An LLM-Integrated BIM Workflow for Rapid Early-Stage Layout Generation of Worksite Trailers

Chao-Hsu Pan[1], Ren-Jie Li[1], Liang-Ting Tsai[2], Cheng-Hsuan Yang[3], and Meng-Han Tsai[1]

[1] National Taiwan University of Science and Technology, Taipei, Taiwan
[2] University of Alberta, Edmonton, Alberta, Canada
[3] RoBIM Technologies Inc., Edmonton, Alberta, Canada
menghan@mail.ntust.edu.tw

Abstract

Worksite trailers are widely used as rapidly deployable prefabricated units, yet their design process must balance standardization for efficiency with customization for diverse functional needs. This study proposes a proof-of-concept early-stage design workflow that integrates a Large Language Model (LLM) with Building Information Modeling (BIM) to support requirement interpretation, historical case reuse, and preliminary layout generation for worksite trailers. The system accepts natural-language descriptions and dimensional constraints, converts them into structured JSON-based parameters, and connects them to BIM generation in Autodesk Revit.

To reduce repetitive modeling effort and improve design reuse, the workflow includes three modules: Historical Case Retrieval, Unit-Based Spatial Assembly, and Conversational Parametric BIM Modeling. Historical cases are matched through a weighted multi-criteria process based on function, spatial composition, occupancy, dimensions, and special-space requirements. Retrieved cases may be reused directly or decomposed into reusable spatial units for recombination. When no suitable precedent is available, or when further refinement is needed, users can generate or modify layouts through natural-language commands that are translated into executable BIM operations.

Prototype demonstrations show that the workflow can generate editable preliminary configurations, support iterative refinement, and improve early-stage design reuse. However, the system remains a proof-of-concept prototype, and formal time-efficiency testing, user studies, and large-scale retrieval benchmarks have not yet been conducted.

Keywords –

off-site manufacturing, Large Language Model (LLM), human–machine interface (HMI)

1 Introduction

Worksite trailers are a prevalent form of prefabricated modular construction, widely deployed as temporary accommodation units, offices, and command centers in construction sites and remote working environments due to their rapid deployment and transportability [1]. In principle, modularity should benefit from standardization to enable efficient delivery and time savings [2]. In practice, however, diverse usage scenarios create a persistent tension between "standardization for efficiency" and "customization for flexibility," where customer-specific requirements often challenge standardized production logic [3]. Even seemingly minor changes in spatial proportions, functional zoning, or dimensional adjustments can still require substantial modeling effort, partly because current BIM tools often provide limited user-centered support for rapid early-stage adaptation [4]. Meanwhile, high investment costs in digital hardware technologies may further discourage stakeholders from adopting advanced digital workflows at scale [5].

Current workflows typically address this tension through manual model modification or template-based extension, yet a lack of project-to-project knowledge transfer remains a persistent barrier [6]. Past cases are commonly scattered across drawings, model files, and individual experience, and a large portion of construction data remains unstructured and difficult to retrieve and reuse [7]. Consequently, case reuse still depends heavily on manual interpretation and rework, limiting the potential for digital data to optimize assembly processes and reduce waste in offsite contexts [8]. In early-stage design, stakeholders often need not a fully detailed model, but a usable preliminary model that can be generated quickly and refined iteratively; language-guided early layout generation has therefore become an emerging direction [9]. Prior work also indicates that integrating BIM with DfMA-oriented parametric and algorithmic collaboration can strengthen client engagement during

offsite configuration, but practical workflows for fast customization are still developing [10].

Building on these challenges, this study integrates Large Language Models (LLMs) with BIM-based modeling to support requirement understanding and early-stage spatial configuration generation for worksite trailers. The system accepts natural-language descriptions and dimensional constraints, translates them into structured and executable parameters, and uses these parameters to drive BIM modeling operations, enabling iterative refinement toward a usable configuration [11]. To address fragmented knowledge organization and limited reuse, the framework further incorporates a structured representation and matching mechanism for historical cases to support retrieval and reuse. Unlike one-shot generative layout methods, the proposed approach emphasizes stepwise refinement, reusable case knowledge, and direct editability in the BIM environment. Accordingly, the objectives of this study are to: (1) develop an LLM-centric requirement interpretation workflow that converts natural-language inputs into executable design parameters; (2) establish a reusable design knowledge workflow by structuring historical configuration cases for retrieval and adjustment; and (3) implement a BIM-based prototype to demonstrate an end-to-end pipeline—from requirement input to the generation of a usable early-stage configuration—thereby validating the feasibility of language-driven BIM for worksite trailer design.

2 Methodology

2.1 System Overview

This study proposes an early-stage worksite trailer design workflow that integrates Large Language Models (LLMs) with Building Information Modeling (BIM). The primary goal is to rapidly generate a usable initial configuration during the early design phase while supporting iterative adjustment and convergence. As shown in Figure 1, the proposed workflow comprises four stages: requirement input and interpretation, historical case retrieval and matching, parameterized model generation and update, and data saving and output. Through this structure, case reuse and parameterized BIM modeling are unified into a single coherent process, enabling consistent model creation whether users start from a retrieved seed layout or directly specified parameters.

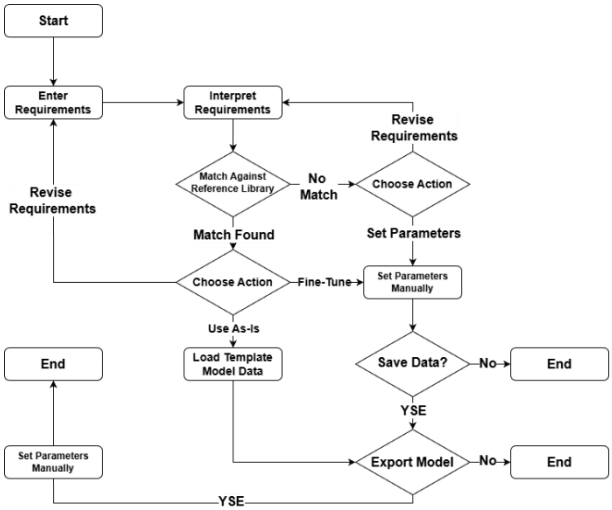

Figure 1. Framework of the proposed approach.

2.2 System Architecture

Figure 2 illustrates the system architecture of the proposed workflow. The framework is composed of three core modules: Historical Case Retrieval, Unit-Based Spatial Assembly, and Conversational Parametric Modeling. These modules do not function independently; instead, they are coordinated through a shared data mechanism based on JSON representations and a centralized ProjectStore for global state management.

Within this architecture, the Historical Case Retrieval module provides reusable reference cases based on user-specified requirements. The Unit-Based Spatial Assembly module enables the decomposition and recombination of reusable spatial units from retrieved or predefined cases. The Conversational Parametric Modeling module interprets user instructions and translates them into executable BIM parameters for model generation and update in Autodesk Revit.

A key feature of the proposed system is its common data mechanism. Rather than maintaining isolated intermediate results in each module, the system uses structured JSON data and ProjectStore to record global design states, shared schema definitions, reusable unit information, and BIM-related parameter mappings. This design supports multi-turn interaction, traceable parameter updates, and consistent synchronization between semantic interpretation and geometric model execution. The resulting BIM parameters are then exported to the Revit environment for model creation and modification.

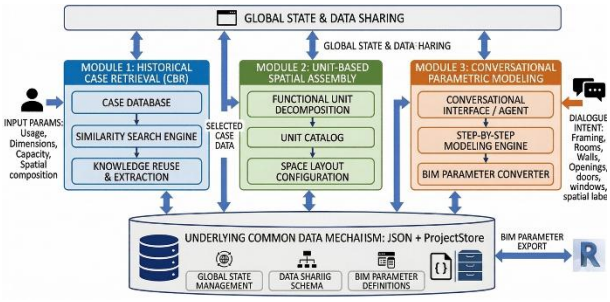

Figure 2. System architecture of the proposed workflow

2.3 Historical Case Retrieval and Reuse

To improve retrieval consistency, each historical case is stored in a structured JSON-based representation containing key attributes such as function type, spatial composition, occupancy, approximate dimensions, and special-space requirements. Candidate cases are then scored through a weighted multi-criteria matching process based on these attributes. To reduce semantic inconsistency, room names are normalized into canonical categories, and partial similarity is allowed for related room types. The final ranking is generated from the aggregated similarity score, and the top-ranked cases are presented as candidate seed layouts for further reuse and refinement.

When a similar case is identified, the retrieved configuration is treated as a seed layout. The selected case is loaded as an editable set of design parameters, allowing users to adopt it as a starting point and refine it according to project-specific needs. This mechanism reduces modeling ambiguity, shortens the exploration cycle, and enables structured reuse across recurring worksite trailer scenarios.

If the retrieved seed layout still deviates from the user's requirements, the system supports iterative customization through parameter editing. Such parameterization not only supports geometric adjustment but also ensures that subsequent BIM updates remain consistent with the retrieved case context. The system does not require users to provide a complete specification at the beginning; requirements may be supplied progressively. In general, richer and more explicit inputs yield more complete retrieval conditions, which improves similarity ranking and increases the stability of case matching.

In the current prototype, retrieval effectiveness is examined mainly through qualitative inspection of returned cases rather than a formal large-scale benchmark, because the present case library is relatively small and manually curated.

2.4 Unit-Based Spatial Assembly

In addition to whole-case retrieval, the proposed workflow supports unit-based spatial assembly. This module allows users to decompose historical cases into reusable spatial units, such as bedrooms, washrooms, kitchens, office areas, or other functional spaces, and recombine them into a new configuration. As a result, the system supports not only case retrieval, but also case decomposition and case recombination.

The extracted spatial units preserve not only geometric properties but also basic semantic information, including space type, unit dimensions, functional role, and relative layout relationships with adjacent spaces. This enables the system to perform semantically informed spatial recomposition rather than simple geometric block placement.

The system also supports case decomposition preview. After a case is retrieved, users may inspect its internal composition and review which spatial units it contains before deciding which parts should be preserved or discarded. This function helps users better understand the internal structure of a precedent case and provides a basis for subsequent recombination.

A further advantage of this module is that spatial units may be selected across different cases. For example, users may choose a bedroom from one case, a washroom from another, and a kitchen from a third, and then reorganize these units into a new solution. This extends design knowledge reuse beyond rigid template-based reuse and allows a finer-grained design recombination process.

After the desired units are selected, the assembled result is converted into a new design basis for downstream modeling. The recombined configuration can then be connected to subsequent processes such as outer frame generation, room allocation, interior wall generation, opening placement, and space labeling. In this sense, Unit-Based Spatial Assembly serves as an intermediate design-restructuring mechanism between historical case knowledge and parameterized BIM generation.

2.5 Parameterized Layout Generation

When case matching is insufficient or no suitable case exists, users may directly specify model parameters, and the system generates an initial configuration through the same parameterized workflow. This design ensures that a usable early-stage model can still be produced even when the case library provides limited coverage.

In the parameterized modeling stage, the system drives BIM generation and updates using an editable set of model data. The process sequentially performs: (1) overall envelope dimension setting, (2) room space partitioning, (3) interior wall layout, (4) door/window

and other opening placement, and (5) room labeling. User instructions are interpreted through task-specific prompts and converted into structured JSON outputs, which are then mapped to internal data objects and BIM parameters.

After each parameter change, the system updates the BIM model and provides visual feedback, enabling low-cost iterative refinement and rapid convergence of the initial configuration. This approach allows the generated model to remain editable and traceable throughout the design process.

Finally, the workflow includes two post-processing options. First, users may store the generated result as a new case to enrich the case library. Second, they may export the BIM model as a baseline for subsequent detailing and engineering coordination.

3 Implementation

The proposed workflow has been implemented as a functional proof-of-concept prototype within a Building Information Modeling (BIM) environment. Autodesk Revit is adopted as the modeling platform, and the system operates as a Revit add-in that uses the Revit API (Application Programming Interface) to transform structured design data into corresponding BIM elements. The current prototype uses an API-based Large Language Model, GPT-4o mini, for requirement parsing and command interpretation. During development, another lightweight model was also tested; however, GPT-4o mini was selected because it provided sufficiently stable parameter extraction performance at a lower cost. This implementation enables an end-to-end pipeline from natural-language requirement input to BIM model generation.

To connect semantic interpretation and BIM execution, the prototype employs a unified structured data format as an intermediate layer. Rather than relying on a single universal prompt, the system adopts a task-specific prompt design strategy. Different prompts are used for subtasks such as requirement extraction, room partitioning, wall operations, opening placement, dimension modification, and intent routing. Each prompt explicitly requires a single valid JSON (JavaScript Object Notation) object without additional explanatory text, and the returned content is parsed into predefined internal data structures for storage and subsequent BIM execution.

In the implemented system, Historical Case Retrieval serves as an important starting point for early-stage design. When a user enters a request such as “I need a dorm unit for 3 people, including a kitchen, bedroom, and bathroom,” the system first analyzes the request and extracts key conditions, such as function, capacity, required spaces, and required dimensions. Figure 3 presents the case retrieval and selection interface, where the system displays candidate historical cases, matching levels, and selected projects based on the parsed conditions for further comparison and confirmation. This mechanism allows users to quickly identify potentially relevant configurations from previous cases, reducing the burden of manually searching and reviewing existing designs, while enabling early-stage design to begin from precedent knowledge.

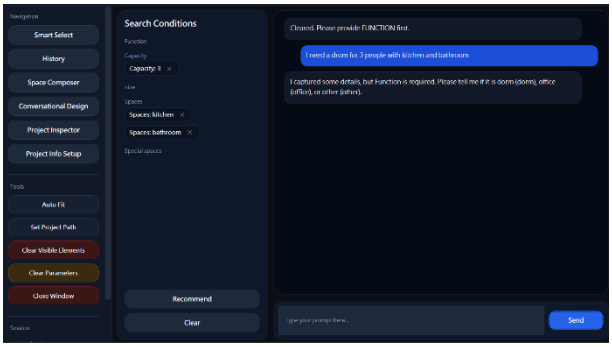

Figure 3. Case retrieval and selection interface

In addition to presenting candidate cases, the system also provides more detailed matching information to help users understand why each project is recommended. Figure 4 shows the matched project results together with their associated attributes, including project function, spatial composition, capacity, dimensional conditions, and their degree of correspondence with the user’s requirements. Through this information, users can not only review the recommended cases, but also understand how each candidate aligns with the intended design conditions, thereby improving the transparency and interpretability of case comparison and selection.

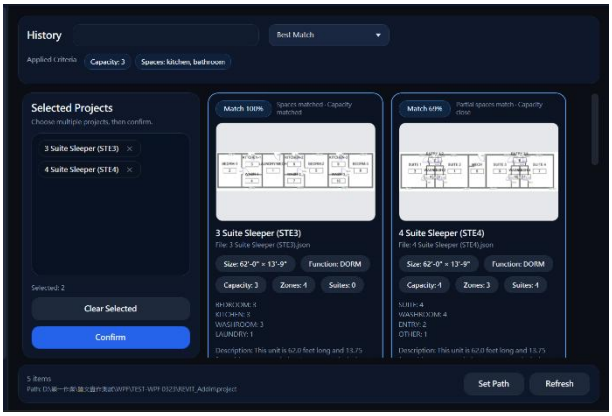

Figure 4. Matched project results and associated attribute information

After candidate cases are obtained, the system further supports Unit-Based Spatial Assembly. Instead of only allowing whole-case reuse, the system can decompose

selected cases into smaller reusable spatial units, such as bedrooms, washrooms, kitchens, and service spaces. Figure 5 illustrates the preview of decomposed units from historical cases. This allows users to inspect the internal composition of each case, determine which spatial units should be retained, and identify which parts may be replaced. In this way, historical cases are no longer treated merely as complete templates, but are transformed into finer-grained design resources.

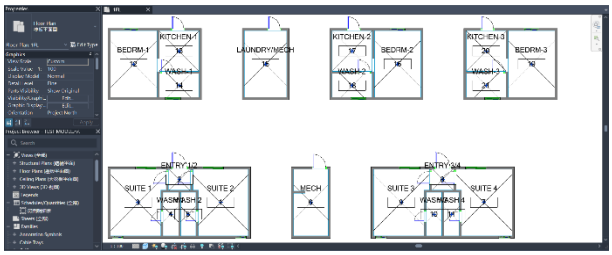

Figure 5. Preview of decomposed reusable spatial units from selected historical cases

Furthermore, the system supports selecting and recombining spatial units across different historical cases. Users may choose rooms or functional units from multiple projects and reorganize them into a new customized configuration. Figure 6 shows a recombined layout assembled from reusable spatial units originating from different precedent cases. This demonstrates that the system supports not only case reuse, but also flexible spatial recombination for early-stage customization.

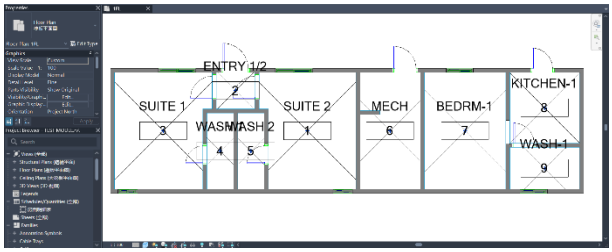

Figure 6. Recombined layout assembled from reusable spatial units

When Historical Case Retrieval and Unit-Based Spatial Assembly cannot provide a satisfactory solution, or when users need to further refine the configurations produced by the previous modules, the system can activate the third module, namely Conversational Parametric BIM Modeling. This module allows users not only to move beyond existing precedent cases, but also to directly describe spatial requirements in natural language or issue localized modification commands for an existing configuration. The system then progressively translates these inputs into executable modeling parameters and corresponding BIM operations, thereby supporting model creation, modification, and iterative refinement.

To achieve this, the system does not rely on a single universal prompt. Instead, it adopts a task-specific prompt design strategy. Dedicated prompts are defined for different subtasks, including outer-frame extraction, room-segmentation extraction, interior-wall command interpretation, wall trimming, opening placement, opening resizing, wall-thickness modification, and intent routing. These prompts explicitly require the model to output only a valid JSON object without any additional explanatory text. The returned content is then immediately parsed into predefined internal data structures for storage and subsequent BIM execution. If the returned output is empty, malformed, or violates predefined geometric or modeling constraints, the system does not execute the corresponding modeling action and instead requests a revised instruction. After user input is processed, the extracted information is written into the shared data structure, and corresponding modeling actions are triggered based on the interpreted parameters. At present, the prototype supports parameterized operations covering overall building dimensions, room count and target sizes, internal spatial subdivision through wall placement, opening placement (e.g., doors and windows), and room label generation for identification and model management. Figure 7 presents the input interface of the third module, where users issue natural-language instructions for direct generation or further refinement, while Figure 6 shows the recombined layout that may serve as the basis for subsequent conversational refinement.

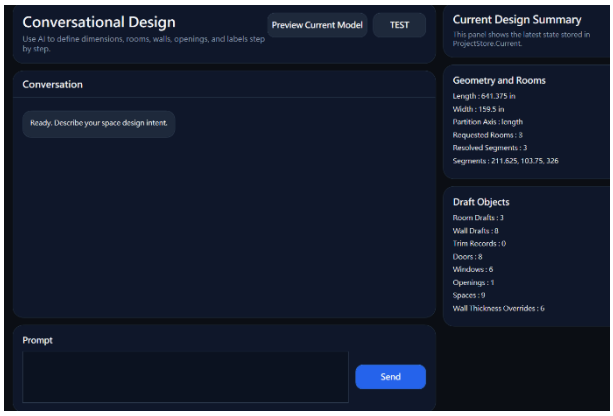

Figure 7. Input interface of the conversational parametric BIM modeling module

Representative examples further illustrate how natural-language commands are translated into executable modeling actions. For overall dimensions, an input such as “Create a building with a total length of 526.875 inches and a width of 159.5 inches” is parsed and stored as envelope parameters in the structured data,

forming the basis for subsequent modeling. For room specification, an instruction such as “Divide it into three rooms with lengths of 211.625, 103.75, and 211.5 inches” updates the room count and assigns target dimensions to each room for later spatial partitioning.

For interior subdivision, the system supports wall placement commands to control room separation. For example, a vertical wall instruction such as “In Room 1, add a vertical wall 98.5 inches from the right side” is interpreted as a vertical partition wall with a specified offset relative to the room boundary, whereas a horizontal wall instruction such as “In Room 3, place a horizontal wall 60.5 inches above the bottom edge” is converted into a corresponding horizontal wall parameter. These wall definitions are recorded in the structured data together with orientation and relative offset information, enabling subsequent BIM generation.

For opening placement, the system allows users to specify doors, windows, or other openings through natural-language descriptions. For instance, a command such as “On the top wall of Room 1, put a door in the center of the right half” is parsed to determine the target room, host wall, relative placement region, and opening type. The system then generates an opening record containing wall references, positional information, and size parameters for BIM execution.

The system also supports label and identification updates. For example, an instruction such as “Rename the label of Room 1 to BEDRM-1” updates the corresponding room identification fields in order to maintain naming consistency within the BIM model. Such label adjustments not only improve model management, but also help align the generated result with subsequent design communication and engineering coordination.

All parsed design information is persistently stored and reused during subsequent modeling steps. The BIM modeling module sequentially reads these structured data and uses the Revit API to generate or update the building envelope, space partitions, walls, openings, and labels. Through this step-by-step update process, the model provides immediate visual feedback after each parameter change, thereby enabling low-cost iterative refinement and rapid convergence. Overall, the third module is capable not only of generating a new model directly from textual requirements when no suitable precedent exists, but also of serving as an effective mechanism for detailed refinement after the first two modules have produced an initial configuration. This prototype implementation demonstrates the feasibility and repeatability of natural-language-driven BIM configuration generation in an operational modeling environment.

4 Discussion and Future Work

This study investigates the integration of Large Language Models (LLMs) with Building Information Modeling (BIM) to support early-stage layout design for worksite trailers. The proposed workflow demonstrates that natural-language interaction can be effectively linked to structured parameter generation and BIM-based modeling, enabling preliminary configurations to be developed more directly than in conventional manual workflows. For worksite trailer projects, where repeated customization is often required under dimensional and transport constraints, this approach offers a practical means of reducing the need to begin each design task from scratch.

A key strength of the proposed workflow lies in its integration of Historical Case Retrieval, Unit-Based Spatial Assembly, and Conversational Parametric BIM Modeling within a unified data mechanism. Through this structure, prior design cases are not treated merely as static references, but can instead be retrieved, examined, reorganized, and further refined according to project-specific needs. This makes the workflow particularly valuable in situations where both design reuse and flexibility are required, especially when user requirements do not fully correspond to any single precedent case.

At the same time, several limitations remain in the current study. The existing case library is still limited in scale and diversity, which affects the breadth of retrieval and the stability of case-based reuse. In addition, the present validation is based mainly on prototype implementation and representative demonstrations. Formal time-efficiency testing, user studies, and quantitative comparisons with conventional workflows have not yet been conducted. Although task-specific prompts and validation rules are applied to improve execution reliability, ambiguous or incomplete user input may still lead to inconsistent interpretation results. Furthermore, the current prototype focuses primarily on orthogonal layouts for container-type modular units and does not yet incorporate code compliance checking or deeper integration with downstream construction processes.

These observations suggest that the system should currently be understood as a proof-of-concept prototype rather than a fully deployable practical platform. Its main contribution lies in demonstrating that LLM-based requirement interpretation, reusable design knowledge, and BIM-based model generation can be organized into a coherent early-stage design workflow. This provides a meaningful foundation for subsequent development toward more robust, scalable, and practically applicable modular design support systems.

Future work should therefore focus on strengthening both the technical robustness and practical applicability

of the workflow. One important direction is the expansion and systematic organization of the historical case library. A broader and more diverse collection of cases would improve retrieval quality, enhance recommendation stability, and support more reliable reuse of spatial units across different design scenarios. Another important direction is the adoption of more rigorous evaluation methods, including time-efficiency analysis, controlled user experiments, and comparisons with conventional design workflows, in order to better assess the actual benefits of the system in practice. Such studies would also help clarify its effectiveness for users with different levels of design expertise. In addition, the interpretation of natural-language input can be further improved through stronger rule-based validation, better handling of vague or approximate expressions, and more advanced semantic reasoning mechanisms. In the longer term, the workflow may be extended beyond single-unit layouts to support larger-scale planning scenarios, such as multi-unit trailer arrangements and mixed-function camp configurations, together with building regulation checking, structural considerations, and mechanical-electrical-plumbing (MEP) integration.

5 Conclusion

This study proposed an LLM-integrated BIM workflow for rapid early-stage layout generation of worksite trailers. By connecting natural-language requirement interpretation with structured parameter processing and BIM-based modeling, the workflow supports the creation of editable preliminary configurations while also enabling the reuse and adjustment of prior design knowledge.

The prototype implementation confirms the feasibility of this approach and suggests that language-driven BIM workflows can provide a useful foundation for improving accessibility, efficiency, and design reuse in modular construction. Overall, this research offers an initial step toward more responsive and knowledge-supported design methods for worksite trailer planning.

References

[1] Smith, R. E. (2010). *Prefab Architecture: A Guide to Modular Design and Construction*. John Wiley & Sons.

[2] Gutiérrez, N., Negrão, J., Dias, A., and Guindos, P. (2024). Bibliometric Review of Prefabricated and Modular Timber Construction from 1990 to 2023: Evolution, Trends, and Current Challenges. *Sustainability*, 16(5), 2134.

[3] Marchesi, M., and Matt, D. T. (2017). Design for Mass Customization: Rethinking Prefabricated Housing Using Axiomatic Design. *Journal of Architectural Engineering*, 23(3), 05017004.

[4] Park, D. Y., Choi, J., Ryu, S., and Kim, M. J. (2022). A User-Centered Approach to the Application of BIM in Smart Working Environments. *Sensors*, 22(8), 2871.

[5] Hedayati, E., Kolaei, A. Z., Khanzadi, M., and Amiri, G. G. (2025). Implementation of hardware technologies in offsite construction (2014–2023). *Automation in Construction*, 170, 105948.

[6] World Economic Forum. (2016). *Shaping the Future of Construction: A Breakthrough in Mindset and Technology*. World Economic Forum.

[7] Wu, C., Li, X., Guo, Y., Wang, J., Ren, Z., Wang, M., and Yang, Z. (2022). Natural language processing for smart construction: Current status and future directions. *Automation in Construction*, 134, 104059.

[8] Gbadamosi, A.-Q., Mahamadu, A.-M., Oyedele, L. O., Akinade, O. O., Manu, P., Mahdjoubi, L., and Aigbavboa, C. (2019). Offsite construction: Developing a BIM-Based optimizer for assembly. *Journal of Cleaner Production*, 215, 1180–1190.

[9] Leng, S., Zhou, Y., Dupty, M. H., Lee, W. S., Joyce, S. C., and Lu, W. (2023). Tell2Design: A Dataset for Language-Guided Floor Plan Generation. *arXiv* preprint arXiv:2311.15941.

[10] Bakhshi, S., Chenaghlou, M. R., Rahimian, F. P., Edwards, D. J., and Dawood, N. (2022). Integrated BIM and DfMA parametric and algorithmic design based collaboration for supporting client engagement within offsite construction. *Automation in Construction*, 133, 104015.

[11] Qiu, Z., Liu, J., Wu, Y., Liu, P., Qi, H., Liang, H., and Xia, Y. (2025). LLM-based framework for automated and customized floor plan design. *Automation in Construction*, 180, 106512.