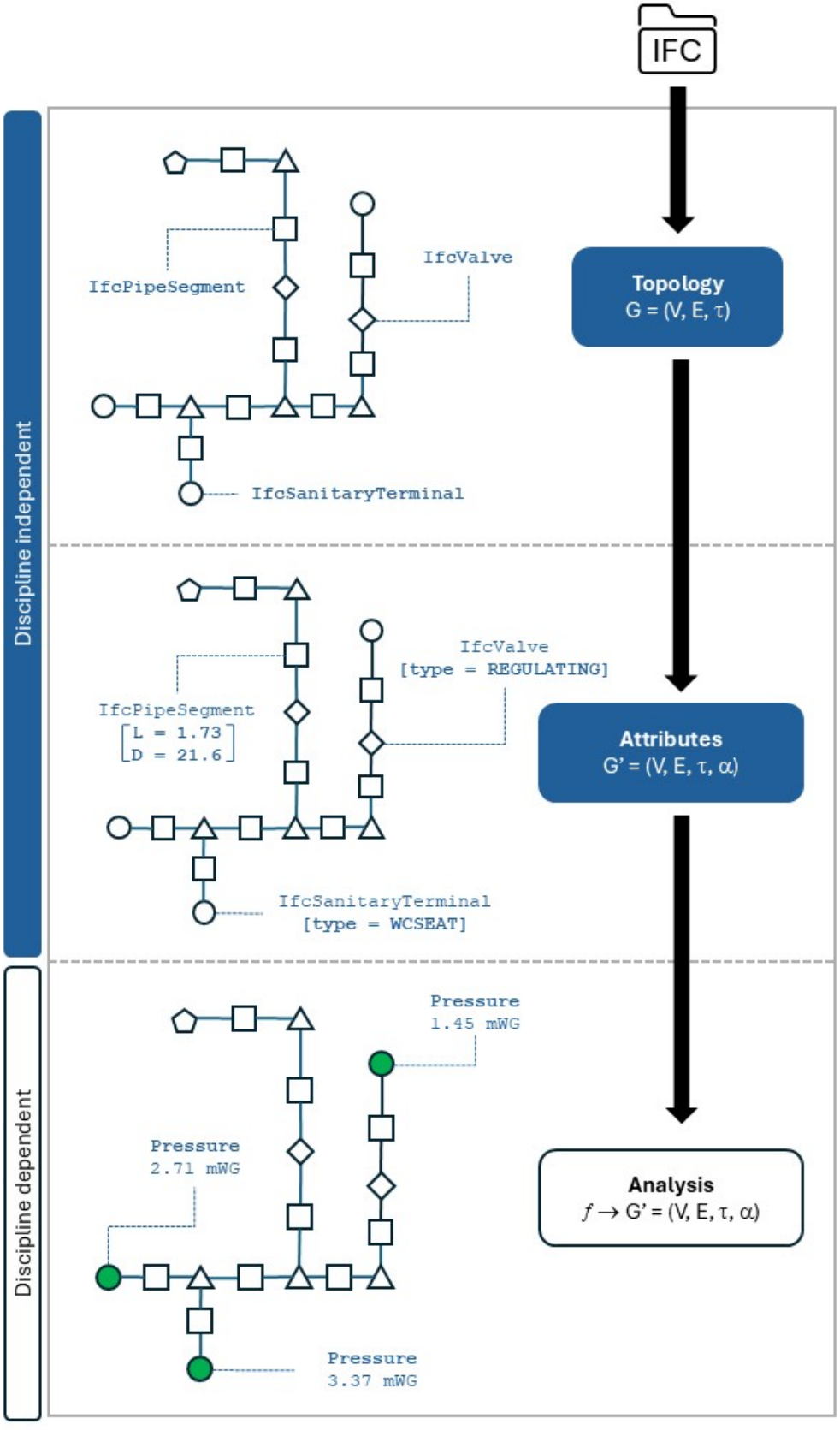


Figure 2: Overview of the graph-based framework demonstrating the sequential generation of topological (G) and attributed (G') graph representations, through the application of an analysis function from an original IFC model.

3.3.1. Topology supply classification

The extent to which system connectivity can be extracted directly from IFC relational structures varies across disciplines and model quality levels.

Three topology supply classes are proposed:

- **T1—Port-based connectivity.** Topology is fully encoded in IFC port relations (`IfcRelConnectsPorts` and `IfcRelNests`) and can be extracted directly without geometric inference. This is the case for MEP disciplines, where authoring tools generate port connectivity as part of system routing. When port relations are absent or incomplete due to export configuration errors or modelling omissions, geometric reconstruction becomes necessary as a fallback.
- **T2—Geometry-based reconstruction.** Topology must be inferred from geometric proximity or intersection analysis. The dual-extraction strategy of Xiao et al. [10] and the mesh-based approach of Han et al. [9], both discussed in Section 2.5, exemplify this class.
- **T3—Parallel analysis model.** Topology is encoded in a separate analysis model distinct from the physical element representations. Structural systems in IFC follow this paradigm, with connectivity defined through `IfcRelConnectsStructuralMember` for structural elements and `IfcRelConnectsStructuralActivity` for loads, rather than through the port-based mechanism used by MEP systems [41].

The instantiated case study (Section 4) operates under T1 conditions. The graph construction algorithm here exposed therefore relies exclusively on port-based relations, inheriting the determinism and computational efficiency that this approach affords.

3.3.2. Graph construction algorithm

In a T1 class, the Topology Layer resolves the two-level IFC port connectivity mechanism described in Section 2.3: port-to-component association via `IfcRelNests` and port-to-port connection via `IfcRelConnectsPorts`, into component-level adjacency through three phases:

1. **Indexing phase.** All instances of `IfcRelNests` are traversed to build a lookup table $M : P \to V$, mapping each distribution port $p \in P$ (the set of all `IfcDistributionPort` instances) to its parent component $v = M(p)$. This mapping is many-to-one, since multiple ports may belong to the same component.
2. **Resolution phase.** Each instance of `IfcRelConnectsPorts` defines a connection between two ports (p_1, p_2). For each such instance, the algorithm retrieves the corresponding components $v_1 = M(p_1)$ and $v_2 = M(p_2)$. If $v_1 \neq v_2$, an undirected edge $\{v_1, v_2\}$ is added to E. Duplicate edges are ignored by set semantics, ensuring that G remains a simple graph.
3. **Node classification.** Each vertex in G is assigned a type through τ according to its IFC entity class: source vertices (e.g., `IfcTank`), intermediate vertices (e.g., `IfcPipeSegment`, `IfcPipeFitting`, `IfcValve`), and terminal vertices (e.g., `IfcSanitaryTerminal`). The entity inheritance hierarchy described in Section 2.3 provides the semantic basis for this classification.

```
INPUT:  ifc_model
OUTPUT: graph G = (V, E)

BEGIN
  // Indexing phase
1.  M <- empty dictionary       // port -> component
2.  FOR each nest IN ifc_model.by_type("IfcRelNests"):
3.      FOR each port IN nest.RelatedObjects:
4.          M[port] <- nest.RelatingObject

  // Resolution phase
5.  E <- empty set
6.  FOR each conn IN ifc_model.by_type("IfcRelConnectsPorts"):
7.      v1 <- M[conn.RelatingPort]
8.      v2 <- M[conn.RelatedPort]
9.      IF v1 != v2:
10.         E <- E U { {v1, v2} }

11. V <- {v : v appears in some edge in E}
12. G <- Graph(V, E)
13. RETURN G
END
```

Figure 3: Pseudocode for the graph construction algorithm.

If the port-to-component mapping is not indexed (i.e., each lookup scans all `IfcRelNests` instances), the complexity becomes much higher, which is acceptable for small residential models but scales poorly for large MEP systems.

The *ifc-hydro* prototype implements this naive variant; for the case studies evaluated, topology extraction is not time consuming, but the indexed formulation described above, and shown in pseudocode in Figure 3, is required for scalability.

The algorithm produces an undirected graph reflecting physical connectivity. For tree-structured distribution systems, such as single-source drinking-water systems, the resulting graph is connected and acyclic, forming a tree.

Selecting a source vertex $r \in V_{\text{source}}$ as root induces a natural orientation, allowing interpretation of edges as directed from source to terminals. This induced orientation is used by the Analysis Layer (Section 3.5) but does not alter the underlying undirected graph structure.

For looped or meshed networks (common in HVAC, fire-protection or recirculating hot-water systems), the graph contains cycles and no unique root-to-leaf paths exist; the analysis implications of this are discussed in Section 3.5.2.

3.4. Attributes layer

The Attributes Layer enriches the topological graph with the physical and semantic properties required for engineering analysis. Its input is the graph G produced by the

Topology Layer and the parsed IFC model; its output is the attributed graph $G' = (V, E, \tau, \alpha)$, where τ assigns component types and α assigns engineering property vectors to each vertex.

3.4.1. Property extraction

For each vertex in G, the Attributes Layer traverses the internal structure of the corresponding IFC entity to populate α. The specific properties depend on the component type $\tau(v)$:

- **Linear elements**, under `IfcFlowSegment` (e.g., `IfcPipeSegment`, `IfcDuctSegment`, `IfcCableCarrierSegment`): length and cross-sectional dimensions, extracted from the geometric representation (`IfcExtrudedAreaSolid` for example), and material designation.
- **Fittings**, under `IfcFlowFitting` (e.g., `IfcPipeFitting`, `IfcDuctFitting`, `IfcCableCarrierFitting`): type classification via `PredefinedType` (e.g., `BEND`, `JUNCTION`, `TRANSITION`), loss coefficient, and inlet/outlet dimensions derived from the fitting itself or adjacent linear elements.
- **Terminals**, under `IfcFlowTerminal` (e.g., `IfcSanitaryTerminal`, `IfcOutlet`, `IfcLamp`): classification of fixture type (e.g., `SHOWER`, `WCSEAT`, `WASHHANDBASIN`) and associated demand parameters (e.g., design flow rate, fixture weight).
- **Controllers**, under `IfcFlowController` (e.g., `IfcValve`): type classification via `PredefinedType` (e.g., `ISOLATING`, `REGULATING`), loss coefficients, and dimensional parameters from the element itself or adjacent elements.
- **Source elements**, under `IfcFlowStorageDevice` (e.g., `IfcTank`): elevation and capacity parameters.

In the instantiated case study (Section 4), elevation data for all components is extracted from `IfcLocalPlacement` or using IfcOpenShell's embedded utilities, enabling the computation of height differences required for pressure calculations in gravity-fed systems or lift calculations in pumped systems.

3.4.2. Attribute supply classification

Not all properties required for engineering analysis are available in the IFC schema or reliably exported by authoring software. The completeness of α achievable without user intervention varies systematically across property types. This variation can be characterised through four attribute supply classes, analogous to the rule complexity taxonomy used for ARC [16]:

- **A1—Direct.** The required property exists as a standard IFC attribute or in a standard property set and is correctly exported. Example: fixture type from `PredefinedType` of `IfcSanitaryTerminal`.
- **A2.1—Derived from component geometry.** The required property can be computed from the geometric representation of the component itself. Example: pipe length from `IfcExtrudedAreaSolid` depth; cross-sectional diameter derived from the profile definition of an `IfcExtrudedAreaSolid`.
- **A2.2—Derived from spatial relationships.** The required property requires spatial intersection or containment analysis across multiple entities. Example: determining which HVAC terminals serve which thermal zones requires resolving spatial containment between `IfcFlowTerminal` and `IfcSpace`, which is not encoded in port-based connectivity.
- **A3—External input required.** The required property is absent from the IFC export and must be supplied from external lookup tables or direct user specification. Example: internal pipe diameter (only nominal/external diameter exported), fitting loss coefficients (defined in `Pset_PipeFittingCommon` but rarely populated by authoring software [7]), roughness coefficients, and electrical circuit parameters.

The proportion of each class in a given discipline determines the achievable degree of automation. Equivalently, the completeness of α obtainable without external input.

3.4.3. Schema-to-analysis mapping and IFC export limitations

The Attributes Layer implements a schema-to-analysis mapping that handles all four attribute supply classes through a combination of direct extraction, geometric derivation, and lookup tables. When all required properties fall under A1 or A2.1, the analysis can proceed fully automatically. When A3 mappings are needed, the framework maintains lookup tables, for example, mapping nominal diameters to internal diameters for specific pipe materials, or associating fitting types with loss coefficients from normative or manufacturer specific tables. These lookup tables constitute a configuration layer that can be updated without modifying the framework's core logic.

Two recurrent gaps illustrate current IFC export limitations. First, authoring tools typically export pipe geometry based on the nominal (external) diameter; the internal diameter, which is critical for head loss calculations, requires a material-specific lookup table. Second, the `FittingLossFactor` property in `Pset_PipeFittingCommon` is defined in the IFC schema [41] but not populated by default in major authoring tools [40].

A third limitation concerns entity classification fidelity. The framework's type function τ assigns vertex roles based on IFC entity classes, and the Attributes Layer dispatches extraction logic by entity type. Both mechanisms presuppose that authoring tools export components under their semantically correct classes. However, exporters sometimes substitute domain-specific entities with the generic `IfcBuildingElementProxy`, which exists in the schema for objects without

a suitable IFC representation but is frequently misused to replace entities that do have one.

Noardo et al. [27], inspecting 57 IFC models from practice, found that only four achieved sufficient semantic quality across all assessed dimensions, with widespread proxy substitution rendering entity-based classification unreliable for automated processing. For the proposed framework, this has a compounding effect: proxy substitution simultaneously prevents τ from classifying the vertex and prevents the Attributes Layer from selecting the correct property extraction strategy.

In the current implementation of *ifc-hydro*, unrecognised entity types raise a descriptive error rather than silently propagating incorrect classifications. Pre-processing validation through Information Delivery Specifications (IDS) [64] can mitigate both property gaps and entity misclassification before the framework is invoked, and as IFC export quality improves through such initiatives, the proportion of A3 mappings and proxy substitutions is expected to decrease.

3.5. Analysis layer

The Analysis Layer operates on the attributed graph $G' = (V, E, \tau, \alpha)$ to verify system conformance with design requirements. Unlike the previous two layers, the Analysis Layer is discipline-dependent: its logic, equations, and acceptance criteria are dictated by the applicable technical standard.

3.5.1. General structure

Regardless of the specific domain, the Analysis Layer follows a consistent computational pattern:

1. **Path enumeration.** For each terminal vertex $v_t \in V_{\text{terminal}}$, enumerate the unique path P_j from the source vertex r through G'.
2. **Cumulative calculation.** Traverse each path, aggregating demand parameters and computing losses as a function of $\alpha(v_i)$ along the path.
3. **Conformance check.** At each terminal, compare the calculated performance parameter against the acceptance criteria defined by the applicable standard.
4. **Report generation.** Produce a structured output listing, for each path, the input parameters, intermediate calculations, final results, and a pass/fail assessment.

3.5.2. Analysis complexity classification

The computational complexity of the Analysis Layer varies across disciplines and application types. Three analysis classes are proposed, classified by their computational structure. Table 2 lists representative applications across disciplines for each class.

- **AN1—Network traversal.** Analyses that operate purely on graph topology, producing connectivity, reachability, or dependency relations with no scalar computation. Computationally, these reduce to graph traversal (DFS/BFS). These analyses require only the type function τ, not the attribute function α.
- **AN2—Scalar path accumulation.** Analyses in which a scalar quantity is accumulated along each path and evaluated at terminal vertices, performed in a single forward traversal with no modification of graph attributes. For each path $P_j = (v_0, v_1, \ldots, v_k)$, the analysis computes $w(P_j) = f(\alpha(v_0), \ldots, \alpha(v_k))$, where f is a domain-specific accumulation function. This is the class addressed by *ifc-hydro*.
- **AN3—Iterative sizing.** Analyses in which results feed back into α, requiring iterative recomputation until convergence: compute path quantities (AN2 pattern), evaluate constraints, update α, repeat. These correspond to design problems where the network is not only evaluated but modified.

A structural assumption underpins all three classes: the path P_j from source to each terminal vertex v_t is unique. This holds when G' is a directed tree rooted at the source vertex r, the topology of gravity-fed and pumped distribution systems in which flow direction is determined by the network hierarchy. Under this condition, the demand at every edge is fully determined by the set of downstream terminals, and the accumulation function $w(P_j)$ is well-defined without reference to other paths. AN3 iteration modifies component attributes α (e.g., pipe diameters) and recomputes path quantities, but each iteration still decomposes into independent per-path accumulations; the paths themselves remain structurally decoupled.

Looped networks violate this assumption. When cycles exist in G', multiple paths connect the source to a terminal and the flow distribution among them is underdetermined by topology alone. Resolving the flow split requires simultaneous satisfaction of conservation and constitutive constraints across all loops, the classical Kirchhoff problem, solved in hydraulic practice by the Hardy Cross method and its matrix generalisations. Although Hardy Cross is iterative, its iteration couples all shared edges across loops through a global correction step; the unknown is the flow state of the network, not a component attribute in α. This places looped-network analysis in a fundamentally different computational category from AN3, where iteration updates design parameters along independent paths.

Full physical simulation (computational fluid dynamics, finite element structural analysis, transient hydraulic modelling) is likewise excluded.

The framework's scope boundary can therefore be stated as five conditions: the system is representable as a discrete network graph; the graph is acyclic with respect to the flow direction relevant to the analysis, or reducible to a spanning tree with known flow allocation; computation proceeds via path traversal from source to terminals; results are expressible as scalar accumulation along each path, possibly with iterative feedback to component attributes; and required inputs are derivable from IFC data or associated property tables.

Table 2
Representative applications of the analysis complexity classes across building service disciplines.

Analysis class	Plumbing / fire protection	HVAC	Electrical	Structural†
AN1 — Network traversal	Isolation valve impact mapping; dead-end detection; vent path verification; sprinkler zone delineation‡	Terminal-to-AHU tracing; return path verification	Circuit-to-panel tracing; emergency circuit verification	Load path tracing (connectivity to supports)
AN2 — Scalar path accumulation	Available pressure verification (drinking/hot water, gas); drainage fixture unit verification; sprinkler/standpipe pressure verification‡	Duct pressure drop verification (pre-sized); hydronic circuit pressure loss (series); thermal loss along piping	Voltage drop calculation (branch/feeder); fault current estimation via impedance accumulation	Tributary load accumulation along load paths
AN3 — Iterative sizing	Pipe sizing under pressure constraints; booster pump selection with network feedback; sprinkler pipe sizing‡	Duct sizing (equal friction, static regain); hydronic pipe sizing with pump interaction	Conductor sizing with voltage drop constraints; transformer sizing with load diversity	Member sizing with load redistribution feedback

†Structural applications are listed as conceptual extensions under T3 topology;
‡Sprinkler and standpipe applications assume tree-structured branch configurations; looped ring mains require simultaneous flow balancing outside the AN1–AN3 scope (Section 3.5.2).

3.5.3. Domain instantiation

To instantiate the Analysis Layer for a specific domain, three elements must be defined:

1. **Demand model**: the method for assigning demand to terminal vertices and aggregating demand along paths (e.g., fixture weights and probabilistic flow estimation in drinking-water systems).
2. **Loss model**: the accumulation function f governing performance degradation along the network (e.g., head loss equations for hydraulic systems, pressure drop equations for HVAC ductwork, voltage drop formulas for electrical systems).
3. **Acceptance criteria**: the thresholds against which the computed $w(P_j)$ is judged (e.g., minimum and maximum acceptable dynamic pressure at sanitary fixtures).

Extending the framework to a new discipline requires implementing a new Analysis Layer module (defining these three elements) while reusing the existing Topology and Attributes layers.

3.6. Transferability across mechanical, electrical, plumbing and structural disciplines

The generalisability of each layer is summarized in Table 3 and is characterized by mapping disciplines against the topology, attribute, and analysis classification schemes introduced in Sections 3.3 through 3.5.

All three MEP disciplines share T1 topology through the same port-based connectivity mechanism, making the Topology Layer directly reusable across plumbing, HVAC, and electrical systems. The disciplines diverge primarily in their attribute and analysis profiles. Drinking-water systems (Section 4) present the most tractable case: A1 and A2.1 attributes dominate, with A3 gaps limited to internal diameter, demand and loss coefficients, and the analysis is a single-pass scalar accumulation (AN2). HVAC systems introduce spatial containment queries (A2.2) for space load derivation and typically require iterative analysis (AN3) for true sizing; supply-air duct networks are tree-structured and compatible with path-based analysis, but hydronic piping circuits may include looped configurations that fall outside the AN2 and AN3 scope (Section 3.5.2). Electrical systems face a high proportion of A3 attributes due to sparsely populated circuit parameters, deferring feasibility to improvements in IFC export practice [34].

Structural systems need further investigation for proper characterization: their connectivity is encoded in a parallel analysis model (T3) that current authoring tools do not reliably populate [24], representing a direction for future development.

In all cases, the degree of achievable automation is governed by the topology and attribute classes of the target discipline, reflecting the maturity of IFC export support rather than fundamental limitations of the framework itself.

4. Case study

This section demonstrates the framework through *ifc-hydro*, an open-source Python implementation for the automated analysis of drinking-water systems conforming to NBR 5626 [65].

Table 3
Framework applicability across building service disciplines.
Each discipline is characterised by its topology supply class (Section 3.3.1), dominant attribute supply classes (Section 3.4.2), analysis complexity class (Section 3.5.2), the primary IFC entities involved, and the key limitation governing achievable automation.

Discipline	Topology	Analysis	Dominant attribute classes	Primary IFC entities	Key limitation
Plumbing*	T1	AN2	A1, A2.1, A3	`IfcPipeSegment`, `IfcPipeFitting`, `IfcValve`, `IfcSanitaryTerminal`, `IfcTank`	A3: internal diameter and loss coefficients not exported by authoring tools
HVAC	T1	AN2, AN3	A2.1, A2.2, A3	`IfcDuctSegment`, `IfcDuctFitting`, `IfcAirTerminal`, `IfcFan`, `IfcEnergyConversionDevice`	A2.2: space load derivation requires spatial containment between `IfcFlowTerminal` and `IfcSpace`; AN3 sizing requires iterative feedback
Electrical	T1	AN2	A3 (dominant)	`IfcCableSegment`, `IfcCableFitting`, `IfcOutlet`, `IfcElectricDistributionBoard`	A3: circuit parameters (`Pset_ElectricalCircuitType`) rarely populated; significant user configuration required
Structural†	T3	—	—	`IfcBeam`, `IfcColumn`, `IfcStructuralAnalysisModel`, `IfcStructuralConnection`	T3: parallel analysis model not reliably populated by current authoring tools

*Demonstrated in this paper through *ifc-hydro* [7]. All three MEP disciplines share the same port-based connectivity mechanism (`IfcRelConnectsPorts`, `IfcRelNests`) for T1 topology extraction.
†Structural systems are included as a boundary case. Attribute and analysis classes are not characterised because the T3 topology constraint is binding: the framework cannot be instantiated until authoring tools reliably export the structural analysis model [8, 24].

Two models are used: a synthetic model for controlled demonstration of each framework layer (Section 4.2), and a real single-family residential (SFR) project for evaluation under realistic conditions (Section 4.3). Results are compared to both manual calculation and outcomes obtained from a commercial software (AltoQi Builder) in the first case, and only results from AltoQi Builder, in the second.

Together, these constitute the Demonstration and Evaluation stages of the DSR methodology described in Section 3.1.

The hydraulic domain was selected because it exercises all three layers under favourable IFC mapping conditions: T1 topology, a mix of A1 through A3 attribute classes, and AN2 analysis complexity.

4.1. Building *ifc-hydro*

ifc-hydro (v4.1.0) [18] is a Python library built on IfcOpenShell [66], an open-source C++/Python IFC parser, whose package architecture directly mirrors the framework's three-layer separation. Five modules compose the library: `core` (graph data structure, vector operations, geometry utilities), `topology` (system graph construction), `properties` (attribute extraction for pipes, fittings, and valves), `hydraulics` (design flow, pressure drop, and available pressure calculations), and `visualization` (graph rendering).

4.1.1. Topology Layer

The `Topology` class implements the naive variant of the graph construction algorithm formalised in Section 3.3.2: for each `IfcRelConnectsPorts` instance, it scans all pairs of `IfcRelNests` instances to identify which two nesting groups contain the connected ports, then appends the corresponding parent components (`RelatingObject`) as an edge. The result is an undirected graph $G = (V, E)$ whose nodes are IFC distribution elements and whose edges are physical connections. Nodes are classified by IFC entity type: `IfcTank` as source, `IfcSanitaryTerminal` as terminal, and `IfcPipeSegment`, `IfcPipeFitting`, and `IfcValve` as intermediate.

For gravity-fed tree-structured networks, designating the tank as root induces a natural parent-to-child orientation along every edge, from source toward terminals (leaves), without modifying the underlying undirected structure. This implicit ordering is exploited by the Analysis Layer: depth-first path enumeration from root to each leaf produces ordered sequences of components that the hydraulic calculator traverses sequentially, accumulating demand weights downstream and subtracting pressure losses upstream. Each path therefore represents a complete hydraulic circuit from tank to fixture, with the traversal order directly encoding the physical flow direction.

4.1.2. Attributes layer

Three property classes (`Pipe`, `Fitting`, `Valve`) extract engineering parameters from IFC geometric and semantic representations, following the attribute supply classification defined in Section 3.3.2.

Pipe length (A2.1) is obtained from the extrusion depth of the `IfcExtrudedAreaSolid` representation. Cross-sectional diameter is extracted from the profile geometry, with dual-representation parsing supporting both `IfcCircleProfileDef`

(explicit radius) and `IfcArbitraryClosedProfileDef` (bounding-box diameter derivation). These are A2.1 attributes: derived from geometry but not directly encoded as named properties. Nominal diameters are mapped to internal diameters through manufacturer-specific lookup tables (A3), resolving the gap between the external dimension present in the IFC schema and the internal dimension required by hydraulic equations.

For fittings, `PredefinedType` (A1) classifies the component as `BEND`, `JUNCTION`, or `EXIT`. Direction change angle is computed through bounding-box centre vector analysis using the `Geom` utility class, enabling classification of tee junctions as straight-through (0°/180°) or branch (90°). Equivalent length coefficients (A3) are retrieved from reference tables indexed by type, angle, and nominal diameter.

4.1.3. Analysis layer

Three classes in the `hydraulics` module perform the complete available pressure calculation. Design flow is estimated using the square root method, an empirical approach standard in Brazilian practice: each terminal type is assigned a weight from a normative lookup table, and the design flow at any point is computed as $Q = 0.3\sqrt{\Sigma P}$, where ΣP is the sum of weights of all terminals downstream. Pressure drop is calculated using the Fair-Whipple-Hsiao (FWH) equation for PVC pipes:

$$J = 8.69 \times 10^{-4} \cdot Q^{1.75} \cdot D^{-4.75} \quad (1)$$

where J is the unit head loss (m/m), Q is the design flow (L/s), and D is the internal diameter (m). Linear pressure drop is the product of J and pipe length; local losses at fittings and valves use the equivalent length method.

Available pressure at each terminal is computed by traversing the path from tank to terminal, starting from the gravity potential (water level elevation minus terminal elevation) and sequentially subtracting each component's pressure drop contribution. The entire calculation proceeds on the attributed graph produced by the first two layers, with no back-reference to the original IFC model geometry.

4.1.4. Modelling requirements

Both test models require specific authoring-tool configuration to satisfy the input assumptions of the Topology and Attributes layers. For the Topology Layer, MEP pipe systems must be modelled as connected networks using the authoring tool's native routing functionality, not as independently placed components. The IFC exporter must be configured to write system connectivity, specifically, the `IfcRelConnectsPorts` and `IfcRelNests` relations on which T1 topology extraction (Section 3.3.1) depends. When these relations are absent or incomplete, the graph construction algorithm cannot recover the network topology and raises a descriptive connectivity error identifying the missing relational data.

Table 4
Available pressure comparison in meters of water gauge (mWG) for the synthetic model across three independent calculation methods.

Terminal	*ifc-hydro*	AltoQi Builder	Manual calc.
Shower	1.45	1.41	1.45
Washbasin	2.71	2.61	2.66
Toilet	3.37	3.27	3.34

For the Attributes Layer, the `PredefinedType` attribute of each `IfcPipeFitting` and `IfcValve` must be correctly assigned before export, as the equivalent-length lookup (Section 4.1.3) uses this attribute to select loss coefficients. Missing or generic predefined types prevent attribute enrichment and produce a corresponding error. Similarly, the `PredefinedType` of each `IfcSanitaryTerminal` must distinguish fixture types for demand-weight assignment.

4.2. Synthetic model

A synthetic BIM model of a bathroom drinking-water system was created in Autodesk Revit 2024 (Figure 5) and exported as IFC4 [41]. The model comprises one elevated tank (`IfcTank`), three sanitary terminals (`IfcSanitaryTerminal`: shower, washbasin, and toilet with coupled cistern), pipe segments (`IfcPipeSegment`), pipe fittings (`IfcPipeFitting`: elbows and tees), and valves (`IfcValve`: one gate valve and one pressure-regulating valve). All components use DN 25 mm PVC pipes.

The three layers operated end-to-end without manual intervention.

From the IFC model, the Topology Layer produced a rooted tree with three paths (one per terminal) in approximately one second (Figure 4). The Attributes Layer extracted pipe lengths and nominal diameters from geometric representations, mapped nominal to internal diameters via a manufacturer lookup table (25 mm nominal → 21.6 mm internal), classified all fittings, valves and sanitary terminals by `PredefinedType`, and computed direction change angles at each junction. The Analysis Layer then computed available pressure at each terminal.

To validate these results, the same system was independently calculated using AltoQi Builder, a commercial hydraulic analysis tool widely used in Brazil, and through manual spreadsheet calculations, both employing the same method of weights and FWH equation with identical input parameters.

Table 4 shows the three-way comparison. The maximum absolute difference between *ifc-hydro* and manual calculation is 0.05 mWG (washbasin), and between *ifc-hydro* and AltoQi Builder is 0.10 mWG (toilet and washbasin). These deviations are within the tolerance expected for simplified projects and are attributable to rounding conventions in intermediate flow values and differences in internal parameters used by each tool, discussed in Section 4.3.

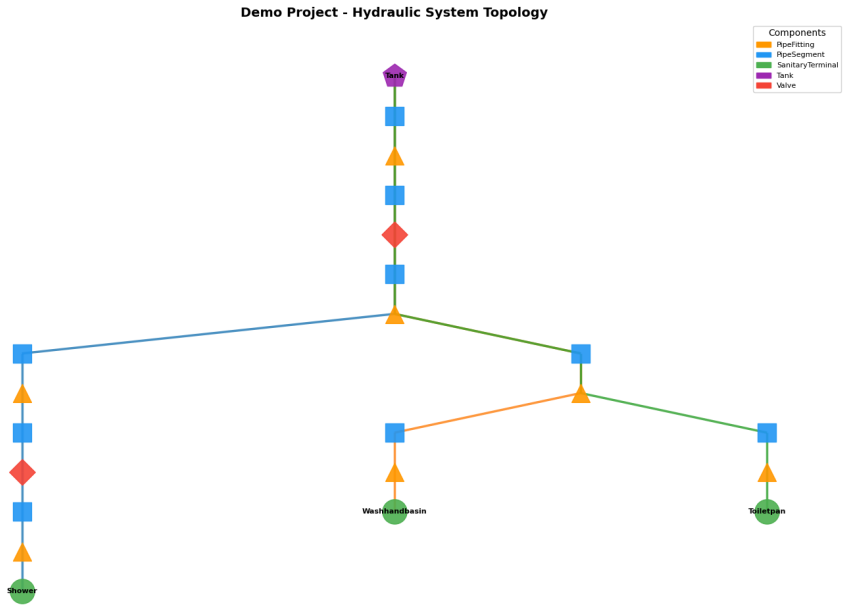


Figure 4: Synthetic system graph produced by the Topology Layer, showing three paths from the elevated tank to the sanitary terminals.

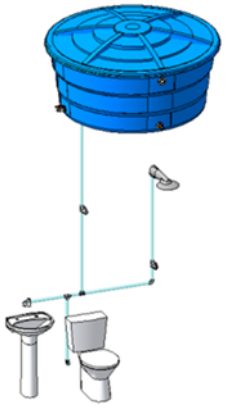


Figure 5: Synthetic model as modelled in Revit 2024.

4.3. Single-family residence

To demonstrate the framework under realistic conditions, *ifc-hydro* was applied to a real-world SFR project obtained from industry, which was modelled and exported from Revit 2024 (Figure 6a) as IFC4. The same project was independently analysed in AltoQi Builder, enabling a two-way validation across all 12 terminals.

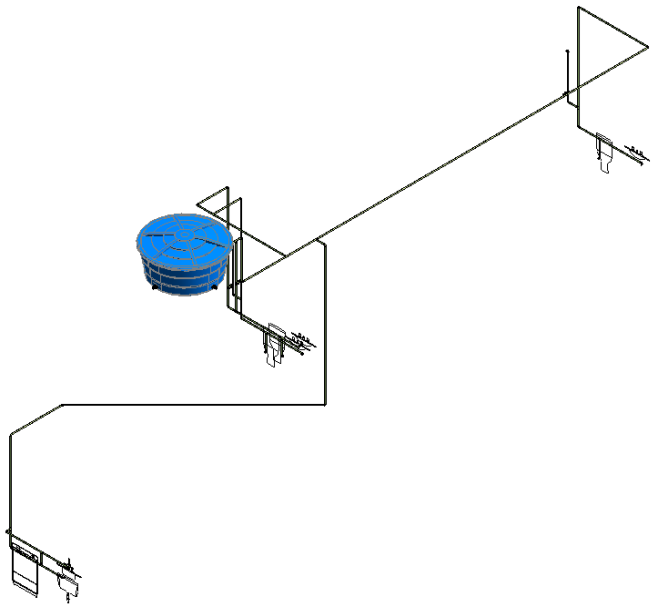
The system comprises one elevated tank (bottom elevation: 7.78 m), 12 sanitary terminals distributed across two floor levels: three showers, three washbasins, three toilets, one kitchen sink, one washing machine, and one laundry tub, connected through a network of pipe segments, fittings, and valves, all using DN 25 mm PVC pipes.

The framework processed the model end-to-end without code changes relative to the synthetic demonstration. The Topology Layer resolved all port-based connections into a 12-path rooted tree in approximately 27 seconds (Figure 6b). The Attributes Layer extracted pipe lengths ranging from 0.052 m to 8.12 m, correctly mapped all nominal diameters to internal values, and classified fittings, valves and sanitary terminals without adaptation. The Analysis Layer computed available pressure at all 12 terminals, with paths ranging from 14 to 28 elements and design flows from 0.095 L/s to 0.636 L/s.

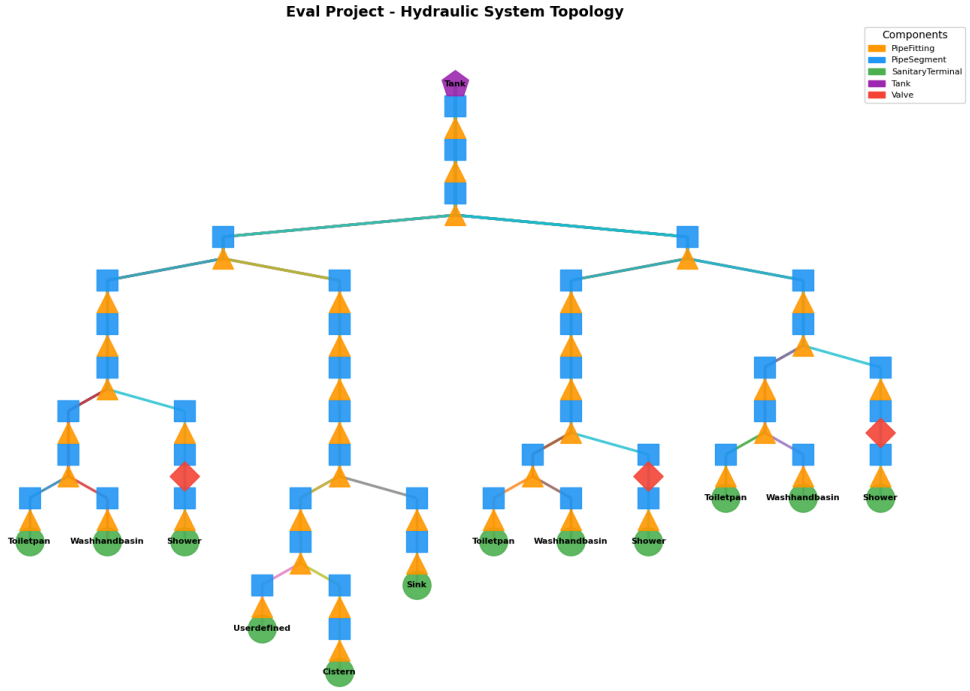
Table 5 compares the available pressure computed by *ifc-hydro* against the independent AltoQi Builder calculation of 12 terminals. The two tools employ the same hydraulic method (square root method) but differ in two input parameters: internal diameter (*ifc-hydro* uses 21.6 mm from the Tigre PVC catalogue; AltoQi Builder uses 22 mm) and effective water level (*ifc-hydro* computes 8.28 m from the tank bottom elevation plus a 0.50 m adjustment; AltoQi Builder uses 7.87 m). Despite these differences, the maximum absolute deviation is 0.20 mWG and the mean absolute deviation is 0.08 mWG.

Critically, both tools produce identical conformance classifications across all 11 compared terminals: all three upper-floor showers are flagged as non-conforming (below the 1.0 mWG mark, practiced as the minimum acceptable value for showers in Brazil), while the remaining terminals pass.

The systematic identification of the same design deficiencies by two independent implementations (one commercial, one framework-based) constitutes the strongest validation evidence in this study. The deviations themselves are attributable to three documented factors: the 0.4 mm difference in internal diameter, which the $D^{-4.75}$ exponent



(a) SFR real-world project as viewed in Revit 2024.



(b) 12-path rooted tree produced by the Topology Layer, with nodes colour-coded by type.

Figure 6: SFR model and resulting graph.

in the FWH equation amplifies on high-demand paths; the difference in effective water level assumptions; and intermediate rounding conventions.

4.4. Discussion

4.4.1. Framework layer behaviour

The Topology Layer operated deterministically in both models. Because the algorithm relies exclusively on `IfcRelConnectsPorts` and `IfcRelNests` graph construction is insensitive to geometric complexity. The same algorithm that

Table 5
Available pressure comparison (mWG) between *ifc-hydro* and AltoQi Builder for the SFR evaluation model. Conformance assessed against minimum pressures prescribed by NBR 5626.

Terminal	Floor	*ifc-hydro*	AltoQi	|Δ|	Status
Toilet 1	Upper	2.18	2.28	0.10	Pass / Pass
Toilet 2	Upper	2.09	2.24	0.15	Pass / Pass
Toilet 3	Upper	2.05	2.04	0.01	Pass / Pass
Washbasin 1	Upper	1.84	1.89	0.05	Pass / Pass
Washbasin 2	Upper	1.76	1.87	0.11	Pass / Pass
Washbasin 3	Upper	1.71	1.65	0.06	Pass / Pass
Shower 1	Upper	0.33	0.41	0.08	Fail / Fail
Shower 2	Upper	0.21	0.35	0.14	Fail / Fail
Shower 3	Upper	0.20	0.17	0.03	Fail / Fail
Kitchen sink	Ground	3.53	3.53	0.00	Pass / Pass
Washing mach.	Ground	2.81	3.01	0.20	Pass / Pass
Laundry tub	Ground	2.58	2.71	0.13	Pass / Pass
Max. |Δ|			0.20		
Mean |Δ|			0.08		

produced a 3-path tree from the synthetic model produced a 12-path tree from the SFR project, with no changes to parameters, thresholds, or heuristics. This is a direct consequence of the T1 topology class: when connectivity is fully encoded in port relations, topology extraction is deterministic.

The Attributes Layer revealed the practical significance of the attribute supply classification (Section 3.4.2). Pipe length (A2.1) and fixture type (A1) were extracted directly. Cross-sectional diameter (A2.1) required dual-representation parsing to handle two distinct geometric profile types. Internal diameter (A3) and fitting/valve loss coefficients (A3) required lookup tables maintained as a configuration layer. The proportion of A3 attributes in drinking-water plumbing (primarily the nominal-to-internal diameter mapping and equivalent length tables) determines the implementation's maintenance burden: when pipe material changes, the tables must be updated, but framework logic remains unchanged.

The Analysis Layer operated as a modular computation on the semantic graph, with no back-reference to the original IFC model geometry. The entire pressure calculation, from weight-based flow estimation through FWH head loss to final available pressure, was performed on the attributed graph produced by the first two layers. This confirms the framework's central design principle: the analysis module consumes a graph, not an IFC file.

4.4.2. Validation and scalability

The three-way comparison in the synthetic model (Table 4) showed maximum deviations of 0.10 mWG, well within acceptable tolerance for simplified projects using the method of weights. The close agreement between *ifc-hydro* and the manual calculation (maximum deviation: 0.05 mWG) confirms that the framework's attribute extraction and analysis logic faithfully reproduce the standard calculation procedure. The slightly larger deviation from AltoQi Builder is consistent with documented differences in tee junction classification and intermediate rounding.

Table 6
Summary of test models and *ifc-hydro* run statistics.

Parameter	Demo. model	Eval. model
Authoring tool / schema	Revit 2024 / IFC4	
Sanitary terminals	3	12
Floor levels	1	2
Tank bottom elev. (m)	3.82	7.78
Max. path length (elem.)	14	28
Min. avail. press. (mWG)	1.45	0.21
Max. avail. press. (mWG)	3.37	4.32
Topology extraction (s)	∼1	∼27
Total analysis pipeline (s)*	<3	∼73

*Excludes visualisation time (∼47 s for the eval. model).

The SFR demonstration extends this validation to a model not designed as a test case. The independent AltoQi Builder calculation (Table 5) produced identical conformance classifications across all compared terminals: both tools flag the same three upper-floor showers as nonconforming and agree on every passing terminal. The maximum absolute deviation between the two tools is 0.20 mWG (mean: 0.08 mWG), attributable to the documented parameter differences described in Section 4.3.

Table 6 summarises the key characteristics of both models and the run statistics, illustrating scalability from a controlled demonstration to a real-world project.

4.4.3. Implementation evolution

Several limitations documented during the initial development [7] have been addressed in *ifc-hydro* v4.1.0, illustrating the framework's capacity for incremental refinement guided by the attribute supply classification.

The nominal-to-internal diameter gap is resolved through a manufacturer-specific lookup table. This moved the internal diameter from an unresolved A2.1 limitation to a managed A3 attribute with explicit configuration.

The geometry extraction methods, originally based on hardcoded index paths into the IFC entity structure, have been replaced by a `Geom` utility class that uses IfcOpenShell's geometry kernel for bounding-box and elevation calculations, improving robustness against variations in representation structure across authoring software versions. The package architecture was restructured from a monolithic script into modular packages aligned with framework layers, facilitating independent testing and extension.

5. Conclusion

This paper has presented a framework for deriving graph-based engineering analysis directly from IFC models. The framework separates the analytical pipeline into three modular layers and draws a structural analogy with Automated Rule Checking: the first two layers perform model preparation (graph extraction and property enrichment), while the third performs rule execution (domain-specific computation and conformance verification).

Three classification taxonomies were introduced to systematise the framework's scope and applicability. The topology supply classification (T1–T3) distinguishes systems whose connectivity is explicitly encoded in IFC port relations from those requiring geometric reconstruction or parallel analysis models. The attribute supply classification (A1–A3) characterises the degree to which the properties required for engineering computation are directly available, derivable from geometry, or dependent on external input. The analysis complexity classification (AN1–AN3) discriminates between network traversal, scalar path accumulation, and iterative sizing, mapping each class to representative applications across plumbing, HVAC, electrical, and structural disciplines (Table 3). Together, these taxonomies provide a structured vocabulary for assessing the feasibility and automation potential of IFC-based analysis in any given discipline.

The framework was demonstrated through *ifc-hydro*, an open-source Python implementation for cold-water plumbing analysis conforming to NBR 5626 [65]. A synthetic model and a real single-family residence were processed end-to-end, with results validated against both manual calculation and AltoQi Builder, a commercial hydraulic analysis tool. The maximum absolute deviation was 0.20 mWG (mean 0.08 mWG), and both tools produced identical conformance classifications across all compared terminals, including the identification of three non-conforming showers.

The generalisability of the first two layers across MEP disciplines was formalised in Section 3.6: the port-based connectivity mechanism is shared across plumbing, HVAC, and electrical systems, and extending the framework to a new discipline requires implementing only a new Analysis Layer module while reusing the existing Topology and Attributes layers.

Future work should pursue four directions.

1. **Multi-discipline extension.** The current demonstration covers drinking-water systems only. HVAC duct pressure-drop verification is the most immediate candidate for a second instantiation, given its shared T1 topology and AN2 analysis structure.
2. **Diameter heterogeneity and authoring-tool coverage.** Both test models use uniform DN 25 mm piping; real systems employ diameter transitions that exercise the Attributes Layer's per-port diameter tracking. Similarly, the geometric property extraction currently traverses representation patterns produced by Autodesk Revit 2024; validating against exports from other authoring tools would isolate framework logic from software-specific representation choices. Adopting Information Delivery Specifications (IDS) as a pre-processing validation step could enforce the modelling requirements the framework depends on, reducing A3 attribute gaps at the source.
3. **Cyclic topologies.** Relaxing the acyclic assumption that bounds the AN2 and AN3 classes would extend coverage to looped distribution networks (e.g., fire-protection rings, recirculating hot-water systems) in which simultaneous flow balancing across shared edges cannot be decomposed into independent path accumulations. This entails a fourth analysis class in which the Analysis Layer delegates to a network solver (e.g., Hardy Cross or gradient method) rather than to path traversal, while the Topology and Attributes layers remain unchanged.
4. **Data-driven inference.** The graph representations produced by the Topology and Attributes layers can serve as input substrates for graph neural networks targeting surrogate modelling or anomaly detection, opening a pathway from rule-based analysis to learned inference on building-system networks.

Declaration of generative AI and AI-assisted technologies in the manuscript preparation process

During the preparation of this work, the authors used ChatGPT, Claude and Gemini in order to assist with code documentation and packaging the *ifc-hydro* repository as a Python library, proofread the writing and improve grammar, spelling and clarity. After using these tools, the authors have reviewed and edited the content as needed and take full responsibility for the content of the published article.

CRediT authorship contribution statement

André Buchmann Müller: Writing – review & editing, Writing – original draft, Visualization, Validation, Investigation, Software, Methodology, Conceptualization.
Fabiano Rogerio Corrêa: Writing – review & editing, Validation, Supervision, Resources, Project administration.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] R. Sacks, C. Eastman, G. Lee, P. Teicholz, BIM handbook: a guide to building information modeling for owners, designers, engineers, contractors, and facility managers, 2018. doi:10.1002/9781119287568.

[2] M. Chowdhury, M. R. Hosseini, D. J. Edwards, I. Martek, S. Shuchi, Comprehensive analysis of BIM adoption: From narrow focus to holistic understanding, Automation in Construction 160 (2024) 105301. doi:10.1016/j.autcon.2024.105301.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580524000372

[3] H. Biswas, T. Sim, S. Lau, Impact of Building Information Modelling and Advanced Technologies in the AEC Industry: A Contemporary Review and Future Directions, Journal of Building Engineering 82 (2024) 108165. doi:10.1016/j.jobe.2023.108165.
URL https://linkinghub.elsevier.com/retrieve/pii/S2352710223023458

[4] T. Singh, M. Mahmoodian, S. Wang, Enhancing Open BIM Interoperability: Automated Generation of a Structural Model from an Architectural Model, Buildings 14 (8) (2024) 2475. doi:10.3390/

buildings14082475.
URL https://www.mdpi.com/2075-5309/14/8/2475

[5] E. Atencio, P. Araya, F. Oyarce, R. F. Herrera, F. Muñoz-La Rivera, F. Lozano-Galant, Towards the Integration and Automation of the Design Process for Domestic Drinking-Water and Sewerage Systems with BIM, Applied Sciences 12 (18) (2022) 9063. doi:10.3390/app12189063.
URL https://www.mdpi.com/2076-3417/12/18/9063

[6] H. Li, J. Zhang, IFC-based Information Extraction and Analysis of HVAC Objects to Support Building Energy Modeling, 2022. doi:10.22260/ISARC2022/0024.
URL http://www.iaarc.org/publications/2022_proceedings_of_the_39th_isarc_bogota_colombia/ifc_based_information_extraction_and_analysis_of_hvac_objects_to_support_building_energy_modeling.html

[7] A. Müller, R. Clemente, T. Ferreira, F. Corrêa, Automação no cálculo de pressões dinâmicas de sistemas prediais hidráulicos com IFC: extração de dados em BIM, Vol. 5, ANTAC, Porto Alegre, 2025. doi:10.46421/sbtic.v5i00.7418.

[8] P. Colajanni, L. Inzerillo, A. Pisciotta, F. Acuto, K. Mantalovas, G. Di Mino, Architectural and Structural Interoperability in the BIM Design Workflow, Buildings 15 (24) (2025) 4540. doi:10.3390/buildings15244540.
URL https://www.mdpi.com/2075-5309/15/24/4540

[9] J. Han, X. Zhou, W. Zhang, Q. Guo, J. Wang, Y. Lu, Directed Representative Graph Modeling of MEP Systems Using BIM Data, Buildings 12 (6) (2022) 834. doi:10.3390/buildings12060834.
URL https://www.mdpi.com/2075-5309/12/6/834

[10] Y.-Q. Xiao, S.-W. Li, Z.-Z. Hu, Automatically Generating a MEP Logic Chain from Building Information Models with Identification Rules, Applied Sciences 9 (11) (2019) 2204. doi:10.3390/app9112204.
URL https://www.mdpi.com/2076-3417/9/11/2204

[11] Y. Li, Q. Zhao, M. Yang, Z. Ma, X. Hei, Advancements and Applications of Industry Foundation Classes Standards in Engineering: A Comprehensive Review, Buildings 15 (16) (2025) 2927. doi:10.3390/buildings15162927.
URL https://www.mdpi.com/2075-5309/15/16/2927

[12] Y. Yu, S. Kim, H. Jeon, B. Koo, A Systematic Review of the Trends and Advances in IFC Schema Extensions for BIM Interoperability, Applied Sciences 13 (23) (2023) 12560. doi:10.3390/app132312560.
URL https://www.mdpi.com/2076-3417/13/23/12560

[13] F. Noardo, T. Wu, K. Arroyo Ohori, T. Krijnen, J. Stoter, IFC models for semi-automating common planning checks for building permits, Automation in Construction 134 (2022) 104097. doi:10.1016/j.autcon.2021.104097.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580521005483

[14] S. Daum, A. Borrmann, Processing of Topological BIM Queries using Boundary Representation Based Methods, Advanced Engineering Informatics 28 (4) (2014) 272–286. doi:10.1016/j.aei.2014.06.001.
URL https://www.sciencedirect.com/science/article/pii/S1474034614000391

[15] A. Borrmann, E. Rank, Topological analysis of 3D building models using a spatial query language, Advanced Engineering Informatics 23 (4) (2009) 370–385. doi:10.1016/j.aei.2009.06.001.
URL https://linkinghub.elsevier.com/retrieve/pii/S1474034609000287

[16] W. Solihin, C. M. Eastman, Classification of rules for automated BIM rule checking development, Automation in Construction 53 (2015) 69–82. doi:10.1016/j.autcon.2015.03.003.
URL https://www.sciencedirect.com/science/article/pii/S0926580515000370

[17] K. Peffers, T. Tuunanen, M. A. Rothenberger, S. Chatterjee, A Design Science Research Methodology for Information Systems Research, Journal of Management Information Systems 24 (3) (2007) 45–77. doi:10.2753/MIS0742-1222240302.
URL https://www.tandfonline.com/doi/full/10.2753/MIS0742-1222240302

[18] A. Müller, ifc-hydro (Mar. 2026). doi:10.5281/zenodo.19022607.

[19] Quaest, Mini Censo CONFEA 2024, Tech. rep., available online: https://www.confea.org.br/censo-do-confea-92-dos-profissionais-estao-empregados. Accessed on 3 Mar. 2026. (2024).

[20] AltoQi, Panorama Precificação de Projetos 2024, Tech. Rep. 5th Edition, available online: https://www.altoqi.com.br/conteudos-gratuitos/ebook-panorama-de-precificacao-de-projetos-2024. Accessed on: 3 Mar. 2026. (2024).

[21] J. P. Martins, A. Monteiro, LicA: A BIM based automated code-checking application for water distribution systems, Automation in Construction 29 (2013) 12–23. doi:10.1016/j.autcon.2012.08.008.
URL https://www.sciencedirect.com/science/article/pii/S0926580512001495

[22] J. Farooq, P. Sharma, S. Ramdas, A BIM-based Detailed Electrical Load Estimation, Costing and Code Checking, International Journal of Electrical and Computer Engineering (IJECE) 8 (5) (2018) 3484–3495, number: 5. doi:10.11591/ijece.v8i5.pp3484-3495.
URL https://ijece.iaescore.com/index.php/IJECE/article/view/11525

[23] ISO, ISO 16739-1:2024 Industry Foundation Classes (IFC) for data sharing in the construction and facility management industries — Part 1: Data schema (Mar. 2024).
URL https://www.iso.org/standard/84123.html

[24] S. Gerbino, L. Cieri, C. Rainieri, G. Fabbrocino, On BIM Interoperability via the IFC Standard: An Assessment from the Structural Engineering and Design Viewpoint, Applied Sciences 11 (23) (2021) 11430. doi:10.3390/app112311430.
URL https://www.mdpi.com/2076-3417/11/23/11430

[25] H. Lai, X. Deng, Interoperability analysis of IFC-based date exchange between heterogeneous BIM software, Journal of Civil Engineering and Management 24 (7) (2018) 537–555. doi:10.3846/jcem.2018.6132.
URL https://journals.vilniustech.lt/index.php/JCEM/article/view/6132

[26] M. Belsky, R. Sacks, I. Brilakis, Semantic Enrichment for Building Information Modeling, Computer-Aided Civil and Infrastructure Engineering 31 (4) (2016) 261–274. doi:10.1111/mice.12128.
URL https://onlinelibrary.wiley.com/doi/10.1111/mice.12128

[27] F. Noardo, K. Arroyo Ohori, T. Krijnen, J. Stoter, An Inspection of IFC Models from Practice, Applied Sciences 11 (5) (2021) 2232. doi:10.3390/app11052232.
URL https://www.mdpi.com/2076-3417/11/5/2232

[28] M. Venugopal, C. M. Eastman, R. Sacks, J. Teizer, Semantics of model views for information exchanges using the industry foundation class schema, Advanced Engineering Informatics 26 (2) (2012) 411–428. doi:10.1016/j.aei.2012.01.005.
URL https://linkinghub.elsevier.com/retrieve/pii/S1474034612000067

[29] H. Wang, P. Xu, H. Sha, J. Gu, T. Xiao, Y. Yang, D. Zhang, BIM-based automated design for HVAC system of office buildings—An experimental study, Building Simulation 15 (7) (2022) 1177–1192. doi:10.1007/s12273-021-0883-7.
URL https://link.springer.com/10.1007/s12273-021-0883-7

[30] H. Sha, P. Xu, Z. Yang, IFC based semi-automated design tool for HVAC central system: A general framework, IOP Conference Series: Earth and Environmental Science 238 (1) (2019) 012074. doi:10.1088/1755-1315/238/1/012074.
URL https://dx.doi.org/10.1088/1755-1315/238/1/012074

[31] D. Jansen, V. Richter, L. Maier, J. Frisch, C. V. Treeck, D. Müller, Open-source framework for automated generation of building energy performance simulation models and beyond from BIM Data, Automation in Construction 179 (2025) 106427. doi:10.1016/j.autcon.2025.106427.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580525004674

[32] E. Kamel, A. M. Memari, Review of BIM's application in energy simulation: Tools, issues, and solutions, Automation in Construction 97 (2019) 164–180. doi:10.1016/j.autcon.2018.11.008.

URL https://linkinghub.elsevier.com/retrieve/pii/S0926580517304958

[33] A. Andriamamonjy, D. Saelens, R. Klein, An automated IFC-based workflow for building energy performance simulation with Modelica, Automation in Construction 91 (2018) 166–181. doi:10.1016/j.autcon.2018.03.019.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580517308282

[34] A. Gupta, A. Cemesova, C. J. Hopfe, Y. Rezgui, T. Sweet, A conceptual framework to support solar PV simulation using an open-BIM data exchange standard, Automation in Construction 37 (2014) 166–181. doi:10.1016/j.autcon.2013.10.005.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580513001623

[35] R. Vieira, P. Carreira, P. Domingues, A. A. Costa, Supporting building automation systems in BIM/IFC: reviewing the existing information gap, Engineering, Construction and Architectural Management 27 (6) (2020) 1357–1375. doi:10.1108/ECAM-07-2018-0294.
URL http://www.emerald.com/ecam/article/27/6/1357-1375/93436

[36] S. Tang, D. R. Shelden, C. M. Eastman, P. Pishdad-Bozorgi, X. Gao, BIM assisted Building Automation System information exchange using BACnet and IFC, Automation in Construction 110 (2020) 103049. doi:10.1016/j.autcon.2019.103049.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580519301645

[37] J. Shi, J. Dao, L. Jiang, Z. Pan, Research on IFC- and FDS-Based Information Sharing for Building Fire Safety Analysis, Advances in Civil Engineering 2019 (1) (2019) 3604369. doi:10.1155/2019/3604369.
URL https://onlinelibrary.wiley.com/doi/abs/10.1155/2019/3604369

[38] J. Wu, J. Chen, G. Chen, Z. Wu, Y. Zhong, B. Chen, W. Ke, J. Huang, Development of Data Integration and Sharing for Geotechnical Engineering Information Modeling Based on IFC, Advances in Civil Engineering 2021 (1) (2021) 8884864. doi:10.1155/2021/8884864.
URL https://onlinelibrary.wiley.com/doi/abs/10.1155/2021/8884864

[39] J. Zhu, X. Wang, M. Chen, P. Wu, M. J. Kim, Integration of BIM and GIS: IFC geometry transformation to shapefile using enhanced open-source approach, Automation in Construction 106 (2019) 102859. doi:10.1016/j.autcon.2019.102859.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580519302468

[40] M. Antunes, K. César, J. Ribeiro, D. Oliveira, J. Carvalho, Analysis of IFC interoperability data schema for project representation, Automation in Construction 166 (2024) 105650. doi:10.1016/j.autcon.2024.105650.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580524003868

[41] buildingSMART International, IFC 4.3.2.0 documentation (2024).
URL https://ifc43-docs.standards.buildingsmart.org/

[42] ISO, ISO 10303-11:2004 Industrial automation systems and integration — Product data representation and exchange — Part 11: Description methods: The EXPRESS language reference manual (Nov. 2004).
URL https://www.iso.org/standard/38047.html

[43] W. Solihin, C. Eastman, Y.-C. Lee, Toward robust and quantifiable automated IFC quality validation, Advanced Engineering Informatics 29 (3) (2015) 739–756. doi:10.1016/j.aei.2015.07.006.
URL https://www.sciencedirect.com/science/article/pii/S1474034615000725

[44] C. Preidel, S. Daum, A. Borrmann, Data retrieval from building information models based on visual programming, Visualization in Engineering 5 (1) (2017) 18. doi:10.1186/s40327-017-0055-0.
URL https://link.springer.com/10.1186/s40327-017-0055-0

[45] W. Mazairac, J. Beetz, BIMQL – An open query language for building information models, Advanced Engineering Informatics 27 (4) (2013) 444–456. doi:10.1016/j.aei.2013.06.001.
URL https://linkinghub.elsevier.com/retrieve/pii/S1474034613000657

[46] Y. Adachi, Overview of partial model query language, in: Concurrent engineering: Enhanced interoperable systems, Vol. 1, CRC Press, 2003, pp. 549–555.

[47] buildingSMART International, buildingSMART data dictionary (bSDD) (2021).
URL https://www.buildingsmart.org/users/services/buildingsmart-data-dictionary/

[48] A. Borrmann, E. Rank, Specification and implementation of directional operators in a 3D spatial query language for building information models, Advanced Engineering Informatics 23 (1) (2009) 32–44. doi:10.1016/j.aei.2008.06.005.
URL https://linkinghub.elsevier.com/retrieve/pii/S1474034608000542

[49] A. Borrmann, S. Schraufstetter, E. Rank, Implementing Metric Operators of a Spatial Query Language for 3D Building Models: Octree and B-Rep Approaches, Journal of Computing in Civil Engineering 23 (1) (2009) 34–46. doi:10.1061/(ASCE)0887-3801(2009)23:1(34).
URL https://ascelibrary.org/doi/10.1061/%28ASCE%290887-3801%282009%2923%3A1%2834%29

[50] B. Hillier, J. Hanson, The Social Logic of Space, 1st Edition, Cambridge University Press, 1984. doi:10.1017/CBO9780511597237.
URL https://www.cambridge.org/core/product/identifier/9780511597237/type/book

[51] A. Ismail, B. Strug, G. Ślusarczyk, Building Knowledge Extraction from BIM/IFC Data for Analysis in Graph Databases, in: Artificial Intelligence and Soft Computing: 17th International Conference, ICAISC 2018, Zakopane, Poland, June 3-7, 2018, Proceedings, Part II, Vol. 10842 of Lecture Notes in Computer Science, Springer International Publishing, Cham, 2018, pp. 652–664. doi:10.1007/978-3-319-91262-2.
URL https://link.springer.com/10.1007/978-3-319-91262-2

[52] J. Zhu, N. Nisbet, M. Yin, R. Wei, I. Brilakis, Releasing the power of graph for building information discovery, Automation in Construction 172 (2025) 106034. doi:10.1016/j.autcon.2025.106034.
URL https://www.sciencedirect.com/science/article/pii/S0926580525000743

[53] J. Zhu, N. Nisbet, R. Kang, Y. Wen, M. Wang, I. Brilakis, Revealing the internal structure of IFC-Graph for efficient querying and knowledge discovery, Advanced Engineering Informatics 70 (2026) 104229. doi:10.1016/j.aei.2025.104229.
URL https://linkinghub.elsevier.com/retrieve/pii/S147403462501122X

[54] Y. Wang, L. Zhang, H. Yu, R. L. Tiong, Detecting logical relationships in mechanical, electrical, and plumbing (MEP) systems with BIM using graph matching, Advanced Engineering Informatics 54 (2022) 101770. doi:10.1016/j.aei.2022.101770.
URL https://linkinghub.elsevier.com/retrieve/pii/S1474034622002282

[55] W. Teclaw, R. Kind, N. Labonnote, IFC Properties Validation Using Deep Graph Neural Network, in: 2024 9th International Conference on Smart and Sustainable Technologies (SpliTech), 2024, pp. 1–6. doi:10.23919/SpliTech61897.2024.10612441.
URL https://ieeexplore.ieee.org/abstract/document/10612441

[56] Y. Pan, M. Wang, L. Lu, R. Lamsal, E. Pärn, S. Zlatanova, I. Brilakis, LLM-enabled multi-agent framework for natural language interaction with graph-based digital twins, Automation in Construction 183 (2026) 106791. doi:10.1016/j.autcon.2026.106791.
URL https://www.sciencedirect.com/science/article/pii/S0926580526000324

[57] Z. Wang, R. Sacks, T. Yeung, Exploring graph neural networks for semantic enrichment: Room type classification, Automation in Construction 134 (2022) 104039. doi:10.1016/j.autcon.2021.104039.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580521004908

[58] C. M. Eastman, J.-m. Lee, Y.-s. Jeong, J.-k. Lee, Automatic rule-based checking of building designs, Automation in Construction

18 (8) (2009) 1011–1033. doi:10.1016/j.autcon.2009.07.002.
URL https://linkinghub.elsevier.com/retrieve/pii/S0926580509001198

[59] Solibri Oy, Understanding checking (2024).
URL https://help.solibri.com

[60] Building and Construction Authority (BCA), CORENET X: The new integrated regulatory process (2024).
URL https://info.corenet.gov.sg

[61] W. Solihin, Z. Liu, Y. Lu, L. Wei, BIM-based automated rule-checking in the AECO industry: Learning from semiconductor manufacturing, Automation in Construction 162 (2024) 105406. doi:10.1016/j.autcon.2024.105406.
URL https://www.sciencedirect.com/science/article/pii/S0926580524001420

[62] D. P. Lacerda, A. Dresch, A. Proença, J. A. V. Antunes Júnior, Design Science Research: método de pesquisa para a engenharia de produção, Gestão & Produção 20 (4) (2013) 741–761. doi:10.1590/S0104-530X2013005000014.
URL http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0104-530X2013000400001&lng=pt&tlng=pt

[63] F. Harary, Graph theory, 4th Edition, Addison-Wesley series in mathematics, Addison-Wesley, Reading, Mass., 1995.

[64] buildingSMART International, Information delivery specification (IDS) (Jun. 2024).
URL https://github.com/buildingSMART/IDS/releases/tag/v1.0.0

[65] Associação Brasileira de Normas Técnicas, ABNT NBR 5626: Sistemas prediais de água fria e água quente — Projeto, execução, operação e manutenção (2020).

[66] IfcOpenShell Contributors, IfcOpenShell 0.8.4 documentation, manual (2026).
URL https://docs.ifcopenshell.org/autoapi/ifcopenshell/index.html

André Buchmann Müller is a civil engineer and Digital Engineering consultant at DB Engineering & Consulting in Brazil, and an M.Sc. candidate in Construction Innovation at Universidade de São Paulo (USP). His research sits at the intersection of computational design and data modelling, with a focus on turning building information into machine-interpretable structures for automated reasoning. His interests are in exploring graph-based abstractions of building systems to support topology reconstruction, rule checking, model-driven engineering analyses and digital-twinning, including hydraulic network assessment for plumbing systems. He develops Python tooling and applies these methods in industry-facing innovation initiatives, targeting engineering workflows from design to operations.

Fabiano Rogerio Corrêa is a lecturer in the Department of Civil Construction Engineering at the Polytechnic School of Universidade de São Paulo, he holds a degree in Mechanical Engineering with an emphasis on Automation and Systems from the University of São Paulo (2002), a master's degree in Mechanical Engineering specializing in Mechatronics from the University of São Paulo (2004), and a doctorate in Mechanical Engineering in the area of Control and Automation (2009). Currently, he dedicates himself to the area of Computational Technology Applied to Civil Construction, conducting research on the following topics: Automation and Robotics in Construction; Impact of Industry 4.0 on Construction, Building Information Modelling (BIM) and Industry Foundation Classes (IFC); Digital Fabrication and Additive Manufacturing (3D Printing) applied to Architecture and Construction; Smart Cities and Big Data.