World Journal of Advanced Research and Reviews
eISSN: 2581-9615 CODEN (USA): WJARAI
Cross Ref DOI: 10.30574/wjarr
Journal homepage: https://wjarr.com/

(Research Article)

Automating the design process for smart building technologies

Ruchit Parekh *

Department of Engineering and Management, Hofstra University, New York, USA.

World Journal of Advanced Research and Reviews, 2024, 23(02), 1213–1234

Publication history: Received on 01 July 2024; revised on 12 August 2024; accepted on 14 August 2024

Article DOI: https://doi.org/10.30574/wjarr.2024.23.2.2461

Abstract

Smart buildings are emerging as complex cyber-physical systems that aim to provide safe, comfortable, energy-efficient, and aesthetically pleasing environments. However, traditional design methods are becoming inefficient due to increasing functional complexity, cost pressures, and market demands. This paper introduces a platform-based design (PBD) methodology for smart buildings. PBD promotes hardware and software reuse on shared infrastructures, enables rapid prototyping, and facilitates extensive design space exploration to optimize performance. We identify, abstract, and formalize key components of smart buildings, presenting a design flow that transforms high-level functional specifications into physical implementations within the PBD framework. To demonstrate the practical application of this approach, we present a case study on the design of an occupancy-responsive heating, ventilation, and air conditioning (HVAC) system.

Keywords: Cyber-physical systems; Design automation; Machine learning; Smart buildings; Control systems; Energy efficiency; Platform-based design; HVAC optimization

1. Introduction

The majority of our lives are spent indoors [1], with indoor environments significantly impacting our health, well-being, and productivity. In the United States, buildings consume 40% of primary energy [2], yet many occupants remain dissatisfied with their built environments [3], even in structures designed to be environmentally friendly [4]. The integration of cutting-edge technologies, including extensive sensing and actuation systems, sophisticated control mechanisms, and big data analytics, has catalyzed the transformation of buildings from basic structures to automated, multifunctional spaces. These "smart buildings" prioritize safety, health, comfort, affordability, and sustainability, while supporting reliable grid operations. The demand for such intelligent buildings has experienced rapid growth globally, with the market doubling triennially in both developed and emerging urban areas [5].

Smart buildings can be conceptualized through three key aspects: components, functions, and outcomes. Components encompass a wide array of interconnected technical equipment and appliances, including traditional systems like HVAC, lighting, and electrical networks, along with their associated sensing and control infrastructure. Emerging technologies such as on-site energy generation and storage systems are also integral components. These elements enable a diverse range of functions. For example, buildings equipped with power meters and energy storage capabilities can participate in demand response programs, while those with occupancy sensing can dynamically adjust indoor conditions (lighting, temperature, and air quality) to optimize energy usage. The functions a building can perform determine its level of intelligence and effectiveness, ultimately facilitating outcomes that benefit the environment, society, and economy, such as improved health, comfort, productivity, and energy efficiency.

Significant research has been conducted on developing smart building functions, including advancements in communication [6], computing [7], and control systems [8]. However, the implementation and integration of these

* Corresponding author: Ruchit Parekh

smart functions have largely remained ad hoc and heuristic [9]. Conventionally, each application is designed and assembled as a standalone system. Consider a scenario where a building owner wants to implement both a demand-controlled ventilation system and an occupancy-responsive lighting system. The ventilation system would require components such as an economizer with a modulating damper, occupancy counting sensors (e.g., cameras), and a controller to manage the system. The lighting system would need wireless passive infrared sensors for occupancy detection and daylight sensors. Despite the potential for shared data (like occupancy information), these systems often cannot integrate due to being sourced from different vendors. For instance, Lennox [10] specializes in ventilation services, while Lutron [11] focuses on lighting control solutions. This "one-function-one-box" approach allows for optimization of individual applications but limits opportunities for component reuse and cost reduction. Its widespread adoption is due to the complexity of smart building applications and the fragmented nature of the technology supply chain.

An emerging alternative is the "application stack paradigm" [12], which enables component sharing across different functions. This approach is facilitated by innovations in building operating systems (BOSs), such as XBOS [7], which provide unified abstractions and controlled access to shared physical resources. In our ventilation and lighting example, this paradigm would allow both systems to utilize occupancy data from a single camera infrastructure, eliminating redundant hardware and software. However, even in this model, the integration of add-on infrastructure often occurs empirically, limiting the potential for cost-effective designs due to insufficient design space exploration (DSE). The advent of BOSs and application stacks is driving a shift in smart building technology from a vertical to a horizontal orientation, allowing independent component procurement and fostering competition among software providers.

The two previously discussed paradigms for integrating smart building functions are encountering growing challenges, necessitating a more structured design approach. These challenges include:

- Cost considerations: Building upgrades are highly cost-sensitive. As buildings grow in size and complexity, the expenses associated with additional sensors and system setup increase significantly. For instance, retrofitting a commercial building's HVAC system for occupancy responsiveness would require occupancy sensors at every entry and exit point on each floor. Implementing more granular control or personalized settings would demand even more sensors, escalating hardware and installation costs. However, substantial savings could be realized by efficiently utilizing existing infrastructure and sharing resources across functions. For example, repurposing security cameras for occupancy detection or leveraging Wi-Fi networks [13] and scheduling systems for occupancy inference could prove cost-effective.
- Escalating functional complexity: The next generation of smart buildings will need to support an expanding array of sophisticated functions, such as automated waste management, self-cleaning systems, personalized environmental control, and optimized space utilization. These functions are intricate, decentralized, and interdependent. Future buildings are expected to provide customized environments tailored to individual occupants' preferences, locations, and activities. Moreover, buildings must respond to both internal occupant needs and external factors like power grid demands and on-site energy generation. This necessitates a comprehensive approach to building management, balancing multiple objectives including occupant comfort, carbon reduction, energy conservation, and grid stability. Integrating these diverse functions requires multi-stage planning and coordination, presenting unprecedented challenges in managing system complexity and interdependencies.
- Adoption barriers for new technologies: Smart buildings represent a multidisciplinary field, involving diverse industrial systems and engineering disciplines. Technological advancements in this domain are equally varied, spanning from sustainable architectural designs and energy-efficient sensor networks to advanced control algorithms and data analytics. However, building owners, who are responsible for integrating these functions, often lack a comprehensive understanding of the synergistic benefits of combining different technologies. This knowledge gap, coupled with the absence of a unified platform for abstracting, modeling, and validating new technologies, hinders their adoption. For example, while commercial occupancy counting systems typically use thermal or video-based solutions, promising research has been conducted on using more cost-effective and privacy-preserving alternatives like CO2 sensors. Machine learning and dynamic systems theory [14] have been applied to enhance the accuracy and responsiveness of CO2-based occupancy detection. Parallel efforts have focused on miniaturizing and reducing the cost of CO2 sensors. Developing optimal smart building solutions often requires diverse expertise from various engineering disciplines. However, current implementation approaches lack systematic methods for exploring and incorporating new technologies across different levels, often resulting in suboptimal designs. Furthermore, buildings are dynamic environments, with changing room uses, occupant preferences, and aging systems. Consequently, building system design is an ongoing process throughout a structure's lifecycle, necessitating a design methodology that not only effectively translates specifications into implementations but also adapts efficiently to changing requirements over time.

Considering these challenges and opportunities, a new paradigm emerges: an automated, structured, and integrated design approach. This paradigm aims to enhance the efficiency and cost-effectiveness of building application prototypes. The design process begins with high-level specifications from the building owner, progressing to the automatic synthesis of functions and their implementations that fulfill these requirements.

This approach encapsulates expertise and experience from various system designers into structured libraries, facilitating future design practices. By organizing interfaces between different libraries coherently and consistently, each library can be independently enhanced while maintaining seamless integration with others. This modularity allows for comprehensive exploration of the design space. The library concept promotes hardware and software reuse on shared infrastructures, enabling extensive design space exploration (DSE) to optimize performance.

Automating the design process is expected to streamline the design flow and simplify the reconfiguration of building systems for diverse applications. Unlike the application stack paradigm, our proposed approach enables principled DSE, resulting in designs with quantifiable benefits in cost savings, comfort, and other metrics. To illustrate, we revisit the lighting and ventilation example. Our paradigm, through DSE, might opt for a WiFi-based occupancy counter over a camera-based system, leveraging existing WiFi infrastructure for a more economical and privacy-conscious solution.

This paper adopts platform-based design (PBD) as a unifying methodology for automated, structured, and integrated building application design. PBD has been successfully applied in various domains, including hardware-software codesign [15], analog circuit design [16], automotive electronics [17], and communication systems [18], both at chip and system levels. The PBD approach consists of two phases:

- Bottom-up phase: Generates libraries by abstracting behavioral models, performance models, and composition rules for components.
- Top-down phase: Comprises optimization steps where a cost function is minimized over the library components, streamlining DSE while focusing on promising solutions.

The paper is structured as follows:

- Section II: Provides a conceptual overview of PBD methodology and our proposed integrated design flow for smart buildings.
- Section III: Details the construction of design libraries in the bottom-up phase.
- Section IV: Elaborates on the top-down design flow.
- Section V: Presents an illustrative case study.
- Section VI: Concludes the paper and discusses future directions.

This structured approach aims to address the complexities of smart building design, offering a more systematic and efficient method for developing and implementing advanced building technologies.

2. PBD Overview and Proposed Design Methodology

2.1. PBD Framework

Platform-based design (PBD) [19] was initially conceived to tackle the growing intricacies of hardware-software codesign in embedded systems. The core principle of PBD is the separation of concerns, particularly distinguishing between function (system purpose) and architecture (implementation method). This separation facilitates more effective exploration of design alternatives. For instance, a video decoder could be implemented entirely in software on a general-purpose CPU or as a hybrid hardware-software solution, with certain functions allocated to a specialized coprocessor for enhanced performance. These design choices are informed by a comprehensive design space exploration (DSE) process, detailed in [20].

PBD's fundamental approach involves:

- Initiating design at the highest abstraction level
- Concealing non-essential implementation details
- Encapsulating key implementation parameters in abstract models
- Constraining DSE to a predefined set of library components

- Progressing through a series of refinement stages from initial specifications to final implementation, utilizing platforms at various abstraction levels [20]-[22]

In the PBD context, a platform is defined as a library of components and their composition rules, used to generate designs at a specific abstraction level. It essentially parameterizes the solution space. The design process in PBD is neither purely top-down nor bottom-up, but rather a "meet-in-the-middle" approach comprising two phases:

- Bottom-up: Constructing a design platform by defining components and their abstractions, encompassing both physical and cyber aspects in cyber-physical systems (CPS) [23].
- Top-down: Formalizing high-level design requirements and mapping them to lower-level platform implementations.

A component (M) in a PBD library is characterized by:

- Input ports (U), output ports (Y), internal variables (X), and configuration parameters (K)
- A behavioral model, represented by dynamic equations or automaton-recognized sequences
- Extra-functional models for non-functional attributes (e.g., cost, performance)
- Labels indicating function and features

The mapping process between abstraction levels is framed as a multi-objective optimization problem, optimizing performance metrics within constraints set by the platform library and design requirements. This mapping must preserve the model's semantics to ensure the validity of verified properties post-implementation.

PBD effectively combines elements of the layered design approach [25], which formalizes vertical abstraction and refinement, with component-based design [26], addressing composition and decomposition at each abstraction level. This synthesis creates a robust framework for tackling complex design challenges in smart building systems.

B. Proposed Design Flow Overview

We adapt the PBD paradigm for the automated design and integration of smart building systems. Our approach incorporates Design Space Exploration (DSE) at various stages of the design process, informing both high-level decisions (e.g., ventilation strategy selection) and low-level choices (e.g., CO2 sensor specifications). The success of this PBD methodology hinges on establishing appropriate intermediate design platforms, their strategic placement, and component selection [27].

Our proposed design flow is structured into three distinct layers

- Function Design Layer
- Module Design Layer
- Implementation Design Layer

Each layer is associated with a specific library:

- Virtual Device Platform
- Module Platform
- Implementation Platform

In our framework, modules serve as fundamental design elements, representing basic functions like computation, sensing, and actuation. A prototype design, composed of interconnected modules, can be simulated in a design environment to verify its functionality before deployment. This modular abstraction decouples high-level functions from the specifics of the implementation platform (e.g., building infrastructure, device drivers, Building Operating System).

More complex functionalities are achieved by combining modules into sophisticated components. We introduce the concept of a "virtual device" - an abstract component built from modules that performs a high-level function. These virtual devices can serve as building blocks for various building functions, both existing and novel.

To efficiently leverage different implementation architectures, we employ architecture templates, including function templates and virtual device templates. These templates define the composition rules for assembling abstract components. Function templates specify how virtual devices combine to achieve certain functions, while virtual device templates outline how modules can be assembled into virtual devices.

Our design flow follows a library-based approach throughout:

- Function Design Layer: Top-level specifications are mapped to a prototype design using virtual devices and function templates from the virtual device library.
- Module Design Layer: The prototype design is further refined into a set of interconnected modules, again using a template-based DSE approach.
- Implementation Design Layer: All abstractions from previous layers are fully instantiated with hardware or software implementations on the target building platform.

This structured, layered approach allows for systematic refinement of the design, from high-level concepts to concrete implementations, while maintaining flexibility and optimizing for efficiency at each stage

3. Bottom-Up Design Flow

Our proposed design methodology employs a "meet-in-the-middle" approach, extensively utilizing library components across all design layers. This framework allows for flexibility: if existing library components fail to meet design specifications or yield unsatisfactory results, new elements can be designed and incorporated into the library. While our focus is primarily on cyber components, models of fixed physical elements (e.g., building structure, equipment, occupants) are included in the design library to provide a comprehensive environmental context.

A crucial aspect of the library system is the set of composition rules that govern the correct assembly of components throughout the design process. These rules fall into two main categories:

1. Port constraints: Ensure proper information types are exchanged between components.

2. Design rules: Guarantee functional correctness and compliance with relevant laws and building codes.

These rules are structured hierarchically, similar to the components themselves. For instance, a high-level rule like "no cameras in bathrooms" is implemented as a design constraint in the module design library, while a more specific rule such as "ZigBee-protocol sensors must be within 5m range" is applied during the final implementation phase as a device placement constraint.

Architecture templates, previously mentioned, represent a special class of composition rules. These become active only when selected for a design implementation and embody a set of structural constraints defining a family of potential solutions for a given functionality.

3.1. Implementation Platform

The foundation of our design library is the implementation platform. In our framework, this consists of a Building Operating System (BOS) and associated device drivers. The BOS is responsible for:

- Scheduling available hardware resources (computing, sensing, actuation, etc.)
- Managing inter-component communication
- Providing resource access through:
    - Application Programming Interfaces (APIs) for sensing and control points
    - Execution environments for computational resources

It's important to note that the actual implementation of the upper-level module platform may vary slightly when deployed in different buildings, depending on the specific BOS in use. This adaptability allows our design framework to accommodate diverse building environments and operating systems while maintaining a consistent overall structure.

3.2. Module Platform

The module platform library L_m is structured as a triplet (C_m, T_m, R_m), comprising modules Cm, virtual device templates Tm, and design rules Rm. Module behavior is typically represented through input-output relationships, with additional attributes or characteristics denoted by module labels. To accurately represent module costs, a dual-component approach is often employed: investment cost (initial setup expenses) and operational cost (ongoing expenses and potential depreciation).

Our design framework incorporates not only models for fundamental functions like sensing, actuation, and computation, but also representations of physical processes and human behaviors. The specific implementation of a module is contingent on the actual platform used. When multiple implementations exist for the same functionality, designers must decide whether to create a single, parameterized module or separate, distinct modules. This decision often hinges on the degree of similarity between implementations and involves balancing granularity with optimality.

Consider, for instance, two humidity sensing modules: one wired and one wireless. Given their significant differences in latency and cost, it may be more advantageous to design them as separate modules. This approach allows for simpler, more accurate latency and cost models for each module, enhancing overall design flexibility and precision.

Table 1 Sensing Module

Modules	**Behavioral Model**		**Extra-Functional Model**
Sensing Device	Input	Output	Cost
TMP36 Temperature Sensors	Temperature: 40 °C - 125 °C	Temperature measurement with ±1°C accuracy	Low ($1.5)
HIH-4030 SparkFun Humidity Sensor	Humidity: 20% RH - 80% RH	Relative humidity measurement with 15% RH accuracy	Low ($16.95)
OPT3001 Ambient Light Sensor	Illuminance: 0.01 - 83 klux	Brightness measurement with 0.5 lux accuracy	Low ($2.48)
Telaire T6615 CO2 Sensor	CO2: 0 - 5000 ppm	CO2 measurement with 75 ppm accuracy	Medium ($98.8)
Efergy E2 classic energy monitor	Power consumption: 5 W - 120 kW	Energy consumption measurement with 3W accuracy	Medium ($120)

We now outline several key module types commonly employed in constructing higher-level functions:

Sensing Modules: These provide fundamental data acquisition services. Typically, a sensing module consists of sensing hardware coupled with its corresponding device driver. Importantly, the concept of a "sensor" extends beyond traditional physical sensors; for instance, a smartphone interacting with a building occupant can function as a sensor. In our framework, we associate each piece of sensing hardware with a distinct sensing module that serves as its abstraction. Table 1 illustrates various sensing module examples, detailing their behavioral models (as input-output relations) and associated costs (as extra-functional properties).

Control Modules: These modules embody decision-making services for control actions, such as setpoint determination. In optimal control approaches like Model Predictive Control (MPC), the module processes a set of constraints and sensor readings, then employs an optimization engine to derive the optimal control action. Another prevalent approach is fuzzy control, where the module utilizes fuzzy logic to compute control actions.

Actuation Modules: These modules implement high-level control directives from control modules on physical building systems. For example, they might execute a command like "adjust airflow volume to 0.236 m3/s." Actuation modules are realized through specific drivers in the Building Operating System (BOS). These drivers encapsulate device-specific logic, providing a standardized interface that abstracts the underlying hardware complexities.

Data Analytics Modules encapsulate the processes of developing and applying predictive and inferential models. These modules often utilize machine learning algorithms to estimate contextual information such as occupancy levels and thermal comfort based on sensor inputs from the building environment.

Each data analytics module is conceptualized as a dual mode switched system:

- Training Mode: This phase involves learning or deriving model parameters from available data, including historical datasets, documentation, and expert knowledge. The output is a set of parameters defining a model for subsequent testing.
- Testing Mode: In this phase, the module applies the trained model to generate outputs based on input data and the established model parameters.

The module handles various data types:

- Categorical (e.g., user identities, event types)
- Ordinal (e.g., thermal preferences, event criticality)
- Measurement (e.g., air temperature, humidity)

Data analytics modules vary in their training requirements, which we classify into three intensity levels: low, medium, and high. For example:

- Low: Threshold-based occupancy inference from PIR and Wi-Fi data
- Medium: Physics-based methods like "sense-by-proxy" [14], requiring hours to a day of training
- High: Data-driven machine learning approaches, often needing several days of data collection

Algorithms can be categorized using various methods, with sample complexity measures like Rademacher complexity being popular for relating performance bounds to data volume [28]. From a practical standpoint, training intensity is crucial as it directly impacts the labor and maintenance efforts required for deployment.

3.3. Virtual Device Platform

The virtual device platform library L_f, analogous to the module platform, is structured as a triple (C_f, T_f, R_f), comprising virtual devices Cf, function templates T_f, and design rules Rf. Each virtual device's implementation architecture is defined by its corresponding template in the module platform library Lm. Virtual devices, like modules, are characterized by input-output relationships. Given the diversity of potential implementations, cost estimations for virtual devices may be expressed as ranges, nominal values, or qualitative assessments.

To illustrate, we present several examples of virtual devices, with their detailed behavioral and extra-functional models summarized in Table 2:

Virtual Occupancy Sensor: This device reports a zone's occupancy status, either as an integer (number of occupants) or a binary value (occupied/unoccupied). Multiple implementation architectures are possible, including CO2-based [14], camera-based, Wi-Fi-based [33], and PIR-based solutions. Each architecture is represented as a distinct virtual sensor in the library.

Virtual Fault Detection Sensor: This device monitors an HVAC system's operational health. Various methodologies have been proposed for HVAC fault detection and diagnosis, utilizing existing building data:

- Analytical-model-based: Relies on explicit system descriptions
- Signal-based: Examines correlations between faults and system outputs
- Knowledge-based or data-driven: Extracts insights and relevant features from building data, considering external environmental factors, internal loads, and mechanical system conditions [39], [40]

Each of these methodological approaches is embodied as a separate virtual fault detection sensor within the library. This diversity allows designers to select the most appropriate fault detection strategy based on the specific requirements and constraints of each building project.

A virtual VAV (Variable Air Volume) controller manages setpoints for a VAV box, such as air flow rate and discharge air temperature. Two common approaches that integrate occupant feedback are rule-based control and Model Predictive Control (MPC). Rule-based controllers use straightforward "if-else" logic for decision-making, while MPC employs predictive modeling and real-time optimization to determine optimal control actions.

The key distinction between the virtual device platform and the module platform is their abstraction level. The module platform comprises basic functions like sensing, actuation, and computation, whereas the virtual device platform encompasses higher-level functions built from these basic components. For example, a virtual occupancy sensor might combine a sensing module with a data analytics module to interpret occupancy information from raw sensor data.

Building functions are created by integrating virtual sensors and controllers into a closed-loop system that interacts with the physical environment and occupants. Function templates describe the interconnections between these components and are tagged with features defining their type and capabilities.

In this system, occupants vote on shared lighting brightness preferences, earning points based on the energy efficiency of their choices. The average vote determines the implemented lighting level, and accumulated points influence lottery-winning probabilities. A mobile app facilitates voting, point tracking, and observation of others' consumption patterns. The platform stores historical data, enabling building managers to model occupant behavior (utility functions) and design incentives to encourage energy-saving practices.

The architecture comprises an environmental component, a virtual sensor, and a virtual controller. These virtual devices can be implemented in various ways, as the underlying modules are not fully specified. For instance, the "utility function estimator" within the virtual sensor could use different parameter estimation techniques.

This example showcases how higher-level building functions can be constructed from virtual devices, which in turn are built from more basic modules. It also demonstrates the flexibility of the proposed design framework in accommodating diverse implementation strategies. For a more comprehensive explanation of this system, readers are directed to [41] and [42].

Table 2 Virtual Devices

Modules	**Behavioral Model**			**Extra-Functional Model**
Object	Method	Input	Output	Cost
Virtual occupancy sensor	CO2-based	Indoor CO2	Occupancy count with high accuracy and delay	Medium
	Camera-based	Illuminance > 50 lux	Occupancy count with high accuracy and no delay	High
	PIR	Human mid-infrared radiation changes	Occupancy status with mid accuracy and no delay	Low
Virtual environmental sensor	Temperature	Dry-bulb temperature	Ambient temperature	Low
	Temperature & humidity	Dry-bulb temperature, humidity	Ambient temperature, humidity	Low
Virtual VAV Controller	Rule-based	Occupancy count with at least mid accuracy, comfort preference, ambient temperature and humidity	HVAC air flow control	Medium

	MPC	Occupancy count with at least mid accuracy, comfort preference, ambient temperature and humidity	HVAC air flow control	High
Virtual lighting controller	Rule-based	Occupancy status with at least mid accuracy and zero delay	Lighting ON/OFF control	Medium
Virtual fault detection sensor	Model-based	Room air temperature, supply air temperature, humidity	If in the abnormal situation	Low
	Data-driven	Temperature, humidity, pressure, flow rate, damper position, fan speed, etc	Fault type and severity	High

4. Top-down design flow

The top-down design process begins with a set of high-level specifications provided by the designer. These specifications encompass both functional and extra-functional requirements for the final design. Functional requirements, such as "implement an occupancy-responsive HVAC system," are represented as a set of predefined features in the design automation tool. Extra-functional requirements, like "total retrofit cost must not exceed $100,000," are expressed as algebraic constraints.

4.1. Function Design Layer

In this initial layer, the input specifications are translated into a prototype design MT. This prototype is constructed by selecting and assembling virtual devices C and function templates T from the library. To ensure the prototype's functional correctness, the features offered by T must encompass all those specified in the functional requirements.

Although the components in this prototype design are not fully realized at this stage, it's still possible to generate a cost estimate cost (MT) based on the individual costs of the selected virtual devices. It's important to note that virtual sensors can be utilized across multiple functions, potentially reducing overall costs.

The mapping process in this layer can be framed as a combinatorial optimization problem. The objective is to identify the optimal combination of virtual devices and function templates that minimizes cost (MT), while adhering to the composition rules that act as constraints. This approach allows for a systematic exploration of design options within the defined parameters, balancing functionality and cost-effectiveness from the earliest stages of the design process.

4.2. Module Design Layer

Building upon the function architecture established in the previous layer, the module design layer aims to further refine the design by incorporating specific modules from the library. This layer provides a more detailed view of the final design, enabling more precise performance and cost estimations during the Design Space Exploration (DSE) process.

This stage also uncovers new opportunities for component reuse that weren't apparent earlier, potentially leading to further cost reductions. For instance, a single multiuse building-in-briefcase (BiB) sensing module [43] might serve the functions of both humidity and CO2 sensing, originally specified as separate virtual devices, provided this doesn't create conflicts between individual module specifications.

The mapping process in this layer, while still template-based, differs from the previous layer in two key aspects:

- It involves selecting both components and their parameters, making it a mixed discrete-continuous optimization problem.
- It allows for the use of simulation techniques to more accurately estimate operational costs.

To manage the potentially infinite parameter space, discretization techniques and empirically based heuristics can be employed to expedite the search for a satisfactory design.

A significant advantage of the Platform-Based Design (PBD) approach is its capacity to identify integration errors early in the design process. While predefined composition errors are avoided through the enforcement of design rules, subtle interaction-based integration errors may still occur. For example, in a smart lighting system using a camera-based occupancy sensor, the system might fail to detect people entering a dark room, violating the intended functionality (occupants present ↔ lights on).

Although components aren't fully instantiated at this stage, such issues can be identified using formal methods like model checking [44] or through simulation techniques. When errors are detected, the DSE process continues iteratively until a validated design is achieved.

This approach allows for a more robust and error-resistant design process, catching potential issues before they become costly problems in the implementation phase.

4.3. Implementation Design Layer

The final layer focuses on determining the specific details for deploying the designed functions in the target building. This process assumes the presence of a Building Operating System (BOS) that simplifies much of the deployment complexity. Key tasks in this layer typically include:

- Deploying the BOS
- Strategically placing and configuring sensing devices

The standardized access to diverse building resources provided by BOSs has significantly simplified module deployment. Consider a retrofit scenario where an intelligent HVAC function is being implemented, requiring the installation of additional sensors. This function aims to conserve energy by dynamically adjusting each room's discharge air temperature setpoint based on occupancy. The required modules can be converted into executable code for the target BOS platform using automated code generation techniques, as demonstrated in previous research [15]. For instance, high-level descriptions like "discharge air temperature setpoint in Room 406" can be automatically translated into unique BACnet identifiers during the code generation process.

The challenge of sensor placement has been addressed through various approaches in literature [45], often framed as either an optimization problem or a constraint satisfaction problem. The solution must adhere to specified design rules and requirements. For example, CO2-based occupancy detection sensors must be positioned near air exhaust vents in mixing ventilation systems [46], or within occupied zones in displacement or underfloor air distribution systems [47], to accurately measure exhaust air CO2 levels.

While beyond this paper's scope, the deployment phase presents additional system design challenges, such as efficient scheduling and partitioning of building functions on the execution platform. Although many building functions don't have strict real-time requirements like other time-critical cyber-physical systems (CPSs), running numerous functions concurrently on the BOS can lead to significant system latencies due to service requests (e.g., data acquisition). The proposed integrated design approach can mitigate these issues by encouraging component reuse at the function design level.

Given recent advancements in system-level design for networks [16], [18] and software [48], [49], we anticipate that the Platform-Based Design (PBD) paradigm will play a crucial role in addressing these challenges within the BOS context, further enhancing the efficiency and effectiveness of smart building implementations.

5. Case study

This section presents a case study to demonstrate the application of our proposed design flow. We consider a scenario where a designer aims to retrofit an existing building's HVAC system to enhance energy efficiency and occupant responsiveness within a reasonable budget. The case study illustrates how the designer's specifications are translated into a concrete hardware and software implementation using the Platform-Based Design (PBD) approach.

Initially, the building designer outlines a set of project requirements, including:

- Targeted energy consumption savings
- Comfort criteria
- Budget limitations
- Expected return on investment timeline

Additionally, a comprehensive building model (e.g., a Building Information Modeling (BIM) file) is provided to the design tool, offering detailed information about the building's structure, occupancy patterns, and other relevant data.

For this case study, let's assume the following specifications:

- Building classification: Commercial
- Total area: 10,000 m^2
- Current yearly energy expenses: $5,000
- Target system for retrofit: HVAC
- Comfort parameters: Static setpoints (20°C–25°C)

Financial constraints:

- Minimum energy savings: 30%
- Maximum payback period: 3 years
- Initial investment cap: $100,000

In this context, comfort requirements are classified as either "static setpoints" or "personalized." The "static setpoints" approach adheres to standardized thermal comfort guidelines, such as ASHRAE 55-2013 [50], maintaining dry-bulb air temperature within a specified range (e.g., 20°C–25°C), possibly with additional relative humidity constraints. This method is cost-effective to implement as it doesn't require extra infrastructure for gathering occupant feedback. However, it may not account for individual preferences or building-specific characteristics.

Conversely, "personalized" control strategies, like Comfy [51], allow for customized comfort ranges based on occupant preferences. These can include personal comfort systems such as heated/cooled chairs [52] or desk fans [53], potentially improving satisfaction and energy efficiency. Personalization can be implemented at group or individual levels [54] and can adapt to changing comfort needs due to environmental or physiological/psychological factors. However, implementing personalized comfort systems requires additional investment in hardware (e.g., comfort devices, sensors) and software (BOS capable of processing occupant feedback).

For this case study, we're using a static temperature range as the comfort requirement. However, it's important to note that our design paradigm can accommodate more complex specifications, including dynamic setpoints and considerations for other comfort factors like humidity and air quality. The design framework is flexible and can be adapted to various specifications. For example, if humidity control were required, the system would incorporate humidity sensors and controllers capable of real-time humidity data processing.

5.1. Function Design Layer

In this initial layer, the mapping engine refines the high-level design specifications into a set of virtual device components. These components are based on function templates selected from the library, forming an architecture instance.

The HVAC templates are:

- APP-HVAC-1: Labeled "static setpoints," this template infers comfort levels from ambient environmental measurements, without direct occupant interaction.
- APP-HVAC-2: Labeled "personalized," this template requires occupant feedback to adjust the comfort range for building control.

The virtual device platform library contains three categories of virtual devices. Each virtual device abstraction includes essential implementation details such as behavior, cost, and performance metrics, enabling the design automation tool to make informed selections.

Virtual Occupancy Sensors: Represented by three variants in the library:

- VS-occup-1: Offers superior occupancy counting accuracy but is typically expensive and may raise privacy concerns.
- VS-occup-2: Implements a "sense-by-proxy" [14] approach, inferring occupancy from CO2 concentration. While more cost-effective than VS-occup-1, it often experiences some latency in occupancy estimation. This

latency, while not critical for HVAC control, makes it unsuitable for applications like lighting control that require immediate response.

- VS-occup-3: Provides binary occupancy status (occupied/unoccupied).

This diverse range of virtual devices allows the design tool to select the most appropriate components based on the specific requirements and constraints of the project, balancing factors such as accuracy, cost, privacy considerations, and response time.

- Virtual Environmental Sensors: The virtual device library includes three environmental sensor options:
    - VS-env-1: A multimodal sensor capable of measuring both temperature and humidity in indoor spaces.
    - VS-env-2: A unimodal sensor dedicated to temperature measurement.
    - VS-env-3: A unimodal sensor specifically for humidity detection.
- Virtual VAV Controllers: Two types of virtual Variable Air Volume (VAV) controllers are available,
    - VC-vav-1: Implements rule-based control, a widely adopted approach in existing buildings. This controller uses straightforward "if-else" logic for decision-making, making it relatively simple to set up and deploy.
    - VC-vav-2: Utilizes Model Predictive Control (MPC), an advanced optimal control strategy gaining significant attention in recent years. While more complex to implement, MPC offers potential benefits:

Requires more comprehensive input data, including temperature, humidity, and occupancy measurements.

Incorporates both a thermodynamic model and a comfort model to compute optimal control actions.

Has demonstrated superior energy-saving capabilities compared to rule-based controllers in previous studies [8], largely due to its predictive and optimization capabilities.

The choice between these controller types involves a trade-off between ease of implementation and potential energy savings. While rule-based controllers are simpler to deploy, MPC controllers offer the promise of greater efficiency through more sophisticated decision-making processes. This diversity in virtual device options allows the design tool to select the most appropriate components based on the specific project requirements, balancing factors such as energy efficiency, implementation complexity, and available resources.

The Design Space Exploration (DSE) process in this layer can be formulated as a discrete optimization problem, aiming to find the optimal mix of function templates and virtual devices. Consider a design library with m function templates and *n* virtual devices. We define:

- $z \in \{0, 1\}$ ^m: A binary array indicating the selection status of each template T_i
- $x \in N$^n+: An array representing the selection of virtual devices, allowing multiple instances of each device
- c_i: The purchase cost associated with each virtual device V_i
- (z, x): The operational cost linked to the chosen function template and virtual devices

The operational cost o (z, x) can be estimated through simulations during library construction. Fig. 1 demonstrates this concept, showing heat and electricity consumption for a simple building using various virtual controllers and occupancy sensors under the APP-HVAC-1 function template. The simulation employs real-world occupancy data and San Francisco weather information.

Key insights from the simulation include:

- An occupancy counter is crucial for Model Predictive Control (MPC) to achieve higher energy efficiency in building control.
- Increasing the accuracy of occupancy counting beyond a certain point does not yield significant additional energy savings.

The operational cost can be derived from the simulated energy consumption data, combined with local energy prices for the building's location. This approach allows for a more accurate estimation of long-term costs and benefits associated with different design choices, enabling the optimization process to balance initial investment against operational efficiency.

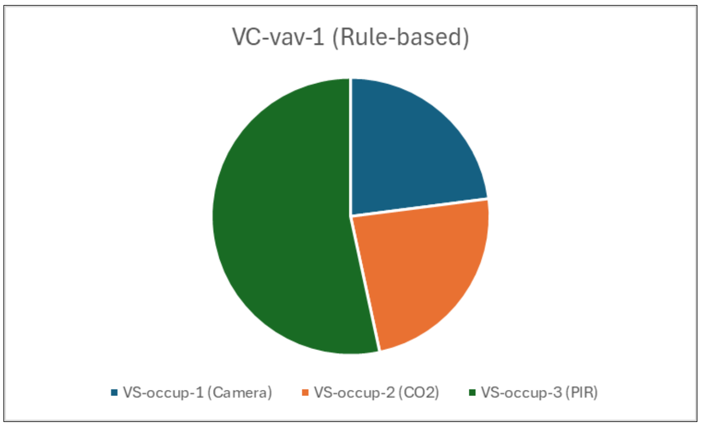

Figure 1(a) Site Heat Energy Consumption

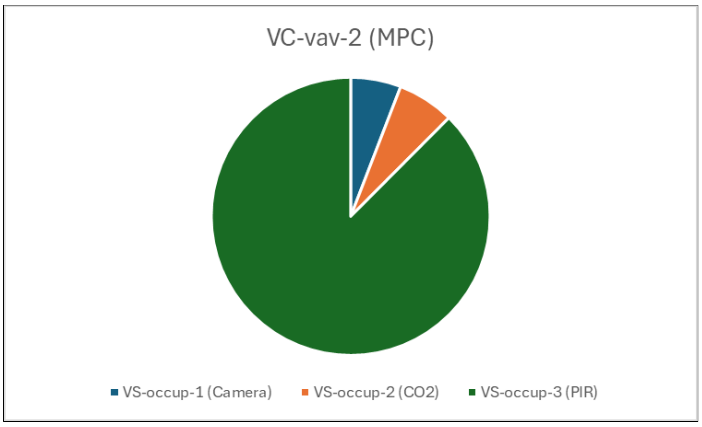

Figure 1(b) Site Heat Energy Consumption

Let's define our key parameters:

- PH: Maximum allowed payback horizon
- ES: Minimum expected energy savings
- SC: Maximum available starting capital
- EB: Current annual energy bill pre-retrofit

For our case, PH = 3 years, ES = 30%, and SC = $100,000 (as per your earlier instruction to use $100,000 instead of $50,000).

We can express the payback horizon for the retrofit investment as:

$$h(z, x) = (\Sigma\ (i=1 \text{ to } n)\ x_i c_i) / (EB - o(z, x))\ (1)$$

where x_i and c_i represent the quantity and cost of each virtual device, respectively.

We can then formulate an optimization problem to minimize the payback horizon:

Minimize $h(z, x)$ (2a)

Subject to:

$$z \mid= \psi, (z, x) \mid= \varphi\ (2b)$$

$$h(z, x) \leq PH\ (2c)$$

$$o(z, x) \leq (1 - ES)\ EB \quad (2d)$$

$$\Sigma(i=1 \text{ to } n)\ x\_i\ c\_i \leq SC \quad (2e)$$

In constraint (2b), the |= operator denotes a "satisfies" relationship between binary variables and their properties. Here, z |= ψ ensures all required features are included in the design, while (z, x) |= φ ensures adherence to composition rules.

If this optimization problem yields an infeasible result, the design automation tool can analyze the conflicting specifications and offer recommendations to the designer. Otherwise, the design process advances to the next stage.

Based on the virtual device attributes in our example library, the mapping engine's selected implementation is illustrated in Fig. 2.

This approach allows for a systematic exploration of the design space, balancing multiple objectives and constraints to find an optimal solution that meets both functional requirements and economic considerations.

5.2. Module Design Layer

This layer's primary goal is to realize each virtual device from the upper layer using specific modules and predefined templates from the module library.

5.2.1. Virtual VAV Controller Templates

For simplicity, we use identical identifiers for both virtual devices and their architecture templates. The module platform offers two types of VAV controller templates.

- VC-vav-1: A rule-based controller that processes real-time measurements of temperature, humidity, and occupancy, along with predefined comfort parameters. It adjusts air flow rate and temperature based on encoded rules such as:

"Minimum airflow rate proportional to occupant count"

"Wider temperature range during unoccupied periods"

- VC-vav-2: A Model Predictive Control (MPC) scheme utilizing an optimal control module (CTRL-1). This module formulates and solves a real-time optimization problem to determine optimal control actions. In addition to current environmental data, it requires:
    - Future occupancy predictions (ML-occup-pred)
    - A thermodynamic model for state prediction (ML-phy)
    - Comfort models for various occupancy states (ML-comf)

5.2.2. Virtual Occupancy Sensor Templates

The module library provides three types,

- VS-occup-1: An architecture employing a camera module and computer vision algorithm to count occupants in video streams.
- VS-occup-2: Utilizes a "sense-by-proxy" [14] approach, inferring occupancy from CO2 concentration measurements.
- VS-occup-3: Represents a standard Passive Infrared (PIR) sensing system, detecting binary occupancy status.

These templates offer a range of occupancy sensing solutions, balancing factors such as accuracy, cost, and privacy considerations. The design tool can select the most appropriate option based on the specific requirements of the project.

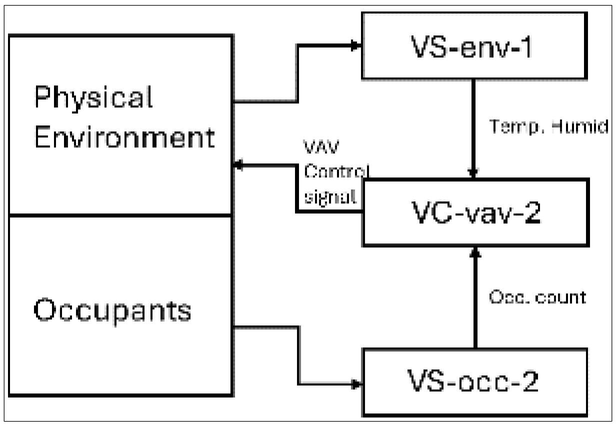

Figure 2 The mapping result after the function design layer

5.2.3. Virtual Environmental Sensor Templates

The module library encompasses various virtual environmental sensors for measuring different environmental aspects.

- A multimodal sensor capable of measuring both temperature and humidity
- Two unimodal sensors: one for temperature and another for humidity measurements

5.2.4. Environmental Sensing Modules

The design library incorporates multiple environmental sensors to fulfill the prototype design's requirements (temperature, humidity, and CO2 measurement). A notable example is Sensing-1, an integrated multimodal Building-in-Briefcase (BiB) device that can perform multiple sensing functions concurrently.

5.2.5. Data Analytics Modules

Our library features eight data analytics modules. Two of these (ML-1 and ML-2) are versatile and applicable to various scenarios, while the others are designed for specific use cases, limiting their application to certain virtual device templates.

- ML-1: Implements a Convex Piecewise Linear Classifier (CPLC) [56], which uses a polygonal envelope represented by a set of linear constraints.
- ML-2: Utilizes a neural network, exemplifying nonlinear classifiers.
- ML-3: Employs a K-Nearest Neighbors (KNN) method, a deterministic approach.
- ML-4: Uses a Markov chain model, providing a stochastic prediction method.

ML-3 and ML-4 are commonly used for occupancy prediction. The stochastic nature of ML-4 allows it to output probability distributions of occupancy counts, enabling stochastic control algorithms that can better manage system uncertainties. In contrast, deterministic methods like ML-3 provide specific future occupancy estimates, making them easily integrable into simple deterministic Model Predictive Control (MPC) schemes for occupancy-responsive climate control.

This diverse range of modules allows for flexible and tailored solutions in smart building design, accommodating various prediction and control needs.

The library also incorporates thermal dynamics models of varying complexity, generally categorized into three types:

- White-box models: These are detailed physical models, such as Energy Plus [57] (represented by ML-7) or Modelica [58]. They offer high accuracy when parameters are correctly identified. However, they often require significant computational resources and setup time. They can be challenging to implement when building documentation is incomplete or when the building environment is subject to frequent changes.
- Black-box models: These are data-driven, non-physical models. They are computationally efficient but rely heavily on the availability and quality of historical data for training to achieve accuracy. ML-2 is an example of a black-box model in our library.

- Gray-box models: These represent a middle ground between white-box and black-box models, typically employing simplified physical models. Examples in our library include:
- ML-5: An abstraction of the temperature model from [55], using a simple bilinear model based on conservation laws.
- ML-6: A thermal circuit model that uses a resistance-capacitance network to simulate heat transfer between different building zones (interiors, walls, windows, etc.).

Gray-box models often strike a balance, providing reasonable accuracy with lower computational demands compared to white-box models. However, they present challenges in parameter estimation.

This diverse range of thermal models allows designers to choose the most appropriate approach based on the specific requirements of the project, available data, computational resources, and desired accuracy. The flexibility to select between different model types enables more efficient and tailored solutions for various building scenarios.

6. Control Modules

The module platform includes several control modules designed to compute control actions from specified inputs. CTRL-1 is responsible for solving an optimization problem to identify the optimal control action. CTRL-2 represents a rule-based control module, as proposed in [59], which adjusts room temperature setpoints based on the current time and the duration of occupancy or vacancy periods. Similarly, CTRL-3 is another rule-based module but applies a different set of rules described in [8] to manage the environment, dynamically altering the flow rate and temperature setpoints according to occupancy changes and occupant comfort preferences.

In the function design layer, VS-env-1, VS-occ-2, and VC-vav-2 have been selected to implement a Model Predictive Control (MPC) scheme. For the virtual occupancy sensor VS-occ-2, the design automation tool opts for ML-8, known as the “sense-by-proxy” module, due to its superior occupancy detection accuracy compared to other machine learning algorithms, as detailed in [14]. For the virtual environment sensor VS-env-1, multiple implementation options exist, such as employing the single module Sensing-1, which is a multimodal BiB sensing solution covering all three sensing needs or using each sensing function individually with the unimodal sensing modules Sensing-2, Sensing-3, and Sensing-4.

For the virtual controller VC-vav-2, CTRL-1 is used because it is the only control module available in our library that fits the VC-vav-2 template. The ML-phy module candidates are ML-5 and ML-6, as CTRL-1 necessitates an explicit algebraic model to frame an optimization problem for the MPC scheme. For modeling comfort regions, both ML-1 and ML-2 are suitable. Although the neural network model (ML-2) might offer a more detailed description of the comfort region through nonlinear boundaries, the polygonal comfort region defined by the CPLC (ML-1) is very compatible with integration into an MPC scheme. As our optimal control module only supports deterministic MPC formulations, the KNN model (ML-3) is a more suitable occupancy predictor than the Markov chain model (ML-4).

The Design Space Exploration (DSE) in this layer is complex, involving optimization across both a discrete search space, for the virtual device templates and modules, and a continuous search space, for the configuration parameters of the selected modules. Various factors, including cost and computational performance, must be considered, rendering the DSE a multiobjective optimization problem. Advanced search and optimization algorithms [60], [61] from other domains can be beneficial for solving this problem. Moreover, simulation techniques can be used during DSE to evaluate trade-offs between design instances, as designs at this stage are executable. Based on the above discussions, the design shown in Fig. 3 is considered the final mapping result for this layer.

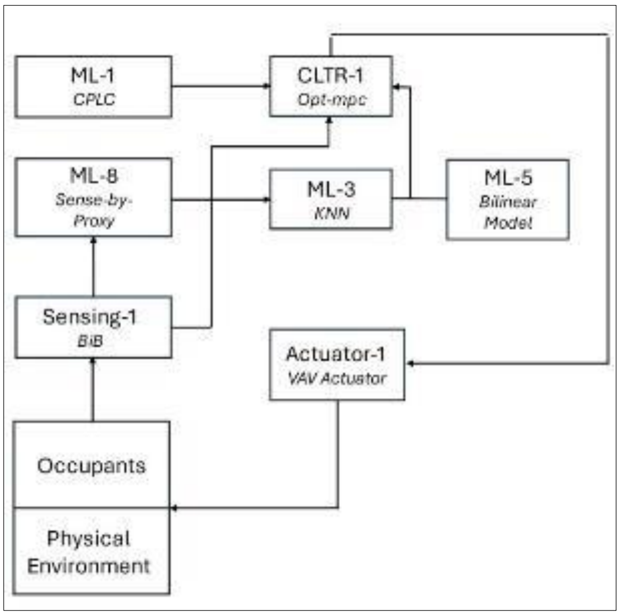

Figure 3 The mapping result after the module design layer

6.1. Implementation Design Layer

In this phase, the focus is on implementing the designed functions into the building infrastructure, which involves both configuring hardware and developing software.

Any sensors required by the design that are not already present in the building need to be acquired, calibrated, and installed in appropriate locations. This process must comply with the constraints outlined in the previous design stages as well as aesthetic and legal considerations. For example, the BiB sensors in our setup must measure CO2 for the occupancy counting algorithm (SBP), as well as monitor temperature and humidity to predict comfort and room dynamics. Therefore, these sensors should be positioned near ventilation outlets and in areas where people frequently spend time. This placement ensures that CO2 measurements accurately reflect the cumulative CO2 levels from occupants, and that temperature and humidity readings correspond to the conditions most occupants experience.

The purpose of software synthesis is to produce BOS-executable code for the data analytics and control modules identified in the module design phase. The Another Tool for Language Recognition (ANTLR) framework [62] can be employed to automatically translate high-level programming languages used for modeling and simulation into a format compatible with BOS.

7. Conclusion and Future Work

This paper has presented an integrated design flow for smart building functions utilizing a Platform-Based Design (PBD) methodology, which facilitates effective Design Space Exploration (DSE) for realizing building functions. A case study focused on designing an HVAC function was used to demonstrate the proposed design flow. This methodology can also be applied to other building services such as water systems, fire protection, Internet, electrical systems, elevators, and security systems, although our examples primarily focused on HVAC and lighting systems. We would like to highlight several challenges not addressed in this paper.

7.1. Populating the Libraries

An important direction for future work is to further enrich the design libraries with additional resources. This involves a) identifying and validating behavioral models that accurately predict component behavior; b) ensuring that library components function correctly in various environments; c) defining parameter variations, operational modes, and constants for each component; and d) establishing component interfaces and composition rules to enable plug-and-play integration of various components. Incorporating resources into the library and maintaining them can be costly. Therefore, the selection of resources to include in the libraries must be strategic. A wide range of models for building automation has already been developed, from occupancy [63] to HVAC systems [64]. As this design paradigm gains traction, vendors will be motivated to populate libraries by creating simulation models of their components. A design ecosystem in which vendors are continually incentivized, possibly through an intellectual property market, to enhance the library with thoroughly vetted models of their components will be crucial for the success of the proposed paradigm.

7.2. Enhancing DSE

A significant challenge in the design flow is the difficulty in accurately assessing the impact of design decisions made early in the process. For instance, the operational cost of a virtual device, assumed to be a known value in our case study, depends not only on its implementation but also on the physical equipment it operates on, making precise estimation challenging. We believe that data-driven techniques, such as those discussed in [65], can address this issue, turning future design flows into a "collective" effort. Cost and performance models, trained with both simulated data from the design environment and operational data from existing buildings, will guide the design flow. In the later stages, the DSE process becomes a mixed discrete-continuous optimization problem, requiring the selection of both candidate modules and their configuration parameters. Thanks to advancements in optimization techniques, mixed-integer programs can now be solved on a large scale using off-the-shelf solvers. However, time-consuming simulation runs are often necessary to evaluate a design's performance, which can hinder effective DSE, especially when the design space is vast. Similar challenges are present in other CPS design domains, such as aircraft environment control systems [66], aircraft electric power systems [23], [67]–[69], and wireless networks [70], [71]. To manage the complexity of the DSE process, an iterative DSE framework has been proposed in recent literature [23], [66]–[71]. The key to an efficient search strategy lies in the feedback from the candidate evaluation engine, which helps narrow down a large portion of the discrete design space based on insights from simulation runs; for a detailed account, refer to Finn et al.'s work [66].

7.3. Toward a Holistic Design Flow

Designing a smart building encompasses not only the implementation of energy-efficient systems like HVAC and lighting but also involves the overall design of the building's form, structure, interior, facade, and cultural expression. This paper does not delve into these latter aspects. The co-design of cyber, physical, and human elements in a building is likely a more complex challenge than what we have discussed; however, we believe the proposed PBD paradigm will be a crucial enabler for a comprehensive future design flow. Achieving this vision will require more effective modeling of cyber, physical, and human interactions, improved interoperability of design tools, and more robust DSE engines that can continuously refine a design based on insights from the design exploration process. Recent advancements in computer-aided design (CAD) and Building Information Modeling (BIM) technologies have yielded significant results in architectural design, such as streamlining the construction process and allowing architects to conduct basic energy performance assessments. However, these capabilities alone are inadequate for the design of future smart buildings. A major hurdle is the integration of models and design tools from various domains, which limits the ability to consider the impact of smart functions on design performance through simulation. An ongoing effort to facilitate model exchange and co-simulation is the functional mockup interface (FMI) standard [72]. This standard is particularly appealing as it enables the coupling of multiple simulation tools in a co-simulation environment. An example of this approach is found in [73], where an HVAC system model is packaged as a functional mockup unit (FMU) and linked to a room model in Energy Plus [57], allowing the performance of the composite system to be evaluated through co-simulation.

7.4. Handling Uncertainties

Smart buildings operate within a complex interplay of cyber components, dynamic physical elements, and human activities, necessitating design and operation that account for various uncertainties. These uncertainties range from operational unpredictability and manufacturing variability to model errors. Operational uncertainty involves the randomness of the environments in which smart building systems function. For instance, building environments are continuously altered by occupant activities, such as heat generation and CO2 emissions, or their interactions with windows, blinds, and lighting. Probabilistic modeling often addresses this environmental uncertainty [74]. The unpredictable nature of operational conditions poses challenges for design verification. Mosalam et al. [75] recently proposed a performance-based engineering method to incorporate uncertainty into building design, where the optimal design decision is made by comparing the expected utility of different designs. It is noteworthy that similar challenges are encountered in integrated circuit (IC) design, which deals with a vast number of functional states and state transitions. There has been significant research in the IC design community on balancing the trade-off between the cost and benefit of test generation, and we believe smart building design can benefit from these innovations.

Manufacturing processes, including the production and assembly of different equipment into a system, also introduce variability. Addressing this variability requires accurate modeling of the manufacturing process and ensuring sufficient design parameter margins to explicitly account for such variability. Additionally, developing models for various cyber and physical components in buildings is a complex task. Current building energy simulation software, such as Energy Plus [57], often relies on idealized physical models to calculate thermal loads, system responses, and energy use by solving physics equations. We anticipate that combining data-driven approaches with physics modeling will result in more accurate and adaptive models.

We view the aforementioned challenges as open questions for future research. It is believed that with the emergence of an open standard for design libraries, the PBD paradigm will significantly transform the design and operation of smart buildings.

Compliance with ethical standards

Disclosure of conflict of interest

No conflict of interest to be disclosed.

References

[1] N. E. Klepeis et al., "The National Human Activity Pattern Survey (NHAPS): A resource for assessing exposure to environmental pollutants," J. Exposure Sci. Environ. Epidemiol., vol. 11, no. 3, pp. 231–252, 2001.

[2] Commercial Buildings Energy Consumption Survey (CBECS), 2015.

[3] M. Frontczak, S. Schiavon, J. Goins, E. Arens, H. Zhang, and P. Wargocki, "Quantitative relationships between occupant satisfaction and satisfaction aspects of indoor environmental quality and building design," Indoor Air, vol. 22, no. 2, pp. 119–131, 2012.

[4] S. Altomonte and S. Schiavon, "Occupant satisfaction in LEED and non-LEED certified buildings," Building Environ., vol. 68, pp. 66–76, Oct. 2013.

[5] World Green Building Trends 2016 Smart market Report, Dodge Data Anal., Inc., New York, NY, USA, 2016.

[6] I. Lobachev and E. Cretu, "Smart sensor network for smart buildings," in Proc. IEEE 7th Annu. Inf. Technol. Electron. Mobile Commun. Conf. (IEMCON), Oct. 2016, pp. 1–7.

[7] G. Fierro and D. E. Culler, "XBOS: An extensible building operating system," in Proc. 2nd ACM Int. Conf. Embedded Syst. Energy-Efficient Built Environ., 2015, pp. 119–120.

[8] S. Goyal, H. A. Ingley, and P. Barooah, "Occupancy-based zone-climate control for energy-efficient buildings: Complexity vs. performance," Appl. Energy, vol. 106, pp. 209–221, Jun. 2013.

[9] P. G. Rowe, Design Thinking. Cambridge, MA, USA: MIT Press, 1991.

[10] Lennox. Accessed: Jul. 25, 2017. [Online]. Available: [http://www.lennoxcommercial.com/products/indoor-air-quality/demand-control-ventilation/] (http://www.lennoxcommercial.com/products/indoor-air-quality/demand-control-ventilation/)

[11] Lutron. Accessed: Jul. 25, 2017. [Online]. Available: [http://www.lutron.com/en/US/Pages/default.aspx] (http://www.lutron.com/en-US/Pages/default.aspx)

[12] J. Taneja, A. Krioukov, S. Dawson-Haggerty, and D. Culler, "Enabling advanced environmental conditioning with a building application stack," in Proc. Int. Green Comput. Conf. (IGCC), Jun. 2013, pp. 1–10.

[13] R. Jia, M. Jin, H. Zou, Y. Yesilata, L. Xie, and C. Spanos, "MapSentinel: Can the knowledge of space use improve indoor tracking further?" Sensors, vol. 16, no. 4, p. 472, 2016.

[14] M. Jin, N. Bekiaris-Liberis, K. Weekly, C. Spanos, and A. M. Bayen, "Sensing by proxy: Occupancy detection based on indoor CO2 concentration," in Proc. 9th Int. Conf. Mobile Ubiquitous Comput. Syst. Services Technol. (UBICOMM), 2015, pp. 1–14.

[15] What is Project Delivery Partners, How to Implement it. R. Parekh. & Hofstra University. (2023). ResearchGate https://doi.org/10.13140/RG.2.2.10185.11360.

[16] L. Carloni, F. De Bernardinis, A. L. Sangiovanni-Vincentelli, and M. Sgroi, "The art and science of integrated systems design," in Proc. 28th Eur. Solid-State Circuits Conf. (ESSCIRC), 2002, pp. 25–36.

[17] M. Di Natale and A. L. Sangiovanni-Vincentelli, "Moving from federated to integrated architectures in automotive: The role of standards, methods and tools," Proc. IEEE, vol. 98, no. 4, pp. 603–620, Apr. 2010.

[18] A. Bonivento, L. P. Carloni, and A. Sangiovanni-Vincentelli, "Platform based design for wireless sensor networks," Mobile Netw. Appl., vol. 11, no. 4, pp. 469–485, 2006.

[19] A. Ferrari and A. Sangiovanni-Vincentelli, "System design: Traditional concepts and new paradigms," in Proc. Int. Conf. Comput. Design (ICCD), Oct. 1999, pp. 2–12.

[20] K. Keutzer, A. R. Newton, J. M. Rabaey, and A. Sangiovanni-Vincentelli, "System-level design: Orthogonalization of concerns and platform-based design," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 19, no. 12, pp. 1523–1543, Dec. 2000.

[21] M. Sgroi, A. L. Sangiovanni-Vincentelli, F. De Bernardinis, C. Pinello, and L. P. Carloni, "Platform-based design for embedded systems," Tech. Rep., 2005.

[22] A. Sangiovanni-Vincentelli, L. Carloni, F. De Bernardinis, and M. Sgroi, "Benefits and challenges for platform-based design," in Proc. 41st Annu. Design Autom. Conf., 2004, pp. 409–414.

[23] P. Nuzzo, A. L. Sangiovanni-Vincentelli, D. Bresolin, L. Geretti, and T. Villa, "A platform-based design methodology with contracts and related tools for the design of cyber-physical systems," Proc. IEEE, vol. 103, no. 11, pp. 2104–2132, Nov. 2015.

[24] P. Nuzzo, A. Sangiovanni-Vincentelli, X. Sun, and A. Puggelli, "Methodology for the design of analog integrated interfaces using contracts," IEEE Sensors J., vol. 12, no. 12, pp. 3329–3345, Dec. 2012.

[25] J. Sztipanovits and G. Karsai, "Model-integrated computing," Computer, vol. 30, no. 4, pp. 110–111, Apr. 1997.

[26] E. A. Lee and Y. Xiong, "System-level types for component-based design," in Proc. Int. Workshop Embedded Softw., 2001, pp. 237–253.

[27] A. Sangiovanni-Vincentelli, "Quo vadis, SLD? Reasoning about the trends and challenges of system level design," Proc. IEEE, vol. 95, no. 3, pp. 467–506, Mar. 2007.

[28] P. Bartlett and S. Mendelson, "Rademacher and Gaussian complexities: Risk bounds and structural results," J. Mach. Learn. Res., vol. 3, pp. 463–482, Nov. 2002.

[29] J. Álvarez, J. L. Redondo, E. Camponogara, J. Normey-Rico, M. Berenguel, and P. M. Ortigosa, "Optimizing building comfort temperature regulation via model predictive control," Energy Buildings, vol. 57, pp. 361–372, Feb. 2013.

[30] I. Hazyuk, C. Ghiaus, and D. Penhouet, "Optimal temperature control of intermittently heated buildings using model predictive control: Part II–Control algorithm," Building Environ., vol. 51, pp. 388–394, May 2012.

[31] D. Kolokotsa, "Comparison of the performance of fuzzy controllers for the management of the indoor environment," Building Environ., vol. 38, no. 12, pp. 1439–1450, 2003.

[32] Y.-J. Wen and A. M. Agogino, "Wireless networked lighting systems for optimizing energy savings and user satisfaction," in Proc. IEEE Wireless Hive Netw. Conf. (WHNC), Aug. 2008, pp. 1–7.

[33] H. Zou, B. Huang, X. Lu, H. Jiang, and L. Xie, "A robust indoor positioning system based on the procrustes analysis and weighted extreme learning machine," IEEE Trans. Wireless Commun., vol. 15, no. 2, pp. 1252–1266, Feb. 2016.

[34] S. Katipamula and M. R. Brambley, "Review article: Methods for fault detection, diagnostics, and prognostics for building systems—A review, part II," HVAC R Res., vol. 11, no. 2, pp. 169–187, 2005.

[35] X. Dai and Z. Gao, "From model, signal to knowledge: A data-driven perspective of fault detection and diagnosis," IEEE Trans. Ind. Informat., vol. 9, no. 4, pp. 2226–2238, Nov. 2013.

[36] Y. Yu, D. Woradechjumroen, and D. Yu, "A review of fault detection and diagnosis methodologies on air-handling units," Energy Buildings, vol. 82, pp. 550–562, Oct. 2014.

[37] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches," IEEE Trans. Ind. Electron., vol. 62, no. 6, pp. 3757–3767, Jun. 2015.

[38] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part II: Fault diagnosis with knowledge-based and hybrid/active approaches," IEEE Trans. Ind. Electron., vol. 62, no. 6, pp. 3768–3774, Jun. 2015.

[39] D. J. Cook and S. K. Das, "How smart are our environments? An updated look at the state of the art," Pervasive Mobile Comput., vol. 3, no. 2, pp. 53–73, 2007.

[40] A. Purarjomandlangrudi, A. H. Ghapanchi, and M. Esmalifalak, "A data mining approach for fault diagnosis: An application of anomaly detection algorithm," Measurement, vol. 55, pp. 343–352, Sep. 2014.

[41] I. C. Konstantakopoulos, L. J. Ratliff, M. Jin, C. Spanos, and S. S. Sastry, "Smart building energy efficiency via social game: a robust utility learning framework for closing–the–loop," in Proc. 1st Int. Workshop Sci. Smart City Oper. Platforms Eng. (SCOPE) Partnership Global City Teams Challenge (GCTC)(SCOPE-GCTC), Apr. 2016, pp. 1–6.

[42] I. C. Konstantakopoulos, L. J. Ratliff, M. Jin, S. S. Sastry, and C. J. Spanos, "A robust utility learning framework via inverse optimization," IEEE Trans. Control Syst. Technol., vol. 26, no. 3, pp. 954–970, May 2017.

[43] K. Weekly, M. Jin, H. Zou, C. Hsu, A. Bayen, and C. Spanos (2014). "Building-in-briefcase (BiB)." [Online]. Available: https://arxiv.org/abs/1409.1660

[44] E. M. Clarke, Jr., O. Grumberg, and D. Peled, Model Checking. Cambridge, MA, USA: MIT Press, 1999.

[45] M. Jin, R. Jia, and C. Spanos, "APEC: Auto planner for efficient configuration of indoor positioning system," in Proc. 9th Int. Conf. Mobile Ubiquitous Comput. Syst. Services Technol. (UBICOMM), 2015, pp. 100–107.

[46] W. J. Fisk and A. T. De Almeida, "Sensor-based demand-controlled ventilation: A review," Energy Buildings, vol. 29, no. 1, pp. 35–45, 1998.

[47] S. Schiavon, F. Bauman, B. Tully, and J. Rimmer, "Room air stratification in combined chilled ceiling and displacement ventilation systems," HVAC R Res., vol. 18, nos. 1–2, pp. 147–159, 2012.

[48] A. Sangiovanni-Vincentelli and G. Martin, "Platform-based design and software design methodology for embedded systems," IEEE Design Test Comput., vol. 18, no. 6, pp. 23–33, Nov./Dec. 2001.

[49] W.-M. Hwu, K. Keutzer, and T. G. Mattson, "The concurrency challenge," IEEE Design Test Comput., vol. 25, no. 4, pp. 312–320, 2008.

[50] Thermal Environmental Conditions for Human Occupancy, Standard 55-2013, ASHRAE Standard, 2013.

[51] Comfy. Accessed: Jul. 25, 2017. [Online]. Available: https://comfyapp.com

[52] W. Pasut, H. Zhang, E. Arens, and Y. Zhai, "Energy-efficient comfort with a heated/cooled chair: Results from human subject tests," Building Environ., vol. 84, pp. 10–21, Jan. 2015.

[53] S. Schiavon, B. Yang, Y. Donner, V. W.-C. Chang, and W. W. Nazaroff, "Thermal comfort, perceived air quality, and cognitive performance when personally controlled air movement is used by tropically acclimatized persons," Indoor Air, vol. 27, no. 3, pp. 690–702, 2017.

[54] T. C. T. Cheung, S. Schiavon, E. T. Gall, M. Jin, and W. W. Nazaroff, "Longitudinal assessment of thermal and perceived air quality acceptability in relation to temperature, humidity, and CO2 exposure in Singapore," Building Environ., vol. 115, pp. 80–90, Apr. 2017.

[55] A. Kelman and F. Borrelli, "Bilinear model predictive control of a HVAC system using sequential quadratic programming," IFAC Proc. Vol., vol. 44, no. 1, pp. 9869–9874, 2011.

[56] Y. Zhou, D. Li, and C. J. Spanos, "Learning optimization friendly comfort model for HVAC model predictive control," in Proc. IEEE Int. Conf. Data Mining Workshop (ICDMW), Nov. 2015, pp. 430–439.

[57] D. B. Crawley et al., "EnergyPlus: Creating a new-generation building energy simulation program," Energy Buildings, vol. 33, no. 4, pp. 319–331, Apr. 2001.

[58] M. Wetter, "Modelica-based modelling and simulation to support research and development in building energy and control systems," J. Building Perform. Simul., vol. 2, no. 2, pp. 143–161, 2009.

[59] V. L. Erickson, M. Á. Carreira-Perpiñán, and A. E. Cerpa, "Occupancy modeling and prediction for building energy management," ACM Trans. Sensor Netw., vol. 10, no. 3, p. 42, 2014.

[60] H. Abdeen et al., "Multi-objective optimization in rule-based design space exploration," in Proc. 29th ACM/IEEE Int. Conf. Autom. Softw. Eng., 2014, pp. 289–300.

[61] H.-Y. Liu, I. Diakonikolas, M. Petracca, and L. Carloni, "Supervised design space exploration by compositional approximation of pareto sets," in Proc. 48th Design Autom. Conf., Jun. 2011, pp. 399–404.

[62] ANTLR. Accessed: Jul. 25, 2017. [Online]. Available: http://www.antlr.org/

[63] X. Luo, K. P. Lam, Y. Chen, and T. Hong, "Performance evaluation of an agent-based occupancy simulation model," Building Environ., vol. 115, pp. 42–53, Apr. 2017.

[64] M. Wetter, W. Zuo, T. S. Nouidui, and X. Pang, "Modelica buildings library," J. Building Perform. Simul., vol. 7, no. 4, pp. 253–270, 2014.

[65] J. Ma and J. C. P. Cheng, "Estimation of the building energy use intensity in the urban scale by integrating GIS and big data technology," Appl. Energy, vol. 183, pp. 182–192, Dec. 2016.

[66] J. Finn, P. Nuzzo, and A. Sangiovanni-Vincentelli, "A mixed discrete-continuous optimization scheme for cyber-physical system architecture exploration," in Proc. IEEE/ACM Int. Conf. Comput.-Aided Design, Nov. 2015, pp. 216–223.

[67] P. Nuzzo et al., "A contract-based methodology for aircraft electric power system design," IEEE Access, vol. 2, pp. 1–25, 2014.

[68] N. Bajaj, P. Nuzzo, M. Masin, and A. Sangiovanni-Vincentelli, "Optimized selection of reliable and cost-effective cyber-physical system architectures," in Proc. Design Autom. Test Eur., Mar. 2015, pp. 561–566.

[69] D. Kirov, P. Nuzzo, R. Passerone, and A. Sangiovanni-Vincentelli, "ArchEx: An extensible framework for the exploration of cyber-physical system architectures," in Proc. Design Autom. Conf., Jun. 2017, pp. 1–6.

[70] D. Kirov, P. Nuzzo, R. Passerone, and A. Sangiovanni-Vincentelli, "Optimized selection of wireless network topologies and components via efficient pruning of feasible paths," in Proc. Design Autom. Conf., 2018.

[71] A. Moin, P. Nuzzo, A. L. Sangiovanni-Vincentelli, and J. M. Rabaey, "Optimized design of a human Intranet network," in Proc. Design Autom. Conf., Jun. 2017, Art. no.30.

[72] T. Blochwitz et al., "Functional mockup interface 2.0: The standard for tool independent exchange of simulation models," in Proc. 9th Int. MODELICA Conf.; vol. 76. Linköping University Electronic Press, Munich; Germany, Sep. 2012, pp. 173–184.

[73] T. Nouidui, M. Wetter, and W. Zuo, "Functional mock-up unit for co-simulation import in EnergyPlus," J. Building Perform. Simul., vol. 7, no. 3, pp. 192–202, 2014.

[74] C. M. Stoppel and F. Leite, "Integrating probabilistic methods for describing occupant presence with building energy simulation models," Energy Buildings, vol. 68, pp. 99–107, Jan. 2014.

[75] K. M. Mosalam, U. Alibrandi, H. Lee, and J. Armengou, "Performance-based engineering and multi-criteria decision analysis for sustainable and resilient building design," Struct. Saf., vol. 74, pp. 1–13, Sep. 2018