A Dynamo-Based Framework for Streamlining MEP Design Workflows in Autodesk Revit

Sadia Akter [1], Mst Mithila Afroz Hridy [1], Preonty Paul [1], Erin Purvis [1], Bryan Kuhr [*]
*Corresponding Author: bkuhr@sbc.edu
The Margaret Wyllie Engineering Program
Sweet Briar College
Sweet Briar, VA, USA

***Abstract*— Autodesk Revit is heavily relied upon by modern mechanical, electrical, and plumbing (MEP) design firms for system modeling and coordination; however, many essential workflows remain repetitive and manually intensive. Significant engineering time is consumed, and opportunities for human error are introduced by these tasks, particularly within smaller firms that lack access to proprietary or paid automation tools. As project complexity and sustainability requirements continue to grow, a strong need is created for efficient, consistent, and accessible automation solutions within standard Revit workflows. This work presents the development of automated MEP design workflows using Dynamo within Autodesk Revit, with a focus on space placement from linked architectural models and the structured investigation and documentation of sanitary plumbing riser diagram automation for future development. The methodology is centered on the identification of repetitive manual processes and their translation into structured Dynamo scripts that extract, transform, and apply data across linked Revit models. For space placement, architectural rooms are identified, relevant parameters are extracted, the data is reformatted to match MEP requirements, and spaces are automatically assigned with consistent parameter mapping. For plumbing riser diagrams, applicable standards, calculation logic, and workflow structure are documented using IAPMO-based data to outline how pipe sizing and flow calculations could be automated in future work. Both workflows are evaluated using sample and client-provided Revit models and are validated through comparison of automated results against manually completed designs where applicable. Testing is focused on accuracy, consistency, Revit version compatibility, and overall time savings. The results demonstrate that targeted Dynamo automation significantly reduces repetitive modeling effort, improves design consistency, and supports reliable, real-world MEP engineering workflows. User documentation and training materials are also included to ensure long-term usability and maintainability of automation tools.**

***Keywords*— Dynamo, Autodesk Revit, MEP design, Automation, BIM, Workflow Optimization.**

I. INTRODUCTION

Building information modeling (BIM), which is a digital process for creating and managing building designs using a 3D model with embedded information, has fundamentally changed how engineering teams approach multi-disciplinary design, yet many workflows within tools such as Autodesk Revit remain manual and repetitive. Revit is widely used in MEP (mechanical, electrical, and plumbing) practice for coordinating complex system models, but routine tasks such as updating parameters (data like names, numbers, or properties attached to elements) across linked models, assigning spaces (defined areas in an MEP model used for analysis, similar to rooms in architectural models), and generating schematic diagrams still require significant manual effort [1]. These inefficiencies lead to inconsistencies, increased project timelines, and reduced engineering productivity [2], [3].

Dynamo, Revit's built-in visual programming environment, provides a practical solution by enabling engineers to create graph-based workflows that interact directly with Revit. Instead of writing code, Dynamo uses "nodes," which are small functional blocks that perform specific tasks such as selecting elements, extracting data, or creating new objects. These nodes are connected together to form a workflow. This allows for the automation of processes like data extraction (pulling information such as room names or locations from a model) and data manipulation without requiring advanced programming knowledge [4]. For high-frequency, low-complexity tasks, such automation can significantly reduce manual input while improving consistency across project deliverables.

One important application of this is space placement, which refers to automatically creating MEP spaces based on architectural room data. This typically involves identifying rooms in a linked architectural model (a connected model that shares information with the MEP model), extracting key parameters such as room names and numbers, and using centroid points (the geometric center of each room) to place corresponding MEP spaces accurately within the model. By automating this process, engineers can avoid repetitive manual placement and reduce errors in coordination between disciplines.

Despite its potential, the adoption of Dynamo within MEP workflows remains limited. Developing reliable automation scripts often requires technical familiarity beyond what many

practicing engineers possess, and existing tools are frequently project-specific and insufficiently documented for broader use [4].

This paper addresses this gap by presenting two targeted Dynamo-based workflows: automated MEP space placement from linked architectural models and a structured framework for sanitary plumbing riser diagram automation. Both workflows are developed and evaluated within a real-world MEP engineering environment, with an emphasis on usability, consistency, and practical implementation.

II. Background and Motivation

This work was developed in collaboration with Staengl Engineering, a Charlottesville-based MEP consulting firm specializing in multi-family residential, commercial, and mixed-use building design. Like many firms of similar size, engineers at Staengl are responsible for both system design and production modeling, which means time spent on repetitive BIM tasks directly reduces the time available for engineering analysis and decision-making.

MEP systems represent a significant portion of overall construction cost, typically ranging from 40% to 60% of total project value [5]. Consequently, inefficiencies in modeling and documentation can lead to increased labor costs and a higher risk of coordination errors during construction [6]. Through discussions with practicing engineers at Staengl, two workflows were identified as primary sources of inefficiency: the manual placement and parameter mapping of MEP spaces from architectural room data, and the generation of sanitary plumbing riser diagrams based on IAPMO fixture unit calculations. Additional tasks, such as circuit renumbering and fixture placement, were also noted, though their individual impact was comparatively smaller. While prior applications of Dynamo in BIM environments have demonstrated its capability for automating data-driven modeling tasks, many implementations remain highly project-specific and lack the structure or documentation required for broader adoption [7]. This limits their practical use of day-to-day engineering workflows.

Fig. 1. Sample architectural floor plan used as the basis for MEP space generation. The layout illustrates room boundaries, spatial organization, and functional zones that are extracted from the linked architectural model and used to automate MEP space placement and parameter mapping within the Revit environment.

The objective of this work is to develop automation workflows that are technically effective and practical, maintainable, and accessible to engineers without extensive programming experience. By focusing on real production challenges and emphasizing usability, this study aims to bridge the gap between theoretical automation capabilities and their application in MEP design practice.

III. Overview of the Automation Framework

This study presents an automated workflow for generating and managing MEP spaces in Autodesk Revit using Dynamo. The framework is designed to reduce manual modeling effort, improve cross-disciplinary consistency, and enable a more automated and consistent design process. The workflow follows a structured sequence consisting of:

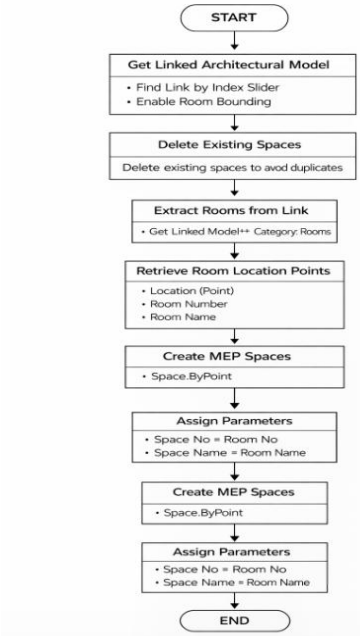

Fig.2. Illustrates the overall workflow of the proposed automation framework.

The following sections describe the process using Revit-specific language and commands:

A. Selection of the Linked Architectural Model

This step identifies and selects the architectural model linked to the MEP project. The Select.GetDocuments node retrieves all linked Revit documents, while the Watch node provides visualization of available models. A Number Slider is used to select the desired document, which is extracted using List.GetItemAtIndex. In typical implementations, the architectural model is located at index 0. This step establishes the primary data source for all subsequent operations.

The node flow can be described like this: Select.GetDocuments (documents) → List.GetItemAtIndex (list, index from Number Slider) → Watch (output visualization)

B. Extraction of Architectural Room Data

In this step, room elements are extracted from the linked architectural model. The Categories node is configured to "Rooms," and All Elements of Category is used to retrieve all room instances. Relevant parameters, including room name and room number, are extracted using Element.GetParameterValueByName. The Room.Location node is used to compute centroid points representing the geometric center of each room. These centroid points serve as the basis for spatial placement and coordination. The resulting dataset includes room names, room numbers, and centroid locations, forming the foundation for automated space generation.

Node Flow: Categories (Rooms) → All Elements of Category → Element.GetParameterValueByName (Name/Number) + Room.Location (center points)

C. Pre-Processing and Model Cleanup

To ensure repeatability and prevent duplication, previously generated MEP spaces are optionally removed prior to execution. Existing spaces are identified using All Elements of Category (set to "Spaces") and deleted using Element.Delete. This step enables iterative use of the workflow by allowing updated space layouts to be generated without overlapping existing elements, supporting changes during early-stage design development.

Node Flow: Categories (Spaces) → All Elements of Category → Element.Delete (remove existing spaces)

D. Determination of Space Placement Locations

Centroid points obtained from room locations are filtered to remove invalid or null values. The List.FilterByBoolMask node is used to isolate valid placement points that can be used for space creation. These centroid points serve as spatial references for MEP space placement and ensure alignment between architectural and MEP models. Using centroid-based placement provides a consistent and automated method for positioning spaces within the model.

Node Flow: Filtered Rooms → Room.Location → List.FilterByBoolMask (valid centroid points used for space creation and tagging)

E. Space Creation within the MEP Model

MEP spaces are generated using the Space.ByPointAndLevel node, which converts centroid points into valid Revit space elements. A target level is selected using the Levels node in combination with a Number Slider and List.GetItemAtIndex, ensuring correct vertical placement within the building model. This approach enables rapid generation of spaces across different design configurations, allowing users to modify inputs such as level selection or linked model choice without repeating manual modeling steps.

Node Flow: Filtered Points + Levels → List.GetItemAtIndex (level from Number Slider) → Space.ByPointAndLevel

F. Parameter Mapping Between Rooms and Spaces

To maintain consistency between architectural and MEP models, room parameters are transferred to the generated spaces. Room names and numbers are extracted and assigned to corresponding space elements using Element.SetParameterByName. The following mappings are applied:

- Room Name → Space Name
- Room Number → Space Number

This direct parameter transfer reduces manual data entry and ensures alignment between disciplines.

Node Flow: Filtered Rooms → Element.GetParameterValueByName (Name/Number) → Element.SetParameterByName (applied to Spaces from Space.ByPointAndLevel)

G. Execution and Quality Verification

Upon execution, the workflow performs space deletion (if enabled), room data extraction, centroid calculation, space generation, and parameter assignment. Following execution, results are verified to confirm correct placement, level assignment, and parameter mapping.

A Revit space schedule is used to validate the generated outputs, confirming that space names and numbers are correctly aligned with architectural data. Any discrepancies, such as missing or invalid inputs, are identified and resolved through manual adjustment. This verification step ensures that the automation process produces accurate and reliable results while maintaining consistency across the model.

IV. Framework Preconditions and Constraints

The framework is intended for early design stages, when architectural rooms are defined, but MEP spaces have not yet been fully developed. Running the script at later stages may conflict with existing or manually modified spaces.

The current implementation supports a single execution per project. Re-running the script may result in duplication or unintended overwriting of space elements. While a cleanup step can remove previously generated spaces, reliable multi-run functionality has not yet been implemented.

The *Space.ByPointAndLevel* node requires manual reinitialization prior to execution. Users must disconnect and

reconnect inputs (or reinsert the node) to ensure proper operation.

Accurate level selection must be configured before execution. Incorrect index selection may result in misplaced or missing spaces, highlighting the dependency on proper input configuration.

V. Technical Challenges

During the development of the Dynamo-based automation framework, several technical challenges were encountered. These challenges were primarily related to software compatibility, data structure alignment, and the generation of valid Revit elements. Addressing them was essential to ensuring a reliable and repeatable workflow.

One challenge involved identifying and installing the correct Dynamo packages (e.g., Clockwork, Rhythm), as some nodes failed due to version incompatibilities or missing dependencies. This was addressed by aligning package versions with the appropriate Dynamo environment and standardizing dependencies across the workflow.

Another difficulty arose when initial implementations successfully generated centroid points but failed to create corresponding MEP spaces. This occurred because geometric points were not converted into valid Revit elements. Implementation of the Space.ByPointAndLevel node allows each point to be associated with a valid level, enabling the creation of functional space elements.

Dynamo's list-based structure also led to misalignment between room data and generated elements, resulting in incorrect or null outputs. To correct this, all datasets (room names, numbers, centroid points, and spaces) were derived from a consistent source and maintained in identical order. Watch nodes were used to verify proper data alignment.

The Space.ByPointAndLevel node occasionally returns null values due to invalid points, incorrect level inputs, or incomplete room data. A validation step using Boolean filtering (Object.IsNull) was introduced to remove invalid inputs and ensure only valid data was used for space creation.

Additionally, incorrect parameter assignment caused spaces to be labeled with mismatched names or numbers. This resulted from inconsistent list indexing and was corrected by maintaining strict alignment between room data and generated space elements.

Element.SetParameterByName also failed in some cases due to incorrect parameter names, improper execution order, or misaligned inputs. This was addressed by verifying parameter names, ensuring list consistency, and adjusting execution sequencing so that parameters were assigned only after successful space creation.

VI. Results:

A. Space Generation and Validation

The proposed automation framework was implemented in Autodesk Revit using Dynamo to streamline MEP space placement based on architectural room data. The workflow extracts room geometry and associated parameters, applies centroid-based placement, and generates corresponding MEP spaces within the model. Space names and numbers are assigned to match architectural data, maintaining consistency across disciplines.

<Space Schedule>

A	B	C	D	E
Space Type	Name	Number	Room: Number	Room: Name
<Building>	MECH/STORAGE	512	512	MECH/STORAGE
<Building>	OFFICE	519	519	OFFICE
<Building>	OFFICE	520	520	OFFICE
<Building>	LAUNDRY	505	505	LAUNDRY
<Building>	UTILITY	8	8	UTILITY
<Building>	WOMENS BATHRO	514	514	WOMENS BATHRO
<Building>	MENS BATHROOM	513	513	MENS BATHROOM
<Building>	JANITOR	515	515	JANITOR
<Building>	COMMUNITY ROO	508	508	COMMUNITY ROO
<Building>	CORRIDOR	525	525	CORRIDOR
<Building>	WAITING AREA	516	516	WAITING AREA
<Building>	UNIT 1C	103	103	UNIT 1C
<Building>	UNIT 1A.2 ACCESS	104	104	UNIT 1A.2 ACCESS
<Building>	UTILITY	522	522	UTILITY
<Building>	UNIT 3B	102	102	UNIT 3B
<Building>	UNIT 1D MARKET R	101	101	UNIT 1D MARKET R
<Building>	MECH ROOM	509	509	MECH ROOM

Fig. 3. Space schedule showing alignment of space names and numbers with architectural rooms generated by Revit.

The schedule confirmed that space names and numbers were correctly aligned with the corresponding architectural rooms, providing a clear validation of parameter mapping and data consistency. A formal evaluation of the workflow was completed using a sample building model with a linked architectural file provided by the client. Results show that MEP spaces were successfully generated for all architectural rooms when the script is working properly.

During early implementation, centroid point generation alone did not produce valid Revit elements, resulting in the absence of spaces in schedules and model views. This limitation was resolved using the Space.ByPointAndLevel node, which enabled the successful conversion of geometric points into functional MEP space elements.

B. Efficiency and Error Reduction

The automated workflow is intended to reduce the manual effort associated with space creation and tagging. In a traditional process, these tasks require repeated user input and careful coordination, and the time required on each project varies based on model complexity and user experience [7], [8]. This includes manually placing each space, entering or

verifying associated parameters, and applying tags across relevant views. By contrast, the Dynamo-based workflow consolidates these steps into a single automated execution, eliminating repetitive user interactions and reducing opportunities for input-related errors. From a workflow perspective, this shifts the user's role from active modeling to output verification, which represents a clear improvement in efficiency and consistency even in the absence of explicit time measurements.

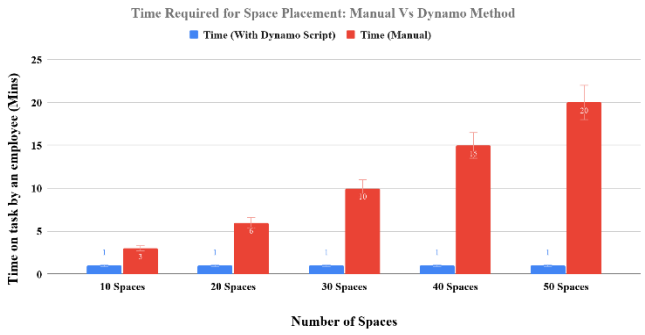

Fig. 4. Comparison of employee time required for manual and Dynamo-based space placement for buildings with varying numbers of spaces. Error bars represent ±10% of the measured time to account for potential variability in workflow execution

Following this, a performance comparison was conducted to evaluate the efficiency of the proposed workflow. All reported values are in minutes and have been rounded for consistency of presentation. As shown, the time required for manual space placement increases significantly with the change in the number of rooms, reflecting the repetitive and labor-intensive nature of the task. In contrast, the execution time of the Dynamo script remains relatively stable within the tested range. It should be noted that a modest increase in workflow time is expected in practical applications as building scale grows, primarily due to the additional effort required for reviewing generated outputs in more complex models [7]. Despite this, the automated approach achieves substantial time savings in multi-level models, with little currently observable disparity as building complexity increases.

Beyond efficiency improvement, the framework contributes to a measurable reduction in common modeling errors. Issues such as missing or duplicated spaces, inconsistent naming conventions, and incorrect tag placement are largely mitigated, as elements are generated directly from architectural source data. As mentioned earlier, similar errors were encountered due to data misalignment and inconsistent parameter mapping and were resolved through strict enforcement of list consistency and data correspondence across all workflow stages.

Improvements were also observed in parameter assignment reliability, following adjustments to execution sequencing and validation of parameter naming conventions. This ensured that parameter mapping occurred only after successful element creation and proved essential in achieving consistent and accurate data transfer.

Ultimately, the results demonstrate that resolving underlying technical constraints, particularly those related to data integrity, element generation, and workflow sequencing, was essential to the development of a stable and repeatable automation process. Once addressed, the framework consistently produced accurate outputs, thus reducing the need for manual intervention.

VII. Discussion

One of the main advantages of the proposed framework is improved coordination between architectural and MEP models. In manual workflows, updates to the architectural model often require repeated adjustments in the MEP model, which can be both time-consuming and prone to inconsistencies [10]. By directly referencing architectural room data, the script allows space parameters to be updated in a few seconds. This enables users to reflect design changes efficiently by re-running the workflow, reducing rework, and improving consistency between disciplines.

The framework also enhances the ability to explore design alternatives. In traditional workflows, generating multiple layout options requires repeating the same sequence of steps [8]. With the Dynamo-based approach, users can regenerate spaces by modifying inputs such as level selection or linked models, without rebuilding the model. This supports faster iteration and allows designers to evaluate multiple scenarios with less effort.

While the workflow demonstrates clear benefits, its performance may vary depending on project conditions. Factors such as model complexity, data quality, and user experience can influence the overall efficiency and level of manual intervention required [7]. For instance, a small office building necessitates less sorting through outputs than a high school or airport. However, internal validation confirms that the script consistently retrieves and applies correct room data, and the overall process shifts from manual modeling to output verification.

The approach is also scalable and can be applied to larger and more complex projects. As project size increases, the reduction in repetitive tasks and improved consistency become more significant, making the workflow particularly valuable in large-scale or multi-level building designs [7].

VIII. Future Work

A. Space Placement

One immediate extension of the current workflow is the automation of space tag placement. The existing script handles space creation and parameter mapping but stops short of

annotation, leaving tagging as a remaining manual step in the documentation process.

A more significant limitation, noted in the Framework Preconditions and Constraints section, is that the script is designed for single execution at the start of a project. It works by clearing all existing spaces and regenerating them from the current architectural room data. This is effective during initial setup but becomes problematic once MEP content has been associated with those spaces, since rerunning the script would remove elements that other parts of the model depend on. Future development should focus on incremental update logic that detects new or modified rooms and regenerates only the affected spaces, making the workflow viable across all stages of a project rather than just the earliest ones.

B. Riser Diagram

The IAPMO-based calculation framework developed in this study lays the groundwork for full riser diagram automation but does not yet produce executable script output. The logical next step is a Dynamo script that reads fixture data from the model, applies fixture unit assignments per IAPMO standards, sizes pipe segments based on accumulated flow loads, and generates standardized schematic outputs. Because building configurations, fixture types, and applicable code requirements vary considerably across projects, a modular architecture will be necessary to make this practical across the range of conditions encountered in real MEP work.

IX. CONCLUSION

This study developed and validated a Dynamo-based automation workflow for MEP space placement in Autodesk Revit, built around the operational needs of a practicing MEP firm. The workflow extracts room geometry and parameters from a linked architectural model and generates corresponding MEP spaces using centroid-based placement via the Space.ByPointAndLevel node, and transfers room names and numbers directly to the created elements. Validation through Revit space schedules confirmed consistent parameter alignment across both sample and client-provided models, with discrepancies largely eliminated after the introduction of Boolean filtering and input validation steps.

Performance testing demonstrated that script execution time remained stable as room count and floor count increased, while manual placement time grew proportionally with model scale. In multi-level buildings, this difference was most pronounced, with the automated approach producing results that would have otherwise required substantial repeated input from the engineer. The node-based interface kept the workflow accessible to practitioners without programming experience, which was a practical requirement given the firm's staffing context.

The parallel development of an IAPMO-based sanitary riser diagram framework contributes to a documented calculation structure for a workflow that currently lacks any accessible automation solution within standard Revit practice. Together, both contributions offer small and mid-sized MEP firms a grounded, maintainable entry point into Dynamo automation, one built from real production constraints rather than theoretical capability alone.

X. ACKWOLEDGMENT

The team would like to thank Staengl Engineering and Sweet Briar College for sponsoring this. We sincerely appreciate the guidance of Galen at Staengl Engineering for supervising and mentoring our work. We also thank Robert Lovaglio and Devin Wright their valuable technical support and insights throughout the project.

XI. REFERENCE

[1] Autodesk, *Autodesk Revit Documentation*, 2024.
[2] C. Eastman, P. Teicholz, R. Sacks, and K. Liston, *BIM Handbook: A Guide to Building Information Modeling*, 3rd ed. Hoboken, NJ, USA: Wiley, 2018.
[3] B. Becerik-Gerber, S. Rice, and B. Boyer, "The perceived value of building information modeling in the U.S. building industry," *Journal of Information Technology in Construction*, vol. 17, pp. 532–543, 2012.
[4] M. Solihin and C. Eastman, "Classification of rules for automated BIM rule checking development," *Automation in Construction*, vol. 53, pp. 69–82, 2015.
[5] Q. Wu, L. Chen, P. Shi, W. Wang, and S. Xu, "Identifying impact factors of MEP installation productivity: An empirical study," *Buildings*, vol. 12, no. 5, p. 565, May 2022.
[6] M. Oraee, M. R. Hosseini, D. J. Edwards, H. Li, E. Papadonikolaki, and D. Cao, "Collaboration barriers in BIM-based construction networks: A conceptual model," *International Journal of Project Management*, vol. 37, no. 6, pp. 839–854, Aug. 2019.
[7] G. Rocha and L. Mateus, "Using Dynamo for automatic reconstruction of BIM elements from point clouds," *Applied Sciences*, vol. 14, no. 10, p. 4078, Jan. 2024.
[8] Y. H. Teo *et al*., "Enhancing the MEP coordination process with BIM technology and management strategies," *Sensors*, vol. 22, no. 13, p. 4936, Jun. 2022.
[9] Autodesk, *Dynamo Primer*, 2023.
[10] S. Park, M. Shin, J. Y. Jang, B. Koo, and T. W. Kim, "Automated process for generating an air conditioning duct model using the CAD-to-BIM approach," *Journal of Building Engineering*, vol. 91, p. 109529, Aug. 2024.