Sustainable Built Environment, Vol. 225*,* 2026, 1–11
https://doi.org/10.54941/ahfe1007893

D-CodeWeaver: Integrating Real-Time Code Compliance Analytics into Modular Timber Housing Design

Halil Erhan[1], Esmaeil Mottaghi[1], Elton Gjata[2], and Yann Tregoat[2]

[1]Simon Fraser University, British Columbia, Canada
[2]Perkins&Will Vancouver, British Columbia, Canada

ABSTRACT

Regulatory constraints play a conclusive role in shaping housing design outcomes, yet their operational integration into early-stage workflows remains limited. This paper introduces D-CodeWeaver, an interactive computational design system that incorporates prioritized regulatory logic directly into parametric design environments to support early-stage decision-making in modular housing projects. Rather than pursuing comprehensive automated code compliance, D-CodeWeaver formalizes selected high-impact regulatory requirements as parametric design rules that operate alongside geometric modelling. The system enables designers to generate modular building blocks, allocate residential unit mixes, and receive continuous spatial and analytical feedback on regulatory implications as design configurations evolve. Developed in collaboration with industry partners, the system integrates modular massing logic and unit-distribution strategies within a unified design analytics interface. By evaluating both building-level and site-level conditions during design exploration, D-CodeWeaver supports rapid “what-if” reasoning without relying on external compliance-checking tools. This work contributes a human-centred approach to regulatory integration, demonstrating how regulatory knowledge can serve as a design-support mechanism that enhances early-stage, data-informed exploration while preserving the designer’s control.

Keywords: Modular timber housing, Real-Time compliance design analytics, Parametric rule integration, Algorithmic unit distribution, Human-Centred regulatory integration

INTRODUCTION

Building regulations fundamentally shape the built environment, dictating critical parameters of form, density, safety, and spatial organization. While adherence to these constraints is mandatory, integrating regulatory reasoning into early-stage design workflows remains a persistent challenge in architectural practice. In conventional workflows, compliance is frequently treated as a post-design verification step, forcing designers to make foundational spatial decisions without clear visibility into their regulatory implications. Additionally, understanding the range of possible parameters requires deep subject-matter expertise. Junior designers might lack this experience, while designers at any level may not be aware of what’s feasible to create. This latency introduces significant uncertainty, often resulting in

Received February 6, 2026; Revised April 3, 2026; Accepted April 22, 2026; Available online July 20, 2026

costly late-stage redesigns when initial concepts fail to meet codified safety or zoning standards.

This disconnect is particularly detrimental in the context of modular and industrialized housing, where project viability relies heavily on rapid iteration, scalable solutions, and the precise coordination of repetitive unit layouts. In these high-speed delivery models, separating design authoring from compliance verification creates a critical bottleneck. When regulatory reasoning is isolated from the design environment, architects must mentally translate abstract legal texts into geometric configurations, a process that increases cognitive load and inhibits exploratory “what-if” reasoning. Consequently, regulations are often perceived solely as restrictive boundaries rather than as generative parameters that can actively inform design possibilities. Furthermore, it’s not only about regulations but also about knowledge and capabilities to understand the manufacturing-focused constraints they impose.

This paper proposes a shift in this approach, arguing that early-stage regulatory reasoning should be spatial, interactive, and selective rather than exhaustive or fully automated. Instead of aiming for comprehensive code coverage, we argue that integrating high-impact constraints directly into the modelling environment can enhance decision-making without limiting creativity. To demonstrate this approach, we present D-CodeWeaver, an interactive computational design system that embeds prioritized regulatory logic within a parametric workflow for modular timber housing. By creating a unified environment for site configuration, massing generation, and unit distribution, the system provides real-time visual feedback on regulatory and performance conditions. This work contributes a human-centred framework for compliance, demonstrating how regulatory knowledge can serve as a design-support mechanism that enhances data-informed exploration while preserving designer control.

BACKGROUND AND MOTIVATION

Compliance with building regulations is a fundamental requirement in the Architecture, Engineering, and Construction (AEC) industry, as it ensures safety, accessibility, sustainability, and overall building performance. However, compliance checking (CC) necessitates evaluating design proposals against complex regulatory frameworks that vary across jurisdictions, building typologies, and approval stages. Due to their complexity and ambiguity, regulatory constraints are often addressed late in the design process, increasing the likelihood of redesign, approval delays, and reduced design flexibility (Pauwels et al., 2017; Preidel & Borrmann, 2018; Nowak et al., 2023). This challenge is particularly evident in housing and modular construction contexts, where repetitive unit layouts, fast project timelines, and cost constraints require feasibility assessment and iterative decision-making.

To reduce the cognitive and operational demands of manual compliance review, Automated Code Compliance Checking (ACC) has been widely investigated in the AEC domain (Dimyadi & Amor, 2013). ACC systems aim to compare digital building models against regulatory constraints using computational methods to reduce time, cost, and error (Eastman et al., 2009; Kasim et al., 2013; Nguyen & Kim, 2013). Early ACC research focused on encoding regulatory requirements into executable rule structures, such as decision tables and expert systems. While these approaches demonstrated the feasibility of automated checking, they required extensive manual effort to encode, maintain, and update regulatory knowledge (Fenves et al., 1995; Eastman et al., 2009). Furthermore, such systems were often implemented as hard-coded rule engines, making them difficult to adapt to evolving regulations and reducing transparency in regulatory interpretation (Fenves et al., 1995; Preidel & Borrmann, 2018).

The subsequent adoption of Building Information Modelling (BIM) enabled more structured compliance checking through standardized building representations and rule execution frameworks. Industry Foundation Classes (IFC) became a central data schema for BIM-based compliance checking, supporting interoperability and standardized rule evaluation (Eastman et al., 2008; Preidel & Borrmann, 2018). National-scale implementations, such as Singapore's CORENET system, have successfully demonstrated the practical benefits of automating well-defined constraints using BIM/CAD models (Khemlani, 2005).

Despite these technological advancements, most ACC systems remain oriented toward post-design verification rather than early-stage design exploration (Dimyadi & Amor, 2013; Nowak et al., 2023). Existing ACC workflows are typically implemented as external checking systems that operate separately from core design environments (Preidel & Borrmann, 2018; Eastman et al., 2009). Designers are required to transfer design intent into compliance tools, interpret system feedback, and revise designs accordingly. This separation heightens cognitive demands and redirects designers from exploratory ideation to the administration of compliance processes (Nowak et al., 2023). As a result, regulatory constraints are often treated as restrictive conditions rather than interactive design parameters, limiting opportunities for rapid "what-if" exploration, scenario comparison, and creative trade-off evaluation.

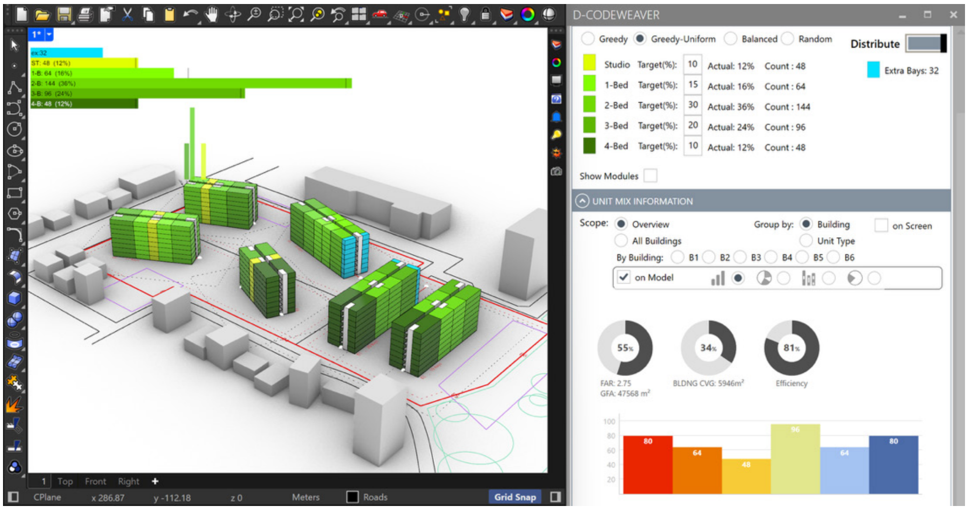
These limitations motivate the need for compliance approaches that are directly integrated into design environments and function as assistive systems rather than comprehensive or fully automated verification tools. Embedding prioritized regulatory constraints within the design workflow can support early-stage decision-making, enable rapid "what-if" exploration, and preserve designers' focus on conceptual development—particularly in modular housing contexts, where repeatability and scalability amplify the value of early regulatory awareness.

D-CODEWEAVER: OVERVIEW

In collaboration with industry partners, D-CodeWeaver was developed to embed regulatory compliance, industry practices, and manufacturing considerations directly into early-stage modular housing design workflows. The system provides real-time code feedback inside a parametric modelling environment, supporting rapid, creative, and data-informed exploration while emphasizing sustainability, constructability, and flexibility. Rather than offering comprehensive automated code checking, D-CodeWeaver integrates prioritized regulatory requirements as modular parametric design rules. This enables designers to focus on high-impact constraints, such as massing limits, density, spacing, and fire-related conditions, during early design phases, while deferring detailed verification to later stages. Compliance is thus treated as an assistive and exploratory design feature rather than a post-design validation task.

System Design

D-CodeWeaver is developed as an interactive design analytics system that integrates modelling, rule-based evaluation, and data visualization to support early-stage design decision-making (**Figure 1**). Rather than separating modelling, regulatory evaluation, and performance assessment into distinct tools, D-CodeWeaver provides a unified interface that enables designers to iteratively specify, modify, and evaluate design configurations.


D-CODEWEAVER

○ Greedy ● Greedy-Uniform ○ Balanced ○ Random

🟨	Studio	Target(%):	10	Actual: 12%	Count : 48
🟩	1-Bed	Target(%):	15	Actual: 16%	Count : 64
🟩	2-Bed	Target(%):	30	Actual: 36%	Count : 144
🟩	3-Bed	Target(%):	20	Actual: 24%	Count : 96
🟩	4-Bed	Target(%):	10	Actual: 12%	Count : 48

Extra Bays: 32

Show Modules ☐

UNIT MIX INFORMATION

Scope: ● Overview ○ All Buildings

By Building: ○ B1 ○ B2 ○ B3 ○ B4 ○ B5 ○ B6

Group by: ● Building ○ Unit Type

☐ on Screen

☑ on Model

FAR: 2.75

GFA: 47568 m²

BLDNG CVG: 5946m²

Efficiency

Figure 1: D-CodeWeaver is a system integrated into CAD modelling tools (Rhino + Grasshopper) to generate designs for modular high-rise timber housing that are industry-driven, construction- and regulation-ready.

Interface Overview

The D-CodeWeaver interface consists of three main tabs:

- Context for site setup and geometric constraints
- Code for regulatory parameters and compliance controls
- Data for unit distribution and performance analytics.

Each tab groups related controls and visual outputs, allowing designers to focus on specific aspects of the design problem while maintaining awareness of the overall system state.

Context Tab: Site and Geometric Configuration

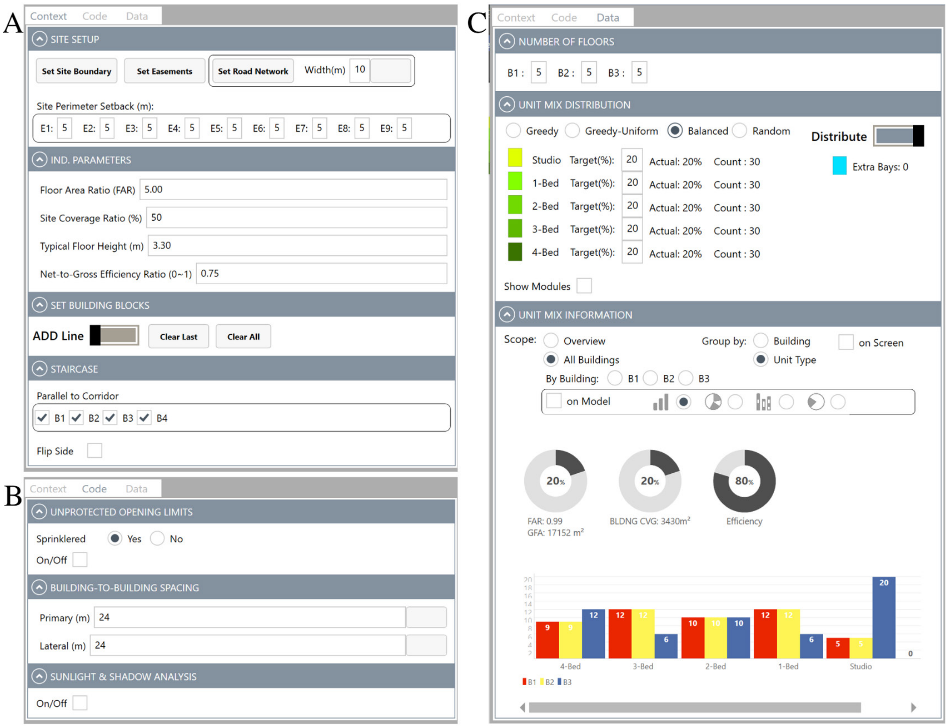
The Context Tab supports the initial specification of site conditions and geometric constraints that define the buildable design space (**Figure 2**). Through this interface, designers can establish the spatial context in which regulatory compliance and building generation are evaluated.

Key interface components in this tab include:

Site Setup: Enables defining site boundaries, easements, and road networks. Designers select or draw geometries directly in the model and register them through interface controls.

Per-Edge Setback Controls: Numeric input fields allow specification of setback distances for each site edge, enabling irregular site configurations and precise regulatory compliance.

Zoning Parameters: Allows designers to enter zoning and development parameters, including Floor Area Ratio (FAR), site coverage ratio, typical floor height, and net-to-gross efficiency ratio. These parameters define maximum allowable limits for building generation and compliance evaluation.



Width(m) 10

Site Perimeter Setback (m):

E1: 5 E2: 5 E3: 5 E4: 5 E5: 5 E6: 5 E7: 5 E8: 5 E9: 5

Floor Area Ratio (FAR) 5.00

Site Coverage Ratio (%) 50

Typical Floor Height (m) 3.30

Net-to-Gross Efficiency Ratio (0~1) 0.75

Parallel to Corridor

☑ B1 ☑ B2 ☑ B3 ☑ B4

Flip Side ☐

Sprinklered ● Yes ○ No

On/Off ☐

Primary (m) 24

Lateral (m) 24

On/Off ☐

B1 : 5 B2 : 5 B3 : 5

○ Greedy ○ Greedy-Uniform ● Balanced ○ Random

Studio Target(%): 20 Actual: 20% Count : 30

1-Bed Target(%): 20 Actual: 20% Count : 30

2-Bed Target(%): 20 Actual: 20% Count : 30

3-Bed Target(%): 20 Actual: 20% Count : 30

4-Bed Target(%): 20 Actual: 20% Count : 30

Extra Bays: 0

Show Modules ☐

Scope: ○ Overview ● All Buildings

By Building: ○ B1 ○ B2 ○ B3

Group by: ○ Building ● Unit Type

☐ on Screen

☐ on Model ● ○ ○ ○

Figure 2: A) Context tab illustrating site setup, zoning parameters, building block generation, and staircase configuration controls. B) Code tab showing regulatory controls for floor limits, unprotected opening limits, building spacing, and environmental analysis. C) Data tab indicating unit mix controls and real-time performance analytics visualizations.

Building Block Setup: Provides tools for generating modular building blocks by allowing the user to draw guide lines that define the starting point and direction of each block's central corridor within the buildable envelope.

Based on these guides, the system automatically produces double-loaded modular masses, trims blocks that exceed bay limits or intersect non-buildable areas and assigns default vertical cores and floor counts when drawing concludes (**Figure 3**).

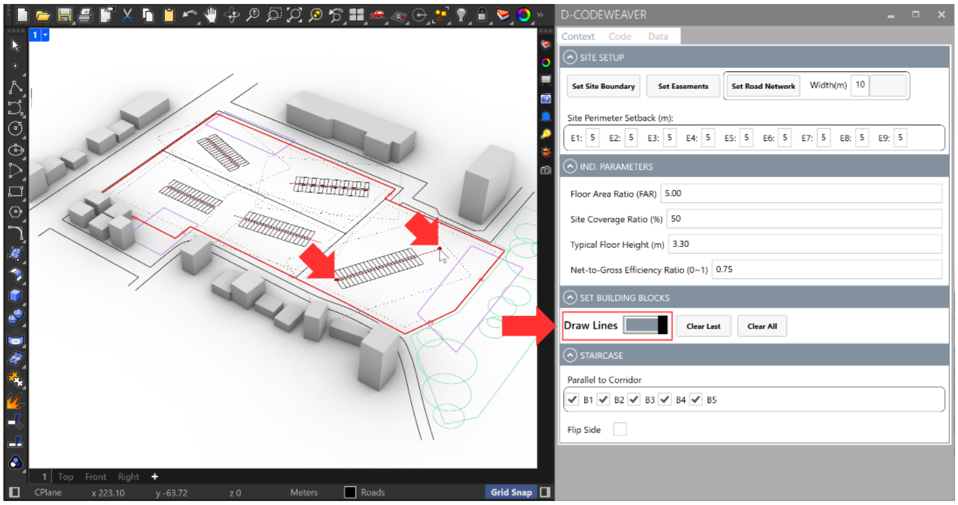
Staircase Configuration: Enables selection of staircase orientation (e.g., parallel to the corridor) and flipping of staircases for selected building blocks, to support circulation layouts. This tab establishes the geometric and contextual foundation for all subsequent compliance checks and performance analyses.

Code Tab: Regulatory Parameters and Compliance Controls

Instead of hidden automated validations, the Code Tab transforms regulatory constraints into clear, interactive parameters that designers can actively manage (**Figure 4**). The main interface elements include:

Unprotected Opening Limits: Provides controls to enable or disable unprotected opening limit evaluation and to specify sprinkler conditions. Based on these inputs, the system determines the maximum allowable percentage and area of unprotected openings for each façade, as a function of boundary distance and sprinkler condition, in compliance with applicable building codes.

Building-to-Building Spacing: Enables the specification of primary and lateral spacing between building blocks. The system generates clearance buffers and removes or flags non-compliant building configurations.



Width(m) 10

Site Perimeter Setback (m):

E1: 5 E2: 5 E3: 5 E4: 5 E5: 5 E6: 5 E7: 5 E8: 5 E9: 5

Floor Area Ratio (FAR) 5.00

Site Coverage Ratio (%) 50

Typical Floor Height (m) 3.30

Net-to-Gross Efficiency Ratio (0~1) 0.75

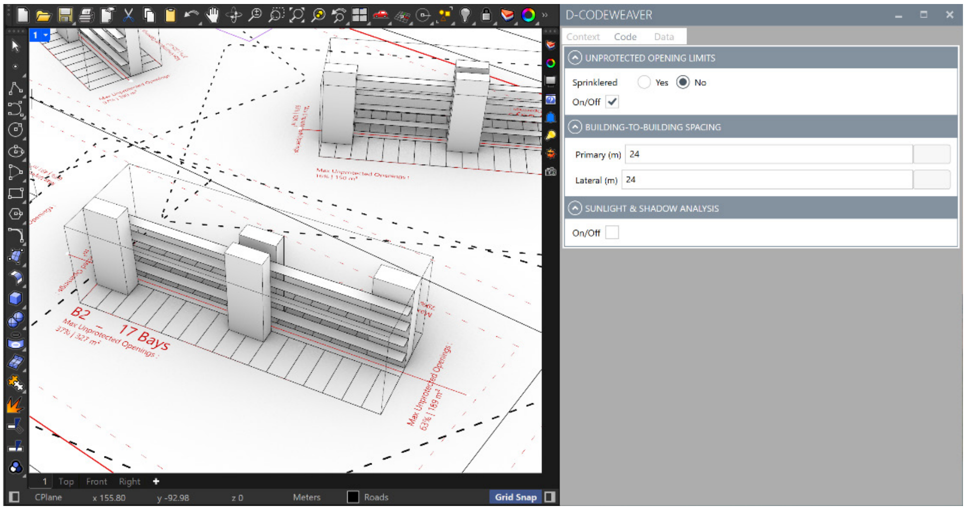
Parallel to Corridor

☑ B1 ☑ B2 ☑ B3 ☑ B4 ☑ B5

Flip Side ☐

Figure 3: Building Block Setup Panel of D-CodeWeaver illustrating guideline–based block generation and automatic trimming of over-limit bays.

Sunlight and Shadow Analysis: Allows designers to perform solar analysis within the design environment to evaluate environmental constraints. This feature provides a heatmap-based visualization of sunlight exposure for each building block and highlights the effects of shadows on adjacent structures. The Code Tab transforms regulatory requirements into adjustable design parameters, encouraging exploratory interaction rather than late-stage verification.



D-CODEWEAVER

UNPROTECTED OPENING LIMITS

Sprinklered ○ Yes ● No

On/Off ☑

BUILDING-TO-BUILDING SPACING

Primary (m) 24

Lateral (m) 24

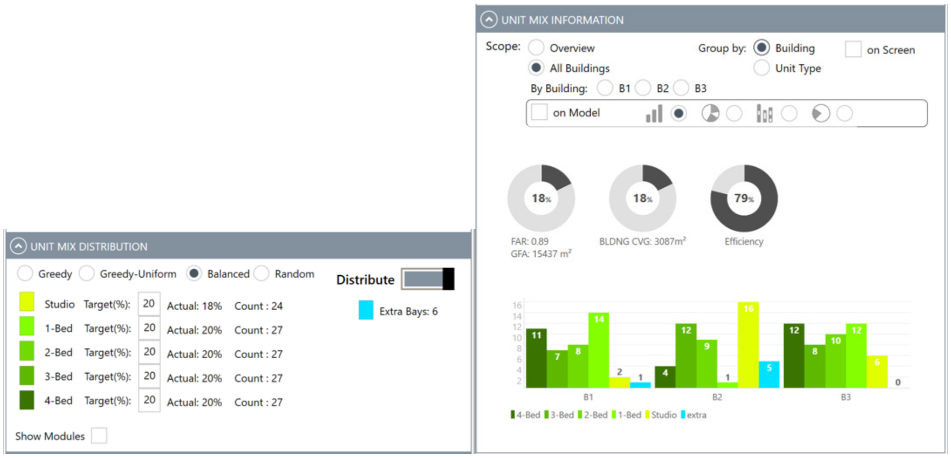
SUNLIGHT & SHADOW ANALYSIS

On/Off ☐

Figure 4: Unprotected Opening Limits are set and controlled to evaluate fire-control choices.

Data Tab: Design Analytics and Visualization

The Data Tab enables designers to define and manage unit-mix distribution targets and outcomes and provides an integrated analytics interface for reviewing and interpreting performance and compliance metrics derived from the current design state. It allows designers to define target unit-type percentages and apply four different distribution strategies, with the system calculating actual distributions and highlighting deviations from targets. This tab also supports multi-scale evaluation through interactive data visualizations. (**Figure 5**).



UNIT MIX DISTRIBUTION

○ Greedy ○ Greedy-Uniform ● Balanced ○ Random

Studio	Target(%):	20	Actual: 18%	Count : 24
1-Bed	Target(%):	20	Actual: 20%	Count : 27
2-Bed	Target(%):	20	Actual: 20%	Count : 27
3-Bed	Target(%):	20	Actual: 20%	Count : 27
4-Bed	Target(%):	20	Actual: 20%	Count : 27

Extra Bays: 6

Show Modules ☐

UNIT MIX INFORMATION

Scope: ○ Overview ● All Buildings

Group by: ● Building ○ Unit Type

☐ on Screen

By Building: ○ B1 ○ B2 ○ B3

☐ on Model

FAR: 0.89
GFA: 15437 m²

BLDNG CVG: 3087m²

Efficiency

Figure 5: (Left) Unit Mix Distribution Panel for setting target unit mixes and visualizing actual distributions across different allocation strategies. (Right) Unit Mix information panel for switching between site-wide and per-building performance views, with interactive charts showing FAR, coverage, efficiency, and unit distributions.

Primary features include:

Number of Floors Panel: Allows designers to set the number of floors independently for each generated building block. The system validates these inputs against maximum permitted limits and egress distance requirements, adjusts them to the nearest allowable value if necessary, and updates building efficiency and gross floor area calculations in real time (**Figure 2** C).

Unit Mix Distribution Panel: This panel enables designers to specify target percentages for each residential unit type and to select among four distribution strategies. Based on the chosen strategy, the system allocates units across all generated building blocks, computes the resulting actual unit mix, and highlights deviations from the specified targets (**Figure 5** Left).

A greedy allocation (Cormen et al., 2009) strategy was adopted as the foundational approach due to its computational efficiency, deterministic behaviour, and suitability for real-time design interaction, enabling unit assignments to be made incrementally without global optimization overhead. Within this framework, a round-robin allocation scheme (Brucker, 2007) was employed to cyclically distribute units across building blocks, preventing early concentration and supporting balanced site-wide distribution.

For capacity-driven scenarios, a First-Fit Decreasing (Johnson, 1973; Coffman et al., 1996) heuristic was applied, prioritizing larger units during allocation to maximize capacity utilization. Together, these strategies allow design intentions to be encoded directly in the allocation logic, while remaining simple and fast enough for interactive early-stage design. The available strategies include:

Mode 1 - Round-Robin Greedy Allocation: Distributes units cyclically across blocks to achieve a balanced site-wide mix. Larger units are positioned toward the ends of each block sequence.

Mode 2 - Fixed Layout Round-Robin Greedy Allocation: Uses the same allocation logic as Mode 1 but applies a consistent unit arrangement across all floors of each building block.

Mode 3 - First-Fit Decreasing Greedy Allocation: Prioritizes placing larger units on upper floors using a first-fit decreasing approach to maximize capacity utilization.

Mode 4 - Randomized Greedy Allocation: Builds on Mode 3 while adding randomized ordering to generate diverse spatial configurations under the same quantitative targets.

The analytics interface provides control over data visualizations for block-by-block, aggregate, or clustered analyses of unit distributions (**Figure 5** Right).

Multi-Scope Visualization Controls: This part allows designers to switch between aggregated and disaggregated analytical views, including site-wide summaries, cross-building comparisons, and detailed per-building inspections. By restructuring performance data across multiple levels of abstraction, the system supports both high-level assessment and focused evaluation of individual blocks.

Interactive Charts and Indicators: The system provides cross-referenced charts and quantitative indicators that visualize unit distributions, units per floor, FAR utilization, building coverage, and efficiency ratios. These visualizations update dynamically in response to design modifications, maintaining continuous alignment between analytical outputs and evolving design parameters.

On-Model Visualization Toggle: This feature enables the projection of analytical information directly onto the 3D model as annotated overlays. Performance charts and indicators are spatially aligned with their corresponding building elements, allowing designers to interpret quantitative metrics within their geometric and spatial context (**Figure 6**). The Data Tab supports reflective decision-making by allowing designers to evaluate regulatory compliance, spatial efficiency, and functional performance simultaneously.

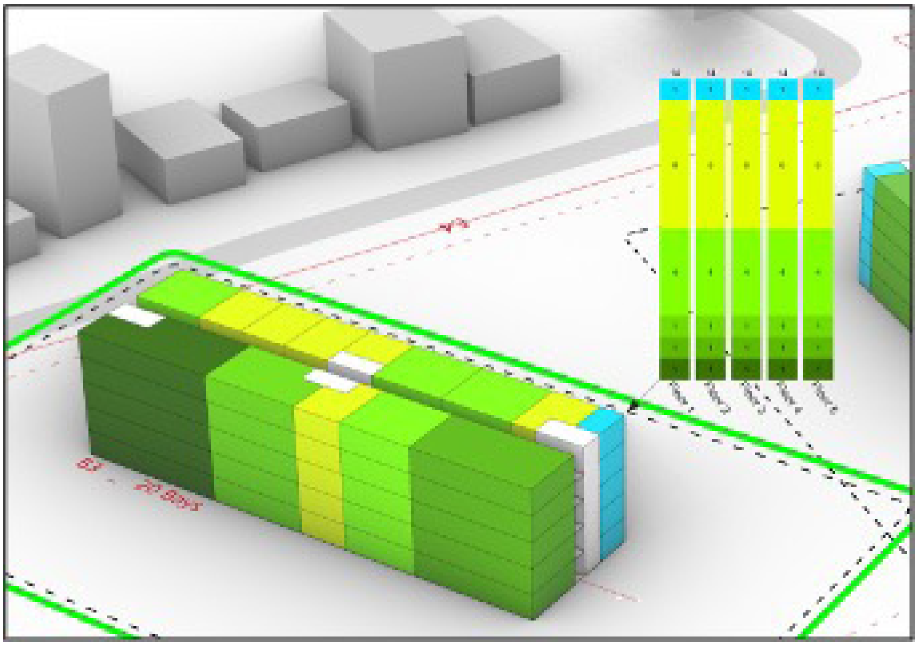


Figure 6: On-Model Visualization: Performance metrics and unit distributions are projected as annotated overlays directly onto the 3D building model.

CONCLUSION

This paper presented D-CodeWeaver, an interactive design analytics system that integrates prioritized regulatory logic directly into early-stage parametric environments for modular timber housing. By shifting compliance from a post-design verification task to an active design parameter, the system enables architects to evaluate feasibility, massing limits, and unit distributions in real-time. This approach addresses the cognitive disconnect in traditional workflows, in which the separation of design and compliance tools often hinders exploratory "what-if" reasoning and delays critical decision-making.

By employing greedy allocation algorithms and unified visual analytics, the system demonstrates that regulatory knowledge can function as a generative support mechanism rather than a restrictive boundary.

However, this research represents an ongoing effort to bridge the gap between computational design and regulatory practice. While the current prototype successfully formalizes high-impact constraints and demonstrates technical feasibility, a full understanding of its impact on professional practice requires rigorous empirical validation. Future work will focus on evaluating D-CodeWeaver through structured studies with expert practitioners to assess its usability and impact on cognitive load. Furthermore, we plan to test the system on real-world project data from our industry partners to verify its scalability and adaptability to complex, site-specific modular housing scenarios. These next steps will be critical in refining the system's interaction model and establishing a robust framework for human-centred regulatory integration in the AEC industry.

ACKNOWLEDGMENT

This work was supported by the DIGITAL Housing Growth Innovation (HGI) program of Canada [412430] and by NSERC Discovery (2022-03108, H. Erhan, PI). The authors declare that GenAI Grammarly was used for editorial support.

REFERENCES

Brucker, P. (2007). Scheduling Algorithms (5th ed.). Springer.

Cheng, C. P., Lau, G. T., & Law, K. H. (2007). Mapping regulations to industry-specific taxonomies. In Proceedings of the 11th International Conference on Artificial Intelligence and Law (ICAIL '07) (pp. 59–63). Association for Computing Machinery.

Coffman, E. G., Garey, M. R., & Johnson, D. S. (1996). Approximation algorithms for bin packing: A survey. Journal of Approximation Theory, 7, 197–224.

Cormen, T. H., Leiserson, C. E., Rivest, R. L., & Stein, C. (2009). Introduction to algorithms (3rd ed.). MIT Press.

Dimyadi, J., & Amor, R. (2013). Automated building code compliance checking: Where is it at? In Proceedings of CIB WBC 2013 (pp. 1–10).

Eastman, C. M., Lee, J., Jeong, Y., & Lee, J. (2009). Automatic rule-based checking of building designs. Automation in Construction, 18(8), 1011–1033.

Eastman, C. M., Teicholz, P., Sacks, R., & Liston, K. (2008). BIM handbook: A guide to building information modeling for owners, managers, architects, engineers, contractors, and fabricators. John Wiley & Sons.

El Kharbili, M. (2012). Business process regulatory compliance management solution frameworks: A comparative evaluation. In Proceedings of the Eighth Asia-Pacific Conf. on Conceptual Modelling (Vol. 130, pp. 23–32). Australian Computer Society.

Fenves, S. J., Garrett, J. H., Kiliccote, H., Law, K. H., & Reed, K. A. (1995). Computer representations of design standards and building codes: US perspective. International Journal of Construction Information Technology, 3(1), 13–34.

Hjelseth, E., & Nisbet, N. (2011). Capturing normative constraints by use of the RASE methodology. In Proceedings of the CIB W78-W102 Conference.

Kasim, T., Li, H., Rezgui, Y., & Beach, T. (2013). Automated sustainability compliance checking process: Proof of concept. In Proceedings of the 13th International Conference on Construction Applications of Virtual Reality (pp. 11–21). Teesside University.

Khemlani, K. (2005). CORENET e-PlanCheck: Singapore's automated code checking system. AECBytes.

Nguyen, T., & Kim, J. (2011). Building code compliance checking using BIM technology. In Proceedings of the 2011 Winter Simulation Conference (pp. 3400–3405). IEEE.

Nowak, S., Aseniero, B. A., Bartram, L., Grossman, T., Fitzmaurice, G., & Matejka, J. (2023). Identifying visualization opportunities to help architects manage the complexity of building codes. IEEE Computer Graphics and Applications, 43(6), 25–38.

Oxel, F. (1998). Life safety issues in hotel/casino occupancies. In Proceedings of the 2nd International Fire Research and Engineering Conference. National Institute of Standards and Technology.

Pauwels, P., de Farias, T. M., Zhang, C., Roxin, A., Beetz, J., De Roo, J., & Nicolle, C. (2017). A performance benchmark over semantic rule-checking approaches in the construction industry. Advanced Engineering Informatics, 33, 68–88.

Pauwels, P., Zhang, C., & Lee, Y.-C. (2011). Semantic web technologies in AEC industry: A literature overview. Automation in Construction, 20(7), 829–843.

Preidel, C., & Borrmann, A. (2018). BIM-based code compliance checking. In A. Borrmann, M. König, C. Koch, & J. Beetz (Eds.), Building information modeling: Technology foundations and industry practice (pp. 367–381). Springer.

Solihin, W., & Eastman, C. M. (2016). Classification of rules for automated BIM rule checking development. Automation in Construction, 68, 69–82.

Yurchyshyna, A., & Zarli, A. (2009). Ontology-based approach for formalization and semantic organization of conformance requirements in construction. Automation in Construction, 18(8), 1084–1098.

Zhang, J., & El-Gohary, N. M. (2016). Semantic NLP-based information extraction from construction regulatory documents for automated compliance checking. Journal of Computing in Civil Engineering, 30(2), 04015014.

Zhang, J., & El-Gohary, N. M. (2017). Integrating semantic NLP and logic reasoning into a unified system for fully automated code checking. Automation in Construction, 73, 45–57.

Zhang, S., Teizer, J., Lee, J.-K., Eastman, C. M., & Venugopal, M. (2022). Building information modeling (BIM) and safety: Automatic safety checking of construction models and schedules. Automation in Construction, 133, 103990.