2026 European Conference on Computing in Construction
Corfu, Greece
July 1215, 2026

LOW-CODE TOOLCHAIN DEVELOPMENT TO ACCELERATE DATA-DRIVEN ENERGY MODELLING

Mariam Ali[1], Cathal Hoare[1], Gabriel Oyeyemi[1], Divyanshu Sood[2], Tiziana Margaria[1] and James O'Donnell[3]
[1]Department of Computer Science and Information Systems, University of Limerick, Ireland
[2]School of Material and Mechanical Engineering, University College Dublin, Ireland
[3]School of Engineering, University of Limerick, Ireland

Abstract

Building operations account for 30\% of global energy consumption, necessitating performant, multi-scale modelling to optimize carbon reduction. While Machine Learning (ML) offers significant promise, significant issues prevent its full adoption. This paper proposes a specific Low-Code/No-Code (LCNC) approach as a solution to democratize analytics software development, allowing stakeholders to focus on their operation needs while removing the programming syntax barrier. We show how we use the WAVE platform, a cloud-based, collaborative LCNC platform based on formal models, for the graphical composition of workflows for multi-scale energy modelling. A LCNC solution to national scale energy modelling is described and compared to traditional implementations.

Introduction

Building operations accounted for 30% of global energy consumption and 27% of energy-sector emissions in 2021 (Skillington et al., 2022). Data-driven decisions are now vital to address the issues of global warming, building comfort and stock availability (Katavoutas et al., 2022). Developing performant methods of modelling and analyzing the built environment across multiple scales, from individual buildings to national building stocks, is essential. Machine Learning (ML) offers a reliable and affordable mechanism to achieve this (Manfren et al., 2022). By fusing various information sources, such as building data, energy infrastructure and population data, and applying ML, planners can perform sophisticated analyses, to identify energy performance trends and optimize large-scale carbon reduction strategies (Ali et al., 2024).

Creating data-driven models using machine learning is non-trivial, particularly within the built environment where data are often fragmented and heterogeneous. As in any domain, a machine learning deployment workflow consists of four main stages: data management, model learning, model verification, and model deployment (Paleyes et al., 2022). In the context of energy modelling, these stages are complicated by the need to integrate disparate sources such as building models, demographic data and utility topologies.

Each stage requires a specific intersection of practitioner skill sets. For instance, data management in this sector requires not only data cleaning but an understanding of physical building properties to ensure that feature selection is based on domain fundamentals. Stakeholders in residential energy modelling report a significant AI Skills Gap (OECD, 2025). This is not merely a lack of coding ability, but a shortage of professionals who possess both energy-modelling knowledge and AI technical proficiency. For SMEs and Government agencies, this gap creates a bottleneck where the potential of data remains unfulfilled, leading to a disadvantage as they plan carbon reduction strategies (Muminova et al., 2024).

Low-Code/No-Code (LCNC) is a visual programming paradigm where developers define an application's logic through workflows, palettes of reusable component, and data models rather than text-based scripts. This paradigm emphasises the business case, which is assembled from pre-configured modules (O'Brien et al., 2025). LCNC provides several advantages for enterprises, including:

- *Democratisation of software development* - LCNC approaches remove the syntax barrier, the need to master programming syntax and, by extension, software engineering skills (Denny et al., 2011). Removing this barrier allows stakeholders in the energy modelling domain to build bespoke solutions. This paradigm shift results in applications and tools that are designed by those who understand the operational needs best.
- *Toolchain development* - by extension, LCNC facilitates the creation of toolchains and workflows where small data manipulation software components trans-form data as it is passed through a chain of established software tools.
- *Lower development costs, higher speed* - the cost of creating bespoke tools is reduced both by improving the ability to combine and reuse code, as well as overcoming the skills gap. This also brings much faster turn around times, as the design happens at the logical level, and changes are quickly made and quickly operational.

Typically, LCNC software is developed in platforms that provide development tools and frameworks (Prinz et al., 2021). Many flavours of LCNC aim at more or less professional outcomes. The WAVE platform is an example of a LCNC platform that follows the eXtreme Model Driven Development approach (Margaria and

Stef-fen, 2020). WAVE addresses many of these needs at the professional level needed for high assurance software. It is a cloud-based, collaborative application development environment that adopts a LCNC paradigm founded in formal models, with graphical composition of hierarchical workflows (control and data flow) on the basis of libraries of components that express the *skills* (or capabilities, or actions, as community-specific terminology) of the domain-specific entities to be coordinated. WAVE is domain-agnostic and employs formal methods to ensure workflow correctness, a capability absent in other platforms. This contrasts with other workflow orchestration tools such as Node-RED, which is limited to event-driven applications and KNIME which focuses on data science applications.

The paper introduces WAVE as a LCNC platform that can enable domain experts to build multi-scale energy models via graphical workflows instead of complex scripts. By showcasing specific workflows for archetype calculation and Building Energy Ratings (BER) prediction, we show that data-driven decarbonization efforts can be accelerated by abstracting from the technical MLOps' complexity. Having explored the background to this work, we introduce the WAVE platform and associated concepts. Finally, we present a use-case that focuses on the development of a multi-scale energy modelling solution.

Background

The need to decarbonize the built environment requires efficient methods for modeling building stocks across multiple scales, from individual thermal components to national energy infrastructure (Ali et al., 2024; Hoare et al., 2022). Historically, this field relied on physics-based approaches that use heat-transfer and thermodynamic equations to simulate physical behavior (Ahmad et al., 2018). However, these models require high-granularity data such as specific geometry and thermal properties (U-values), while being also computationally expensive (Ali et al., 2024; Ahmad et al., 2018).

To address these data gaps, data-driven models have gained prominence. These models leverage machine learning (ML) algorithms to predict energy use without requiring a deep understanding of modelled system (Ahmad et al., 2018; Manfren et al., 2022). To achieve multi-scale capabilities, researchers use Building Energy Modelling (BEM), which employ building archetypes, i.e., representative baseline models capturing typical characteristics of building groups, to scale individual estimates to regional or national levels (Ali et al., 2024). Furthermore, Linked Data approaches admit the integration of heterogeneous information sources, such as energy infrastructure and population data (Hoare et al., 2022). Recent advances have also integrated Geographic Information Systems (GIS) into these workflows to enhance spatial analysis of energy consumption patterns (Ali et al., 2024).

Despite the potential of ML, significant obstacles hinder its widespread adoption, particularly for Small and Medium Enterprises (SMEs) and government agencies. A primary barrier is the AI Skills Gap; firms often struggle to find personnel who possess both domain expertise and the specialized technical skills required to implement complex ML algorithms for energy models (Carioli et al., 2024; OECD, 2025). Smaller organizations often lack the financial resources to compete for data analysts familiar with energy modelling, leading to a competitive disadvantage and unfulfilled potential for the technology (Muminova et al., 2024; Dzhusupova et al., 2022).

Beyond personnel, the Machine Learning Deployment Workflow (MLOps) presents significant architectural and process challenges. Practitioners report that data management, including data discovery, cleaning, and joining dispersed datasets, consumes the vast majority of time in ML projects (Paleyes et al., 2022; Dzhusupova et al., 2022). Data often exist in *functional silos* making large-scale integration difficult (Paleyes et al., 2022; Dzhusupova et al., 2022). Additionally, the opaque nature of ML algorithms creates a *trust deficit* among stakeholders who require interpretability and formal verification to comply with high safety and regulatory standards (de Marco et al., 2025). *Low-Code/No-Code* (LC/NC) development is an emerging technology paradigm that enables individuals with no advanced computing or coding skills, collectively referred to as *Citizen Developers*, to create custom applications through visual drag-and-drop interfaces and pre-configured modules (El Kamouchi et al., 2023; Martinez and Pfister, 2023). By removing the *syntax barrier* these platforms allow domain experts to be closer and participate to code development on the basis of their own professional skills, thereby democratizing software development (Sundberg and Holmström, 2023).

The LC/NC paradigm has demonstrated success across several domains. It has been used to develop solutions for healthcare, for example diagnostic systems and virus tracking applications (Margaria and Schieweck, 2019). Similarly in manufacturing, the digital thread (Margaria and Schieweck, 2019) and automation of inventory management have been developed using LC/NC (El Kamouchi et al., 2023). It has also been used to support the development of decision support systems (Karusseit and Margaria, 2006) and for educational training at many levels (Gossen et al., 2018; O'Brien et al., 2025).

LC/NC offers a mechanism to overcome the skills gap by facilitating rapid tool-chain development (Sundberg and Holmström, 2023). These tools reduce implementation costs because organizations have their own staff to build bespoke models, bypassing the need for expensive IT consultants (Muminova et al., 2024). Platforms such as WAVE, and before it DIME (Boßelmann et al., 2016) and the jABC (Margaria and Steffen, 2009) address multi-scale modeling needs by utilizing hierarchical workflows that coordinate domain-specific skills into complex automated pipelines.

The WAVE architecture

The WAVE platform is described next. The architecture is summarized in Fig. 1. *The Formal Modelling Language* (orange layer in Fig. 1), provides the modelling languages

used in the platform. It is provided and maintained by the WAVE delveloper and support team. The Application Design layer (in blue in Fig. 1) is where the researchers from various communities use the WAVE facilities to design and execute their applications.

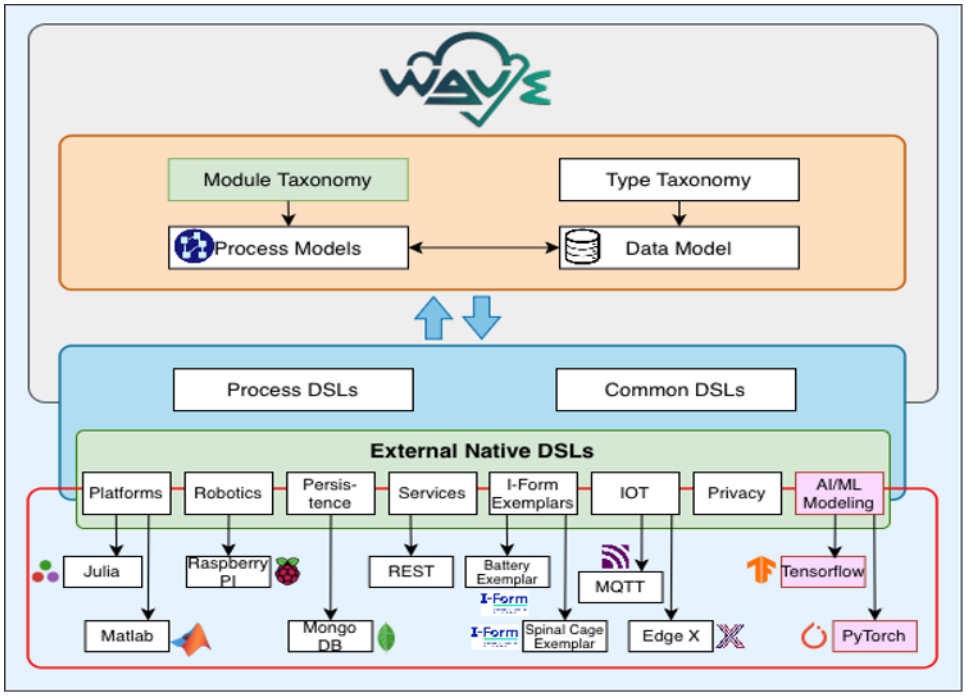

Figure 1: Architecture of the WAVE platform: the layers correspond roughly to different professional figures found in many communities

When LCNC is used to develop code, the final end-to-end logic that solves a business problem (traditionally coded as a script or program) is commonly called the workflow or process of the application under development and is modelled using compositions of components. This workflow describes how the control and the data flows from one step of the workflow to the other. The steps are designed using components called Service Independent Building Blocks (SIBs), each one embedding a domain specific functionality that executes a specific task. The task can range from simple operations, such as converting file formats, to complex ones like connecting to external APIs or training ML models. The complexity of API development, writing scripts for Python libraries such as Keras, scikit-learn, and TensorFlow, is fully abstracted. Users interact only with visual representations that expose the necessary configuration parameters. Built using the *Correctness by Construction* (CoC) approach, each SIB in the WAVE is a pre-validated block with established functional correct-ness and an associated model (Margaria et al., 2005) akin to a digital shadow. As shown in Figure 2, each SIB has well-defined, typed inputs and outputs. For each potential execution outcome there is a separate outgoing branch, labelled accordingly. The label also includes the outputs produced in that case. So even a simple visual inspection identifies what is produced (data, messages) for each specific outcome. Next steps are determined by the outcome of the current SIB, thus the outgoing branches are connected to the ”suitable” subsequent SIBs, defining the control flow. SIBs are thematically organized in palettes of related functionalities.

The collection of SIB palettes pertaining to a given application domain constitutes a Domain Specific Language (DSL) for that application domain. In our case, the DSL concerns the domain of energy consumption analysis, pre-diction and mapping.

The WAVE uses Software Engineering methods and techniques to enable researchers to design, implement and execute verified and tested applications by modelling their workflows using application-domain DSLs. If the DSL and SIBs for a particular workflow are available, designing new applications simply reuses those ready SIBs: a no-code activity. If functions are missing, they need to be either integrated from existing libraries or tools, or programmed afresh. This is the *External Native* DSL, in green in Fig. 1. There, existing or new software components, hardware, APIs or other resources (red layer in the Figure) are made available to the WAVE users as collecitons of SIBs. This is a low-code task, as it concerns a specific, well defined component to be added. This step requires some programming skills, but not complex architectural knowledge, and it is the step that produces the palettes of reusable components.

WAVE will employ formal methods like the GEAR model checker (Steffen et al., 2007) and the MC3 infinite state model checker (Steffen and Murtovi, 2018) to ensure workflow correctness at design time. Those model checkers are specialized for deep diagnostic support and procedural (i.e, hierarchical) systems respectively. While users are modelling their workflow, syntactic and static semantics check will ensure that workflows are valid. For example, verifying the absence of deadlocks and

guaranteeing that critical steps are always executed, or guaranteeing precedences. Additionally, the platform will offer support for integration of new components, execution of the workflows and adequate monitoring and logging mechanisms.

Using the WAVE

When researchers from various communities use the facilities offered by the WAVE platform to design and execute their applications, they reuse over and over again the general purpose components of the Common DSLs and the domain specific DSLs. This way, users produce the do-main and application specific workflows of the Process DSLs. As WAVE has the same common modelling language for its components and workflows, that is shared across any application domain, diverse research communities can share components with other communities, speed-ing up significantly the work of other users. For example, data analytics and visualization components and even workflows are frequently shared across several communities active in diverse research domains such as bioinformatics, history and advanced manufacturing.

Any applications, and many of the features of the WAVE platform itself, are implemented as workflows that use application domain specific libraries of components called AD-DSLs. The specialization to a domain does not require a new workflow language or composition model, but it simply happens by adding collections of skills through libraries of components that provide access to (interface) and execution (on an adequate runtime) of those skills. Robotics applications, data transcription applications, biomedical imaging applications (Brandon et al., 2024) have recently been realized this way.

A slightly more technical view of the components and workflows will be illustrated in the case study.

The feedback from the different communities of practice drives the growth and evolution of the platform itself. This happens through requests for additions, like recently the need of support for larger data files.

Integrating existing capabilities

The encapsulation of external resources providing data (databases, input streams), communication (http, wifi, 5G networks, ORAN), hardware (sensors, actuators, computation, robots), software (analytics libraries, graphics, simulators, geovisualisation systems) and models (AI models, recommenders, ontologies, classifiers, simulation models, digital twins) and other services follows API integration, or uses higher level community-driven services like e.g. the EdgeXfoundry services for edge IoT peripherals and computing tools.

In this way, it is possible to grow the capability of each community in according to their domain's needs, and thus happens in the context of ensured compatibility across communities. This is the key design decision that allows sharing and reuse across the entire WAVE.

Learning Support

LCNC is a new concept for many. The WAVE developers are designing and delivering training materials targeted to specific communities. For example, with education experts, they have produced both a *train the trainer program* (O'Brien et al., 2025) as well as training material on technical skills needed to use the platform. A MOOC on LCNC is currently under production. For example, topics such as hardware integration skills for robotics and advanced manufacturing researchers and workflow modelling skills for application designers have been developed.

Case study: predicting energy performance

A typical prediction methodology for energy performance of urban buildings involves the following steps:

1. Collecting data from various sources such as building stock, census, weather, and geographical data.
2. Developing building archetypes using building stock data to identify representative baseline models.
3. Parametric simulation is then conducted to develop appropriate synthetic data, followed by
4. the development of machine learning models to predict building energy performance.
5. Finally, the urban building energy performance analysis step analyzes the modeling process results for planning and decision-making purposes.

The data collection phase is non-trivial. In this country, it involves ingesting and harmonizing approximately 2.4 mil-lion residential records (in this instance, a GeoDirectory). Other heterogeneous data sources (census data from the national Central Statistics Office and Energy Performance Certificate (EPC) data from the national Sustainable Energy Authority (SEA)) are also ingested and fused with the building records to produce an overarching knowledge graph that captures both the individual data records and their relationships with other records in the graph.

When a similar methodology is applied to other jurisdictions, different data sources are ingested and fused, but the software processes for doing this are similar and suggest that code should be extensively reused with minor modification. However, in a traditional software environment, implementing this ingestion would require coordination between a domain expert (e.g., a housing analyst) and a software engineer to produce the modification. The WAVE platform replaces this requirement with the design of a workflow orchestrating SIBs according to shared, reusable components defined in a domain specific palette of SIBs. We show how to develop these workflows and related SIBs to demonstrate how the WAVE platform can be applied to develop software for multi-scale energy modelling.

Figure 2 represents (part of) the SIB library for multi-scale energy modelling within the WAVE platform. The library is organized into five distinct palettes: *Common*, *Data Ingestion*, *File Parsing and Formatting*, *Arch*

Count, and *Predict BER*. The *Common* palette contains general purpose SIBs: they are not specific to a given domain. The *Data Ingestion* palette contains SIBs to retrieve valid datasets from AWS, Azure, REST APIs and SFTP servers. Once the data is successfully stored into the local workspace, *File Parsing and Formatting* SIBs are used to homogenize data across file encodings and formats. During data engineering phase, *Arch Count* palette SIBs are used to query the Neo4J graph database and retrieve building counts whose features match those specified by the *ArcheTypesCombination* parameter. Finally, *Predict BER* SIBs are used in workflows that utilize pretrained ML models for predicting Building Energy Ratings.

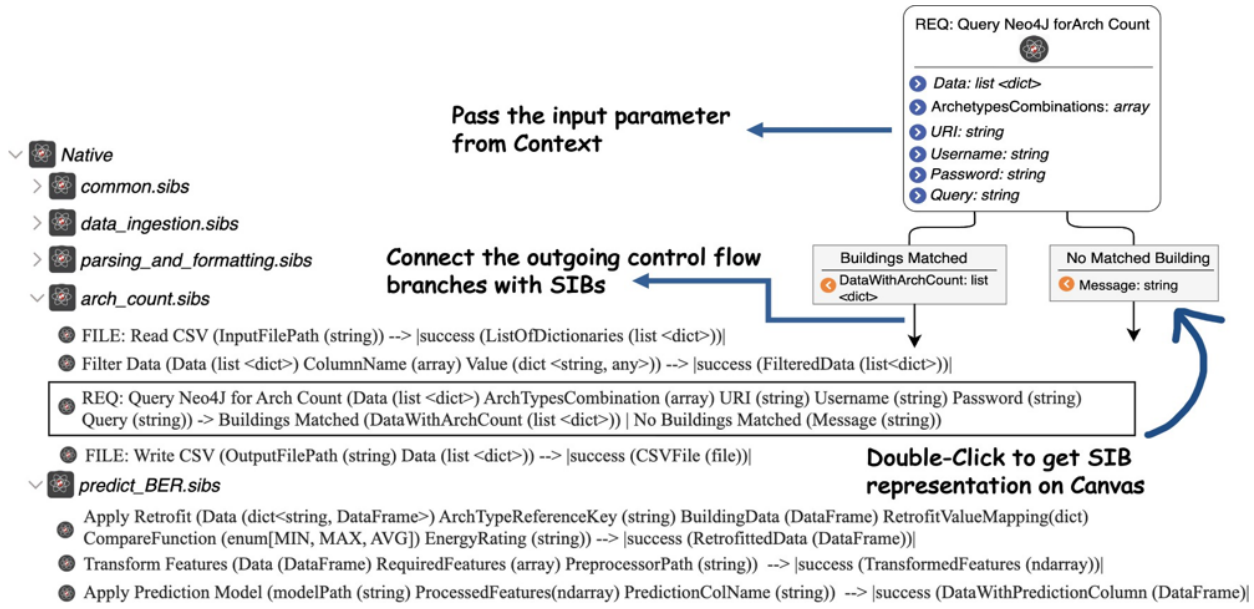

Figure 2:SIB transition from the SIB Library for multi-scale energy modelling into the WAVE Canvas

To create custom data ingestion pipelines, stakeholders select the appropriate SIBs from the palettes and use them to develop workflows. We show and discuss here two of the workflows defined for this case study: (a) Calculate Arch count (Neo4j connection and building count retrieval) and

(b) part of Predict BER (Data transformation and prediction using a pre-trained Machine Learning model). These workflows can become themselves process SIBs, and be reused for similar tasks, for example, for energy modelling in other jurisdictions.

We now show and briefly discuss two of the energy modelling workflows to demonstrate how the WAVE LCNC approach can support the full ML lifecycle.

Anatomy of a workflow

A well-formed workflow is composed of a *Start* node, one or more *End* nodes, one or more Service Independent Building Blocks (SIBs), and both the control flow of the application (solid line connectors) and its data flow (dashed line connectors). The Start node marks the initialization of the workflow with the input parameters passed from the context. Each SIB in the workflow has a name that reflects its purpose, and one or more outgoing control flow labelled *branches* that lead to the successive SIBs. Each branch corresponds to an outcome of the execution (e.g. the branches of the *REQ: Query Neo4J for Arch Count* SIB in Fig.3 are labelled *Buildings Matched*, and *No Matched Building*, respectively. In terms of the data flow, the SIB inputs are indicated in blue, and the outputs are placed on the branches, as they are typically specific to the execution outcome and thus branch specific.

Predict building energy ratings (BER)

This workflow predicts Building Energy Ratings by applying a pre-trained ML model to simulated envelope retrofits. It combines domain independent data-preprocessing SIBs, such as Load Datasets, Filter Data, and Dictionary Mapping, with application-specific ones, like Apply Retrofit. Technical complexities, including file I/O, data transformation, and running ML models via libraries like *os, pan-das*, and *joblib*, are fully abstracted. The visual modules accelerate model comparison and selection, as users can experiment with different ML models on the same or heterogeneous datasets simply by switching configurations in the Apply Prediction Model and Load Dataset SIBs. Additionally, the entire workflow can be encapsulated as a Process SIB, enhancing collaboration and reproducibility.

Calculate arch count workflow

This workflow processes a dataset that contains various archetype combinations such as Building Type, Building Energy Rating, Occupancy and Heating System, and queries a national-scale graph for the count of buildings with similar characteristics. By abstracting the technical details of connecting to Neo4j hosts, the workflow enables straightforward retrieval of building counts. The Neo4j query is passed as a parameter, allowing the matching criteria to be modified without requiring changes to the code files.

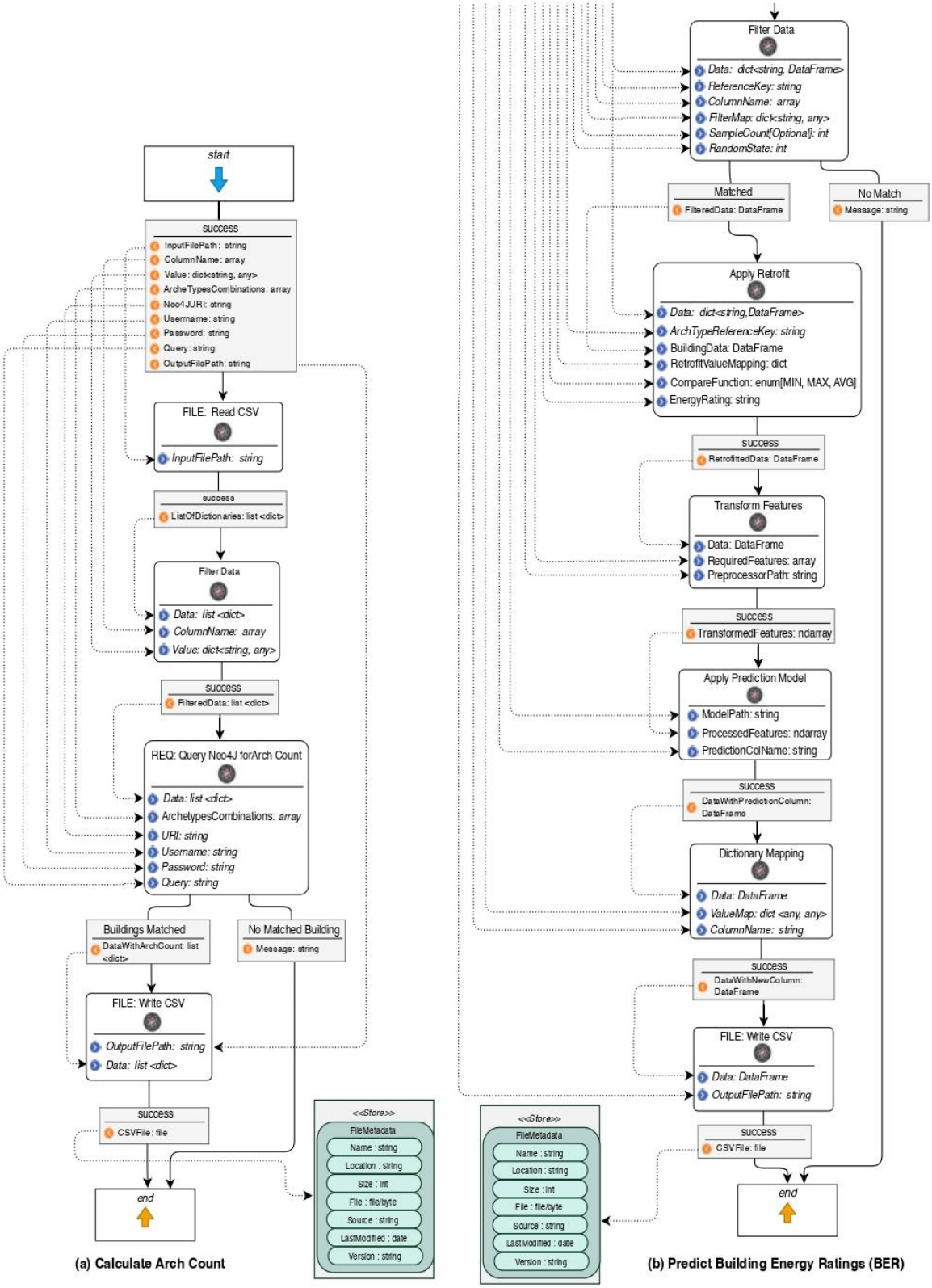

Figure 3:The WAVE workflows for (a) Calculating Archetype Counts, (b) (core part of) Predicting Building Energy Ratings (BER)

Similarly, parameters can be adjusted to calculate Arch counts for datasets filtered by a combination of columns (e.g., where ”BER” = ”A” and ”Occupancy” = 2).

Comparing traditional and LCNC approaches

While comparing the WAVE LCNC approach with a script-based pipeline, an assessment of their respective development time was done. The developer of the original

code scripts is both an expert in energy modelling and a competent programmer. In total, approximately 2000 lines of code were developed, talking approximately 3 days to develop and validate the code. A second developer created the LCNC workflows. Each one was developed and validated in approximately 4 hours. In addition to the reduced development time, the maintainability and reusability of the LCNC code also improved, as the corresponding code is componentized at a fine granularity and complies with the architectural characteristics of SIBs, and the control and data flows are explicitly modelled and thus easy to understand and to debug.

Workflows will be developed to support full electrification scenario modelling such as the effect of heat-pump adoption and EV charging within the housing stock. The resulting demand will be spatially allocated to low-voltage sub-stations, allowing feeder peak load and transformer over-load to be calculated. This will be displayed in multiscale geographical reports such as those in Fig. 4. This demonstrates how graphical workflows enable non-programmers to move from heterogeneous datasets to infrastructure-level planning insights across multiple modelling scales. Moreover, WAVE decouples workflow design from the computation, thereby enhancing execution performance. Workflows are stored in Common Workflow Language for-mat1 and executed on high-capacity server clusters, allowing users with low-specification laptops to initiate large-scale data-processing and machine learning pipelines. The backend manages metrics such as memory usage and CPU cycles through an orchestrator, eliminating the need for users to oversee these technical aspects.

Conclusions

The need to decarbonize the built environment requires a shift toward data-driven modeling. Although machine learning (ML) provides a powerful mechanism for optimizing energy performance across multiple scales, the skill gap around implementation remains a primary barrier to widespread adoption. This gap is characterized not just by a lack of technical coding ability, but by the difficulty of merging domain-specific energy expertise with complex software engineering.

LCNC platforms such as WAVE offer a convenient solution by democratizing software development. By re-moving the syntax barrier yet increasing checks and the use of domain knowledge, Wave allows domain experts who best understand building physics and operational needs - to lead the development of bespoke energy models. As demonstrated, this paradigm reduces development costs, accelerates toolchain creation, and bridges the professional skills gap. Ultimately, leveraging platforms like WAVE ensures that SMEs, policymakers and other stakeholders can fulfill the potential data-driven energy modelling.

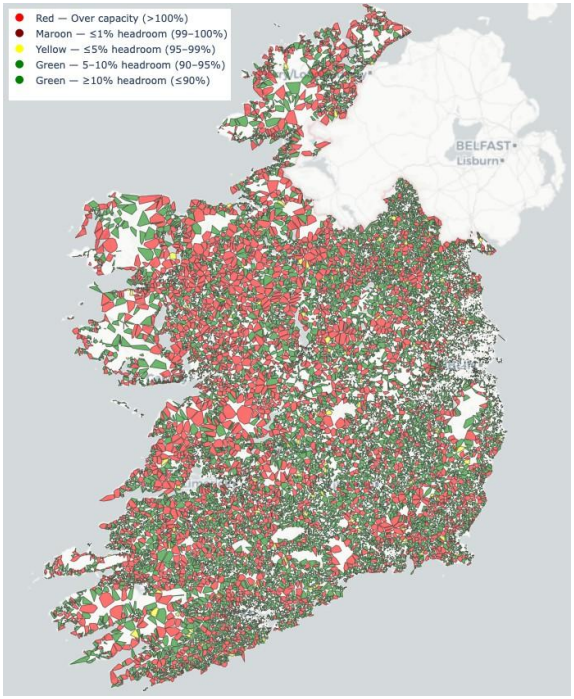

Figure 4: Results of multiscale energy modelling for scenarios including heatpump and EV adoption by residential households.

Acknowledgements

This publication has emanated from research supported by Wave project (ULF-Wave-2025) and partly by a grant from the Sustainable Energy Authority of Ireland (SEAI) un-der the Research, Development and Demonstration Funding Programme Grant No. 22/RDD/822. The opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the SEAI. Grammarly and ChatGPT were used to improve spelling, grammar, punctuation and clarity of some text here-in.

References

Ahmad, T., Chen, H., Guo, Y., and Wang, J. (2018). A com-prehensive overview on the data driven and large scale based approaches for forecasting of building energy demand: A re-view. Energy and Buildings, 165:301–320.

Ali, U., Bano, S., Shamsi, M. H., Sood, D., Hoare, C., Zuo, W., Hewitt, N., and O'Donnell, J. (2024). Urban building en-ergy performance prediction and retrofit analysis using data-driven machine learning approach. Energy and Buildings, 303:113768.

Boßelmann, S., Frohme, M., Kopetzki, D., Lybecait, M., Nau-jokat, S., Neubauer, J., Wirkner, D., Zweihoff, P., and Steffen, B. (2016). DIME: A programming-less modeling envi-ronment for web applications. In Proc. ISoLA 2026, volume LNCS 9953, pages 809–832, Cham. Springer International Publishing.

[1] CWL standard: https://www.commonwl.org

Brandon, C., Boßelmann, S., Singh, A., Ryan, S., Schieweck, A., Fennell, E., Steffen, B., and Margaria, T. (2024). Cinco de bio: A low-code platform for domain-specific workflows for biomedical imaging research. BioMedInformatics, 4(3):1865–1883.

Carioli, P., Czarnitzki, D., and Fernández, G. P. (2024). Evidence on the adoption of artificial intelligence: The role of skills shortage. ZEW-Centre for European Economic Research Discussion Paper, (24-013).

de Marco, G., Niederwieser, E., and Siegele, D. (2025). Explor-ing a multimodal conversational agent for construction site safety: A low-code approach to hazard detection and compliance assessment. Buildings, 15(18):3352.

Denny, P., Luxton-Reilly, A., Tempero, E., and Hendrickx, J. (2011). Understanding the syntax barrier for novices. In Proc. ITiCSE '11, page 208–212, New York, NY, USA. ACM.

Dzhusupova, R., Bosch, J., and Olsson, H. H. (2022). Challenges in developing and deploying ai in the engineering, procurement and construction industry. In Proc. COMPSAC 2022, pages 1070–1075.

El Kamouchi, H., Kissi, M., and El Beggar, O. (2023). Low-code/no-code development: A systematic literature review. In 2023 14th Int. Conf. on Intelligent Systems: Theories and Ap-plications (SITA), pages 1–8. IEEE.

Gossen, F., Kühn, D., Margaria, T., and Lamprecht, A.-L. (2018). Computational thinking: Learning by doing with the cinco adventure game tool. In Proc. COMPSAC 2018, volume 01, pages 990–999.

Hoare, C., Aghamolaei, R., Lynch, M., Gaur, A., and O'Don-nell, J. (2022). A linked data approach to multi-scale energy modelling. Advanced Engineering Informatics, 54:101719.

Karusseit, M. and Margaria, T. (2006). Feature-based modelling of a complex, online-reconfigurable decision support service. ENTCS, 157(2):101–118. Proc. Int. Workshop on Automated Specification and Verification of Web Sites (WWV 2005).

Katavoutas, G., Founda, D., Varotsos, K. V., and Giannakopoulos, C. (2022). Climate change impacts on thermal stress in four climatically diverse european cities. Int. Journal of Biometeorology, 66(11):2339–2355.

Manfren, M., James, P. A., and Tronchin, L. (2022). Data-driven building energy modelling – an analysis of the potential for generalisation through interpretable machine learning. Renewable and Sustainable Energy Reviews, 167:112686.

Margaria, T., Raffelt, H., and Steffen, B. (2005). Knowledge-based relevance filtering for efficient system-level test-based model generation. Innovations in Systems and Software Engineering, 1(2):147–156.

Margaria, T. and Schieweck, A. (2019). The digital thread in industry 4.0. In Proc. iFM 2019, Integrated Formal Methods, pages 3–24, Cham. Springer International Publishing.

Margaria, T. and Steffen, B. (2009). Business Process Model-ing in the jABC: The One-Thing Approach. In Handbook of Research on Business Process Modeling, pages 1–26. IGI Global, Hershey, PA.

Margaria, T. and Steffen, B. (2020). Extreme model-driven development (XMDD) technologies as a hands-on approach to software development without coding. In Encyclopedia of Education and Information Technologies, pages 732–750. Springer.

Martinez, E. and Pfister, L. (2023). Benefits and limitations of using low-code development to support digitalization in the construction industry. Automation in Construction, 152:104909.

Muminova, E., Ashurov, M., Akhunova, S., and Turgunov, M. (2024). Ai in small and medium enterprises: Assessing the barriers, benefits, and socioeconomic impacts. In Proc. ICK-ECS 2024, volume 1, pages 1–6.

OECD (2025). Bridging the AI skills gap: Is training keeping up? OECD Publishing, Paris.

O'Brien, S., Brandon, C., Busch, D., Krumrey, M., Mitwalli,

D. S., Singh, A., Teumert, S., and Margaria, T. (2025). Spe-cialized training in LCNC and AI: from the pedagogical concept to the experience. In Proc. COMPSAC 2025, pages 2394–2403. IEEE.

Paleyes, A., Urma, R.-G., and Lawrence, N. D. (2022). Challenges in deploying machine learning: A survey of case stud-ies. ACM Comput. Surv., 55(6).

Prinz, N., Rentrop, C., and Huber, M. (2021). Low-code devel-opment platforms-a literature review. In AMCIS.

Skillington, K., Crawford, R. H., Warren-Myers, G., and David-son, K. (2022). A review of existing policy for reducing embodied energy and greenhouse gas emissions of buildings. En-ergy Policy, 168:112920.

Steffen, B., Margaria, T., Nagel, R., Jörges, S., and Kubczak, C. (2007). Model-driven development with the jabc. In Bin, E., Ziv, A., and Ur, S., editors, Hardware and Software, Verification and Testing, pages 92–108, Berlin, Heidelberg. Springer Berlin Heidelberg.

Steffen, B. and Murtovi, A. (2018). M3c: Modal meta model checking. In Howar, F. and Barnat, J., editors, Formal Methods for Industrial Critical Systems, pages 223–241, Cham. Springer International Publishing.

Sundberg, L. and Holmström, J. (2023). Democratizing artificial intelligence: How no-code AI can leverage machine learning operations. Business Horizons, 66(6):777–788.