From 2D Drawings to BIM Models: An LLM-Assisted Approach

Jean Viaunel Victor[1] and Pavan Kumar[2] and Shang-Hsien Hsieh[3]

[1]Master Student, Department of Civil Engineering, National Taiwan University, Taiwan
[2]PhD Candidate, Department of Civil Engineering, National Taiwan University, Taiwan
[3]Professor, Department of Civil Engineering, National Taiwan University, Taiwan
viocitovictor@caece.net

Abstract

Building Information Modeling (BIM) has become a central platform for digital building representation; however, the creation and manipulation of BIM models remain largely manual, and accessible primarily to domain experts. At the same time, architectural information is still commonly produced in 2D drawing formats, creating inefficiencies in translating design intent into semantically rich 3D BIM models. Recent advances in large language models (LLMs) offer new possibilities for natural language–based interaction with complex digital systems, yet their integration with BIM for automated and scalable building modeling is still limited. This research addresses this gap by proposing an automated framework that integrates LLMs with BIM systems to support the transformation of 2D architectural drawings into parametric model generation and manipulation. The framework's capability is demonstrated by using a single-floor architectural plan as input; the system enables the scalable generation of multi-storey buildings. The results indicate that LLM-assisted BIM workflows can significantly reduce manual modeling effort and improve efficiency, while remaining computationally tractable for generating multi-storey buildings. This study demonstrates the potential of LLM–BIM integration as a foundation for more intuitive, efficient, and scalable building modeling workflows.

Keywords –
image2BIM, img2BIM, LLM-BIM integration

1 Introduction

Building Information Modeling (BIM) has become a foundational technology for digital building representation and management across the architecture, engineering, and construction (AEC) industry. It is widely regarded as an intelligent, three-dimensional, model-centric process that enables the comprehensive integration of geometric and semantic information across the entire building lifecycle**[1].** By integrating geometric, semantic, and relational information within a unified digital environment, BIM supports improved design coordination, construction planning, and lifecycle management of built assets. As such, BIM has been widely adopted as a central platform for advancing digital transformation in the built environment.

Despite its widespread adoption, the creation and manipulation of BIM models remain largely dependent on manual operations performed within specialized authoring tools. Furthermore, heavily reliant on 2D CAD formats for design communication [2], these processes are time-consuming, require significant domain expertise, and familiarity with software-specific modeling paradigms. Consequently, BIM authoring remains time-consuming, error-prone, and inaccessible to non-expert stakeholders, particularly during the early design stages, where rapid exploration and iteration are crucial.

At the same time, architectural information is still predominantly produced and communicated through two-dimensional (2D) drawings, such as floor plans, sections, and elevations. While these representations effectively convey design intent, they lack the explicit semantic structure and parametric relationships required for downstream BIM-based analysis and automation**[2].** The persistent reliance on 2D documentation creates a disconnect between early-stage design practices and BIM-centric workflows.

Large Language Models (LLMs) provide an opportunity, and a recent review highlights that LLMs in the construction sector demonstrate their high potential, with median accuracies of 89% for building information detection and 91% for auxiliary modeling automation [3]. While earlier studies leveraged LLMs primarily for building information detection[3] and querying 3D spatial relationships through tree-based indexing [4], the contemporary landscape is increasingly defined by generative workflows. Notably, the workflow proposed by[2] orchestrates a multi-agent system to transform abstract textual instructions into imperative Python code for BIM authoring software APIs, enabling the creation

of natively editable models with internal layouts and semantic metadata. Parallel efforts have integrated conversational AI with Visual Programming Languages (VPLs), such as Grasshopper, to allow designers to iterate on forms and materials through natural-language prompts[5][6]. Similarly, [7] proposed the integration of LangGraph and LLM to generate BIM models, and the LSVR-SE workflow demonstrates text-to-BIM conversion for generating compliant Industry Foundation Classes (IFC) files directly from language commands[8].

However, as the field matures, there remains a valuable opportunity to refine the workflows at the technical grounding and orchestration of these generative processes. Existing workflows primarily excel at engaging natural language as input; they often rely on abstract linguistic prompts that may lack the inherent geometric precision and structured data found in existing technical documentation. Furthermore, recent studies indicate that agent-based BIM interactions can remain fragmented, often requiring repetitive re-implementation of core capabilities across different proprietary APIs[9]. This study seeks to build a workflow integrating LLM and BIM to generate BIM models, taking 2D DXF architectural drawings, which provide necessary geometric constraints for professional-grade modelling. Central to this approach is the adoption of the Model Context Protocol (MCP). By utilizing MCP to orchestrate a staged transformation pipeline, this study enables the systematic parsing of 2D primitives and their translation into semantically rich, multi-storey IFC models. The proposed workflow provides an interoperable foundation that moves beyond single-story constraints, offering a scalable bridge between traditional 2D documentation and automated 3D parametric modelling and manipulation.

This research study aims to develop a necessary technical proof-of-concept of a workflow that integrates LLMs with BIM systems to transform the 2D architectural drawings into parametric models. By focusing on the technical feasibility of the proposed integrated workflow, the research study establishes a scalable baseline for automating complex BIM authoring tasks. The proposed framework enables natural language–driven building generation and manipulation, with a specific emphasis on achieving scalability in multi-storey building modeling.

2 Methodology

This study proposes a unified framework integrating LLM with a parametric model from a 2D architectural floor plan. The LLM-driven pipeline for converting two-dimensional DXF architectural drawings into a semantically enriched three-dimensional IFC Building Information Model consists of the LLM orchestrating four core tools: Parse DXF, detect room, Generate IFC, and Create multi-storey building, as shown in Figure 1. Each of these core tools consists of sub-tools such as Read DXF file, followed by Detecting units, normalizing coordinates, and extracting elements, forming a tool Parse DXF as shown in Figure 2.

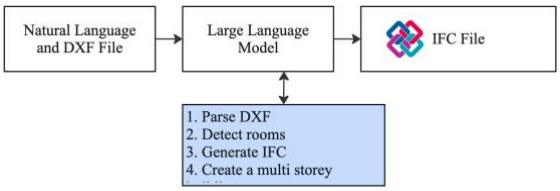

Figure 1. Overview of LLM-driven BIM to automate 2D floor plans into BIM Models

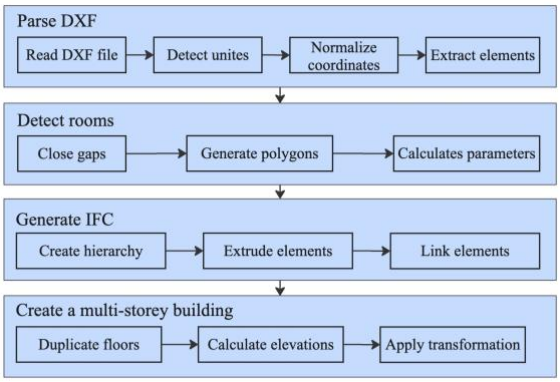

Figure 2. A Detailed Workflow for Generating Multi-Storey BIM Models from DXF Floor Plans

2.1 Parse DXF

This is the first tool that LLM executes in the stack of 4 tools in sequence. In the Parse DXF, the provided DXF file by the user is read with "ezdxf.readfile()" to access the header and model space. Units are detected from $INSUNITS and scaled via "get_dxf_unit_scale()" to millimeters. Layers are mapped using a configurable alias system through "detect_wall_layer_structure()". Elements (walls, columns, windows, doors, slabs) are extracted with specialized functions, preserving semantic XData (e.g., Revit wall types).

2.2 Detect Rooms

Once the elements are extracted from the parsed DXF, the LLM verifies the outputs and calls wall endpoints, which are connected across small gaps (< 100 mm) at doors/windows, using "_find_and_close_openings()". Walls are merged and polygonised with Shapely to form a room. Parameters such as wall start and end points, and locations of doors and windows are then returned to the LLM.

2.3 Generate IFC

Standard IFC files are generated through this tool. A standard IFC4 hierarchy (IfcProject → IfcSite → IfcBuilding → IfcBuildingStorey) is created via "create_combined_ifc()". Furthermore, walls are extruded to floor height (default 3000 mm) through IfcWallStandardCase objects. Doors and windows are modelled as IfcOpeningElement voids. All elements are linked to stories to maintain BIM validity.

2.4 Create Multi-storey Building

The generated IFC file is then used to create a multi-storey building as per the user's request. Parameters, such as the number of walls, windows, doors, and the final IFC model, are returned. The ground floor is treated as a reference floor and can be duplicated for the nth stories. The user can also instruct to specifically choose a certain floor to be the reference floor and duplicate it for the nth storey. Each copy is placed at elevation `$Z_i = i \times H\_f$`, with all elements shifted vertically. Specific stories can optionally be replaced with alternate DXF files (e.g., a lobby or penthouse).

2.5 Intent Interpretation Layer

In the intent interpretation layer, the LLM engages in interpreting the user inputs and orchestrates the tools in sequence through the Model Context Protocol (MCP) Standard. The natural-language requests are converted into structured parameters (floor indices, feature flags, heights). LLM callable tools execute functions such as "convert_dxf_to_ifc()" and "create_multi_storey_building()", using the Model Context Protocol (MCP) standard. Inputs are then validated, and structured results with statistics are returned by the LLM to the user.

3 Example demonstration

This section describes the demonstration of the framework proposed to integrate LLM and BIM to generate an IFC file from the 2D floor plan. A DXF file, as shown in Figures 3 and 4 is used as input along with the user request in Figure 5. The DXF file is a layout on an orthogonal alphanumeric grid and uses standard CAD layers for walls, doors, windows, furniture, sanitary fixtures, and stairs. This simple, structured representation is extremely important as it makes it easier for both humans and algorithms to read.

The framework accepts natural language requests alongside 2D DXF floor plans as input. An LLM interprets the user's intent, extracting structured parameters (e.g., number of storeys, floor height, floor swapping specifications) and mapping them to predefined tool functions through MCP configured for Claude code with Sonnet 4.5. From these tools, architectural elements (walls, columns, windows, doors) are extracted through layer-based classification. The final output is an IFC-compliant 3D BIM model, which is compatible with software such as Revit and ArchiCAD, as shown in Figure 5.

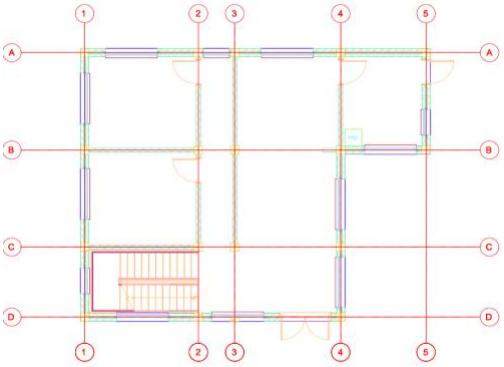

Figure 3. Floor plan 1

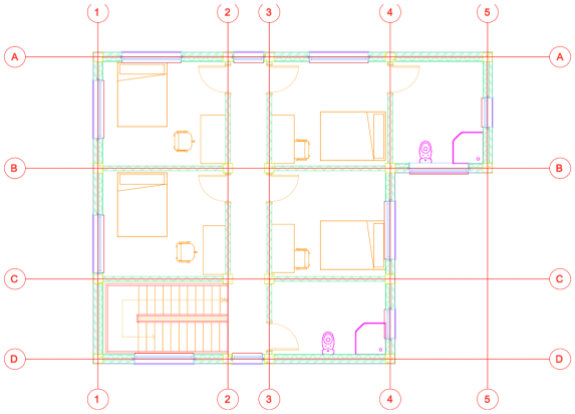

Figure 4. Floor plan 2

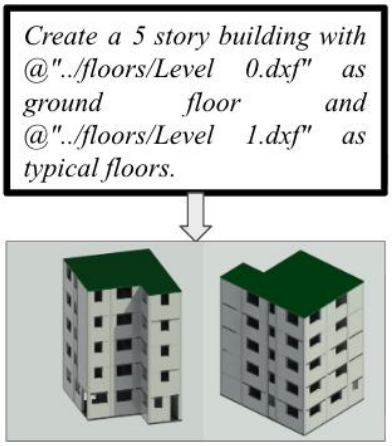

Figure 5. User request as input to the LLM to generate the 3D model.

4 Results & Discussions

This section lays out the result of the example demonstrated using the proposed framework to integrate BIM and LLM to generate IFC files from the 2D floor plans. The DXF file (figures 3 and 4), along with the user request, Create a 5-story building with @"../floors/Level 0.dxf" as ground floor and @"../floors/Level 1.dxf" as typical floors. is provided to the LLM to generate an IFC file. The generated IFC file can be viewed through tools such as Revit, ArchiCAD, etc. The IFC file generated in the example demonstration is shown in Figure 6.

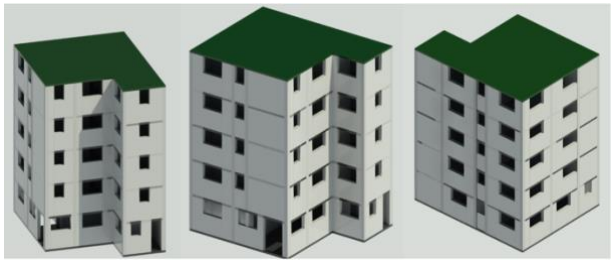

Figure 6. IFC 3D Model Viewed In Revit

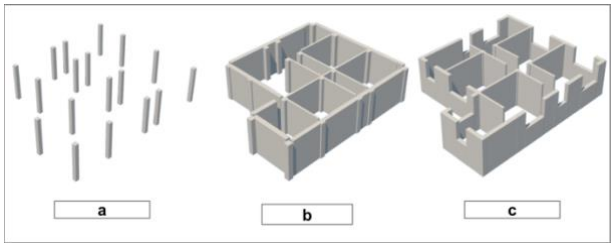

Figure 7. Three-Stage Generative Modeling Process

The transformation follows a staged pipeline in which geometric primitives are extracted and normalized, processed into element-level representations, and then translated into IFC entities with spatial relationships. This separation enables each element type to be handled using element-specific rules while maintaining IFC schema compliance.

Column entities are extracted during Parse DXF through layer-based classification and converted to IFC columns during IFC generation. The intermediate output (Figure 7(a)) visualizes the extracted vertical elements as isolated volumes at their detected plan locations, providing an immediate check that the layer-based extraction has correctly recovered structural placement before enclosure elements are assembled.

Wall geometry is then reconstructed from the extracted DXF wall elements and is then instantiated in IFC using IfcWallStandardCase objects. In the demonstrated input, the DXF plan is an orthogonal layout using standard CAD naming layers (AIA), which supports reliable extraction and simplifies downstream reconstruction. Table 1 shows how layers are matched to IFC classes. Walls are then extruded to the floor height parameter (default 3000 mm), and all generated elements are linked to their respective storeys to maintain BIM validity. The intermediate wall assembly shown in Figure 7(b) illustrates that the system recovers the external perimeter and internal partitions of the layout with minor inconsistencies with the source drawings.

Openings are handled as semantic voids to preserve IFC validity while delegating solid subtraction to BIM applications. Doors and windows are modelled as IfcOpeningElement voids, and the resulting openings are associated with host walls through standard IFC relationships during IFC generation. The Detect Rooms stage further supports topological consistency by closing wall endpoints across small gaps (< 100 mm) at door and window locations, improving enclosure continuity for downstream interpretation. The final stage shown in Figure 7(c) reflects this semantic modelling choice: openings appear consistently placed within the wall system, and BIM viewers render these voids during visualization.

Table 1. Mapping from AIA CAD naming layer to the Architectural Elements

DXF Layer Name	Architectural Element	IFC Class Generated
A-WALL	Wall	IfcWall
A-COLS	Column	IfcColumn
A-DOOR	Door	IfcDoor
A-WIND	Window	IfcWindow
A-SLAB	Floor slab	IfcSlab
A-STAIR	Stair	Not Generated
A-FURN	Furniture	Not Generated

The integration of natural language input with MCP-based orchestration provides a practical interface for specifying conversion parameters (e.g., storey count and height) without requiring direct interaction with BIM authoring tools. In this example, the produced BIM model preserves the orthogonal layout and storey structure of the source drawings, and the resulting IFC file is usable in standard BIM environments for inspection and further editing.

The proposed proof of concept workflow demonstrates architectural scalability through the automated generation of multi-storey models from 2D floor plan data and text, highlighting that efficiency and accessibility are currently qualitative, based on the reduction of discrete manual modelling steps.

At the same time, the results highlight limitations consistent with the current scope of the pipeline. Performance and accuracy are sensitive to the drawing being structured and layer-disciplined, as assumed by the

layer-based extraction approach used in Parse DXF.

In addition, the current implementation prioritizes orthogonal plan reconstruction (as used in the demonstration), and more complex drafting cases (e.g., diagonal walls, irregular junctions, and inconsistent drafting gaps) require additional geometric reasoning and automated quality checks beyond the current gap-closing strategy. Future work shall focus on expanding geometric coverage (e.g., non-orthogonal walls and junction resolution), increasing semantic enrichment of the exported model while preserving IFC4 compliance, and quantitatively comparing this LLM-assisted approach against traditional BIM authoring workflows for both expert and non-expert users.

5 Conclusion

This research study demonstrates a framework that integrates Large Language Models (LLMs) with Building Information Modeling (BIM) to bridge the gap between 2D architectural floor plans and semantically rich 3D parametric models. By automating the transformation of floor plans into accurate multi-storey representations, the proposed system can significantly reduce time-consuming manual work of building 3D models and reduce reliance on manual labour and specialized expertise traditionally required for BIM authoring. The proposed framework utilizes the capabilities of LLM and orchestrates multiple tools based on the inputs, such as DXF files alongside natural language inputs. The framework automates a staged transformation pipeline—extracting geometric primitives, normalizing element-level representations, and instantiating IFC-compliant entities.

The demonstration results confirm that the system can reliably transform standard 2D layouts into multi-storey IFC models that preserve structural placement, wall perimeters, and semantic relationships such as door and window openings. While the current implementation shows high fidelity for orthogonal, layer-disciplined drawings, it also highlights the necessity for advanced geometric reasoning to handle non-orthogonal layouts and drafting inconsistencies. Furthermore, this study establishes technical feasibility, and quantitative performance metrics shall be employed in future work of this study, with a focus on benchmarking the framework using element detection precision and geometric deviation to provide a statistically significant assessment of its robustness compared to existing non-LLM baselines. Ultimately, this study demonstrates that LLM-assisted BIM workflows can significantly lower the barrier to entry for complex building modeling, providing a scalable and intuitive foundation for future advancements in automated AEC digital transformation.

References

[1] S. Lee, M. Uhm, H. D. Jeong, and G. Lee, "Automated conversion and semantic gap filling of material layer annotations from CAD drawings to semantically rich BIM objects," *Advanced Engineering Informatics*, vol. 70, Mar. 2026, doi: 10.1016/j.aei.2025.104199.

[2] C. Du, S. Esser, S. Nousias, and A. Borrmann, "Text2BIM: Generating Building Models Using a Large Language Model-Based Multiagent Framework," *Journal of Computing in Civil Engineering*, vol. 40, no. 2, Mar. 2026, doi: 10.1061/jccee5.cpeng-6386.

[3] J. Liang, Y. Lin, Y. Liu, Y. Huang, and H. Wu, "Pre-training large language models based on Transformer architecture for building industry application: A review," *Tsinghua University*. Nov. 01, 2025 doi: 10.1007/s12273-025-1324-9.

[4] A. Li, P. K. Y. Wong, X. Tao, J. Ma, and J. C. P. Cheng, "An interactive system for 3D spatial relationship query by integrating tree-based element indexing and LLM-based agent," *Advanced Engineering Informatics*, vol. 66, Jul. 2025, doi: 10.1016/j.aei.2025.103375.

[5] J. Ko, J. Ajibefun, and W. Yan, "Generative AI-powered parametric modeling and BIM for architectural design and visualization," in *Proceedings of the Design Society*, Cambridge University Press, pp. 1943–1952. Aug. 2025 doi: 10.1017/pds.2025.10208.

[6] B. El Hizmi, Y. Sterman, and G. Austern, "LLMto3D - Generation of parametric, 3D printable objects using large language models," *International Journal of Architectural Computing*, vol. 23, no. 3, pp. 701–719, Sep. 2025, doi: 10.1177/14780771251353792.

[7] M. Selim, K. Nassar, and O. Hosny, "Automating BIM (IFC) Data Analysis Using LangGraph," in *World Congress on Civil, Structural, and Environmental Engineering*, Avestia Publishing, 2025. doi: 10.11159/icsect25.182.

[8] M. Xu, G. Qi, N. He, and F. Ju, "Language-guided semantic editing in single-view 3D reconstruction," *Visual Computer*, vol. 42, no. 3, Feb. 2026, doi: 10.1007/s00371-025-04349-y.

[9] Hellin, N., Du, C., & Borrmann, A. Natural language information retrieval from BIM models: An LLM-based multi-agent system approach. *Proceedings of the 2025 European Conference on Computing in Construction (EC3).* 2025